Curiosity-driven AI for Science : Automated Discovery of Self-Organized Structures

Mayalen Etcheverry

In partial fulfillment of the requirements

for the degree of

Doctor of Philosophy
of the University of Bordeaux

Graduate school of Mathematics and Computer Science

Major in Computer Science

By **Mayalen ETCHEVERRY**

Curiosity-driven AI for Science: Automated Discovery of Self-Organized Structures

---

IA Curieuse au service de la Science:
Découverte Automatisée de Structures Auto-Organisées

Under the direction of **Pierre-Yves OUDEYER** & **Clément MOULIN-FRIER**

Submitted on September 29, 2023. Defended on November 16, 2023.

Composition of the jury :

Pr. Alan ASPURU-GUZIK	Professor	University of Toronto	Reviewer
Pr. Sebastian RISI	Professor	IT University of Copenhagen	Reviewer
Pr. Melanie MITCHELL	Professor	Santa Fe Institute	Examinator
Pr. Jeff CLUNE	Associate Professor	University of British Columbia	Examinator
Dr. Nicolas Brodu	Researcher	INRIA	Examinator
Dr. Pierre-Yves OUDEYER	Research Director	INRIA	Director
Dr. Clément MOULIN-FRIER	Researcher	INRIA	Director
Dr. Marc NICODEME	Research Engineer	Poietis	Supervisor

Contents

Acknowledgements

Thank you.

To my two academic supervisors, Pierre-Yves Oudeyer and Clément Moulin-Frier, to whom I am immensely grateful, in equal manners, for always giving me the space and trust that I needed to grow as a scientist. To Pierre-Yves, you have provided me with continuous support and invaluable guidance to "think big" since the beginning of my research, and for that, I am truly grateful. To Clément, your immediate engagement and sustained enthusiasm for the project has been truly invigorating and inspiring, and I thank you for that.

To Fabien Guillemot and Marc Nicodème, for offering me this CIFRE opportunity within the Poietis company. To Marc, particularly, for his kindness and for always giving me the freedom to explore and develop my personal research directions.

To all members of the FLOWERS team, past and present, with whom I've had the privilege of sharing this enriching experience. To past fellows – Cédric, Remy, Tristan, Laetitia, Alex Péré, Alex Ten, Benjamin, Chris, Eleni, Masataka – who not only I've shared pleasant moments with but who opened me up to few new perspectives on life. To present fellows – Grgur, Clément, Maxime, Rania, Gautier, Thomas, Isabeau, Marion, Matisse, and others – who have enlightened my last months in the team, despite the darkness of thesis writing: thank you!! To Hélène, Nathalie, and Cécile for their kindness and for bringing their refreshing energy into the team.

Special thanks to Chris Reinke, my mentor when I arrived in the team, who provided me with all the necessary tools to swiftly and quickly immerse myself in the research topic. Special thanks as well to the bachelor students – Lucie, Marion and Théo – and master students – Gautier and Erwan – whose internships I had the chance to co-supervise. In particular, the works of the very talented Gautier Hamon and Erwan Plantec on the Lenia project led to great results and scientific publications presented in Chapter **??**, and I learned a lot from you! I also had the opportunity to collaborate with several engineers – Clément Romac, Mathieu Perie and Jesse Lin – who did a great work for developing the ADTOOL software presented in Chapter 10.

To all members of the Poietis company, both "technos" people – Marc, Thibault, Boris, Mathieu, Antonio, Guillaume, Alain, Alexandre, Guillaume, Mickael, Lucas – and "bios" people – Hélène, Loïc, Camille, Laurence, Vanessa, Etienne, Marine, Diane, Manon, Deepshika – and all others, thank you for your joviality and for the enjoyable lunch breaks in your company. Special thanks to Marc, Dan, Bertrand, Antonio and Fabien for their scientific guidance, as well as to Bruno, Annick, and Karine for their kind support. Thank you also to Pascal Ballet and Arthur Douillet for providing the SimCells simulator presented in Chapter 9.

To Michael Levin, and his amazing team at Tufts University, Boston, that I had the privilege to be part of for a few months. Michael's groundbreaking ideas on cognition, agency, and biology not only shaped many of the research questions presented in Part II of this manuscript but also fundamentally challenged my perspective and understanding of the world. Special thanks to Wes for his warm welcome in the team, as well as to Santosh, Hananel, Hamid, Nick, and Franz for their kindness and for sharing their knowledge.

To my family and friends in Bordeaux, Paris, Basque Country, Croatia and Germany: thank you for always supporting me and for providing much-needed breaks from the seriousness of a PhD life ♡

Last but not least, a big thanks to my chapeautiot for sharing every bit of this journey together, all the ups and downs, from our very deep discussions to those hilarious moments when we completely lost it. You have been my rock and so much more throughout this messy journey, and I hope we can share many more messy years together ♡

Abstract

Curiosity-driven AI for Science: Automated Discovery of Self-Organized Structures

Abstract: Complex systems are very hard to predict and control due to their chaotic dynamics and open-ended outcomes. However, understanding and harnessing the underlying mechanisms of these systems hold great promise for revolutionizing many areas of science. While considerable progress has been made in manipulating and measuring system activity down to the lowest level, there remains a fundamental gap between our knowledge at the micro-level and our ability to control resulting properties on a global scale. Modern machine learning tools offer promising avenues for assisting scientists in navigating the vast space of possible outcomes, especially when aiming for novel or challenging morphological or functional objectives. Nevertheless, current methods tend to constrain and bias the range of events that AI can measure and attempt to influence. This thesis aims to transpose and advance recent computational models of intrinsically motivated learning and exploration with the goal of designing more open-ended forms of AI "discovery assistants" for assisting scientists in mapping the outcome space of self-organizing systems. To that end, several key ingredients are introduced to efficiently shape the discovery process. These include the use of unsupervised learning for representations, meta-diversity search, curriculum learning, and external human guidance, whether environment-based or preference-based. We discuss how these components, when implemented in practice, can help address challenging problems in science. These challenges encompass the search for interesting patterns in continuous models of cellular automata, the investigation of the origins of sensorimotor agency, the exploration of gene regulatory networks behavioral capabilities, and the design of innovative forms of cellular collectives for applications in AI and biology.

Keywords: AI for Science, Machine Learning, Artificial Curiosity, Scientific Discovery, Complex Systems, Collective Intelligence

---

IA Curieuse au service de la Science: Découverte Automatisée de Structures Auto-Organisées

Résumé : Les systèmes complexes sont très difficiles à prédire et à contrôler en raison de leur dynamique chaotique et de leurs vastes espaces de sortie. Cependant, comprendre et exploiter les mécanismes sous-jacents de ces systèmes offre de grandes promesses pour révolutionner de nombreux domaines scientifiques. Bien que des progrès considérables aient été réalisés dans la manipulation et la mesure de l'activité des systèmes jusqu'au niveau microscopique voire nanoscopique, un fossé fondamental persiste entre nos connaissances à l'échelle microscopique et notre capacité à contrôler les propriétés résultantes à l'échelle globale. Les outils modernes d'apprentissage automatique offrent des perspectives prometteuses pour aider les scientifiques à naviguer dans l'espace complexe des sorties du système, en particulier lorsqu'il s'agit d'atteindre de nouveaux buts morphologiques ou fonctionnels difficiles. Néanmoins, les méthodes actuelles ont tendance à restreindre et à biaiser l'étendue des événements que l'IA peut mesurer et tenter d'influencer. Cette thèse vise à appliquer et à développer les modèles computationnels récents d'apprentissage et d'exploration intrinsèquement motivés dans le but de concevoir des "assistants de découverte" IA pour aider les scientifiques à cartographier les résultats potentiels des systèmes auto-organisés. Pour atteindre cet objectif, plusieurs éléments clés sont introduits pour façonner efficacement le processus de découverte. Cela comprend l'utilisation de l'apprentissage non supervisé de représentations, la recherche de méta-diversité, l'apprentissage par curriculum, et l'intégration de guidage humain dans la boucle (par l'introduction de contraintes environnementales ou de préférences). Nous discutons de la manière dont ces composants, lorsqu'ils sont mis en pratique, peuvent contribuer à résoudre des problèmes scientifiques complexes. Cela comprend la recherche de motifs intéressants dans des modèles continus d'automates cellulaires, l'investigation des origines de l'agence sensorimotrice, l'exploration des capacités comportementales des réseaux de régulation génétique et la conception de formes innovantes de collectifs cellulaires pour des applications en IA et en biologie.

Mots-clés : IA pour la Science, Apprentissage Automatique, Curiosité Artificielle, Découverte Scientifique, Intelligence Collective

---

FLOWERS team, INRIA
Université de Bordeaux, 33000 Bordeaux, France.

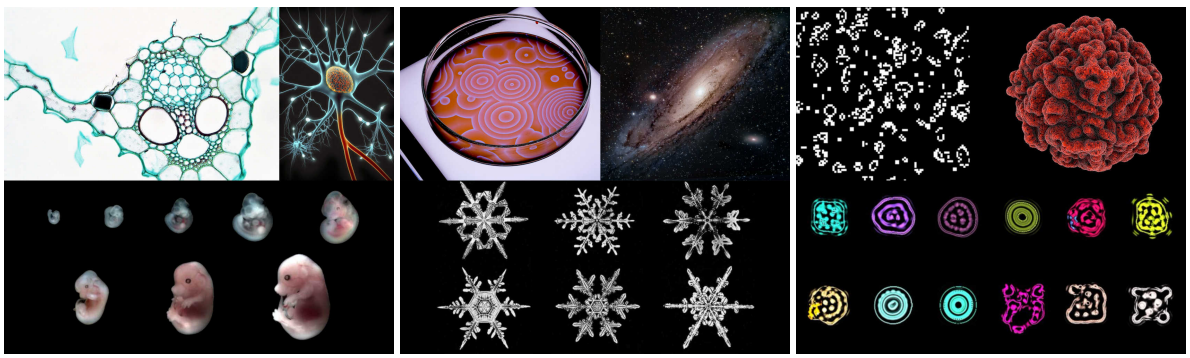
1. Introduction

1.1. Self-Organization and its Role in the Evolution of Forms

The natural world, in all its diversity, showcases various complex patterns and structures that arise from the interactions among its many components. In the living, cells work together to create highly organized biological tissues, organs, and even even full organisms endowed with complex brains (Figure 1.1a). In the inorganic world, water molecules crystalize in fascinating hexagonal snowflakes observable under the microscope, and stars arrange themselves into gigantic flat spiral galaxies with bulging centers visible with a telescope (Figure 1.1b). This inherent propensity of particles, molecules, or cells to self-organize into structured patterns is a phenomenon known as "self-organization".

One key characteristic of self-organizing systems is that one cannot predict the bigger, organized structures simply by looking at the individual parts. For example, the exact shape of a snowflake cannot be predicted by the physico-chemical properties of the water molecules it's made of, nor can the development of an organism be predicted simply by analyzing its genetic code. Remarkably these structures are not the result of some central plan or control; they emerge on their own through self-organization and yet, they rival many human artefacts both in organizational and functional complexity.

Scientists are particularly interested by these organizational properties, but they are often challenging to comprehend or predict intuitively. Even in artificial self-organizing systems, where engineers deliberately program the behavior of individual components and simulate them on computers (Figure 1.1c), scientists still lack an effective understanding of the range of possible behaviors and structures that can emerge, nor how to categorize them or how to predict their evolution. To better understand these phenomena, empirical experimentation with complex dynamical systems have become pervasive across various domains of science.



(a) Self-Organization in the Living World **(b)** Self-Organization in the Inorganic World **(c)** Self-Organization in the Artificial World

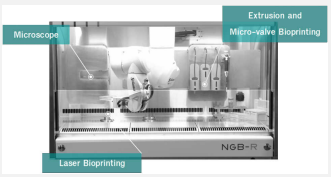
Figure 1.1.: Example of complex dynamical systems with self-organizing dynamics in the natural and artificial world.

1.2. Experimenting with Complex Dynamical Systems for the Discovery of Novel Outcomes

Many of today's scientific grand challenges, that we would like to solve as a society, involve the manipulation, exploration and control of complex self-organizing systems. On the one hand, these challenges span fundamental scientific questions such as unraveling the conditions that led to the emergence of life from a complex milieu of primordial elements [9, 10] or the remarkable regenerative capabilities observed in specific animal species [11]. On the other hand, they span central applicative quests such as the discovery of novel drug compounds [12] or bio-engineered constructs [13, 14] for clinical applications, as well as the design of clean-energy materials [15] or even "living machines" [16] that could potentially offer solutions to the current pressing environmental concerns.

Whereas the space of self-organized shapes, forms and functions has historically been shaped by the interplay of self-organization and natural selection (as well as historical environmental factors), scientific experimentation is now playing a key role in orienting innovation within that space. Not only it can foster the emergence of novel self-organized structures within that space, but it can progressively redefine the boundaries within which self-organization (hence scientific discovery) proceeds.

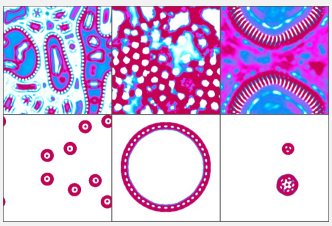
Technological Advances at the Laboratory In the recent years, significant advancements in technology have empowered scientists with the capacity to manipulate and measure self-organizing phenomena at microscopic and nanoscale levels within controlled laboratory environments, fostering empirical experimentation in various scientific domains. For instance, in developmental biology, the mechanisms underlying cellular morphogenesis, the process by which cells undergo structural and organizational changes to form complex tissues and organs, are now extensively studied at the bench. Similarly, in chemistry and materials science, the process of molecular self-assembly, wherein individual molecules autonomously arrange themselves into organized structures, is explored with unprecedented levels precision. Furthermore, the integration of these advanced technological devices into automated robotic platforms has given rise to what are commonly referred to as "self-driving laboratories" (SDL). SDLs deliver many advantages such as precision, reproducibility, parallelization, observability, and data production [17–19].



For instance the Poietis company, with which I was working during my PhD, has developed a laser-assisted bioprinting technology that allows to position cells in three dimensions with micrometric resolution [20].

Computational Modeling Advances Simultaneously, significant progress has been made in the field of numerical self-organization, where researchers have developed abstract models for simulating and exploring organizational aspects within computer-based environments. These models can be categorized into two main types: artificial life systems, which aim to generate and study various life-like behaviors such as agency, cognition, and open-ended evolution; and modeling systems, which use experimental data and scientific knowledge to simulate various physical, chemical, and biological self-organizing phenomena. Historically, the

simulation of artificial worlds with life-like characteristics can be traced back to von Neumann's work on self-reproducing cellular automata in the 1940s [21, 22], while the concept of replicating patterns observed in nature from simple rules and non-organized matter can be traced back to Alan Turing's seminal paper on *"The Chemical Basis of Morphogenesis"* in 1952 [23]. Since these foundational works, a broad range of numerical self-organizing systems have been proposed, such as cellular automata, reaction-diffusion systems, agent-based models and particle systems. These models have been explored across various disciplines, including biology [24–27], physics [28, 29], chemistry [30] as well as artificial life [31–34]. Recently, we observe a renewal of interest around these traditional models with the introduction of novel data structures [35, 36], differentiable programming tools [37], and large-scale simulations [38], bringing new expressivity levels to the simulated patterns and dynamics.



For instance the Lenia system, a continuous extension of the game of life which we will explore extensively in this manuscript, draws inspiration from classical CA and RD systems [39, 40].

Scientific Exploration: Limits of Conventional Search Methods Despite the automatization of the experimental process, the sequence of input parameters which organizes the scientific exploration of these systems remains often generated manually by a few individual or team of experts. Even when experimental budget allows to deploy massive randomized trials with the hope to achieve interesting discoveries, *e.g.* especially used in high-throughput screening for drug discovery, this is often very inefficient due to the complexity and numerous interactions involved. In fact, a significant "knowledge gap" persists between our ability to manipulate the low-level inputs of these systems and our ability to efficiently discover novel and controllable properties at the system level. This gap persists in various domains, such as drug discovery and genetic engineering, where random exploration strategies are inefficient in reaching unexplored regions of chemical space [41, 42] or predicting the effects of genomic changes on transcriptional, physiological, morphological, and behavioral levels [43–45]. As illustrated in Figure 1.2, the manipulation, exploration, prediction and control of complex systems requires working "at the edge of chaos" [46, 47], *i.e.* dealing with excessively large search spaces (if not infinite) and unstable chaotic dynamics. Notably, their exploration and understanding poses many challenges with respect to simpler physical or engineered systems. To navigate the complex data landscape of these systems more effectively, the development of novel computer tools, especially those harnessing the potential of artificial intelligence (AI), holds great promise.

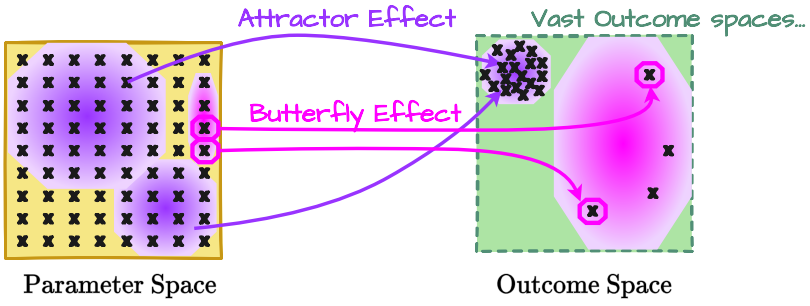


Figure 1.2.: The "knowledge gap". Exploring the outcome space of dynamical complex systems poses many challenges due to their nonlinearity, presence of dynamical systems "attractors", sensitivity to initial conditions (the "butterfly" effect), and very large exploration spaces.

1.3. Machine Learning for Guiding Experimentation: Opportunities and Challenges

Whereas "AI for Science" is traditionally employed in contexts where there is already an experimental database and AI tools are used for analyzing that data (*e.g.* for extracting regularities or making predictions), here, we are rather interested in developing AI tools to *collect new data* in a dynamical system. We specifically target discovery within systems with many interacting entities capable of self-organization, referred to as *self-organizing* or *complex* dynamical systems. The primary goal is to use AI for assisting scientists in their exploration and search for novel discoveries within these systems, which is what we call the *automated discovery* problem[1]. Despite the recent advancements in high-throughput experimental platforms, a common property in many of these systems is that it is expensive in time and energy to conduct experiments. An important requirement of the automated discovery problem, in addition to addressing the many exploration challenges due to the nonlinearity and chaoticity of the dynamics involved, is that exploration must be done under a *limited experimental budget*. This budget is often orders of magnitude smaller than what's typically available in traditional machine learning testbeds, calling for advanced and sample-efficient AI exploration methods.

1: The automated discovery problem is formalized in Chapter 2

There is already a large body of work that proposes to exploit the use of *optimization* and *prediction* machine learning methods to better navigate the high-dimensional parameter space of complex systems, and for accelerating the discovery of self-organized structures in these systems. For instance, as we will see in the next chapter, optimization methods have successfully been employed in various applications within physico-chemical wet systems, such as catalyst design [48], carbon nanotube growth [49] and in the discovery of alloy materials with shape memory [50], among others. Likewise, prediction methods that use and learn surrogate models have been proposed to guide scientific experimentation, for instance to predict the crystallization of supra-molecular compounds [51] or which genes are involved in yeast metabolism [52]. Approaches combining predictive models and optimization were also successfully applied, for instance in biology for maximizing cell-level metabolic yields [53].

However, applying these methods in practice is not straightforward as they often require the prior knowledge of an expert both for the *task characterization* (oracle or reward function that maps the raw observations to a low-dimensional space) and for the *task achievement* strategy (either under the form of a good forward/inverse surrogate or of a good initialization) to successfully navigate the chaotic optimization landscape. A major limitation is that scientists might not even know what to look for in the first place nor how to characterize it from raw observations. More importantly, current methods largely restrict and bias the boundary of events that the AI can measure and try to affect: they are powerful problem solvers but for a narrow span of problems that are imposed in advance by the machine learning engineer or the expert.

1.4. Towards AI-driven Curious Discovery Assistants for Science

A long-standing aspiration for automated discovery is to construct autonomous AI agents that could learn to represent, generate, select, and solve their own problems in order to efficiently explore the vast outcome space of artificial and natural complex systems. For scientists, such AI "discovery assistants" could enhance the likelihood of making novel and interesting discoveries, while using a low budget of experiments [54].

Humans are an incredible source of inspiration for achieving that purpose: despite energy and time limitations, humans develop a diverse skill set from a myriad of possibilities throughout their lives. This inherent ability to autonomously pursue goals in an open-ended manner is a crucial aspect of exploration and learning: humans are *autotelic*[2] learners. Autotelic learning, yet, is often absent from standard AI paradigms. Instead of self-generating and pursuing their own goals, AI agents generally pursue externally defined goals (and their corresponding learning signals), relying on engineers to pre-define the set of tasks and associated rewards.

2: "autotelic" comes from the Greek *auto* (self) and *telos* (goal). Autotelic agents are ones that can generate and pursue their own goals [55]

Drawing inspiration from human open-ended learning abilities, the field of *developmental robotics* has brought together a diverse community of researchers from both AI and developmental sciences to try and reproduce similar learning mechanisms in artificial agents, including physical robots. This interdisciplinary endeavor is centered on the development of computational models rooted in *goal-directed* and *intrinsically-motivated* learning processes, which is also called *autotelic curiosity search*. In contrast to conventional AI algorithms focused on predefined optimization or prediction tasks, developmental robotics is rooted in creating autonomous machines capable of learning to represent, generate, and pursue a diversity of self-generated goals [56]. Several studies in developmental robotics have shown how the integration of these models into robotic agents can foster the development of efficient exploration trajectories while working with limited resources [57, 58].

In the recent years, the developmental robotics field has merged into the broader field of *developmental artificial intelligence* (developmental AI). Developmental AI proposes to combine the autotelic computational models with modern deep learning methods and increased compute power (see Colas et al. [59] for a review). This approach is aimed at fostering adaptive exploration and learning in complex environments with high-dimensional state spaces, providing AI agents with control over their learning experiences and developmental trajectories.

1.4.1. Objective, Approach and Contributions

The general objective of the present research is to make progress toward constructing artificial agents that form efficient AI-driven "discovery assistant" and help addressing various challenging problems in Science.

To this end, we propose to transpose the recent approaches from the field of *developmental AI* to the targeted application of *automated discovery in self-organizing systems*, as they revolve around the same fundamental

objective: exploring and mapping, in a sample efficient manner, the space of possible behaviors of unfamiliar dynamical systems.

In Chapter 2, we provide an overview of important concepts and research directions in AI-driven scientific experimentation on one hand, and developmental AI on the other. As our initial contribution, we introduce a formalization of the *automated discovery* problem and survey the standard AI paradigms employed to address this issue. We then propose a novel perspective on the problem within the context of developmental AI.

Next, the main contributions of the present research are two fold. In the first part of this manuscript (Part I), we present several contributions that target *algorithmic developments* of foundational curiosity-driven exploration algorithms, initially developed to enable open-ended autonomous learning in AI, for their reuse and extensions as digital assistant helping scientists to explore and map new self-organized complex systems.

In the second part of this manuscript (Part II), we shift our focus towards the *practical applications* of these curiosity-driven exploration algorithms, even in their simplest form. We demonstrate their effectiveness in tackling challenging scientific problems around two example applications.

In the third part of this manuscript (Part III), we present *preliminary* experiments and *software* contributions aimed at expanding the practical applications to a broader range of scientific problems, and disseminating the developed algorithms to a broader interdisciplinary audience.

Finally, in the DISCUSSION, we provide several perspectives for future applications in several scientific domains.

Part I: The "Curious Discovery Assistant" Framework

In Chapter 3, we propose to transpose intrinsically-motivated goal exploration processes (IMGEP), a family of algorithms originally developed for learning inverse models in the context of developmental robotics, to guide scientific experiments in complex dynamical systems. In particular, we introduce the Lenia environment, a class of continuous cellular automata (CA) models with rich possibilities for emergence, and discuss the several challenges that their exploration pose in practice.

To address those challenges, we propose to integrate three key algorithmic ingredients within the IMGEP computational framework.

The first proposed ingredient, presented in Chapter 4, is to combine intrinsically-motivated goal exploration with *unsupervised* and *online* learning of goal space representations that are incrementally refined using the data collected by the AI agent during its exploration. We show how this approach forms a promising framework to address the problem of automated discovery of diverse self-organized patterns. On the one hand, it removes the need for human expertise to define such representations, and on the other it proves more effective than several baselines in discovering diverse spatially localized patterns in Lenia.

The second proposed ingredient, presented in Chapter 5, encompasses both a conceptual and algorithmic aspect. First, we introduce the novel concept of *meta-diversity* search, which extends the standard notion of diversity, and where an agent incrementally learns diverse behavioral

characterization spaces and aims to discover diverse patterns within each of them. As one possible implementation of the meta-diversity search concept, we then propose to combine a dynamic and modular model architecture, called HOLMES, for unsupervised learning of *diverse representation spaces* with intrinsically motivated exploration processes operating within the learned spaces. We demonstrate the effectiveness of the proposed approach in learning to characterize and explore diverse pattern niches in the Lenia system, which notably leads to the discovery of novel glider-emitting structures from unexpected pattern niches.

The third proposed ingredient, presented in Chapter 6, is to incorporate *human guidance* into the discovery process, transitioning from a fully automated process to a more *assisted discovery* approach. To that end, we introduce a simple variant of the previous meta-diversity search approach that, leveraging the modularity of HOLMES architecture, biases the AI exploration toward *preferred* representation spaces. Considering two end-user models, respectively interested in two types of diversities (diverse spatially localized and diverse Turing-like patterns), we show that the approach can effectively adapt the search toward these two types of "interesting" diversities with minimal simulated user feedback.

Part II: Use Cases of the Curious Discovery Assistant

In Chapter 7, we show how curiosity-driven exploration algorithms can be used to explore and discover the self-organization of forms of *agency* with *robust sensorimotor behaviors* in continuous cellular automata, which is our first applicative use case. Moreover, we propose an extensive battery of quantitative and qualitative tests to characterize the robustness of the discovered self-organized agents. To our knowledge, this is the first time that "robust sensorimotor agents" have been automatically discovered and systematically characterized (evaluated across a set of robustness tests) in cellular automata. The discoveries themselves constitute one of the major contributions of this chapter, although several novel methodological components are also introduced. Those include the use of gradient descent for local optimization and the use of stochastic perturbations and curriculum learning for the goal sampling strategy.

Additionally, still within Chapter 7, we present Flow Lenia, an extension of the original Lenia system which introduces *mass conservation* in the CA dynamics (Section 7.5). Although this is not a "use case" application but rather a parallel contribution, we discuss its connections with the previous use-case as well as the several benefits it provides for the study of artificial life forms, together with the perspectives it opens towards achieving *open-ended evolution* in such an artificial substrata.

As our second use case, in Chapter 8, we show how the curiosity-driven algorithms can also be very useful to assist *biologists* mapping the space of behaviors of gene regulatory networks (GRNs), even in their simplest version. We demonstrate how the discovered "behavioral catalogs" can in turn be very useful to study forms of *non-genetic developmental robustness* in these biological networks, as well as for the development of *intervention strategies* both in biomedical and bioengineering contexts.

Please note that all the content presented in these applicative chapters originates from publications we recently authored in interdisciplinary

scientific journals (Hamon et al. [•5], Plantec et al. [•6], and Etcheverry et al. [•8]), and which we replicated here with minor revisions for clarity.

Part III: Targeted Reuses of the Curious Discovery Assistant

In Chapter 9, we present a series of *preliminary* numerical experiments as well as *envisaged* real-world experiments, that have been devised within the context of exploring behaviors of cellular collectives at the tissue within *biological morphogenetic systems*. In particular, in the context of the collaboration with the Poietis company, which specializes in the development of a *bioprinting* technology, we present few experimental campaigns that could be envisaged as proof of concept of automated AI-driven exploration in a bioprinter-controlled morphogenetic system.

In Chapter 10, we present the two major *software* contributions of this research. The fist one, ADTOOL, is aimed at making the developed algorithms accessible to a broader community of scientists. The second one, SBMLtoODEjax, proposes a lightweight JAX-based library harnessing advancements in high-performance computing and differentiable programming to bridge the gap between ML research and biological network analysis.

1.4.2. Collaborations

This thesis reflects the outcomes of several interdisciplinary collaborations. My two thesis supervisors from the Flowers Lab, Clément Moulin-Frier and Pierre-Yves Oudeyer, were involved in all collaborations.

First, several collaborations were made within the Flowers Lab. Chris Reinke, a postdoc who mentored me upon my arrival, co-authored the work presented in Chapter 4. I also had the privilege to supervise multiple interns, who all contributed positively to my research. This included mentoring three undergraduate students, Lucie Galland, Marion Schaeffer, and Théo Goix, during their 2-3 month internships, and supervising the work of two outstanding master's students, Gautier Hamon and Erwan Plantec, whose exceptional contributions to the Lenia project resulted in the notable publications presented in Chapter 7. Finally, I collaborated with several engineers from the Flowers Lab, Clément Romac, Mathieu Perie, and Jesse Lin, who worked on developing the open-source and interactive software presented in Chapter 10.

This research also involved several collaborations with Bert Chan, creator of the Lenia system, now working in the Google Brain team in Tokyo. Bert Chan participated to the development of research directions within the Lenia project and co-supervised the works presented in Chapter 7.

In the final year of my Ph.D., I also had the privilege of spending five months at Tufts University in Boston, USA, collaborating with Dr Michael Levin and his team. Dr Levin was a great source of inspiration for several applicative works of this research. Our collaboration led to the results presented in Chapter 8, as well as some preliminary results presented in Chapter 9 and a software contribution presented in Chapter 10.

Finally, this thesis was conducted in collaboration with Poietis, a biotechnology company which specializes in bioprinting and with whom I've

spent part of my time under CIFRE contract. This gave me the opportunity to engage with several biologists and engineers from the company, notably with my supervisor Marc Nicodeme with whom I've devised the envisaged experimental campaigns presented in Chapter 9.

1.4.3. Publications, Code and Other Materials

The work presented in this thesis is based on the following publications[3], as well as accompanying codebases[4] and other materials:

[•1] Chris Reinke, Mayalen Etcheverry, and Pierre-Yves Oudeyer. "Intrinsically Motivated Discovery of Diverse Patterns in Self-Organizing Systems". In: *Eighth International Conference on Learning Representations*. 2020. **ICLR 2020, Oral presentation**
[Blogpost] [Website] [Oral] [Code]
Cited on pages 11, 28, 30, 57, 60, 71, 72, 79, 103, 108, 128, 132, 146, 208, 209, 213, 214, 233, 249

[•2] Mayalen Etcheverry, Pierre-Yves Oudeyer, and Chris Reinke. "Progressive Growing of Self-Organized Hierarchical Representations for Exploration". In: *Beyond 'tabula Rasa' in Reinforcement Learning: Agents That Remember, Adapt, and Generalize*. 2020. **ICLR 2020 Workshop**
[Oral]
Cited on page 81

[•3] Mayalen Etcheverry, Clément Moulin-Frier, and Pierre-Yves Oudeyer. "Hierarchically Organized Latent Modules for Exploratory Search in Morphogenetic Systems". In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 33. 2020. **NeurIPS 2020, Oral presentation**
[Website] [Oral] [Poster] [Code]
Cited on pages 11, 13, 28, 30, 34, 75, 76, 87, 103, 104, 128, 132, 146, 211, 249

[•4] Mayalen Etcheverry, Bert Wang-Chak Chan, Clément Moulin-Frier, and Pierre-Yves Oudeyer. *Meta-Diversity Search in Complex Systems, a Recipe for Artificial Open-Endedness?* 2021. **GECCO 2021 Competition, Runner-up Prize**
[Video] [Blogpost]
Cited on pages 46, 87

[•5] Gautier Hamon, Mayalen Etcheverry, Bert Wang-Chak Chan, Clément Moulin-Frier, and Pierre-Yves Oudeyer. *Learning Sensorimotor Agency in Cellular Automata*. 2022. **In Submission**
[Blogpost] [Website] [Notebook] [Code]
Cited on pages 8, 28, 30, 46, 47, 119, 128, 238, 244

[•6] Erwan Plantec, Gautier Hamon, Mayalen Etcheverry, Pierre-Yves Oudeyer, Clément Moulin-Frier, and Bert Wang-Chak Chan. "Flow-Lenia: Towards Open-Ended Evolution in Cellular Automata through Mass Conservation and Parameter Localization". In: *Proceedings of the Artificial Life Conference*. MIT Press, 2023. **ALIFE 2023, Best Paper Award**
[Website] [Notebook]
Cited on pages 8, 33, 47

3: our publications are referenced with the • symbol throughout this manuscript

4: all code repositories are released under open-source MIT license

[•7] Mayalen Etcheverry, Michael Levin, Clément Moulin-Frier, and Pierre-Yves Oudeyer. *SBMLtoODEjax: Efficient Simulation and Optimization of ODE SBML Models in JAX*. 2023. **Software**
[Paper] [Code] [Documentation] [Tutorials]
Cited on pages 23, 246, 247

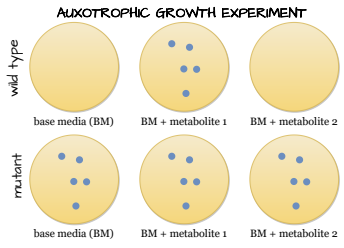
[•8] Mayalen Etcheverry, Clément Moulin-Frier, Pierre-Yves Oudeyer, and Michael Levin. *AI-driven Automated Discovery Tools Reveal Diverse Behavioral Competencies of Biological Networks*. 2023. **In Submission**
[Executable Paper] [Tutorials]
Cited on pages 8, 28, 30, 78, 93

Automated Scientific Discovery in Complex Systems

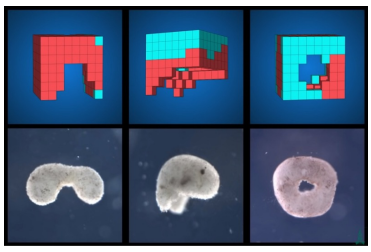
2.

What is the aim of this chapter? This chapter aims to provide an overview of the current approaches in machine learning assisted scientific discovery and to propose a novel *developmental AI* perspective.

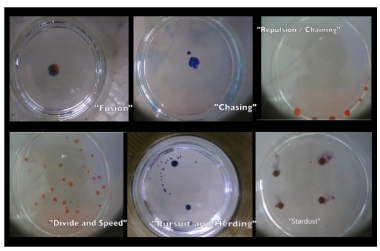
How is this chapter organized? In Section 2.1, we formalize the *automated discovery* (AD) problem, propose possible evaluation criteria, and highlight the primary challenges encountered in the targeted applications. In Section 2.2, we discuss typical use-cases of machine learning agents for guiding scientific experimentation which we call the "standard" AI-driven exploration strategies, and for which we survey a set of recent papers. Then, in Section 2.3, we argue in favor of more open-ended forms of discovery assistants integrated within a broader *developmental AI* perspective, and discuss the fundamental challenges it entails.



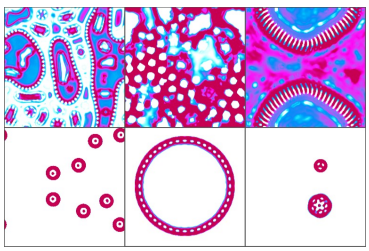
(a) Auxotrophic growth experiments: testing functional genomic hypoheses



(b) Computer-designed organisms: constructing novel functional morphologies



(c) Oil-in-water droplets experiments: discovery of novel protocell behaviors



(d) Continuous cellular automata (CA): studying life and cognition *as-it-could-be*

Figure 2.1.: Examples of real-world and numerical complex systems explored using AI-driven strategies. These include (a) the use of the *Adam* robot scientist to predict genes encoding enzyme catalysts for yeast growth [52, 60]; (b) optimization of *Xenobots* to perform specific functions in their environments [61–63]; (c) exploration of oil-in-water droplet dynamics with a curious robot [64]; and (d) the study of *Lenia* [39, 40] using AI algorithms to uncover diverse self-organized behaviors [•1, •3]. Here, the AI-driven discoveries are useful both for fundamental research, *e. g.* for better understanding genetic pathways, protocell formation and the emergence of life and cognition, as well as for practical applications like the development of functional biological machines. A more detailed discussion of these examples can be found in Section 2.2.

2.1. The Automated Discovery Problem

When exploring a complex system, conducting an *experiment* consists in 1) choosing the experimental parameters, 2) launching an experiment in the system with the input parameters, 3) measuring the outcomes into some numerical vectors describing some of the observed properties, and 4) collecting the results in a database. The *automated discovery* (AD) process consists in having a machine iteratively performing experiments in the complex system, where an AI-driven algorithm is in charge of deciding the experimental parameters while being given a limited budget of N experiments.The final outcome is the database of discoveries which has been collected throughout experimentation, with eventually (but not necessary) a set of learned representations (*e.g.* forward or inverse models of the environment dynamics). In this section, we first formalize the AD procedure as the *exploration process* of an AI-driven[1] *agent* within a complex *environment* (the targeted system). We then discuss, from the perspective of the scientist end-user, possible ways to evaluate the agent's discoveries and learned representations. Finally, we explain the fundamental challenges that arise in the targeted dynamical systems with respect to simpler physical or engineered systems.

1: note that, as we discuss later in the chapter, the "AI-driven" strategy could potentially be influenced by external guidance, typically from a human end-user

2.1.1. Formalism

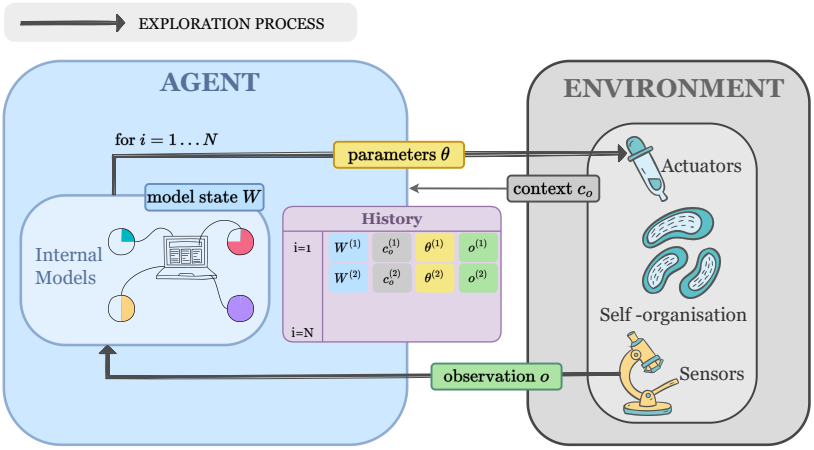


Figure 2.2.: The automated scientific discovery procedure is defined as the sequential interaction between a *agent* and an *environment*. The "agent-environment" perspective is a terminology which we use to emphasize that 1) everything related to the engineering of robotic platforms and to the experimental orchestration software is assumed given as part of the environment (and out of the scope of this thesis), and 2) we are interested here on the engineering on the AI agent and on its capacity to actively shape the discovery process in open-ended and innovative ways.

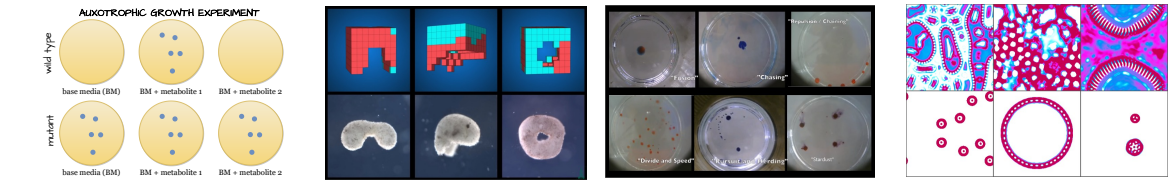
In this chapter, as illustrated in Figure 2.2, we follow a terminology commonly employed in machine learning (and in particular in reinforcement learning) where we call *environment* the complex system that one wishes to explore and *agent* the AI-driven algorithm in charge of generating the sequence of experimental parameters. Here, the automated discovery process is guided by the agent's *exploration process* which actively decides how to interact with the environment. Finally we discuss the role of the *human* in guiding the agent's exploration process, in which we distinguish the human *engineer* and human *end-user*.

The Environment The environment is typically a chemical, biological or numerical system that can be interfaced via some actuators and sensors. An experimental *rollout* in the environment consists in sending some input *parameters* θ to the actuators, letting the system evolve

through time via its inherent self-organizing dynamics, and observing the output *observation* o returned by the sensors. In addition to the input parameters, a rollout in the environment is conditioned by the experimental *context* $c \sim C$, which refers to all the source of incoming variations that are not controlled or known by the experimenter (such as noise, provenance of materials or temperature of the room). The context can be decomposed into the observable context c_o, *i.e.* the one that that is measurable (such as room temperature), and the unknown context c_u (which can be empty in the case of deterministic numerical system or very large in the case of biological systems). Overall, the environment is considered as a black-box[2] mapping $f : \Theta \xrightarrow{(c_o, c_u) \sim C} O$ with complex and self-organizing dynamics. In Table 2.1, we illustrate example actuators, self-organizing phenomena, sensors, context (c_o, c_u), parameters θ and observation o for the environments from Figure 2.1.

2: The term "black box" is employed to highlight the fact that certain internal states and processes of the environment are often hidden from both the agent and the scientist. However, it is not intended as a strict limitation, as certain exploration strategies may leverage knowledge of the underlying dynamics

Table 2.1.: Examples environments from Figure 2.1.
The parameters θ can represent physically interpretable quantities (*e.g.* chemical composition of the media in (a) or droplets in (c)) or more abstract quantities (*e.g.* parameters of the CA rules in (d)).
The observation o are usually raw measurements of the systems dynamics (*e.g.* timelapses in (a-c-d)) but they can also be more human-interpretable (*e.g.* growth curves in (a)) or the result of some post-processing (*e.g.* droplet tracking in (d)).
The observable context c_o can represent physical quantities that influence the resulting dynamics (*e.g.* temperature in (c) was shown to have big impact on the discoveries [64]) but it can also contain prior "knowledge" about the system when available (*e.g.* in (d) the CA rules f could be provided to the agent, especially if f is differentiable, to be exploited during the exploration process).



	(a)	(b)	(c)	(d)
	liquid handlers, robot arms, ...	$\emptyset$, pattern-generator, ...	robotic platforms (oil filling, stirring, ...)	$\emptyset$, image-generator, ...
	yeast growth	voxel interactions	self-propelling	CA evolution
	plate reader, growth curve processing	screen capture, image post-processing, ...	camera timelapse, post-processing, ...	screen captures, post-processing, ...
c_o	strains provenance, ...	$\emptyset$	temperature, ...	$\emptyset$ or f
c_u	patterning noise, ...	Stochasticity, ...	Actuator noise, ...	Noise in the CA, ...
θ	gene knockout, media, ...	voxel configurations, tissue options	concentrations, droplets position, ...	initial grid A^0, CA rules' parameters
o	optical density, extracted features, ...	center of mass distance, voxel's trajectories, ...	raw video $[A^0, ..., A^T]$, droplet trajectories, ...	raw video $[A^0, ..., A^T]$, extracted features, ...

(a) Auxotrophic growth experiments [52, 60]
(b) Computer-designed organisms [61–63]
(c) Oil-in-water droplets experiments [64]
(d) Continuous cellular automata [•3]

The Exploration Process Given an environment and an experimental budget of N rollouts, the *exploration process* is defined as the sequential process that iterates N times[3] through the following steps: 1) Observe the experimental context c_o; 2) Generate the experimental parameters θ; 3) Execute a system rollout $f : \theta \xrightarrow{(c_o, c_u) \sim C} o$; 4) Observe the outcome o; 5) Store the discoveries (c_o, θ, o) in history $\mathcal{H}$. The history $\mathcal{H}$, which contains all the information that has been generated by the exploration process, is returned at the end.

3: or $\frac{N}{n}$ for environments allowing parallel execution of n experimental rollouts

The Agent The agent is the entity in charge of generating the sequence of experimental parameters $\theta_1, \theta_2, \ldots, \theta_N$ that will drive the exploration process. We then refer now to the exploration process as the agent's

exploration process, through which it can *actively* interact with the environment. In the *automated* discovery context, the agent is typically an AI with internal models $\mathcal{M}$ that it uses to generate the parameters $\theta \sim \mathcal{M}_W(\cdot|c_o, \mathcal{H})$, given the observed experimental context c_o, the history of previous discoveries $\mathcal{H}$ and the models' current *state* W [4]. Note that the agent's exploration process often depends on other internal processes. In particular, the agent typically uses some *learning process* in parallel of the exploration process to exploit the discovered data in $\mathcal{H}$ and update its internal states $W \leftarrow \mathcal{M}_W(\mathcal{H})$. Other processes, such as prediction processes via the use of surrogate models or representation processes via the use of encoder models, could be envisaged and are generally included as part of the agent's internal models $\mathcal{M}$ in our formalism. Within that agent-centered formalism, the overall automated discovery procedure is summarized in Algorithm. 1.

4: the state of a model is everything that is needed to save/load the model, *e.g.* the parameters of a neural network

Algorithm 1: The automated discovery procedure

Require: Observable Context Space C_O, Parameter Space Θ,
Observation space O, black-box mapping $f : \Theta, C \rightarrow O$,
experimental budget N, agent's internal models $\mathcal{M}$

Initialization: history table $\mathcal{H}$ and agent's internal state $W^{(0)}$;

Agent's exploration process;
for *i=1 to N* **do**
  Observe context c_o;
  Generate experimental parameters $\theta \sim \mathcal{M}_W(\cdot|c_o, \mathcal{H})$;
  Execute experiment $f(\theta, c_o)$;
  Observe outcome o ;
  Write (W, c_o, θ, o) to history $\mathcal{H}$;
end

Agent's learning processes;
Update internal models $W \leftarrow \mathcal{M}_W(\mathcal{H})$;

return $\mathcal{H}$;

At the end of exploration, the AI-driven agent has learned a map of possible behaviors in the system $\{(\theta_1, o_1), \ldots, (\theta_N, o_N)\}$ and some (optional) internal representations ($W^{(N)}$) in $\mathcal{H}$, which is what scientists are ultimately interested in evaluating (Subsection 2.1.2).

The Human Finally, while not shown in Figure 2.2, the human plays a crucial role in shaping the agent's exploration. We distinguish two types of human influence. Firstly, there is the influence from the *human engineer* which intervenes *before* exploration by defining both the agent's exploration spaces (Θ, O) and internal models ($\mathcal{M}$). Such decisions significantly determine the range of events that the AI agent can observe and impact, thus influencing the exploration process. In Section 2.2, when examining the current landscape of algorithm design practices, we will delve into how the human engineer commonly shapes the agent's discovery process, often concerning prediction and optimization tasks. Secondly, the *human end-user* (*e.g.* scientists) can influence exploration *before* it occurs by furnishing the agent with an initial database $\mathcal{H}$, *during* exploration by providing guidance or feedback, and *after* exploration by assessing final discoveries that might be valuable for other experiments and users. Note that the manner in which the human end-user interacts

with the AI agent is determined by the human engineer. As discussed in Section 2.2, the human end-user is frequently either completely removed from the exploration process or merely functions as an "expert" who provides preliminary data to the agent *before exploration*. Although it is often advocated that an "automated" discovery process should be separated from the human end-user, both in execution and experiment planning, we emphasize that the evolutionary pressures on AI agents are contingent on their utilization and acceptance by human end-users. In the upcoming section, we address how assessing the significance of discoveries must align with a human-centric value system, which might in turn lead to the development of human-in-the-loop AI systems. In Section 2.3, we redefine the roles of the AI agent, human engineer, and human end-user within a broader *developmental AI* perspective. Here, the AI agent mirrors a "developmental learner", and the human engineer and end-user correspondingly embody "evolutionary" and "cultural" processes in AI development, collectively shaping the AI discovery process.

2.1.2. Evaluation

Let's denote R^* an eventual metric that would evaluate the "interestingness" of the discoveries and acquired representations collected by the AI agent in $\mathcal{H}$. Scientists might care about different things when they look at the discoveries and information the AI agent collected, so there could be many ways to measure R^*. Here, we present a selection of such criteria, including those pertinent to this thesis (as indicated by the sidenotes).

Measuring Performance

The first, and maybe more trivial way to evaluate the agent is when scientists have a clear objective and can assess the agent's progress using a quantifiable score. This evaluation method is commonly employed in the AD literature, particularly when examining *knowledge-driven* and *optimization-driven* strategies (see Subsection 2.2.4). In the first scenario, R^* assesses the quality of the agent's learned *representations*: $R^*(\mathcal{H}) = \sum_{x \in \mathcal{D}_{test}} acc(\hat{f}_W(x), f(x))$. Here, $\hat{f}_W$ represents what the agent has learned (a prediction model with learned parameters W), f represents the expensive-to-evaluate function the agent aims to approximate (*e.g.* the environment forward dynamics), acc is some accuracy score function (*e.g.* classification accuracy), and $\mathcal{D}_{test} = \{(x, f(x))\}$ represents a set of hold-out test samples. In the second scenario, R^* typically evaluates the quality of the *discoveries* made by the agent: $R^*(\mathcal{H}) = \max(r(o_1), \cdots, r(o_N))$. In this case, r is a reward function that gauges how closely an outcome o aligns with a predefined target (based on specific measurable attributes).

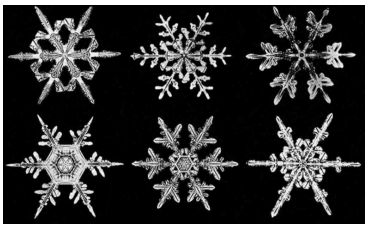


Figure 2.3.: What defines a snowflake? Wilson Bentley's initial detailed photographs of snow crystals beautifully exemplify the common knowledge that "no two snowflakes are identical".

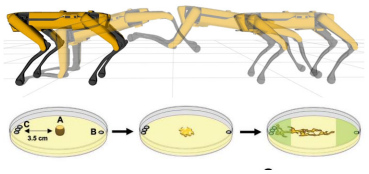


Figure 2.4.: What defines an agent? Does a quadrupled robot capable of complex motions [65] or a slime mold capable of using mechanosensation to decide where to grow [66] qualify more as an agent?

In both cases, designing a predefined accuracy or reward function as a scalar value can be demanding, especially when scientists are aiming for complex objectives. To illustrate this, consider the example of growing a target "snowflake" pattern. There are numerous ways to grow (and thus evaluate) such a pattern (Figure 2.3), and defining a proxy metric to evaluate performance is non trivial. Other scientific problems are even harder to characterize, for instance the concept of *agency* while intuitive

and central in artificial intelligence and life sciences, presents significant challenges in both definition and quantitative evaluation [5] (Figure 2.4).

5: We investigate this question in Chapter 7 and Chapter 8

Measuring Exploration

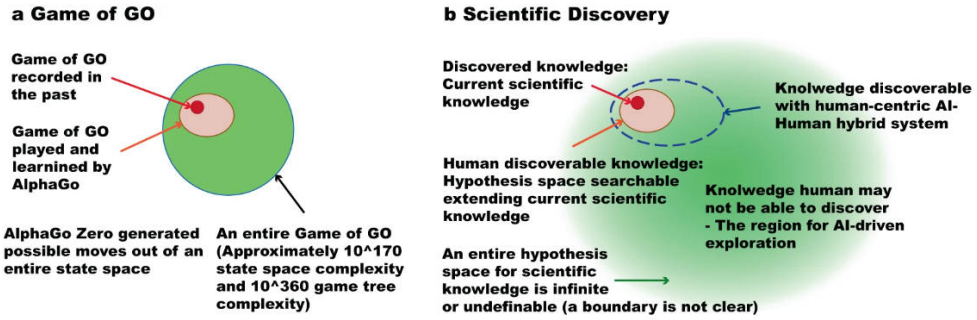
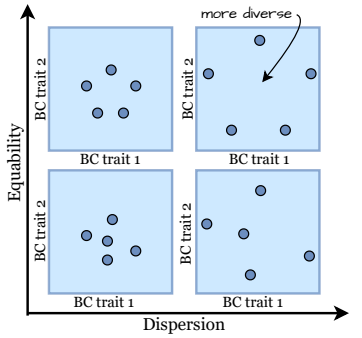


Figure 2.5.: A possible space of exploration for AI Scientists. Image is taken from [54].

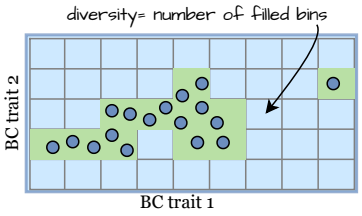
In the context of scientific discovery the space of possible behaviors is huge and complex as opposed to very big but finite, well-defined, and often monolithic state-space in games [54]. While the ability of AI agents to uncover entirely novel and unexpected behaviors (depicted as the green area in Figure 2.5) is highly desirable, assessing their "exploration" in practice is far from straightforward. Unlike conventional task-based evaluation, the focus here is on gauging the "extent" of the discoveries collected by the agent in $\mathcal{H}$. Though rarely employed in the AD literature, several task-agnostic exploration proxies R^* could be envisaged. For instance, metrics revolving around quantifying *entropy* (drawn from information theory and commonly employed in thermodynamics) or *diversity* (prominent in ecological and genetic research) could be used to evaluate the reach of agent discoveries $o_1, \cdots, o_N$. Detailed insights into these concepts and a comprehensive review of entropy and diversity metrics can be found in Leinster [67]. These concepts and metrics have also been adapted in AI and robotics, applied to assess task-agnostic exploration in artificial agents. For example, they have been employed to measure the entropy or coverage of the distribution within state space [68–70], as well as the diversity of the distribution within higher-level behavior space [71, 72]. In this thesis, we focus on adapting metrics that quantify *diversity* to evaluate the AD agent discovery process. The agent's discoveries $o_1, \cdots, o_N$ are projected into a higher-level *behavioral characterization* (BC) space, creating a set of discovered effects $(z_1, \cdots, z_N) \in BC$. Diversity measures the level of distinctiveness or novelty within this set.

Three main classes of diversity metrics are explored:

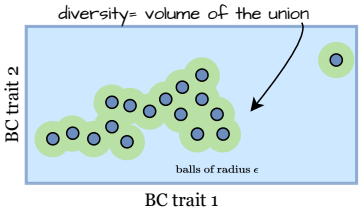
- *Distance-based* metrics define diversity based on the magnitudes of dispersion and variability among pairwise dissimilarities $dist(z_i, z_j)$ between the discovered $\{z\}$ points [73]. Many functional forms and parametrization can be envisaged, making their application and interpretation difficult in practice[6].
- *Binning-based* methods quantify diversity by discretizing the entire BC space into P bins and counting the filled bins during exploration: $R^*(\mathcal{H}) = \sum_{i=1}^{P} \delta_i$ where $\delta_i = 1$ if the i^{th} bin is filled (*i.e.* at least one discovered effect z_i falls in that bin) and $\delta_i = 0$ otherwise. Although less common in biology, they are prevalent in diversity-driven machine learning approaches [72, 74][7]. However these methods depend on the chosen binning strategy, which can be mitigated by analyzing its impact on results.
- *Threshold-based* metrics measure diversity through the volume of the union of hyperballs (or other geometric objects) centered on the discovered effects: $R^*(\mathcal{H}) = volume(\bigcup_{z_i \in BC} Ball(z_i, \epsilon))$ [75]. These metrics were recently proposed to address the limitations of distance-based and binning-based metrics [71] but are complicated to implement in high dimensions[8].



(a) Distance-based metrics



(b) Binning-based metrics



(c) Threshold-coverage metrics

Figure 2.6.: Diversity metrics

A major (but rarely discussed) limitation of these metrics is that they all rely on the prior definition of an analytical BC feature space which should formalize the "interesting" degrees of behavioral variation in the system. While this may be straightforward in simpler engineered or robotic systems (such as a 2D maze's $x-y$ space), it becomes notably challenging for self-organizing systems. Let's take again the example of snowflakes: while it's conceivable to devise descriptors like the number, length, and angle of a snowflake's branches, discovering diverse patterns within this space might not truly reflect what we intuitively consider as diverse (which might instead include subjective interpretations of snowflake variability or aesthetics). Furthermore, systems manipulated by scientists to generate snow crystals could produce many other structures that wouldn't contribute to diversity within a "snowflake" BC space but that might still be interesting in expanding our scientific knowledge. In the context of automated discovery, we propose that diversity metrics should involve human end-users to identify BC spaces that align more closely with the human intuitive notion of diversity [9].

6: For instance distance-based metrics do not respect the intuitive constraint that adding more points to a set should not decrease its diversity. We only use such metrics when measuring diversity of a small set of points.

7: In this thesis we mainly use binning-based metrics to evaluate diversity

8: We only use this metric when possible, *e.g.* in 2d BC spaces

9: We investigate these extensions Chapter 6

Measuring Robustness

The automated discovery process (Algorithm. 1) generated a parameter database ($\theta \in \mathcal{H}$), which, for some of them, might have resulted in the emergence of interesting patterns or behaviors observed under specific experimental conditions ($o \in \mathcal{H}$). Often, in the context of scientific discovery, one is interested in characterizing the degree of *robustness* of the discovered parameters/patterns to a variety of perturbations. These perturbations can be uncontrolled (*e.g.* system noise) or externally imposed (*e.g.* specific cues or constraints from human end-users), and already induced during the exploration phase or completely novel (in

which case on rather talks about *generalization*). Robustness is generally characterized as the degree of variation (sensitivity) in functionality (*e.g.* performance drop) or phenotypic trait (*e.g.* distance in BC space after perturbation) under a distribution of tested environmental changes: $R^*(\theta) = -\sum_{\xi \in \mathfrak{X}} sensitivity(f(\theta, \xi))$ with $\mathfrak{X}$ the distribution of tested perturbations.

Defining suitable tests for evaluating robustness can once again be challenging, especially within complex systems. It requires devising sensitivity metrics and a battery of tests, which is not straightforward[10]. Furthermore, these empirical tests can be resource-intensive as each tested perturbation ξ necessitates an experimental rollout $f(\theta, \xi)$, adding to the experimental budget N. The human end-user can again play a pivotal role by interacting with the discovered patterns, dynamically introducing perturbations based on observations to evaluate robustness or generalization[11].

10: In Chapter 7 and Chapter 8, we propose a battery of empirical tests formulated within a continuous cellular automata and gene regulatory network system to assess the robustness and generalization of discovered behaviors

11: In Chapter 7, an interactive web demo is used to enable end-users to replay and interact with the discovered structures, testing them against a range of freely-drawn perturbations

Measuring Interestingness

The evaluation of *open-ended* agents is an ongoing and dynamic topic in the field of AI [76, 77]. Within this debate, Soros et al. [76] accentuate the substantial influence of *subjectivity* on the evaluation of open-ended systems. The concept of *interestingness* is tightly coupled with that of open-endedness. Similarly in the context of automated discovery, what we expect from our discovery agents is to generate more and more behaviors that *we* deem interesting. There are at least three types of evaluation protocols that could be envisaged in the context of automated discovery:

- The "human committee" metric: a possible evaluation protocol involves having a panel of humans (such as expert scientists) *rating* the AI agent's discoveries. This concept aligns with the "Nobel Turing Challenge" recently proposed by Kitano [54], which suggests that AI-driven discoveries should rival with "top" human scientists discoveries and therefore be evaluated in the same way, aiming for major advancements in fields like biomedicine and environmental sciences, while assessing the potential risks and ethical issues of these discoveries.
- The "impact factor" metric: another possible evaluation protocol would be to measure the *reuses* of the discoveries by the scientific community. Consider the case of AlphaFold, the recent AI breakthrough that addressed a long-standing biological challenge known as the "protein folding problem" [78]. While the primary purpose of this AI tool is data analysis rather than data collection, it's interesting to see how the model predictions were shared with the public and made easily accessible to anyone via an online platform (AlphaFold DB)[12]. Via the platform, the AlphaFold team was able to track the number of researchers who accessed and downloaded the protein structures (more than 500,000 by July 2022), serving as a good proxy of its impact.
- The "model of interest" metric: alternatively, the use of *prediction models* to infer human interest could be pursued. For instance, in the context of measuring performance, diversity or robustness,

12: In Chapter 8, we propose that similar platforms could be used to share the dataset of discoveries made by the AD agents to the scientific community. We discuss how this could lead to several kinds of reuses in the context of biological network understanding and control.

one could ask a human to score performance or design specific tests, and learn a model to automatically test future discoveries. Implicit models of human interest could also be envisaged. Zhang et al. [79], for instance, recently introduced the use of language models (LM) to simulate human notions of interest. While not explicitly trained to predict human preferences, it is likely that LMs have *internalized* some sort of human-interest understanding as they have been exposed to extensive human-generated data. Nevertheless, the application of these models to automated discovery remains uncertain, largely because human language has not yet been shown to empower exploration in domains like molecular or bio-engineering.

In conclusion, evaluating the "interestingness" of the agent discoveries is a very hard problem. In the context of scientific discovery, some desirable properties could be that the discovered outcomes are 1) performant to solve some "interesting" tasks, 2) diverse to expand our knowledge of the system, 3) robust to withstand perturbations in the environment, 4) useful to assist solving other problems in the scientific community. In the context of *open-ended* discovery, evaluation is likely to become *subjective* and *adaptive* to the agent/environment evolution, *i.e.* involving human end-users interacting with the discoveries and providing feedback that is not limited to a predefined set of tasks but constantly evolving and aligned with their current value system. This, in turn, is likely to lead to the development of human-in-the-loop AI system where humans work in tandem with the AI to shape the discovery process and expand scientific knowledge.

2.1.3. Exploration Challenges

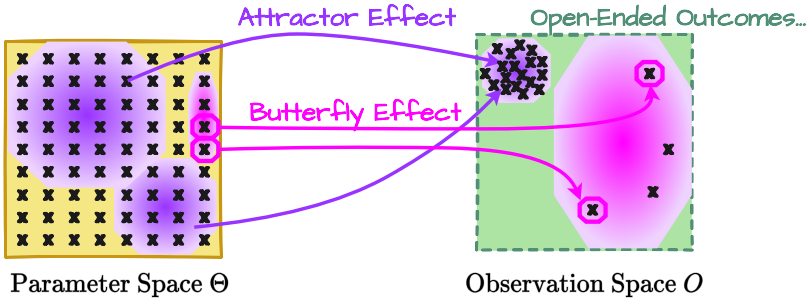


Figure 2.7.: Typical properties of the $f : \Theta \mapsto O$ mapping in the targeted complex systems.

The manipulation, exploration and control of complex systems requires dealing with excessively large and open-ended search spaces, which poses many challenges with respect to simpler physical or engineered systems. We discuss some of these challenges in this section.

Non-linearity, redundancy and variability of the $f : \Theta \mapsto O$ mapping
As shown in figure Figure 2.7, exploring certain regions in the outcome space can be extremely difficult, if not impossible, due to the non-linear, redundant, and stochastic nature of the mapping $f : \Theta \mapsto O$. *Non-linearity* implies that changes in the parameter space Θ have disproportionate effects in the observation space O, which can lead to the well-known "butterfly effect" of complex systems. *Redundancy*

means that some regions in O can be reached from multiple points in Θ, causing an "attractor effect", while others are only reachable through specific paths. Because redundancy is often heterogeneously distributed in O, exploration strategies that uniformly cover the input space Θ will preferentially reach points in areas of high redundancy in O (purple area in the figure) and miss out on areas of low redundancy (pink area in the figure). Finally, *stochasticity* refers to the fact that the same input θ can lead to multiple outputs o in O, and several experimental rollouts may be required to account for this variability.

Vast Exploration Spaces The state space of targeted systems is often vast and potentially infinite, unlike the finite spaces found in most games and engineered systems (as shown in Figure 2.5). While access to the system is limited through sensors and actuators, which impose finite input and output spaces Θ and O, these spaces can still be extremely vast. Observations can span different spatial and temporal scales and there is no compact representation or description of the state space, unlike simpler engineered systems. Hence, whereas advances in technology are likely to provide finer control and observability, the challenge remains for AI agents to extract meaningful information and to construct tractable internal representations of the high-dimensional system outcomes.

Limited Exploration (and Evaluation) Budget Despite advancements in high-throughput experimental platforms, the experimental budget in complex system science remains significantly lower than that in typical machine learning testbeds. Each experimental rollout can be very costly and time-consuming. In the example systems from Figure 2.1, few minutes might be enough to observe oil-droplets dynamics (c) but several hours might be needed to observe the formation of cellular structures (b).

Failure of *passive* exploration strategies Passive exploration strategies, such as grid search and random search, are widely used for automated discovery in complex systems and science domains. These strategies aim to uniformly cover the known input space Θ without making assumptions about the mapping f or outcome space O. Random search is commonly applied in numerical systems like cellular automata [80], agent-based models [81], and hyper-parameter search for deep neural network[13] training dynamics [83]. It is also used in high-throughput screening for material design [84, 85] and drug discovery [86, 87] in chemical wet systems. Grid search is the primary methodology in biological tissue engineering but is mostly used to assess the impact of a small range of parameters [88, 89] rather than for discovering novel forms of life. However, despite their extensive use, brute-force exploration strategies are sample-inefficient for covering the range of possible outcomes O due to the aforementioned challenges.

Exploration "at the edge of chaos"

Non-linearity and redundancy of the mapping between action and effects are phenomena that are already observed in simpler physical or engineered dynamical systems, such as robotic environments where they have been extensively studied [71]. However, these phenomena are often drastically exacerbated in the considered complex systems where one usually talks of exploration "at the edge of chaos" [46, 47]. Can we transpose the autonomous exploration strategies deployed in robotics context to systems with chaotic dynamics, emergent outcomes and complex state spaces?

13: While rarely formulated as such, neural network training can be reinterpreted as an example of self-organizing system "rollout" where nodes operations are low-level elements whose states are updated by the weight update rule during the backwards pass [82], such that θ here is the optimization hyper-parameters (learning rate, activation function, etc.)

2.2. Standard AI paradigms

Different AI-driven approaches have been proposed in the recent years for actively guiding the agent exploration, and can generally be categorized

into *knowledge-driven*, *optimization-driven* or *diversity-driven* exploration strategies.

2.2.1. Knowledge-Driven Strategies

The first (and perhaps most famous) "robot scientist", which pioneered the use of fully automated experimentation processes combining artificial intelligence and robotic platforms, was called Adam and developed by King et al. [52] (Figure 2.8). Adam was capable of autonomously (i) generating functional genomic hypotheses about the yeast Saccharomyces cerevisiae (of the form 'gene X encodes the orphan [14] enzyme Y') and (ii) testing the hypotheses experimentally to compare the outcome phenotype (growth or no growth) with and without knockout of the hypothetic gene. Given a limited experimental budget Adam was capable of identifying several accurate hypotheses, more than a random strategy and a "cheapest trial" strategy (always choosing the cheapest experiment in terms of monetary cost); and even performed comparably to human graduate students. Here, Adam's experimental strategy was a form of *active learning*: it selected experiments in a manner that minimized the total expected cost of identifying the genes that code for all the orphan enzymes in the pathway. To do so, Adam had an internal *surrogate model* $\hat{f}$ (in which a big part was hard-coded with prior knowledge) that was used to infer the outcome z (binary "growth" or no "growth") given input parameters θ (categorical vector with the gene to knockdown and the growth media to use).

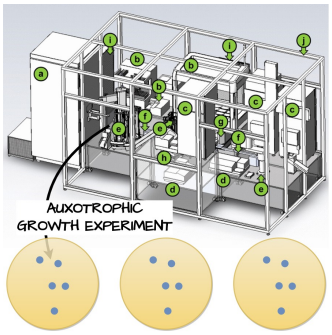


Figure 2.8.: Adam the Robot Scientist. Adam's hardware included fully automated a) freezer, b) liquid handlers, c) incubators, d) plate readers, e) robot arms, f) plate slides, g) plate centrifuge, h) plate washer, i) air filters, and j) plastic enclosure. Adam measures growth curves (phenotypes) of selected microbial strains (genotypes) growing in defined media (environments) and was used to identify genes encoding orphan enzymes in *Saccharomyces cerevisiae*. Adapted from [60].

14: orphan enzymes catalize reactions that are thought to be essential for yeast growth but their encoding gene(s) are not (yet) known

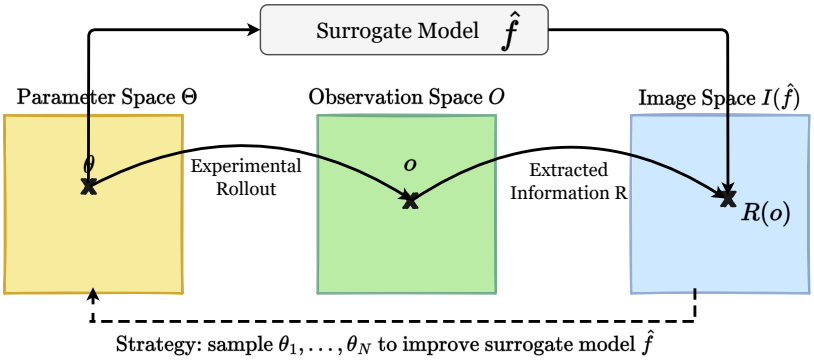


Figure 2.9.: Knowledge-driven exploration strategies

More generally, in active learning θ is often called a *query* and $z = R(o)$ a *response*, and the objective of the agent is to select queries $\theta_1, \ldots, \theta_N$ that would be the most informative for improving the surrogate $\hat{f}$. We call these strategies *knowledge-driven* because here the specific discoveries collected by the agent ($\{\theta_i, o_i\} \in \mathcal{H}$) is not really what interest scientists here. Instead they are generally interested in reusing the acquired knowledge (final state of the learned surrogate $W^{(N)}$) for *prediction* purposes.

The main design choice in knowledge-driven strategies concerns the choice of the criteria for selecting the next query θ based on current knowledge in $\hat{f}$. Typical strategies select candidate queries that maximize the uncertainty of the model (such as *entropy*, *margin score*, or *least confidence*) or uncertainty of a committee of models (so-called query-by-committee methods such as *vote entropy*, *consensus entropy* and *KL divergence*), or that maximize some expected model change (such as

expected gradient length or *information gain*). We refer to [90, 91] for a blogpost and thorough review on active learning query strategies.

Other design choices typically concern: (i) the details of the surrogate model used ($\hat{f}$ can take many forms and be used for classification, prediction or clustering), (ii) the degree of reliance on prior knowledge, (iii) the type of queries θ that specify each experiments, and (iv) the type of data z that experiments yield. Finally, at the end of exploration knowledge-driven strategies are generally evaluated by assessing the *task-specific performance* (classification or regression accuracy) of the learned surrogate model on a test database which is typically taken from previously performed experiments (see Subsection 2.1.2).

Whereas Adam was the first application of a knowledge-driven strategy in a fully-automated setup, active learning (AL) has been used across many domains of science for maximizing predictive performance of a model subject to a fixed annotation budget. Those include applications for molecular crystallization prediction [51], biological network construction [92], materials science [93] and drug discovery [94].

Limitations In practice, when the experimental budget is limited and when one does not have the opportunity to explore and compare alternative AL strategies, the success of these approaches often rely on some prior expert knowledge [95]. Expertise is typically applied either when designing the base surrogate model and query selection strategy or by providing a prior database of large and representative examples. Without this, learning of the surrogate model is likely to suffer from the *cold start problem*, where poor selection of data in the early phases of active learning can propagate their harm throughout the whole learning, or from failure to learned from *skewed data distribution* [96]. In the targeted systems, skewed distributions are very likely to emerge as instances present in areas of high redundancy in O are likely to be often sampled whereas instances present in areas of low redundancy are unlikely to be well represented in the discoveries. However, when such expertise is not available, one could couple these methods with some prior task-agnostic exploration phase in order to reveal a distribution of representative instances in the outcome space O and to facilitate learning of the surrogate $\hat{f}$.

2.2.2. Optimization-Driven Strategies

Another typical use-case of AI for guiding scientific experimentation is when scientists have a clear target in mind that they want to achieve, typically a certain morphology or functionality. A reward function R is defined, and some optimization method is used to find the parameters θ maximizing the reward, which is assumed to drive the system as close as possible to the target. A recent famous example was the computer-assisted design of so-called *xenobots*, described by the press as the first ever "living robots" (Figure 2.10). Xenobots are a new form of "organisms" created from frog cells that have been specifically designed by scientists from university of Vermont and Tufts university to accomplish specific functions in their environment. Those include moving autonomously [61],

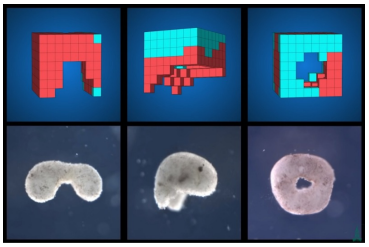


Figure 2.10.: Xenobots. Adapted from [61].

collecting micro-particles [62] or even self-reproducing [63], where organisms shown a life span of approximately 10 days. Here the exploration happened in computer-based environment, and the optimal designs were then manufactured by the hands of biologists. Interestingly it is an evolutionary algorithm (EA) that was used to find the right parameters θ leading to these functional xenobots. In this case, the evolutionary algorithm is an exemple of AI-driven exploration process that we include in the family of *optimization-driven* strategies.

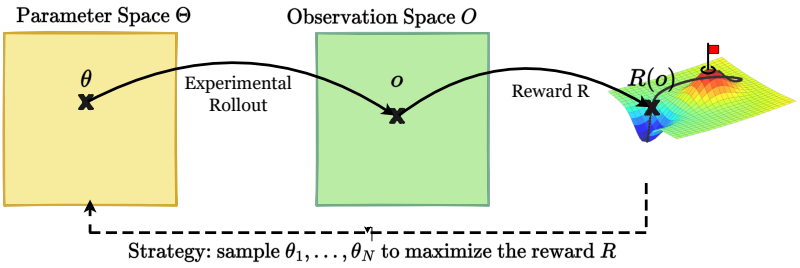


Figure 2.11.: Optimization-driven exploration strategies

More generally, assuming a known reward function R used to score the discoveries in O, the objective of the agent is to select parameters $\theta_1, \ldots, \theta_N$ leading to the highest reward $max(R(o_1), \ldots, R(o_N))$. We call these strategies *optimization-driven* because the agent's focus here is on acquiring a specific competence, the one of finding the parameters θ that will self-organize a structure with target properties in the environment. They are four main families of optimization-driven strategies.

Evolutionary algorithms (EA) or *population-based* strategies, used in the xenobot example, is a family of black-box optimization which evolves a population of candidates through iterative generations, and select the fittest offsprings based on the reward function. It has been widely used across many domains, for instance to perform computation in cellular automata [97], to synchronize motion toward target functionalities in robotic swarms [98–100] or target shapes in biomolecular robot swarms [101], and for the design of new catalysts [102] or nanoparticles [103] in chemistry.

Stochastic gradient descent (SGD) is another family of powerful optimization methods, commonly used in machine learning to optimize deep neural networks, but rarely used in self-organizing systems as it requires the environment dynamics f and reward function R to be differentiable. However, with the recent progress in differentiable programming we observe a rise of *differentiable self-organizing systems* [37], such as the recent neural cellular automata (NCA) which have been used to train cells to grow and regenerate a desired shape [104, 105], self-organize in a texture [106], classify handwritten digit in a decentralized way [107] and perform complex image segmentation tasks [108]. Since then, differentiable self-organizing systems have gained traction across several science domains including molecular dynamics, [109], fluid dynamics [110] and biological network analysis [•7].

Reinforcement learning (RL), another family of optimization method commonly used in machine learning and robotics, has also been transposed for scientific discovery applications such as microbial strain design [111], control of nuclear fusion plasma [112], optimization of chemical reaction yield [113] or self-assembly of "limbs" for the design of artificial

agents [36]. Here, the environment's dynamics integrate a policy Π_θ which actions directly influence the system dynamics, and the exploration process seeks to optimize parameters θ of the RL policy Π_θ.

Finally, *model-based* optimization methods rely on the use of surrogate models $\hat{f}$ allowing them to select promising parameters θ by evaluating the surrogate model instead of the expensive real system. In most cases, the surrogate is directly used as an approximation of the objective function $\hat{f} \approx R \circ f$. This is the case in *bayesian optimization* (overviewed in [114]), which alternates between exploitation (selecting experiments θ in areas where $\hat{f}$ expects maximum reward) and exploration (sampling θ in areas where $\hat{f}$ is the most uncertain about). Bayesian optimization has been used to optimize behavior of a reaction-diffusion numerical system [115], yield of chemical reaction [116, 117], and production of Bose-Einstein condensates [118] or alloys [50] with target properties. There are also several papers which suggested augmenting EA, SGD and RL approaches with reward prediction models. For instance, EA coupled with surrogate model was used to optimize catalytic activity of materials [48] and growth of carbon nanotubes [49][15] and model-based RL was used for biological sequence design [119]. In other cases, the surrogate is not used as a direct task-prediction model but rather as an intrinsic-reward model $R_i = R_{aux} \circ f$ which, in addition to the extrinsic reward R, rewards the agent for experiencing dissonance (or resonance) on an auxiliary prediction task with respect to its current knowledge and expectations (as in knowledge-driven strategies). These approaches are often coined as "curiosity-driven" [120], but as we will discuss in the next section, curiosity-driven exploration encompasses a broader set of approaches and this is more precisely a form of *knowledge-based* intrinsic motivation (KB-IM) [121]. In the AD literature, Thiede et al. [122] proposed combining RL optimization with KB-IM surrogates rewarding for either prediction errors, novelty or surprise to encourage exploration of the solution space in molecular design tasks[16].

15: In [49], the target growth rate is updated through experimentation (as the maximum experimentally observed) which is a form of *curriculum learning*

16: In [122], the RL agent manipulates SELFIES, a string-based representations of molecules [123]. There is a huge portion of works in ML for science that explores the space of molecular representations, but we do not include them in this chapter as these spaces are generally not the outcomes of self-organizing environment with local "physics" (and differ from other molecular dynamics environments [109] or artificial combinatory chemistries [124, 125]). SELFIES strings lie somewhere in between, as they are the result of a minimal but formal grammar (automaton), whose derivation rules can be seen as a form of basic "physics".

In addition to the choice of optimization method, important design choices typically concern the degree of reliance on prior knowledge for the choice of the reward function and initialization (starting parameter $\theta^{(0)}$ or population of parameters in the case of EA). At the end of exploration, optimization-driven strategies are generally evaluated by assessing the *task-specific performance* which typically scores of the best discovered parameter and is eventually coupled with a measure of *robustness* to a distribution of test perturbations (see Subsection 2.1.2).

Limitations In practice, the successful application of these methods also relies on some prior expert knowledge in providing either a "good-enough" initialization or base surrogate for model-based approaches. Due to the chaotic nature of the $\Theta \to O$ mapping, many initializations are likely to get trapped in local minima (or diverge) in O while other locations might by luck make the optimization a success. These "good-enough" initializations are unknown and vary depending on the target endpoint in O. Reversely, given an ensemble of previously-reached points in O, some new targets might be easy to reach but others might be very hard (deceptive or sparse reward) or even impossible. Without prior knowledge of such feasible targets, one might consider coupling the optimization with some *curriculum learning* in order to progressively

adapt the target or with some *intrinsically-motivated* (also called *curiosity-driven*) exploration process. However, the few works that propose to do so in the AD literature either rely on simple linear curriculum [49] which might not scale to more complex problems, or leverage knowledge-based IMs [122] where training of the surrogate might also not scale to other problems. Indeed the KB-IM surrogate provides a (unique) source of intrinsic reward on which the agent has no control and that is likely to slowly evolve throughout exploration (and highly depend on the initial dataset due to the same cold start and skewed-data problems discussed in Subsection 2.2.1). In general, objective-driven agents do not explicitly control which reward function they maximize, which greatly limit their capacity to explore the state space and to discover open-ended repertoire of self-organized structures.

2.2.3. Diversity-Driven Strategies

The last big family of experimentation strategies, while not so common in the AD literature, is what we refer to as *diversity-driven* strategies. Here, at the difference of optimization-driven strategies, scientists do not have a clear objective in mind. In fact, they might not even know what they are looking for in the first place, or might have some intuitions about it but do not know how to express it into a computable score function. This was for instance the case of the so-called "curious robot" by Grizou et al. [64] which was used to explore the dynamics of an oil-droplet system (Figure 2.12). By investigating the dynamics of these lipid droplets, scientists hope to gain insights on the conditions that led to the emergence of the first protocell on Earth, a central question for understanding the origins of life. Typically here, scientists do not know how the possibly-emerging structures should behave or look like, and they would rather be interested in constructing a map of the diverse kinds of behaviors that can self-organize in these systems, in otder to expand their knowledge of the system. To do so, Grizou et al. [64] proposed to equip the robot with an *intrinsically-motivated discovery goal exploration process* (IMGEP), which is a recent-family of diversity-driven approaches that aim to discover a diversity of behaviors in a dynamical system by targeting a diversity of self-generated goals [57]. Controlling the initial oil-mixture of the droplets and using only a low budget of experiments, the IMGEP was shown to enable the discovery of a variety of "life-like" self-propelled droplet behaviors including movement, grouping, division, fusion, chemotaxis, and many others (Figure 2.12).

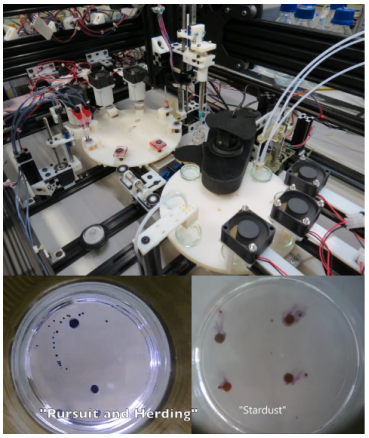


Figure 2.12.: A "curious robot" capable of discovering a diversity of self-propelled behaviors in an oil-in-water droplet system, a promising model to study protocell formation. Adapted from [64].

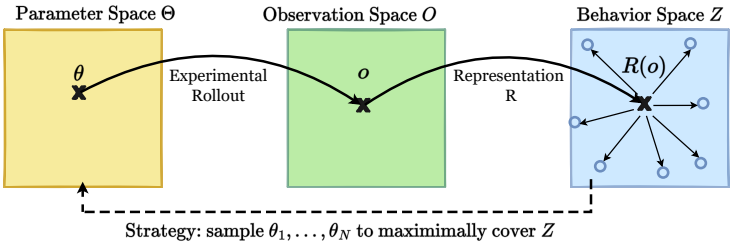


Figure 2.13.: Diversity-driven exploration strategies

More generally, assuming a representation function $R : O \to Z^2$ mapping high-dimensional outcomes to a behavioral space Z, and a distance metric $\mathcal{D}(z_1, z_2)$ characterizing how behaviorally-similar two embeddings

$(z1, z2) \in Z$ are, diversity-driven strategies aim to select parameters $\theta_1, \ldots, \theta_N$ leading to the highest diversity in behavior space Z, given a certain diversity metric (see Subsection 2.1.2). There are two main families of diversity-driven algorithms, both share similarities but have emerged from distinct contexts.

The first set of approaches, which was formulated within the context of evolutionary computation (EC), is known as *Novelty Search (NS)*. In contrast to fitness-oriented EA approaches, NS proposes to focus on maximizing the *novelty* of the discovered behaviors instead of their fitness [126, 127]. NS evolves a candidate parameter archive iteratively, selecting candidates from the previous generation that exhibited the most novel behaviors, updating their parameters through mutation and crossover operations, and evaluating the outcomes of the resulting solutions. Here novelty of a point $z \in Z$ is measured as the average behavioral distance $\mathcal{D}$ to its k nearest neighbors in Z. Throughout exploration, the concept of "novel" evolves leading previously novel behaviors to become less novel over time, which results in the progressive coverage of the behavioral space. Although not objective-driven, NS was shown to beat fitness-oriented methods for the resolution of pre-defined tasks characterized by sparse or deceptive reward, notably in robotics contexts such as maze solzing [127]. Building on these results, recent works have proposed to apply NS to guide experimentation in self-organized systems, though they were evaluating the discoveries on task-specific performance metrics. Gomes et al. [128] used NS to control swarm robotics systems in collective aggregation and energy-sharing tasks. Cazenille et al. [129] used the MAP-Elites algorithm [130] to optimize biomolecular robot swarms towards a target shape (as in [101]). MAP-Elites belongs to the family of *Quality Diversity (QD)* approaches, another well-known family of approaches from the EC literature which proposes to optimize both for *novelty* and *local quality* to discover a population of both behaviorally diverse and locally high-performing parameters [131, 132].

"Diversity-driven" approaches

Novelty Search (NS): coming from the evolutionary computation field, NS approaches suggest to "abandon" the objective and instead focus on maximizing novelty of the discovered behaviors [126, 127].
Intrinsically Motivated Goal Exploration Process (IMGEP): coming from developmental robotics, IMGEPs are centered around the notion of *goals* (intrinsic objectives) and aims to achieve a diversity of self-generated goals [57, 58].
Several parallels can be drawn between NS and GEP algorithms [133]. Notably, random goal exploration variants of IMGEP were shown to behave equivalently than NS [134].

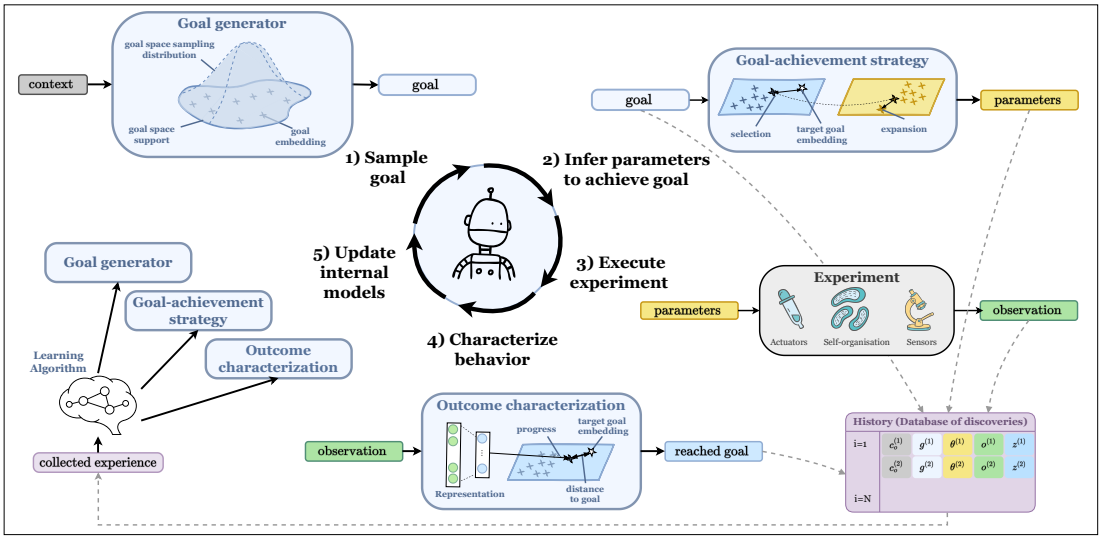


Figure 2.14.: Intrinsically-motivated goal exploration processes (IMGEP): family of *autotelic* machine learning approaches that generates a sequence of experiments to explore the parameters of a dynamical system by targeting a *diversity* of self-generated *goals*.

"Curiosity-driven" approaches

Several approaches in the AD literature employ the term "curious", both in optimization-driven [122] and diversity-driven [64] contexts. *Curious* generally means that the agent is endowed with some computational model of *intrinsic motivation*, but they are many ways to build these IM models. Oudeyer and Kaplan [121] organize IM approaches in two prominent families:
Knowledge-based IMs (KB-IMs) use prediction-based models [120] or count-based models [70] of IMs to reward for prediction errors, novelty, surprise, progress in forward predictions or information gains.
Competence-based IMs (CB-IMs) introduce an explicit goal-selection mechanism and reward agents for achieving self-generated goals, biasing goal selection towards intrinsic proxies such a novelty or progress in goal reaching [57].

The second set of approaches, which was originally formulated within the context of developmental robotics, is known as *Intrinsically-Motivated Goal Exploration Process (IMGEP)* [57, 58]. Similarly than NS, IMGEP agents directly explore a space of abstract representations Z that compactly characterize outcomes of the system, often called a *goal space*. At the difference of NS, the notion of *goal* is central in IMGEPs: a goal $g = (z_g, \mathscr{L}_g)$ is defined by a compact goal-embedding $z_g \in Z$ and a goal-achievement loss function $\mathscr{L}_g = \mathscr{D}(\cdot|z_g)$ measuring distance toward the goal. Therefore, instead of fully "abandoning" the concept of objective, an IMGEP is *autotelic* meaning that it targets a diversity of self-generated objectives (goals and their corresponding learning signal), generally allocating a small budget of experiments per target goal to try and achieve it (Figure 2.14). Moreover, the concept of *intrinsic motivations* (IMs) (or *curiosity*) is also central in IMGEPs: goals are self-generated according to intrinsic goal-selection mechanisms. At the difference of knowledge-based intrinsic motivations (KB-IMs) which use a single intrinsic reward function but do not represent goals, IMGEPs belong to the fa mily of *competence-based* intrinsic motivations (CB-IMs) where the notion of competence naturally corresponds to goal achievement. CB-IMs can be understood as a form of autotelic curiosity, *i.e.* curiosity explicitly based on the pursuit of goals as. KB-IMs, on the other hand, are generally used for optimization purposes: in-fine they do not seek to learn an accurate surrogate model (like knowledge-driven exploration strategies) nor to find diverse solutions to diverse problems (as in CB-IMs), but rather aim to find a solution maximizing the external reward. While CB-IMs approaches were also shown to aid optimizing tasks with sparse or deceptive rewards [135], their primary purpose is to learn a repertoire of diverse skills (where a skill is defined as the association of a goal g and a policy θ achieving that goal) using a limited budget of experiments [58]. In the AD literature, IMGEP were shown useful to guide scientific experimentation in various domains. Grizou et al. [64] used a simple random goal exploration variant of IMGEP to reveal diverse behaviors in the oil-droplet system, and Terayama et al. [136] usedan IMGEP-like approach (with learned surrogate model and Stein novelty metric for goal sampling) to discover diverse light-absorbing molecules. Both shown that the IMGEP found a much higher diversity than a random search baseline, while being given the same experimental budget.

Limitations At the end, only a few works proposed to use diversity-driven approaches in the AD literature [64, 128, 129, 136], and even less were really interested in revealing a diversity of behaviors and relied on very simple IMGEP variants [64, 136]. A main limitation of these approaches is that they all assume that the notion of "diversity" can be captured and measured within a single predefined representation space Z. Therefore, the exploration space is constrained *a priori* to a small and bounded set of reachable goals such that the autonomy and open-endedness of the exploring agent is restricted. This limits the scope of the final discoveries, in particular in the targeted self-organizing systems where the degrees of behavioral variations are emergent (unknown a priori) and potentially open-ended. A second limitation of existing approaches is that they rarely evaluate the *robustness*. Whereas they

might be able to identify conditions leading to the self-organization of novel structures with interesting properties, such as droplets with novel self-propelled behaviors [64] or molecules with novel optical properties [136], they might have trouble generalizing to novel situations in the environment such as stochasticity or external perturbations.

Works related this thesis All the works presented in this thesis falls within the framework of Intrinsic Motivation Goal Exploration Processes (IMGEPs), detailed in Chapter 3, and aim to address the aforementioned challenges in the context of automated discovery in complex systems. In Chapter 3, we propose to integrate IMGEPs with *unsupervised learning* of the goal space representation R, using a variational auto-encoder (VAE) [137] to embed the high-dimensional observation space into a compact goal space, and removing the need for expert knowledge to define such representations. In Chapter 5 we further introduce the concept of *meta-diversity* where an IMGEP continuously learns diverse representation spaces while searching to discover diverse patterns within each of them, with the aim to design more open-ended forms of AI "discovery assistants". In Chapter 6 we propose to integrate *human-guidance* to shape the IMGEP discovery process toward the initially unknown preferences of an external end-user. In Chapter 7, we show how the IMGEP can integrate *curriculum learning* to progressively sample goals of increasing difficulty as well as stochastic *environmental design* and *gradient descent* to efficiently achieve these goals under various perturbations. Finally in Chapter 8, while using a very simple IMGEP variant, we formalize a procedure to empirically evaluate the robustness of the discovered behaviors integrating key insights from the emerging field of basal cognition[17]. These works were applied in continuous cellular automata models where they were shown to assist the search for interesting patterns [•1, •3] and forms of self-organized agency with robust sensorimotor behaviors [•5], as well as in continuous models of gene regulatory networks where they were shown to uncover a wide spectrum of possible phenotypes [•8].

17: Basal cognition studies the fundamental mechanisms that enable organisms across all domains of life (from prokaryote cells to plants and animals) to sense and act in their environment to meet existential goals [45, 138]

Other related works Falk et al. [139] applied the IMGEP-VAE approach proposed in [•1] to discover diverse behaviors (including previously unknown ones) in physical non-equilibrium models of coupled oscillators. Sudhakaran et al. [140] proposed a goal-conditioned exploration strategy where the target goal g is sent to a neural cellular automata system within the input parameters $\theta = (g, \phi)$ (where ϕ are the NCA rules), and searched for a single rule ϕ capable of achieving various goals $z_g \in Z$ (target patterns) depending on the provided context g (one-hot vector written into the NCA cell states). Whereas all approaches in this thesis are population-based IMGEPs (POP-IMGEPs), *i.e.* mapping one parameter set θ per reached goal g, this draws similarity with recent RL-IMGEP approaches where the outcome of exploration is a single goal-conditioned policy [55]. However, Sudhakaran et al. [140] used a small and predefined set of target goals (emoji patterns), therefore qualifying more as "multi-objective" than "diversity" driven approaches.

2.2.4. Survey

All the works presented in this section are presented in Table 2.2*. The table provide the main design choices of these works according to the:

1. *Outcome Characterization*: type of representation used to characterize the system outcomes $R : O \rightarrow Z$; *i.e.* choice of the type of encoder R and latent dimensionality Z (black) and whether the representation is fixed or learned throughout exploration (gray).
2. *Task Selection*: type of task formulation strategy, *i.e.* whether these works follow a knowledge, optimization, hybrid or diversity-driven paradigm (black) and what specific task sampling strategy they use (gray)
3. *Parameter Generation*: type of strategy used to infer (or optimize) parameters $\theta \in \Theta$ according to the task, *i.e.* whether it is population-based[18], model-based[19] or surrogate-based[20] (black) and what specific optimization method is used (gray).
4. *Human Guidance*: whether they use human guidance or not along one of these three dimensions (black), and the specific way the human influences exploration (gray).
5. *Evaluation of Discoveries*: type of evaluation criteria (from Subsection 2.1.2) that is used to evaluate the final discoveries (black) and more specifically what scientists were interested in achieving (gray).

Note that this survey is far from exhaustive, and instead aims to review a selection of recent papers that are representative of the field. From this analysis, two noteworthy patterns emerge. First, a vast majority of works are structured around a precise and formal problem on which they evaluate the discoveries (task-specific performance). Here the use of AI is intended to benefit complex problem-solving tasks, but the scope of tasks remain relatively narrow and predetermined by machine learning experts or engineers. Secondly, while these methods rely a lot on expert inputs both for the task definition and for task-specific prior knowledge, almost none rely on human guidance during exploration. In the next section, we propose to redefine the roles of the AI agent, human engineer, and human end-user within a broader *developmental AI* perspective. In this perspective, AI agents aren't just designed for solving narrow tasks; they must learn to represent, generate and pursue their own goals. The AI self-generated goals should both foster the *bold exploration* of unseen outcomes at the system-level, while also remaining aligned with what humans consider *interesting*. Achieving this requires the AI agent to continually adapt its internal models (governing the overall exploration process and decision-making) and to incorporate feedback from external human end-user.

18: population-based strategies are non-parametric inverse models that directly use the current points in $\mathcal{H}$ to infer the most promising candidate θ (*e.g.* K-Nearest Neighbors Algorithm)

19: model-based strategies are parametric inverse models that are learned on $\mathcal{H}$ and used to generate the most promising candidate θ. Note that this "model-based" RL which means using a forward predictive model (what we call here surrogate-based)

20: surrogate-based strategies are parametric forward models that are learned on $\mathcal{H}$ and used to "cheaply" try several candidates $\{\theta\}$ and select the predicted best one

* Abbreviations used in the table: Genetic Algorithm (GA), Cellular Automata (CA), Stochastic Gradient Descent (SGD), Deep Reinforcement Learning (DRL), k-Nearest Neighbors (kNN), Behavioral Characterization Space (BC), Evolutionary Algorithm (EA), Explore-Exploit (EE), Bayesian Neural Network (BNN), Kernel Density Estimation (KDE), Reaction Diffusion (RD), Knowledge-Based Intrinsic Motivation (KB-IM), penalized logP (plogP), quantitative estimate of druglikeness (QED), NeuroEvolution of Augmenting Topologies (NEAT), Support Vector Machine (SVM), Co-variance Matrix Adaptation Evolutionary Strategy (CMA-ES), Expected Improvement (EI), Gaussian Process (GP), Expected Cost (EC), Bose-Einstein Condensate (BEC), Knowledge Gradient (KG), Support Vector Regression (SVR), Neural Network (NN), Multi-dimensional Archive of Phenotypic Elites (MAP-Elites), Random Forest (RF), Carbon NanoTubes (CNT), SN (Stein Novelty), SD (Stein Discrepency), max (maximum), sampl (sampling), optim (optimization), knowl (knowledge), var (variance)

Table 2.2.: Survey of papers from Section 2.2, organized by the type of explored complex system (numerical ALife, numerical modeling, swarm robotics, physico-chemical or biological), summarizing the main algorithm ingredients.

Paper	Outcome Characterization	Task(s) Selection	Parameter Generation	Human Guidance	Evaluation of Discoveries
Numerical ALife CS					
Mitchell et al. [97]	1d-fitness (fixed)	optimization (fixed)	pop-based (GA)		task-specific performance (CA compute tasks)
Mordvintsev et al. [104]	1d-loss (fixed)	optimization (fixed)	model-based (SGD)		task-specific performance (CA target shape)
Pathak et al. [36]	1d-reward (fixed)	optimization (fixed)	model-based (DRL)		task-specific performance (agent locomotion)
Reinke et al. [•1]	8d-encoder (learned)	diversity (hypercube sampl)	pop-based (kNN+mutation)		diversity (CA BC coverage)
Etcheverry et al. [•3]	Nd-encoder (learned+grown)	meta-diversity (preference-based+ hypercube sampl)	pop-based (kNN+mutation)	task generation (preference feedback)	diversity (CA BC coverage) interestingness (preferred diversity type)
Hamon et al. [•5]	2d-encoder (fixed)	diversity (curriculum sampl)	model-based (SGD)		robustness (sensorimotor generalization)
Sudhakaran et al. [140]	1d-loss (fixed)	multi-objective (fixed)	model-based (SGD)	task generation (goal selection)	multi-task performance (goal-guided CA target shape)
Numerical modeling CS					
Kriegman et al. [61]	1d-fitness (fixed)	optimization (fixed)	pop-based (EA)		task-specific performance (xenobot locomotion)
Treloar et al. [111]	1d-reward (fixed)	optimization (fixed)	model-based (DRL)		task-specific performance (target microbial levels)
Degrave et al. [112]	1d-reward (fixed)	optimization (fixed)	model-based (DRL)		task-specific performance (tokamak magnetic control)
Hase et al. [115]	1d-regressor (learned)	optim-knowl (EE sampl)	model-based (BNN+KDE)		task-specific performance (RD target behavior)
Thiede et al. [122]	1d-regressor (learned)	optim-knowl (EE sampl)	model-based (KB-IM DRL)		task-specific performance (pLogP, QED)
Etcheverry et al. [•8]	2d-encoder (fixed)	diversity (hypercube sampl)	pop-based (kNN+mutation)		diversity + robustness (navigation competency tests)
Falk et al. [139]	4d-encoder (learned)	diversity (hypercube sampl)	pop-based (kNN+mutation)		diversity (Kuramoto BC coverage)
Swarm robotics CS					
Baldassarre et al. [98] Trianni and Nolfi [99]	1d-fitness (fixed)	optimization (fixed)	pop-based (EA)		task-specific performance (swarm movement)
Duarte et al. [100]	1d-fitness (fixed)	optimization (fixed)	pop-based (NEAT)		task-specific performance (homing, . . .)
Gomes et al. [128]	100d-encoder (fixed)	diversity (novelty sampl)	pop-based (kNN+NEAT)		task-specific performance (aggregation, energy sharing)
Physico-chemical CS					
Duros et al. [51]	1d-classifier (learned)	knowledge (max entropy)	surrogate-based (SVM)	outcome characterizat° (human-data pretrain)	diversity (crystal space coverage) task-specific performance (classification accuracy)
Aubert-Kato et al. [101]	1d-fitness (fixed)	optimization (fixed)	pop-based (CMA-ES+NEAT)		comparison with expert task-specific performance (bio-robots target shape)
Kreutz et al. [102]	1d-fitness (fixed)	optimization (fixed)	pop-based (GA)		comparison with expert task-specific performance (catalyst activity)
Salley et al. [103]	1d-fitness (fixed)	optimization (fixed)	pop-based (GA)		task-specific performance (target nanoparticle shape)
Zhou et al. [113]	1d-reward (fixed)	optimization (fixed)	model-based (DRL)		task-specific performance (max reaction yield)
Christensen et al. [116]	1d-regressor (learned)	optim-knowl (EE sampl)	model-based (BNN+KDE)		task-specific performance (max E-product yield)
Shields et al. [117]	1d-regressor (learned)	optim-knowl (max EI)	model-based (GP)	outcome characterizat° (human-data pretrain)	task-specific performance (max reaction yield)
Wigley et al. [118]	1d-regressor (learned)	optim-knowl (min EC, max var)	model-based (GP)		comparison with expert task-specific performance (BEC production)
Xue et al. [50]	1d-regressor (learned)	optim-knowl (KG)	model-based (SVR)		task-specific performance (alloy thermal hysteresis)
Corma et al. [48]	1d-regressor (learned)	optimization (fixed)	pop-based (NN-surrogate GA)		task-specific performance (epoxide yield)
Nikolaev et al. [49]	1d-regressor (fixed)	optimization (moving target)	pop-based (RF-surrogate GA)		task-specific performance (max CNT growth)
Cazenille et al. [129]	1d-fitness, 1d-BC (fixed)	quality-diversity (MAP-Elites)	pop-based (CMA-ES+NEAT)		task-specific performance (bio-robots target shape)
Grizou et al. [64]	2d-encoder (fixed)	diversity (uniform sampl)	pop-based (kNN+mutation)		diversity (oil-droplet BC coverage)
Terayama et al. [136]	2d-encoder (fixed)	diversity (SN sampl)	surrogate-based (RF)		diversity (discovered molecules SD)
Biological CS					
King et al. [52]	1d-classifier (learned)	knowledge (min EC)	surrogate-based (logical model)	outcome characterizat° (expert knowledge)	task-specific performance (classification accuracy)
HamediRad et al. [141]	1d-regressor (learned)	optim-knowl (max EI)	model-based (GP)		task-specific performance (lycopene production)
Radivojević et al. [53] Zhang et al. [142]	8×1d-regressor (learned)	Optim-knowl (EE sampl)	surrogate-based (ensemble)		task-specific performance (synthesis rate)

2.3. Problem Reformulation: the Developmental AI Paradigm

Developmental AI aims to model children learning and, thus, takes inspiration from the mechanisms underlying autonomous behaviors in humans. After discussing essential mechanisms enabling open-ended learning in humans (Subsection 2.3.1), we discuss how baking similar mechanisms into artifical agents seems to be a key step for the development of autonomous learning agents and how it enabled sample efficient learning of high-dimensional motor skills in robotic contexts (Subsection 2.3.2). Finally, we discuss perspectives and challenges for transposing developmental AI approaches to the target automated discovery problem (Subsection 2.3.3).

2.3.1. Motivation: Humans are Open-Ended Learners

Humans are an incredible source of inspiration for AI: despite their limitations in energy and time, humans develop an extensive repertoire of diverse skills from an almost infinite array of potential possibilities throughout their lives. As babies, they learn simple things like recognizing objects, and as they grow, they learn more complex things like talking and interacting with others. Most of the time, humans are not motivated by external rewards but spontaneously explore their environment to discover and learn about what is around them. While we are far from fully understanding what makes humans efficient open-ended learners in such a short period of time, several mechanisms have been extensively investigated by psychologists and cognitive scientists, and appear pivotal in enabling effective learning in humans, particularly in children.

Firstly, humans exhibit a propensity for *goal-directed* behaviors such that the concept of *goal* holds central importance in theories of human motivation [143, 144]. Several psychological studies have delved into the use of this concept, leading Elliot and Fryer [143] to propose a general definition: *"A goal is a cognitive representation of a future object that the organism is committed to approach or avoid"*. Pursuit of goals, also called *teleonomy* in biology, is likely to be play a crucial role in enabling humans and living organisms in general to structure their behavior into organized developmental trajectories [145].

A second (and related) notion at the core of human development is the one of *intrinsic motivations*, known in everyday language as *curiosity*. Humans are goal-directed learners but what's really impressive is that they can come up with their own problems (goals) to solve. Notably, Chu and Schulz [146] argue that "children's exploratory play may be characterized by nothing so much as their tendency to invent novel goals and problems for themselves". This behavior seems to be driven by intrinsic motivations, a set of brain processes that motivate humans (and other animals) to explore for the mere purpose of experiencing novelty, surprise or learning progress [147–152]. Therefore humans can be qualified as *autotelic i.e.* agents that are both intrinsically motivated and goal-conditioned, generating their own goals and learning signals [55].

Finally, human are inherently *social* learners: not only they are capable of generating their own goals but they are constantly adapting thier goals and responses based on the environment around them. Since birth, humans enter a culture that strongly shapes their development [153]. Then, throughout their lives, they benefit greatly from social interactions and are in fact intrinsically motivated to communicate and cooperate with their peers [154–156].

(a) The curious child

(b) The robot as child

Figure 2.15.: The "playground experiment": (a) drawing inspiration from children exploratory play, (b) pioneer works in developmental AI illustrate how mechanisms of curiosity-driven exploration enable a robotic agent to successively develop object affordances and vocal interaction with its peers [157, 158].

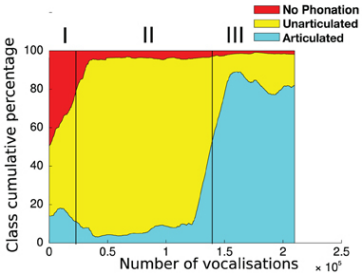


Figure 2.16.: Emergence of developmental stages in an IM agent exploring a realistic model of the human vocal tract: transition from no phonation (red), to exploring unarticulated sound (yellow) and articulated sounds like cannonical babbling (blue), akin to the developmental stages observed in children [159]

Figure 2.17.: An intrinsically motivated agent learns what effects can be producted by its action and uses it to explore its environment leading to nested tool use discovery [58]

2.3.2. From Theory to Practice: Developmental AI

At the crossroads of the 20^{th} and the 21^{st} centuries, the field of *developmental robotics* emerged, bringing together a diverse group of researchers from AI and developmental sciences. Inspired by the aforementioned theories from developmental psychology and cognitive sciences, their aim was to create computational models based on these theories and to integrate them into artificial systems, including physical robots. Within developmental robotics, innovative frameworks were proposed to model development including the previously-mentionned IMGEP ones combining intrinsically motivated and goal-directed learning processes [57, 58]. More recently, the developmental robotics field has merged into the broader field of *developmental artificial intelligence* (developmental AI). This integration incorporates modern deep learning methods and processing capabilities into the developmental artificial systems. Developmental AI essentially strives to "build machines that learn like children", or at least replicate aspects of how children learn and explore. Unlike traditional machine learning models that focus on solving specific tasks, developmental AI focuses on proposing computational models that enable efficient exploration of the environment without relying on external rewards and working with limited resources. Interestingly, the integration of these models into artificial agents was shown to be useful for both theoretical investigations, such as constructing cognitive models of child development, and practical achievements, such as building robots and AI systems that can efficiently learn in open-ended environments.

On the theoretical front, computational models of intrinsic motivation (IMs) using artificial exploring agents are increasingly being used to explain various aspect of human cognitive development, such as the emergence of developmental transitions between object manipulation, tool use and speech [160]. For instance, Moulin-Frier et al. [159] explained the progression of human vocal behavior through distinct developmental stages as an intrinsically motivated, competence-progress based goal-exploration process (Figure 2.16). Such computational accounts of curiosity-driven learning have led to novel hypotheses on the role of curiosity in the evolution of language [151]. Similarly, a recent study comparing various computational models of utilities functions in motivational systems has shown that models that take into accound the learning progress explain the best observed behaviors of human adults in free exploration games [161].

On the empirical front, these computational models have demonstrated their capability to enable efficient exploration mechanisms when embedded in artificial agents, including physical robots, operating in complex environments and using only a limited budget of interactions [56]. For instance, Forestier et al. [58] shown how a humanoid robot, which initially

knows nothing about its environment, can explore its body movements and progressively discover how to interact with the various objects and tools in the scene (Figure 2.17). Similarly, autotelic curious mechanisms where shown to enable the autonomous and efficient aquisition of other sensorimotor skills including locomotion [57, 162], soft object manipulation [163, 164], visual skills [165], vocal interactions [157, 158] and nested tool use [166].

More recently, leveraging the recent advancements in AI algorithms, computing power, and processing capabilities, developmental AI has shown to scale the proposed approaches to high-dimensional complex environments. These advancements have been integrated within the AI agents in diverse ways. Those include the integration of unsupervised representation learning techniques like variational auto-encoders (VAEs) [137] for the autonomous learning of compact representations from high-dimensional image-based states [74, 168–170] and for the detection of controllable areas in the image-based state space [171]. Deep reinforcement learning (RL) optimization techniques have also been incorporated to better handle higher dimensional inputs and perturbations [135, 172] (see Colas et al. [55] for an extensive review). Language has been integrated as a powerful cognitive tool to support the imagination of goals [173] and (large) language models have recently been used to enhance the goal representation and generation of autotelic agents [59]. More broadly, developmental AI concepts are gaining interest in the AI community, and modern ML tools are shown to enhance the computational models of developmental AI. For instance, the use of learning progress based curriculum was shown to foster the emergence of generally capable agents in rich open-ended environments [167] (Figure 2.18), and the use of LLM was shown very promising to automatically generate a learning curriculum for enhancing exploration progress [174] and human interest [79] in Minecraft-like environments. However, these approaches often focus on the problem of sequential decisions in deep RL settings, incurring a big cost on sample efficiency.

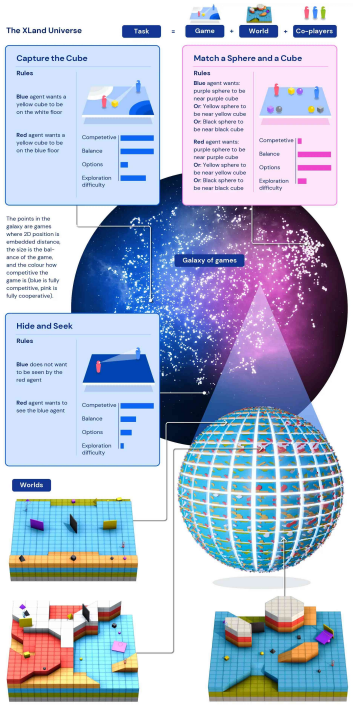
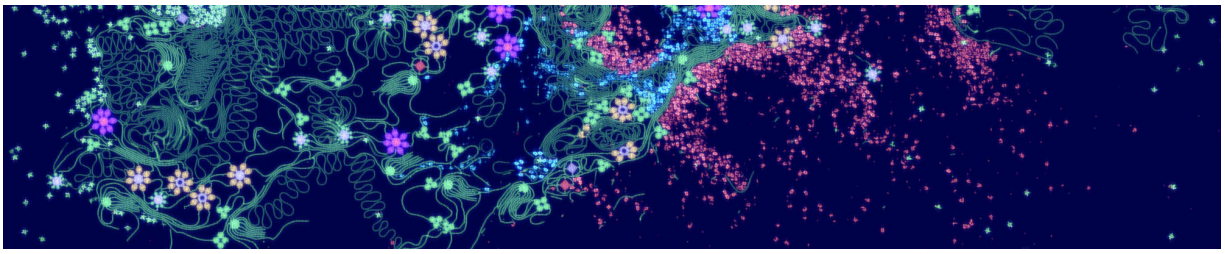
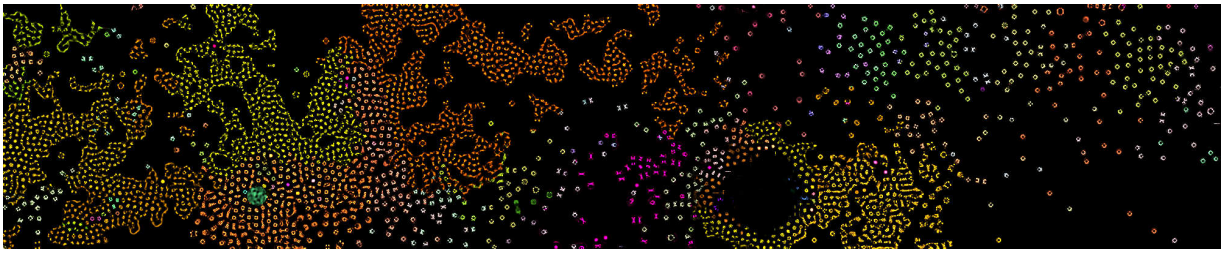


Figure 2.18.: XLand procedurally-generated environments. Image taken from [167].



(a) ALIEN: particle-based artificial life environment (`https://alien-project.org/`)



(b) Flow Lenia: CA-based artificial life environment [•6]

Figure 2.19.: Large-scale simulation of artificial life environments made of millions of particles (or cells). While the particle (or cell) dynamics are the result of simple physical laws, they can give rise to a very rich and hardly predictable diversity of collective behaviors.

One natural path forward for developmental AI approaches is to move from toy robotic or simulated environments to exploration spaces where creative open-ended search could have *practical applications* for humans. They are notably two applications that seem particularly promising: educational technologies to foster curiosity and creativity in young children [175] and automated discovery tools to foster the likelihood of making "interesting" discoveries [•3]. In the former, there are already several works that reuse developmental AI models into intelligent tutoring system and were shown to improve skills acquisition in students [176, 177], with recent works even integrating modern AI tools such as large language models [178]. This thesis aims to introduce, formalize and contribute to the second type of application - specifically the potential of using developmental AI to organize scientific experimentation. The question we aim to answer is: **Can we construct artificial agents, combining autotelic curious exploration mechanisms with the processing capacities of recent AI algorithms, to form efficient AI-driven "discovery assistant" and help addressing various challenging problems in Science?**

2.3.3. Developmental AI: Practical Applications for Assisting Humans and Scientific Discovery

Developing AI algorithms that will work in tandem with human scientist to help them explore complex and unfamiliar problem spaces and maximize the likelihood of making "interesting" discoveries in these spaces could be extremely powerful [54, 179]. In the history of Science, most of what humans consider to have been "interesting" discoveries were in fact *serendipitious* discoveries: the result of a creative process more than a objective-driven process [180]. Expanding on the "child-scientist" theory, we argue that automated discovery would benefit from being formulated as a developmental AI process rather than an optimization process (as standardly done) and discuss its fundamental challenges.

Curiosity: the fuel for discovery In history, there are numerous example of groundbreaking discoveries which were driven by fundamental curiosity-driven questions or simple "accidents", that then led to practical applications. For instance, Alexander Fleming's discovery of penicillin in 1928, the first antibiotic, arose from observing accidental mold growth on his culture plates and realizing that it prevented bacterial growth. More recently, biochemists Jennifer Doudna and Emmanuelle Charpentier's breakthrough in CRISPR-cas9 technology, the "genetic scissors", originated from fundamental biology research exploring bacterial defenses without a clear drug-related external objective. In a recent study, Spector et al. [181] found that 80% of the "most transformative" medicines approved in the United States between 1985 and 2009 stemmed from fundamental discoveries about biological processes or diseases, without a predetermined drug path.

The "child-scientist": conceptual analogy Jean Piaget, a swiss psychologist and father of theories of children cognitive development, is particularly known for his *"child as little scientist"* theory: the child is actively exploring its body movements in the same way that the scientist

is actively choosing the next experiment to perform in order to augment its understanding of the natural world [182]. This theory, which was the first to describe child development as a succession of developmental stages marked by intrinsically motivated experiences, notably influenced the many works in developmental psychology and artificial intelligence previously mentionned [183]. Reciprocally, american psychologist and philosopher Alison Gopnik (and many others) also defended the *"scientist as child"* theory [184]. This theory, emphasizing the *creative* and *exploratory* nature of scientific thinking and the connections it shares with the natural curiosity and exploration of children, forge a strong conceptual link between the two types of exploration [149].

(a) The child as little scientist

(b) The AI scientist as child

Figure 2.20.: Automated discovery as a curiosity-driven developmental process.

The "child-scientist": proposed algorithmic analogy Building on the "scientist as child" perspective, and as a natural extension of the "robot as child" perspective proposed by developmental robotics (Figure 2.15), the "AI scientist as child" naturally comes as alternative formulation to the automated discovery process (Figure 2.20). Following the analogy in the context of automated discovery, we suggest to view the AI agent as a *developmental learner*, the human engineers as the equivalent of *evolutionary processes* and the human end-user as the equivalent of *cultural processes*. The engineer must find ways to design (or evolve) AI agents with efficient exploration mechanisms and ways to interact with the human end-user. The human end-user shapes the AI developmental trajectories with guidance and instructions, and affect evolutionary pressures on the AI at a higher-level by influencing the human engineer practices (Figure 2.21). This dynamic is expected to foster a significant degree of interaction between the AI agent and human scientists, forging new frontiers in collaborative scientific exploration [54]. In Chapter 3, we will discuss how we can reuse the IMGEP algorithmic framework originally developed for the learning of inverse models in developmental robotics, to implement the "AI scientist as child" in practice. As we will see, both applications share the same underlying fundamental challenges: discovering maximally-diverse outcomes in highly-redundant dynamical systems within a limited budget of experiments.

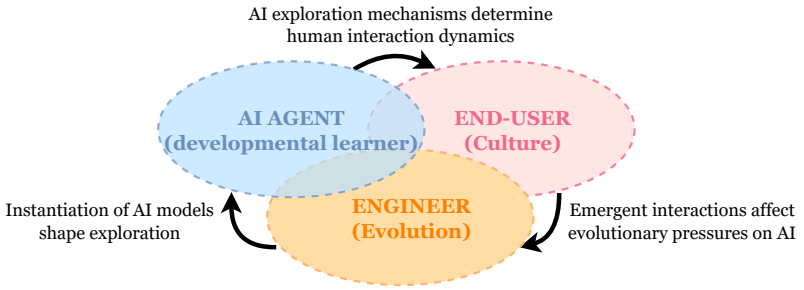


Figure 2.21.: Roles of the human engineer and human end-user in shaping the AI-driven scientific discovery

Key challenges for autotelic exploration in self-orgazing systems However, fundamental challenges remain for transposing existing algorithms to the highly complex and emergent exploration spaces of self -organized systems. Despite the testbed environments used to evaluate artificial autotelic agents becoming more and more complex in machine learning, as examplified by the XLand 3D environment that can generate many different tasks (Figure 2.18), environments remain mostly *engineered*: they are designed by engineers with specific factors

of variation in mind like the position of objects, terrain topology and so on. But, when we look at the self-organized structures found in nature that scientists want to study, there's no pre-set plan: it's rather about large collective of low-level elements (particles or cells) coming together on their own to make something bigger (Figure 2.19). This distinction between microstates (smaller, fine-grained description) and macrostates (where emergent behavior occurs on a larger scale), while central in theories of complex systems and emergence, is challenging to apprehend in practice. In fact, any system can be described in many state spaces (in particular biological systems) and what defines a macrostate depends on what aspects of a system an observer decides to observe. Creative exploration is all about finding new ways to interact with the environment. Here it is about finding new state spaces in which to imagine goals, and about finding efficient space traversal strategies to achieve those goals. Human civilization is a great example of creative process which found completely new state spaces to explore, such as the language space or the internet space, which turned out to be particularly useful spaces for humans to imagine, pursue and achieve new goals in the physical world. Recent approaches in developmental AI haves considered state spaces that were either fixed and monolithic (*e.g.* descriptors of the state-space in games) or human familiar spaces assumed to help exploration (*e.g.* language space for exploration of the 3D world). But what is a good state space for exploring molecular self-assembly or cellular morphogenesis? Scientists are only starting to explore those systems and do not know really what are interesting state spaces to explore. On top of that, even for systems where scientists want to explore specific state spaces, devising efficient traversal strategies in those spaces can be very challenging. Indeed, making progress toward a (macro) goal in a certain space requires finding ways to drive the (chaotic) dynamics of the environment toward self-organizing structures whose emergent properties align with the target goal, which shifts the goal-achievement process to something that we might call "engineering emergence" and which is much less intuitive than standard problem-solving capacities of current AI agents.

2.4. Summary

In this section, we presented the automated discovery problem along with the associated challenges for evaluation and exploration (Section 2.1). We then provided an overview of conventional AI paradigms designed to address this problem, summarizing the existing approaches in Table 2.2. We notably introduced the IMGEP computational framework, a recent family of diversity search algorithms for autotelic learning in artificial agents (Figure 2.14). Then, we presented the developmental AI paradigm which, at the intersection of developmental robotics and modern AI techniques, builds on the IMGEP computational framework to design intrinsically-motivated agents that can generate and pursue their own goals, *i.e. autotelic agens* (Section 2.3). Unlike standard optimization-driven or knowledge-driven strategies that rely on externally provided goals, autotelic agents discover and learn to represent their own goals from their experience of the physical world. This adaptability to the physical world reflects the concept of Piaget's "child as scientist" developmental psychology theory, which emphasizes children's capacity to shape their

learning journeys akin to how scientists adjust their experimental paths to deepen their understanding of the natural world [182].

In the following part of this manuscript, we will delve into the details of the IMGEP computational framework and explore its application in creating autotelic learning agents within the context of automated discovery. We will also discuss the challenges surrounding the definition of IMGEP's internal models in practical scenarios, for which we propose a range of algorithmic contributions. These contributions have led us to formulate the framework of what we call the "curious discovery assistant", which we propose as an alternative framework for addressing the automated discovery problem in Science.

The "curious discovery assistant" framework

Intrinsically-Motivated Discovery of Diverse Self-Organized Structures

3.

What is the aim of this chapter? In many complex dynamical systems, artificial or natural, one can observe self-organization of patterns emerging from local rules. Cellular automata, like the Game of Life (GOL), have been widely used as abstract models enabling the study of various aspects of self-organization and morphogenesis, such as the emergence of spatially localized patterns. However, findings of self-organized patterns in such models have so far relied on manual tuning of parameters and initial states, and on the human eye to identify "interesting" patterns. In this chapter, we formulate the problem of automated discovery of *diverse* self-organized patterns in high-dimensional complex dynamical systems using a continuous extension of the Game of Life called Lenia as testbed environment. We propose to transpose intrinsically-motivated machine learning algorithms (IMGEPs), initially developed for learning of inverse models in developmental robotics (as illustrated in Figure 3.1), as computational framework for experimentation.

How is this chapter organized? In Section 3.1, we present the IMGEP computational framework and its core principles in a robotic context. In Section 3.2, we present the Lenia environment, a class of continuous cellular automata models that will serve as experimental testbed for the first part of this manuscript. Finally in Section 3.3, we propose to transpose the IMGEP framework for the automated discovery of a *diverse* set of self-organized patterns in such system, and identify the main challenges for implementing the IMGEP internal models in practice.

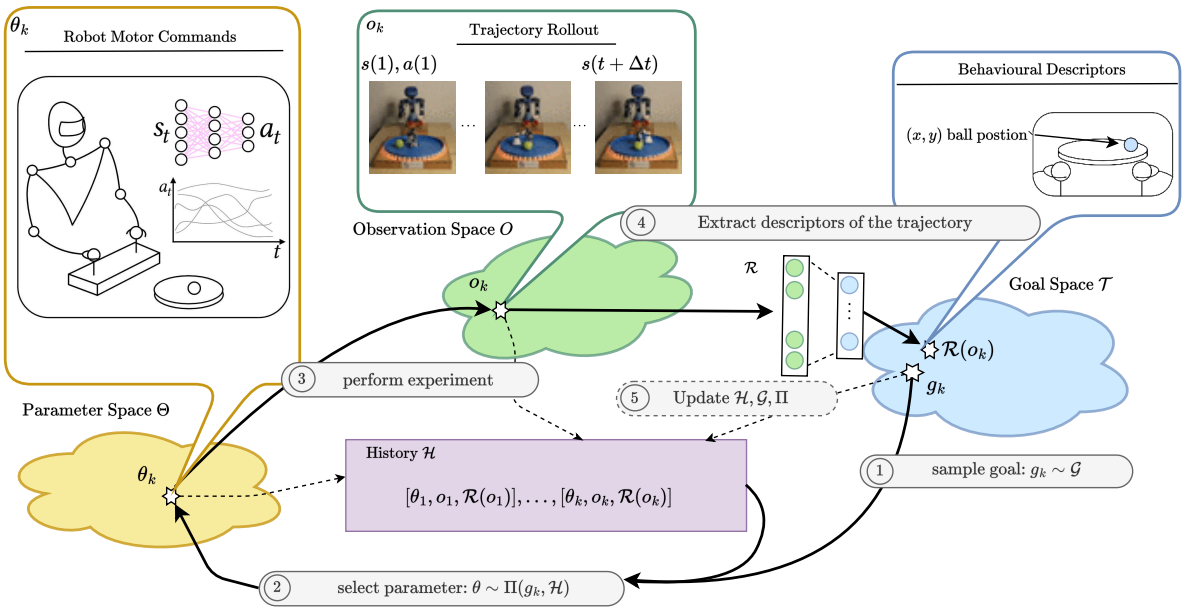


Figure 3.1.: IMGEPs were designed to enable robots to explore, learn and control what effects can be produced by their actions. See the corresponding video of Forestier and Oudeyer [166] where a robot is intrinsically motivated to learn how to interact with objects.

3.1. The IMGEP Computational Framework

IMGEPs are a recent family of diversity search algorithms, inspired from the way children self-develop open repertoires of skills and learn world models, which were initially designed to enable autonomous robots to explore and learn what effects can be produced by their actions, and how to control these effects [57, 186]. As indicated by their names, IMGEPs are *goal-directed* and *intrinsically-motivated*, with the aim to create *autotelic* learners capable of representing, generating and pursuing their own goals [58]. Before diving into their application to the discovery of self-organized structures, let's first explain the *core principles* of IMGEP illustrating them in a robotic experiment. As we will see, the two domains share many properties.

IMGEP agents are goal-directed As illustrated in Figure 3.1, IMGEPs self-define goals in a *goal space* $\mathcal{T}$ that represents important features of the outcomes of actions, such as the position of the different objects in the scene. Goal-directed agents are conditioned on this compact representation of the goal and update their behavior to make progress towards the goal, *e.g.* moving an object toward the goal-position. This leads to the following formalization of goals [55]:

A goal $g = (z_g, \mathcal{L}_g) \in \mathcal{T}$ is defined by a compact goal-embedding $z_g \in Z$ and a goal-achievement loss function $\mathcal{L}_g$ measuring progress toward the goal.

This definition of goals is quite general and encompasses a wide diversity of goal representations. We refer to [55] for a review of possible goal representations, but in this thesis we define goals as *target features of states*. In this scenario, a state representation function R maps the observation space O to an embedding space $Z = R(0)$. Goal embeddings z_g are target points in Z that the agent should reach. In the example of robotic object manipulation from Figure 3.1, z_g represents target object coordinates, but it could be other things like target agent position, etc. Note that the representation is often predefined, but it could also be learned. This is the case for visual goals, where the representation function R is usually implemented by a generative model trained on a database of visual states from the environment [74, 168–170]. For this type of goals, the goal-achievement function $\mathcal{L}_g$ is based on a distance metric $\mathcal{D}$ which is most of the cases shared across goals. This results in a *goal-conditioned loss function* such that $\mathcal{L}_g = \mathcal{D}(\cdot|z_g)$. For instance, one can directly define $\mathcal{D}$ as the distance between features of the current state and the target goal embedding: $\mathcal{D}(\cdot|z_g) = dist(R(o)|z_g)$ or define some binary achievement when distance falls below a pre-defined threshold: $\mathcal{D}(\cdot|z_g) = 1$ if $dist(R(o)|z_g) < \epsilon, 0$ otherwise.

IMGEP agents are intrinsically-motivated To sample goals from $\mathcal{T}$, IMGEP use a *goal-generator* module $\mathcal{G}$. Sampling new goals is not trivial: some goals might be straightforward to attain (hence not very interesting) but others might be very hard (and reachable only after the agent mastered a sequence of easier goals), or even impossible (and hence not desirable

to target). In developmental robotics, CB-IMs were used to automatically organize the sampling of goals- *i.e.* of learning signals - and enable the sample-efficient mastery of complex goals. Common ways to sample goals in the IMGEP literature is to estimate the *learning progress* (LP) or *novelty* of the agent in different regions of the goal space, and to bias sampling towards areas of high absolute learning progress [57, 166, 187] or high novelty [170], or hybrid LP-novelty based selection [171]. In the example of robotic object manipulation from Figure 3.1, LP estimate could measure whether the agent is making progress to move the chosen goal object toward the target goal position *e.g.* whether the distance between goal and actual reached position decrease over time on average. Novelty estimate on the other hand could learn a density model on the history of reached object positions, and skew sampling toward unexplored areas of the goal space. Many other selection mechanisms could be envisaged to organize the agent exploration automatically, which is in fact part of a broader and active area of research known as *automatic curriculum learning* (ACL).

Automatic curriculum learning (ACL) is defined as *"the family of mechanisms that automatically adapt the distribution of training data by adjusting the selection of learning situations to the capabilities of learning agents"* [188]. Within ACL, automated goal selection is often formalized as a competence-based intrinsic motivation (CB-IM).

Interested readers can refer to Portelas et al. [188] and Romac et al. [189] for a review and a benchmark of ACL methods in deep reinforcement learning contexts.

IMGEP agents reuse knowledge The data collected by the IMGEP agent thoughout exploration is used to improve the goal-sampling and goal-reaching strategies in two ways. First, the information collected while aiming at a particular goal is systematically reused to achieve other goals faster. This cross-goal learning is something that has recently been called *hindsight learning* [190]. Learning by hindsight, agents can reuse a initially-failed trajectory as an informative trajectory to learn about another goal, thus making the most out of every trajectory. More generally, the information collected is stored in history $\mathcal{H}$ and is reused by to update its internal models throughout exploration.

IMGEPs implement two parallel processes: a goal-directed exploration (*exploration process*) and an offline data exploitation (*learning process*) which uses the collected data to improve goal-reaching behaviors.

POP-IMGEPs versus RL-IMGEPs There are two main versions of IMGEPs, known as *population-based* (POP-IMGEPs [58]) and *reinforcement learning-based* (RL-IMGEPs [55]), which mainly differ in the way the goal-achievement strategy $\Pi(g, \mathcal{H})$ and agent memory are defined. In POP-IMGEPs, an explicit memory of the history $\mathcal{H} = \{\theta, o\}$ of experiments and observations is collected. The history is directly-used to infer the most promising candidate θ given a target goal g, using a non-parametric inverse model Π such as k-Nearest Neighbors algorithm. In RL-IMGEPs,

Π is a goal-conditioned parametric model which is trained using deep reinforcement learning techniques, and used to infer the most promising canditate θ given a target goal g. RL-IMGEPs approach generally focus on the problem of sequential decisions in MDPs, and therefore consider experimental budgets N that are much bigger than the ones in automated discovery. Therefore, the history $\mathcal{H} = \{\theta, o\}$ is often regularly emptied, keeping only a memory buffer with a subset of selected discoveries on which the policy Π is regularly trained.

POP-IMGEPs return the all history of discoveries $\mathcal{H}$ associating one set of experiment parameters θ per reached goal g whereas RL-IMGEPs return a goal-conditioned policy Π (called to generate parameters θ) which has internalized the capacity to achieve various goals.

IMGEP pseudocode Algorithm. 2 presents the general pseudo-code of the IMGEP, where the exploration and learning process happen in parallel. The IMGEP exploration operates in two phases: N_{init} interventions are first sampled randomly from Θ to initially populate history $\mathcal{H}$, and then the remaining interventions are generated through the goal-directed process. The sampling and pursuit of goals (highlighted in bold) is the main novelty with respect to the standard automated discovery procedure (Algorithm. 1). It relies on several key internal models: the goal representation (R) which encodes observations (o) into the IMGEP goal space ($\mathcal{T}$), the goal generator module ($\mathcal{G}$) which samples goals from the goal space, and the goal-conditioned optimization policy (Π) that generates candidate parameters to achieve the current goal. In parallel, the IMGEP learning process regularly updates the IMGEP internal models with the information collected throughout exploration.

Algorithm 2: Intrinsically Motivated Goal Exploration Process

Require: Parameter Space Θ, Observation space O, Representation R
Require: Experimental budget N

Initialize empty history table $\mathcal{H}$
Initialize goal space $\mathcal{T}$ and goal generator module $\mathcal{G}$
Initialize goal-conditioned optimization policy Π

IMGEP EXPLORATION
for *i=1 to* N **do**
  if $i < N_{init}$ **then** `// Initial random iterations`
    Sample random parameter $\theta \sim \mathcal{U}(\Theta)$
  else `// Goal-directed IM iterations`
    Sample target goal $g \sim \mathcal{G}(\mathcal{H})$
    Infer experiment parameters to achieve goal $\theta \sim \Pi(g, \mathcal{H})$
  Execute experiment with θ and observe system outcome o
  Characterize behavior $z = R(o)$
  Write (θ, o, z) to history $\mathcal{H}$

IMGEP LEARNING
  Update goal space $\mathcal{T}$ with $\mathcal{H}$
  Update goal generator module $\mathcal{G}$ with $\mathcal{H}$
  Update goal-conditioned optimization policy Π with $\mathcal{H}$

return $\mathcal{H}$

3.2. Testbed Environment: the Lenia system

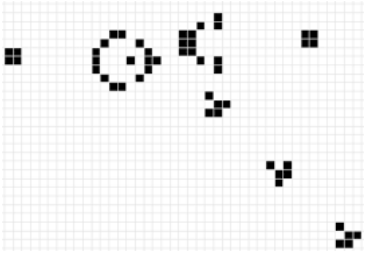
In this section, we dive back into the world of self-organization. After providing some background on cellular automata, we present the Lenia system [39, 40], a recently-proposed generalization of the game of life which was shown to generate a wide range of complex behaviors and life-like structures. We then discuss what makes Lenia a particularly interesting "toy" experimental testbed for automated discovery tools.

3.2.1. Backgound: CA and GoL

Cellular automata (CA) are mathematical models that have been widely used to simulate complex systems or processes. They are particularly interesting as they are rich abstract computational models - some are even Turing-complete *i.e.* capable of universal computation - and yet their dynamics can be described with only a simple and compact set of rules. Their formalization goes back to Von Neumann's and Stanislaw Ulam seminal ideas in the 1940s, which introduced the concept of cellular automata in order to create a theoretical model for a self-reproducing machine [21]. Von Neumann's work was notably motivated by his attempt to understand biological evolution and self-reproduction. Whereas Von Neumann was never able to actually build his self-reproducing machine, his concept of cellular automata became widely influential. Notably, the ability of CA to exhibit complex behavior based on only a few rules made them a primary choice for modeling complex phenomena in various scientific fields, including biology, physics and chemistry. They also have become an especially fundamental concept in the field of artificial life.

In its classic form, a CA is is a theoretical machine that consists of elements called "cells". Each cell has a value, or state, and is connected to certain neighboring cells so that they form a one- or multidimensional lattice: $A = \{a_x\}$, where $x \in \mathcal{X}$ is the position of the cell on the grid and a_x is the state of the cell. The states of the cells change at discrete time-steps $A^{t=1} \longrightarrow \cdots \longrightarrow A^{t=T}$. The new state of a cell is computed based on the states of its neighbors: $a_x^{t+1} = f(\mathcal{N}(a_x^t))$, where $\mathcal{N}(a_x^t)$ is the neighborhood of the cell (including itself) and f is a local update rule which is shared at the different locations. The dynamic of the CA is thus entirely defined by the initialization $A^{t=1}$ (initial state of the cells in the grid) and the update rule f (how a cell updates based on its neighbors).

The Game of Life (GoL), introduced in the seventies by the mathematician John Conway, is probably the most famous example of cellular automaton. GoL is composed of a 2D lattice of infinite size, where cells are either "dead" ($a_x = 0$) or "alive" ($a_x = 1$). At each time step, every cell interacts with its 8 neighbors and can survive, die or give birth according to very simple rules f. At the beginning of a simulation, one chooses how many and which cells are alive and then observes the patterns formed by the cells as they "divide". Despite its extreme simplicity, it was shown that various complex structures can emerge in it. One of the best-known of these patterns is the *glider*: a small spatially-localized pattern capable of moving diagonally across the space. Gliders are important pattern that can be used to transmit information over long distances and that can be combined (or rather "collided") together to create more complex



(a) Glider gun shooting *gliders*

(b) Sir Robin

Figure 3.2.: (a) The glider gun, a relatively complex structure that periodically emits gliders. (b) Sir Robin, the first elementary knightship, discovered in 2018 as a result of partial manual discovery and heavy computer search (see blog-post)

.

structures such as the well-known *glider gun*, an infinite growth pattern and important indicator that GoL could function as a Turing machine (Figure 3.2a). Gliders have been extensively studied and even proposed as a computational model of an autopoietic system [191, 192]. While the glider is quite easy to find, the possibilities for emergence in GoL are very wide and players are still discovering novel patterns (Figure 3.2b).

3.2.2. Lenia: a continuous extension of the Game of Life

Lenia is a recent generalization of Conway's Game of Life rules proposed by Bert Chan in 2018 [39]. Lenia generalizes GoL to the continuous domain by:

- allow cells to take any value between 0 and 1 (continuous states)
- extend the neighborhood to a circular neighborhood of variable radius R (continuous space)
- weighting the neighbors influence by a radially symmetrical kernel K and smooth mapping function G (smooth updates)
- add the value back to the site x with incremental update factor dt (continuous time)

More precisely, Lenia's update rule ($A^t \longrightarrow A^{t+1}$) is defined as

$$A^{t+1} = \left[A^t + dt\ G(K * A^t)\right]_0^1$$

It can be decomposed in four steps:

1. First cells *sense* their neighborhood through a convolution filter K which defines a parametrized kernel formed by b concentric rings:

$$K_C(r) = \exp\left(\alpha - \frac{\alpha}{4r(1-r)}\right), \text{with } \alpha = 4 \text{ (Kernel core)}$$

$$K_S(r;\beta) = \beta_{\lfloor Br \rfloor} K_C(Br \bmod 1), \text{with } \beta = (\beta_1, \ldots, \beta_b) \text{ (Kernel shell)}$$

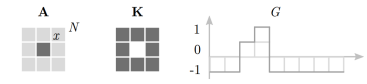
$$K = \frac{K_S}{|K_S|}$$

   Note that Lenia grid A is a torus where neighborhood is circular, such that pixels on the top border are neighbors of the pixels on the bottom border, and same applies for left and right borders.
2. Second, cells convert this sensing into a *growth update* (which can be positive, negative or no growth) through a mapping G which defines a parametrized function:

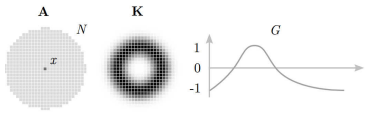
$$G(u;\mu,\sigma) = 2\exp\left(-\frac{(u-\mu)^2}{2\sigma^2}\right) - 1$$

3. Then, cells modify their state by adding a small portion dt of the scalars obtained to their current states.
4. Finally, the state is clipped between 0 and 1 (to prevent negative matter or concentrating too much matter in very small regions)

Note that other kernel cores and growth fields have been proposed in [39], but we did not use them in our works hence do not detail them here.

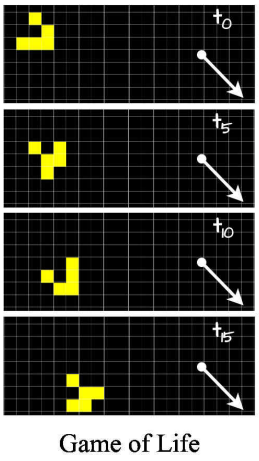


(a) GoL discrete rules

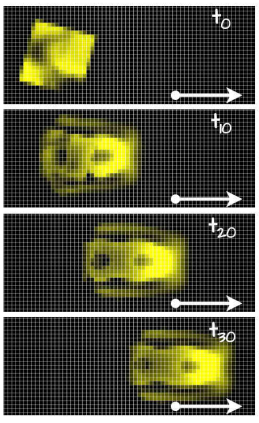


(b) Lenia continuous rules

Figure 3.3.: Rules of GoL and Lenia, taken from Chan [40]



Game of Life



Lenia

Figure 3.4.: Gliders in GoL and Lenia

Lenia parameter space When performing experimentation in Lenia, we generally fix the lattice resolution (*e.g.* 256×256), the total number of steps T (*e.g.* $T = 250$) and number of concentric rings b (*e.g.* b=3). Lenia controllable parameters θ include the initial grid pattern ($A^{t=1}$) and a (4+b)-dimensional set of parameters ϕ controlling Lenia's update rule (Figure 3.5). Note that Lenia is not a single CA *per se* but rather defines a *class* of continuous cellular automata (CA) where each CA instance is defined by a set of parameters ϕ that conditions the CA rule f_ϕ. Once the parameters ϕ conditioning the update rule has been chosen, the system is a classical CA where the initial grid pattern $A^{t=1}$ will be updated. In fact, GoL can be implemented as one specific instantiation of Lenia.

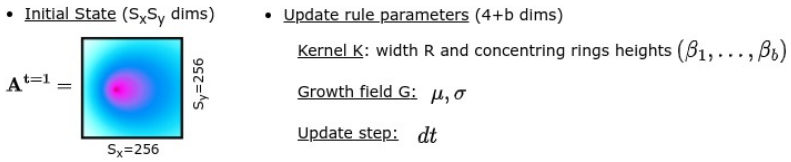


Figure 3.5.: Lenia parameters θ

Lenia: a highly-redundant mapping The parameter space Θ is high-dimensional but a vast area of the parameters will tend to produce "dead" patterns (with all cells being zeros or ones), which is the main attractor of Lenia. Naively exploring the parameters with random or systematic grid will tend to fall into this area and miss out more interesting structures. Similarly finding patterns that spread over the whole (infinite patterns connected by borders) is very easy to find, but finding of spatially-localized patterns (SLPs) is much harder. Finding of moving SLPs, such as the Lenia "gliders" depicted in Figure 3.4, is even more challenging.

3.2.3. Lenia extensions

The works presented in the first part of this manuscript (Part I) were all conducted with the original version of Lenia [39] as above presented. Yet, since these works, Lenia has gained popularity notably in the Artificial Life community where many works are now using it for research (Figure 3.6). Among those, several extensions of the original Lenia "universe" have been proposed (including ours). We discuss those extensions below.

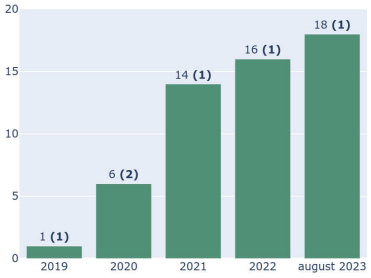


Figure 3.6.: Publications citing Lenia (Chan [39, 40]) between 2019 and now. Numbers in parentheses denote ours.

Multi-Kernels Lenia [40] Inspired from Multiple Neighborhoods CA (MNCA, see blogpost), this extension proposes to have multiple neighborhoods instead of a single one, each one with their own *sense* and *udpate* functions, which allows to introduce more complex interactions between the cells of the grids and produce interesting in dynamics. In Lenia, this was implemented by having K rules *i.e.* multiple kernels K_k each parametrized with relative radius $r_k R$ and corresponding growth mapping G_k. The resulting "sense" (K_k) and "update" (G_k) functions are executed in parallel, and then a fraction dt of the weighted average of the results by factors $\frac{h_k}{\sum_k h_k}$ added back to the state (Figure 3.7). In this variant the number of parameters is $\phi = (4 + b)K + 2$, with 2 global parameters (R, T) and 4+b per rule where (r_k, h_k) replaces the original (R, T).

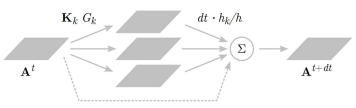


Figure 3.7.: Multi-kernels Lenia, where multiple K_k and G_k are feed into $\sum$ by factors h_k. Taken from [40].

Multi-Channels Lenia [40] Inspired from Neural CA (NCA [104]), this extension proposes to have several communicating grids $A = \{A_c\}$ instead of one, which are called "channels". Intuitively, we can see channels as the domain of existence of a certain *type* of cell. Each type of cell has its own physics : it has its own way to interact with other cells of its type (intra-channel influence) and also its own way to interact with cells of other types (cross-channel influence). In practice, this mean associating each rule (K_k, G_k) with a source channel c_s (the one that is "sensed") and a target channel c_t (the one that is "updated"). The resulting update rule is shown in Figure 3.8. Denoting the number of channels C, the number of intra- and cross-channel kernels K_i and K_c, the total number of parameters is $\phi = (4 + b)K + 2$ with $K = K_i C + K_c C(C - 1)$ rules.

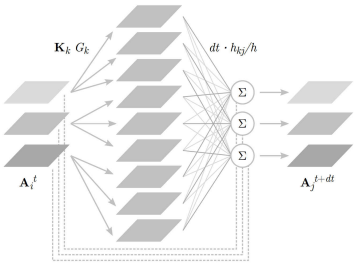


Figure 3.8.: Multi-channels Lenia, where separated channels A_i pass through K_k and G_k, feed into multiple $\sum$ updating channel A_j. Taken from [40].

LeniaChem [•4] LeniaChem[1] is an extension of multi-channels Lenia that we proposed in the context of the Minecraft Open-Endedness Challenge [193] in order to "grow" 3D artifacts in Minecraft. This variant used 3-dimensional grids A_c with C continuous *hidden* channels (one per "block" in Minecraft) and one categorical *visible* channel (with decided what type of block was grown in the world). Intuitively, we can interpret this variant as a form of "artificial chemistry" where each block in minecraft represented a "chemical specie" and the continuous channels represent the "chemical potential" of the associated specie, such that "matter" is created in zones of higher chemical potential with $a^t = argmax_{j \in C} A_j^t$ (Figure 3.9). To prevent the participation of "air" blocks (empty cells) in the update, we also applied an alive-masking step as in NCA [104]. Finally, we replaced the concentric-ring kernels with hyperrectangle kernels generated with Compositional Pattern Producing Networks (CPPN [194]), the idea being to allow for more expressivity in the rules (although resulting in *anisotropic* updates). This variant was implemented in Pytorch, enabling computational speed-ups on GPUs. Here the total number of parameters $\phi = (6 + g)K + 1 + C(C - 1)$ where the $6 + g$ rule parameters are $(\mu_k, \sigma_k, h_k, R_{xk}, R_{yk}, R_{zk})$ and the g parameters of the kernel CPPN genome (variable), and a $C(C - 1)$-dimensional one-hot vector determines which species interact together such that $K = C(intra) + N_i(cross)$ with N_i the sum of this one-hot vector.

1: LeniaChem variant is used in Section 5.5

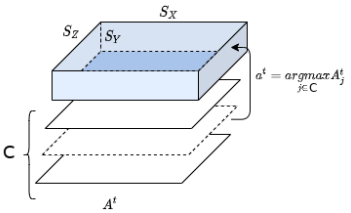


Figure 3.9.: In LeniaChem, Lenia's 3D grid represents the "petri-dish" where chemical "species" evolve in time [•4]

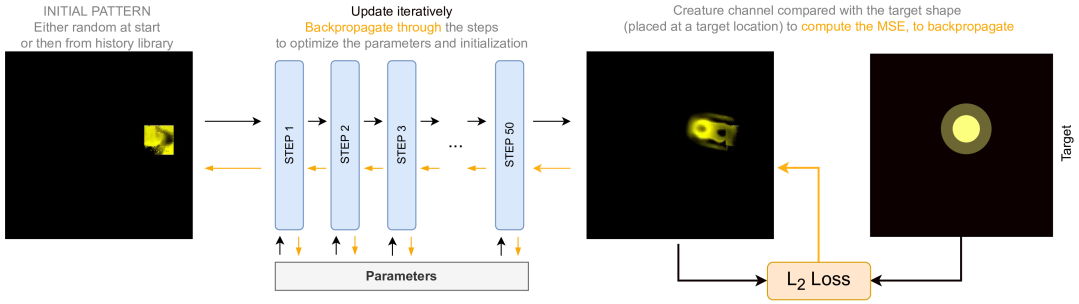
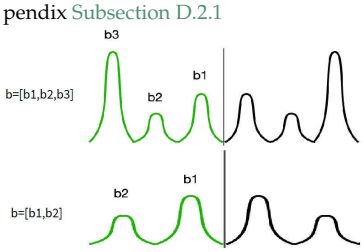


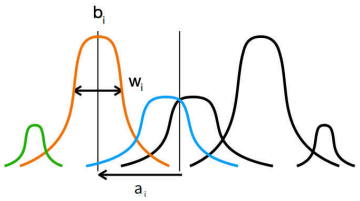
Figure 3.10.: In differentiable Lenia, the CA parameters can be optimized via backpropagation of the gradient [•5].

Differentiable Lenia [•5] Differentiable Lenia[2] is an extension which we proposed in order to make the Lenia system differentiable. Due to the locality and recurrence of the update rule, there is a close relationship between CA and recurrent convolutional networks [35]. In fact, we can see a rollout in Lenia as applying a recurrent neural network to an initial state. If (some of) the network parameters are differentiable, backpropagation can be done by "unfolding" the Lenia rollout and applying a loss at certain time step(s) like in NCA [104] (Figure 3.10). Whereas the original Lenia was close to differentiable (notably in its pytorch implementation), the shape of the kernels is not totally differentiable and not very flexible, as the number of rings b is given by the length of a list which cannot be differentiated. While we can fix b (as in the works presented in the first part of this manuscript where $b = 3$), this limits the shape of kernels and hence limits possibly emergent patterns. To address these issues, we proposed to define kernels as a sum of b overlapping gaussian rings parametrized by heigh b_i, width w_i and center a_i (Figure 3.11). These symmetric "free kernels" allow more flexibility while being very similar to the original ones, allowing to maintain *isotropic* updates and to recover quasi-original kernels (hence to simulate some of the original Lenia gliders). Here the total number of parameters is $\phi = (4 + 3b)K + 2$.

2: Differentiable Lenia variant is used in Chapter 7 and detailed in Appendix Subsection D.2.1



(a) Original Lenia Kernels



(b) Differentiable Lenia Kernels

Figure 3.11.: (a) Original Lenia kernels are parametrized by heights of concentric rings. (b) Differentiable Lenia kernels are defined by a sum of gaussian rings [•5].

Flow Lenia [•6] Flow Lenia[3] is a recent extension that we proposed for adding *mass conservation* to the Lenia system. The main idea of Flow Lenia is to reinterpret the output of the *growth update* $U^t = G(K * A^t)$, which is originally used to *create* new mass (or remove existing one), as an *affinity field* $F^t \approx \nabla U^t$ which is instead used to *move* the distribution of "matter" toward high affinity regions in space. In practice, Flow Lenia uses the reintegration tracking method [195] depicted in Figure 3.12. Flow Lenia can be seen as a grid-based approximation of a particle system (with infinite number of particles) and which has the property of conserving the total mass. Whereas conservation of mass can be seen as a strong *constraint* on emergence, it is hypothesized to be a key driver for diversity and open-ended evolution (OEE) [196, 197]. In Section 7.5, we will see that Flow Lenia opens many interesting perspectives toward OEE both for defining physical constraints in the environment (hence opportunities for emergence) and for having diversity and evolutionary activity emerge *within* the CA (which is not the case in original lenia where different "genomes" ϕ cannot coexist in the same grid A). Flow Lenia is implemented in JAX [198] and has $\phi = 7K + 4$ parameters.

3: Flow Lenia is discussed in Section 7.5

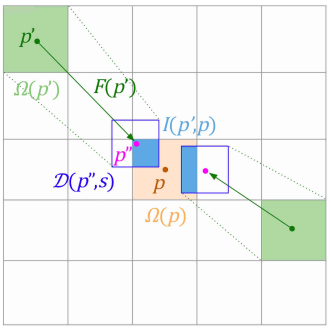


Figure 3.12.: Calculation of incoming matter to cell $p \in \mathfrak{X}$ in Flow Lenia [•6]. Mass contained in cell at location $p' \in \mathfrak{X}$ is moved to $p'' = p' + dt \cdot F^t(p')$ and diffused to a square distribution $\mathfrak{D}$ (emulating brownian motion to avoid concentrating too much matter in p). The proportion of mass from p' arriving in p is given by the integral of $\mathfrak{D}$ on the cell domain of p, $\Omega(p)$, denoted as $I(p', p)$.

Others Other extensions include *Asymptotic Lenia* which replaces the growth update $U^t = G(K*A^t)$ by a *target* update $U^t = T(K*A^t) - A^t$, reinterpreting the result of "sensing" as a "target" rather than a increase [200]. *RD Lenia* reformulates asymptotic Lenia as a reaction-diffusion system composed solely of diffusion and spatially local diffusion terms [201]. *Glaberish* is another extension that proposes to split Lenia's growth function G into two "genesis" and "persistence" functions analogous to GoL's birth and survival rules [202]. Finally, *Particle Lenia* is a recently-proposed reformulation of Flow Lenia, where moving matter is represented as a population of particles rather than a scalar field [199] (Figure 3.13).

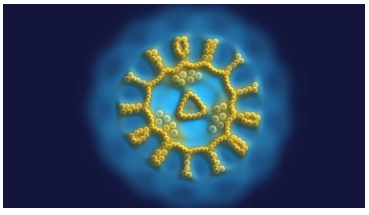
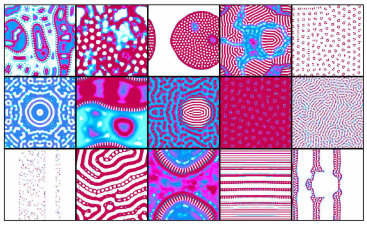


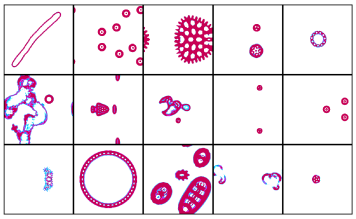
Figure 3.13.: Example pattern in Particle Lenia. Image is taken from [199].

3.2.4. Lenia: an interesting testbed for automated discovery

Whereas many interesting patterns have been found in Lenia, its variants and in CA in general, finding of these structures has so far relied heavily on manual tuning of parameters and initial states, and on the human eye to identify "interesting" patterns, limiting their discovery and analysis. In this thesis, we suggest to use Lenia as an experimental testbed for *automated* discovery because we believe it has many interesting properties.



(a) Turing-like Patterns (TLP)



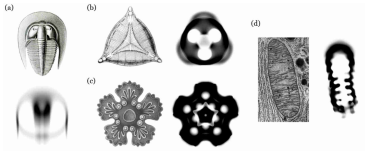
(b) Spatially-Localized Patterns (SLP)

Figure 3.14.: Example of diverse patterns that can emerge in Lenia. Those patterns were discovered by the approach presented in Chapter 5.

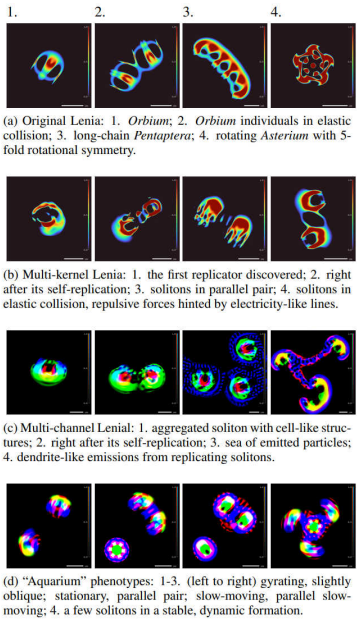
Unbounded possibilities for emergence CA are known to enable the emergence of considerable complexity from set of simple rules, and Lenia is no exception. The space of structures and dynamics that can possibly emerge in Lenia is extremely vast, complex and high-dimensional. We illustrate some of these patterns in Figure 3.14. As we can see, patterns range from *Turing-like patterns* (TLP) characterized by an unlimited spatial growth and resembling reaction-diffusion pattern-formation of fronts, spirals, and stripes reminiscent of biological tissues to *spatially-localized patterns* (SLP), autonomous stable patterns that show interesting behaviors such as locomotion and metamorphosis. Complex systems like Lenia are great candidates for open-endedness in the sense that they offer unbounded possibilities for emergence. At the same time, they are very hard to explore and navigate for us human as the space of possibly-emergent patterns is unknown and challenging to apprehend in practice. Automating the long-term discovery of novel and increasingly complex structures in these systems is an exciting research field: it may even be one way to achieve "open-endedness" in artificial "worlds" [76], and we argue it should be a central quest for automated discovery.

Open scientific question #1: the quest for *open-endedness*

Can we devise an open-ended form of AI discovery assistant whose search is able to reveal this unbounded possibility for emergence?



(a) Life-like appearances in Lenia



(b) Life-like behaviors in Lenia

Figure 3.15.: Intriguing similarities between Earth and Lenia Life. Images are taken from [39, 40].

Interesting life-like structures Not only Lenia can generate a high diversity of complex self-organized patterns from local update rules but it seems that some of them share intriguing resemblances with biological forms of micro-organisms, both in appearance and dynamical behavior (Figure 3.15). Structures include spatially localized organization and symmetries like bilateral, radial and rotational symmetries. Behaviors include regular modes of locomotion like stationary, directional, rotating, gyrating; irregular behaviors like chaotic movements and metamorphosis (shape-shifting); and even advanced behaviors like individuality (*e.g.* robustness to collision), self-replication, or pattern emission (akin to GoL's "guns"). Those patterns are particularly interesting for scientists, not only because they "look like" biological organisms but because they provide an artificial substratum to test theories about the origins of "life" and "agency". However, these patterns are very hard to characterize and find in Lenia, limiting their discovery and analysis. Beyond Lenia, there is a growing consensus that explicitly searching for *agency* in self-organized

systems could allow to "connect the dots" for understanding cognition across "diverse intelligences" (*i.e.* collectives spanning disparate sector of life such as unicellular organisms, organs, plants and animals), which is known as the field of *basal cognition* [138, 203]. Whereas talking of "agents" is unconventional when referring to *e.g.* molecular, bacterial or cellular collectives; several empirical evidences seem to suggest that treating these entities as such could open pioneering opportunities for top-down control in many natural and artificial complex systems [45]. However, these ideas remain quite conceptual at the moment and automating the long-term discovery and characterization of agents in unconventional self-organizing substrates is another very exciting research field. We believe that it could be another central quest for automated discovery, perhaps even leading on the path toward "open-endedness".

Open scientific question #2: the quest for *agency*

Can we efficiently find the parameters leading to the emergence of agential structures?

A "toy" yet rich testbed for automated discovery In conclusion, we believe Lenia is a great testbed which is rich enough to support emergent open-ended and agential behaviors while simple enough to study these scientific questions explicitly, and in a computer-based environment. Lenia features the richness of Turing-complete models and is therefore well suited to test the performance of pattern exploration algorithms for unknown and complex systems. The fact that CA models have been used to study self-organization in various disciplines, ranging from physics to biology and artificial life, also supports their generality, suggesting that successful automated discovery approaches in those systems might transpose to other computational or wet high-dimensional systems.

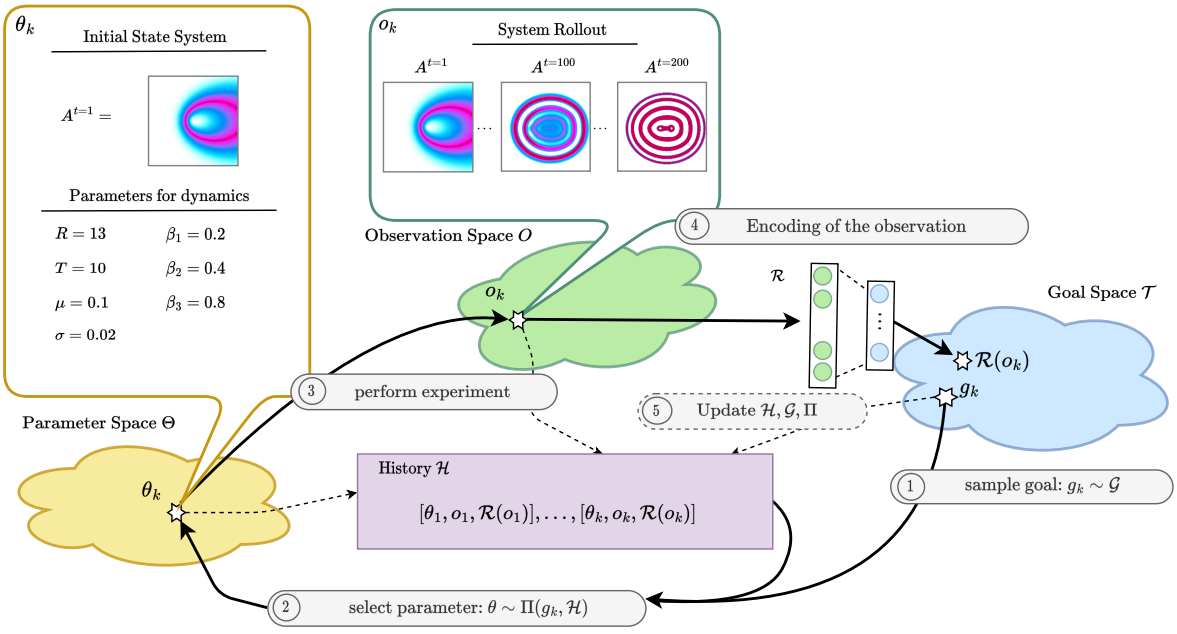


Figure 3.16.: Transposing IMGEPs to assist the scientific discovery of diverse self-organized patterns in complex dynamical systems.

3.3. Problem Definition: IMGEPs for the Discovery of Diverse Self-Organized Structures

The problem we formulate in the first part of this manuscript is: **How to *automatically* find a *diversity* of *interesting* self-organized patterns in the Lenia system using a *limited budget* of experiments?**

This objective shares the same fundamental challenges that the ones addressed in the developmental robotics: achieve sample efficiency in high-dimensional continuous action and outcome spaces. Here, sample efficiency is important to enable the later use of such discovery algorithms for physical systems [64], where experimental time and costs are strongly bounded. We propose to leverage the successful ingredient of recent intrinsically motivated learning algorithms, *i.e.* goal-directed exploration and diversity search, by transposing them to the target application. Transposing IMGEPs to this new application is quite straightforward, as depicted in Figure 3.16. This enable us to more formally pose the problem we target within the context of automated discovery as follows:

Problem Definition: IMGEP diversity-driven search

How can we use IMGEPs to reach a maximally *diverse* set of *interesting* points in the goal space $\mathcal{T}$ using a *limited budget* of experiments?

IMGEPs are a general family of algorithms, such that many variants could be envisaged in practice for the choice of for the goal space representation (R), goal sampling ($\mathcal{G}$) and goal-achievement strategies (Π). We detail these challenges below and introduce the several algorithmic contributions we will present in the next chapters to try and address these challenges. Those contributions are all formulated within the IMGEP computational framework, proposing various algorithmic extensions.

3.3.1. How to characterize "relevant" features of the outcomes?

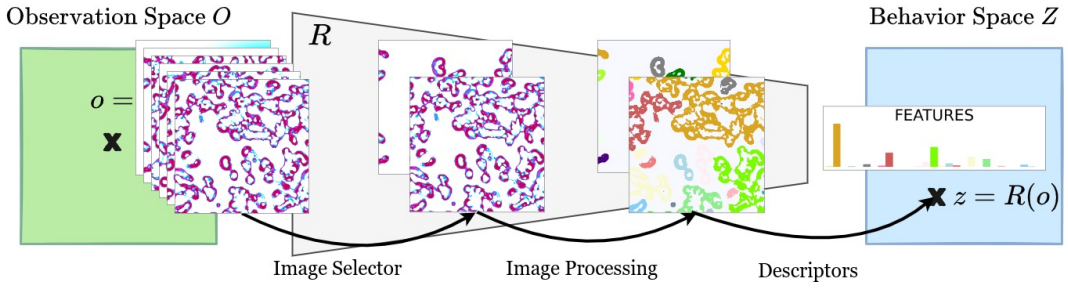


Figure 3.17.: How to process the multi-scale high-dimensional system outcomes into compact representations?

A first challenge consists in determining a representation of patterns, possibly through learning, enabling to incentivize the discovery of diverse interesting patterns. Such a representation guides exploration by

providing a measure of (di-)similarity between patterns. This problem is particularly challenging in domains where patterns are high-dimensional as in Lenia. In such cases, scientists have a limited intuition about what useful features are and how to represent them.

A first possibility is to define such representation R manually, by selecting pre-defined features, *e.g.* using computer vision algorithms to detect the positions of a pattern and its form.

Another possibility is to use representation learning methods to learn autonomously goal spaces for IMGEPs. Representation learning, and more specifically unsupervised feature learning, aims at finding low-dimensional explanatory factors representing high-dimensional input data [204]. It is a key problem in many areas in order to understand the underlying structure of complex observations.

In Chapter 4, we discuss the limitations of using predefined representations, notably their reliance on prior expert knowledge. We suggest that *online* and *unsupervised* representation learning offers an alternative way to automatically encode high-dimensional observations into compact latent code, removing the need for human expert knowledge to define such representations.

In Chapter 5, we suggest that *monolithic* representations R are limited to capture the diverse factors of variations that could emerge in complex systems, whether they are carefully crafted or unsupervisedly learned. To adress this limitation, we suggest to use a *dynamic* and *modular* model architecture for online and unsupervised learning of *diverse* representations $\{R_k\}$.

3.3.2. How to generate "interesting" goals?

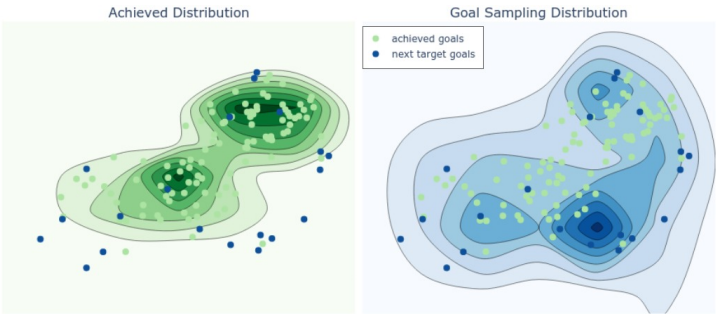


Figure 3.18.: How to select *novel*, *interesting* and *feasible* goals?

Goal Definition As we saw in Section 3.1, a goal $g \in \mathcal{T}$ is defined by a compact goal embedding $z_g \in Z$ and a loss function $\mathcal{L}_g$. In this thesis we define goals as *target features of states*, such that we use the latent space of the representation R as embedding space $Z = R(O)$. That way goal embeddings z_g are target points in Z that the agent should reach. For the loss $\mathcal{L}_g$, we simply use the euclidean distance in embedding space Z, although other (more informative) losses could be envisaged. For instance, in Chapter 7 we will see how the loss function can incentivize not only closeness to the goal position, but other aspects of goal "achievement" such as robustness to a distribution of perturbations.

Given this definition of goals, a second challenge consists in automating the goal generation process to efficiently discover interesting patterns within the goal space $\mathcal{T}$ with a limited budget of experiments. We argue that in the context of automated discovery (and in general intrinisically-motivated goal setting systems), a "good" goal is one that is both *creative* - *i.e.* novel and interesting - and *learnable*, *i.e.* achievable given the agent current knowledge. In this thesis, we use very simple goal-generation mechanisms ($\mathcal{G}$) yet that pursue those dimensions.

In Chapter 5 and Chapter 8, we generate goals *uniformly* in the goal space, but where sampling bounds adapt to the agent discoveries instead of being predefined. Adaptive bounds incentivize sampling outside of the envelope of currently reached goals, which can be seen as a simple form of novelty-based IM incentive [133]. In Chapter 6, we propose to bias the goal sampling toward *interesting* regions of a modular goal space as defined by an external end-user, using simple (and sparse) preference feedback requests. In the first part of this manuscript, as our main aim is to generate a *diversity*, and not to achieve specific goals precisely, we found these simple goal selection mechanisms to be quite efficient. As discussed in Chapter 5, a more critical part however is to define the compact goal representation, as it tends to strongly bias the final discoveries.

However in Chapter 7, in the second part of this manuscript, we have a more precise idea of what we are searching for: patterns with higher-level *functionality* that we can characterize with some predefined proxies but that are very hard to find. In this case we show that the goal generation $\mathcal{G}$ plays a crucial role, and we propose to generate goals $g = (z_g, \mathcal{L}_g)$ following a *curriculum*. The curriculum is defined both in the sampling of goal embeddings z_g (such that we do not sample goals to close nor to far from the set of already reached goals) and in the definition of the goal-achievement loss $\mathcal{L}_g$ (introducing progressively harder environmental conditions to achieve the goal).

3.3.3. How to effectively achieve goals?

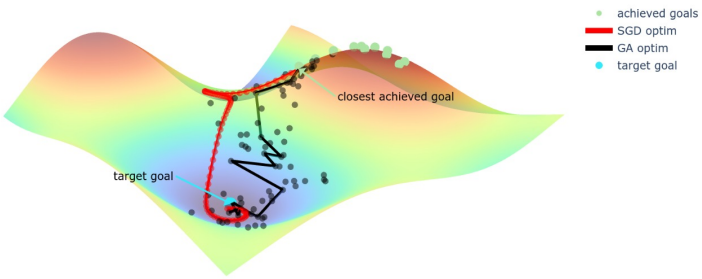


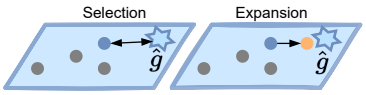
Figure 3.19.: How to make progress toward a goal while using a very small budget of experiments?

Given a goal $g = (z_g, \mathcal{L}_g)$, a third challenge consists in defining the goal-achievement strategy (Π) . Π should help making progress toward the goal while using only few experiments. As we saw in Section 3.1, they are two main ways to implement the goal-achievement strategy Π. The first one, used in RL-IMGEPs, is *model-based*: akin to goal-conditioned RL, it uses a goal-conditioned parametric model Π that is trained on the previous interactions with the environment. In this thesis, we intentionally use low experimental budgets ($10^2 \lesssim N \lesssim 10^4$) to be compatible with applications in physical experimental setups similar to [64]. Therefore,

we only use *population-based* goal achievement strategies (POP-IMGEPs) which leverage independent experiments allowing the use of memory-based sample efficient methods. In POP-IMGEPs, which maintain an archive of previous samples, Π can be decomposed into two steps[4]:

1. the *selection operator* which chooses one or several sample (θ, z) from the archive $\mathcal{H}$ on which to expand to make progress toward g. Selection operates in goal space $\mathcal{T}$.
2. the *expansion operator* that adds M new samples to $\mathcal{H}$, built from the selected samples, again with the aim to make progress toward g ($M \ll N$ being the budget allocated per goal for local optimization). As it is not possible to sample directly in $\mathcal{T}$, expansion operates in .

4: The "selection-expansion" terminology is taken from Chenu et al. [133], analogous to the terminology used in evolutionary population-based algorithms



Population-based goal achievement Π

In this thesis, for the *selection* operator, we simply use a k-Nearest Neighbor (kNN) algorithm to select the most promising samples from history $\mathcal{H}$, *i.e.* the ones whose reached goal z was the closest to the target goal z_g according to the target loss $\mathcal{L}_g$.

For a selected (θ, z), a common approach for *expansion* is to simply apply a small random mutation to the corresponding θ with the hope that it will reach a new goal z' close to the original point z and, by chance, closest to the target point z_g. This simple strategy is the one we used throughout all the works presented in the first part of this manuscript. In fact, as in Grizou et al. [64], we allocate a minimal budget of $M = 1$ experiment per sampled goal to avoid spending too much time on specific individual goals. However, at the difference of Grizou et al. [64] which uses low-dimensional parameter space Θ, in Lenia directly randomizing and mutating the initial state $A^{t=1}$ is not very efficient to produce structured effects. In Chapter 4, we propose to use Compositional Pattern Producing Networks (CPPNs) [194] to randomly generate and mutate Lenia initial states. Given this novel parametrization, we use the simple kNN selection and random expansion variant, which is equivalent to performing Novelty search in the goal space Z. Again, here, our aim is not to achieve specific goals but to rather to find a maximally diverse set of discoveries in Z.

However, in Chapter 7, we use more-advanced *gradient descent* optimization for the expansion, and a local budget of $M = 100$ experiments per goal. In this chapter we aim to find specific patterns which are in fragile equilibrium in Lenia, such that random mutations of the experiment parameters θ (Lenia update rules) can easily break this equilibrium and make the optimization harder. Gradient descent, on the other hand, can be efficient to optimize sensitive parameters in complex chaotic and has already been successfully applied with cellular automata [37]. However, it requires the environment dynamics f and loss function $\mathcal{L}_g$ to be differentiable. Other evolutionary algorithms (EAs) strategies, which do not require differentiable environment dynamics, could also be envisaged for local optimization (Figure 3.19). EAs were shown to enable the optimization of high-dimensional neural networks reaching equivalent performances than gradient descent [205, 206], but they often require much more samples M to recover the precision of the gradient.

3.3.4. How to update internal models?

We have discussed possible implementations for the IMGEP internal models governing the IMGEP *exploration process*, namely the goal space representation (R), goal-sampling strategy ($\mathcal{G}$) and goal-achievement strategy (Π). Another open question when it comes to deploy these models in practice concerns the IMGEP *learning process*, *i.e.* the choice of a strategy to *update* the internal models based on the new discoveries. These updates should lead to changing dynamics in the IMGEP discovery process, which in turn should lead to changing dynamics in the environment hence in the discoveries. In this thesis, we consider three types of possible strategies for updating the models.

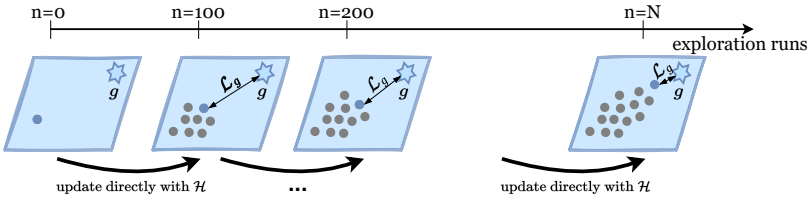


Figure 3.20.: Non-parametric learning

The first one, and most common strategy in population-based IMGEPs is what we refer to as *non-parametric* learning. This is the case, for instance, in novelty-based goal sampling strategies $\mathcal{G}$ that use non-parametric models to estimate density probabilities from the history of reached goals, and then use this estimate to sample new goals in low-density (unexplored) areas. Another common example is the use of k-Nearest Neighbors (kNN)algorithm for the *selection* operator in Π. As shown in Figure 3.20, kNN model learning is directly based on the expansion of the history $\mathcal{H}$: as exploration progress and more points are added to $\mathcal{H}$, kNN gets better and better at selecting promising candidates (here candidates that minimize L_g in Z). Note that non-parametric does not means no-parameters at all, for instance the number of neighbors k in kNN is a parameter (that could potentially be updated during exploration). Rather, it means that the effective complexity of the model linearly grows with the lenght of the database.

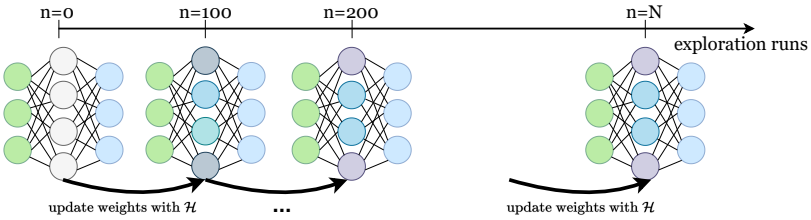


Figure 3.21.: Sequential weight updates

The second, and most common strategy when internal models of the IMGEP are implemented as parametric model (*e.g.* neural networks or gaussian processes) is what we refer to as *sequential weight updates*. Here, parameters of the models (weights) are updated periodically based on the observation collected in the history $\mathcal{H}$. This is for instance what we used in Chapter 4 to update the weights of a neural representation R by periodically training it on the IMGEP self-collected data. Here, the weights of the network are updated in time but not is neural capacity

(number of parameters) such that the model effective complexity is constant O(parameters).

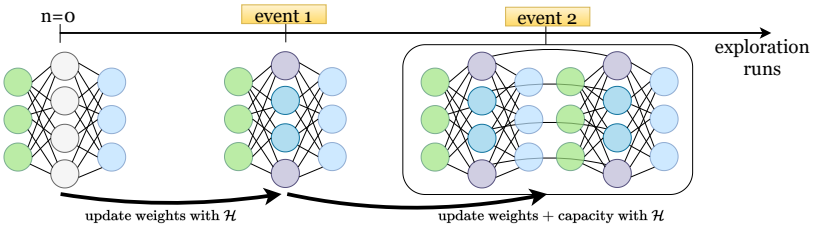


Figure 3.22.: Self-adaptive updates (weights and/or topology)

Finally, there is a third strategy that we refer to as *self-adaptive updates*. Here internal models are updated progressively by the agent without relying on a fixed external strategy but rather on internal events/signals (*e.g.*, catastrophic forgetting, learning progress plateau, etc.). Moreover, not only do these updates modify the model parameters (weight updates) but they can also allocate new parameters (topology updates) *e.g.*, by adding more layers to a neural network or more components to a mixture of Gaussians. While this strategy is more flexible, it is also more complex than standard weight updates, as the engineer must decide *when* the current capacity is insufficient, *where* to expand the capacity, and *what* initialization to give to the new parameters; in addition to the standard question of *how* to update existing parameters (*e.g.*, choice of a training loss and optimizer for NN). While this update strategy is not common in RL-IMGEPs and POP-IMGEPs, nor in standard machine learning, it is increasingly proposed, notably in continual learning scenarios where expanding capacity over time is often crucial. In this thesis, we propose in Chapter 5 a self-adaptive strategy for deciding when, where, and what to expand in the IMGEP internal representation R.

3.3.5. How to integrate human guidance?

At last, but not least, a central question for the human engineer concerns the integration of the human *end-user* in the IMGEP exploration (or learning) loop. In the case of human-in-the loop exploration systems, not only the engineer must find ways to design (and update) the agent internal models but it must find ways for these models to efficiently interact with the human end-user in order to efficiently align the discovery process with the end-user preferences. Human-AI collaborations is a broad and active area of research in many scientific domains, and they are obviously many ways in which human feedback can be integrated in the AI decision-making process [207].

In Chapter 6, we propose several ways in which the human could influence one (or several) of the main IMGEP components: the goal space representation (R), the goal sampling strategy ($\mathcal{G}$), goal-achievement strategy (Π) or learning strategy (model updates), resulting in four possible roles for the human (see Figure 6.2 in Chapter 6).

The experiments presented in Chapter 6 focus on one possible role that we call *preference-based* guidance. Here, the AI is seen as a *discovery assistant*: it cannot predict what a future (unknown) end-user will find interesting in the complex system, but it aims to find diverse discoveries enabling

to quickly adapt the search as soon as the human end-user expresses preferences, through simple and sparse feedback.

In the second part of this manuscript, we explore a second type of human-guidance that we call *environment-based*. Here, the human directly intervenes in the environment to provide cues (or constraints) in order to assist (or obstruct) self-organization toward the goal in the system. For instance, in Chapter 7 we propose to integrate various elements in the Lenia environment such as obstacles, food or attractor objects while remaining within the CA paradigm. In Chapter 8, where we investigate the behavioral abilities of another type of complex system (gene regulatory networks), we propose that classical experiments from behavioral sciences, originally testing various navigation competencies that living agents often exhibit, can also serve as a source of inspiration to define these environmental constraints or cues.

Finally, in Chapter 10, we present the ADTOOL software package that was developed, among other things, to facilitate human interaction with the exploration process via the implementation of custom jupyter interfaces both for vizualisation and feedback.

3.4. Summary

In summary, we presented in this chapter the IMGEP computational framework and its core components (goal-directed exploration, self-generation of goals and reuse of knowledge) in a robotic context (Section 3.1). We then presented the Lenia system, and explained why it is a particularly interesting testbed for automated discovery (Section 3.2). Then, we proposed to apply IMGEP diversity-driven search for the automated discovery of diverse self-organized structures in complex systems like Lenia (Section 3.3). Finally, we identified the main challenges for implementing the IMGEP internal models in practice, as well as for integrating human end-users in the exploration process. IMGEP is a general family of algorithmic processes and all the "challenges" listed above can be positioned within the broader challenges of representation learning (Subsection 3.3.1), automated curriculum learning (Subsection 3.3.2), (black-box) optimization (Subsection 3.3.3), continual learning (Subsection 3.3.4) and human-AI collaboration (Subsection 3.3.5); which are all active and exciting directions of research that go beyond the scope of this thesis.

In this first part of the manuscript (Part I), while only addressing a narrow aspect of these vast challenges, we will propose several key ingredients within the IMGEP computational framework with the aim to design more *open-ended* forms of AI "discovery assistant" in the Lenia system. In the next chapter we focus on the first ingredient that we propose in that direction: the *online* learning of *unsupervised* representations to target a diversity of self-generated goals. In the following chapters, we will go more in depth into the concept of *meta-diversity* search and *human guidance*, which we propose as other key ingredients toward the design of open-ended discovery assistants in self-organizing systems.

Learning of representations to sample goals

4.

What is the aim of this chapter? In this chapter, we propose to combine intrinsically-motivated goal exploration, as presented in the previous chapter, with *unsupervised* and *online* learning of goal space representations. Goal space representations describe the features of patterns for which diverse variations should be discovered, and are hard to define by hand before exploration. Here, we introduce an extension of the IMGEP algorithm which incrementally learns a goal representation on the data collected throughout exploration, removing the need for human expert knowledge to define such representations.

How is this chapter organized? In Section 4.1, we discuss the limitations of standard IMGEP approaches relying on predefined (engineered or pretrained) representations. In Section 4.2, we propose an *online* goal space learning IMGEP variant which we refer to as IMGEP-OGL. In Section 4.3, we present one possible implementation of IMGEP-OGL that leverages two key ingredients that were shown useful in recent developmental robotics works: (i) the use of deep neuronal encoders for unsupervised learning of goal representations from raw pixel perception, and (ii) the use of structured primitives to generate initial states in continuous cellular automata models. Finally, in Section 4.4 and Section 4.5, we discuss and compare various approaches to define and learn goal space representations from the perspective of discovering diverse patterns, and in particular diverse spatially localized patterns. We show that IMGEP-OGL is more efficient than several baselines and equally efficient as a system pre-trained on a hand-made database of patterns.

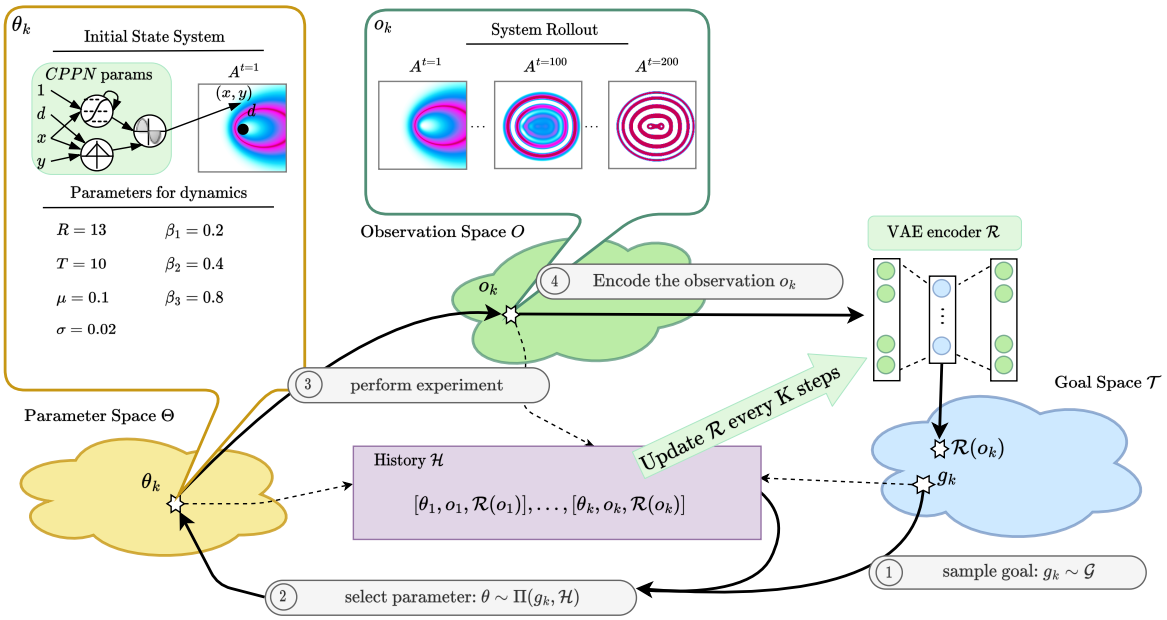


Figure 4.1.: Overview of the proposed approach in Reinke et al. [•1]. The main novelties with respect to the standard IMGEP implementation (Figure 3.16) are shown in light green. They include the use of an online-learned VAE encoder (R) for encoding the outcomes and for defining the goal space ($\mathcal{T}$), as well as the use of CPPN primitives to parametrize the initial state $A^{t=1}$.

4.1. Motivation: Limits of Predefined Goal Spaces

IMGEPs agents self-define goals in a goal space $\mathcal{T}$ that represents important features of the outcomes of their actions, here features of the experimental outcomes. This approach enables the discovery of diverse and novel effects within these goal representations. The goal space $\mathcal{T}$ is often defined as the embedding space Z of a predefined representation ($Z = R(0)$). Although these methods have demonstrated their efficacy in enabling high-dimensional robots to efficiently acquire various skills (as discussed in Subsection 2.3.2), a limitation arises from the reliance on predefined engineered representations of goal spaces. This approach assumes that engineers possess a clear understanding of the key descriptors characterizing high-level behavior, often introducing bias towards a specific type of diversity. For instance, navigation skills might benefit from seeking diversity in $x - y$ space of body position [57, 208] whereas manipulation skills might benefit seeking diversity in the $x - y$ space of object positions or trajectories within the scene [166, 190].

A significant challenge emerges when attempting to learn goal spaces in scenarios where only low-level perceptual measurements are available to the learner (*e.g.*, pixel data from visual states). Recent IMGEP works proposed to address this challenge by using *deep variational autoencoders* (VAEs) [137] to *automatically* learn goal space representations from raw visual states of the environment. For instance, Péré et al. [74] demonstrated that an IMGEP exploring within a goal space made of VAE-learned features led to comparable efficiency than manually crafted goal features for revealing a diversity of controllable effects. Another study by Laversanne-Finot et al. [168] proposed to use β-VAEs representations [209] in environments that included "distractor" objects, *i.e.* objects that cannot be controlled by the robot. β-VAEs is a simple VAE extension which was shown to enhance interpretability and disentanglement of the latent variables in simple generative contexts [210]. Laversanne-Finot et al. [168] shown that the disentangled latent variables improved the efficiency of exploration when compared to traditional VAE representations that lacked sufficient structure for effective exploration. However, both works were demonstrated in simple simulated visual environments, with one robotic arm and few objects as unique source of variation.

However, a main limitations of these works, is that training was done passively and in an early stage on a pre-collected set of available observations [74, 168]. Consequently, while the resulting representation R is not engineered, it remains "predefined" from the agent's perspective, as the agent has no control on it throughout exploration. This greatly restricts the autonomy and flexbility of the learning agent. To illustrate, consider a simple robotic scenario where a new object is introduced midway through exploration. If this object was absent from the precollected dataset used for training representation R, it is likely that the learned features in Z are completely insensitive to this new source of incoming variation, or that they are sensitive to it but in a very unstructured way.

In the unfamiliar self-organizing substrates that we are targeting in this thesis, relying on engineered or pretrained representations is even more limiting. Engineered representations relies on human prior expertise

of what constitute important high-level descriptors of patterns that could emerge in the system, and how to effectively characterize them. Pretrained representations on the other hand requires access to an existing database of representative experiments, *e.g.* experiments pre-collected by a human scientist. Due to the complexity of the targeted systems, the set of precollected observations can hardly be representative of the actual diversity of patterns that the system can produce, limiting the possibilities to discover novel patterns beyond the distribution of pretraining examples. In Lenia for instance, widespread infinite-growth patterns (TLPs) are readily found and likely to be prevalent in pre-collected data. However, spatially-localized patterns (SLPs), which are of particular interest to scientists but much harder to find, might not be optimally represented (and discovered) using descriptors that are learned on TLPs. Instead, we would like the IMGEP agent to be able to *adapt* its representation of the discovered patterns throughout exploration, enabling to adapt goals to novel possibilities in the environment.

4.2. Problem reformulation: IMGEP with Online Goal Space Learning

To remove the reliance on human expertise or human pre-collected data, and to enable more flexibility in the learning agent's discovery process, we propose an *online* goal space learning IMGEP (which we refer to as IMGEP-OGL). Novelties in comparison to the standard IMGEP procedure (Algorithm. 2) are highlighted in bold in Algorithm. 3.

Algorithm 3: IMGEP-OGL

Require: Parameter Space Θ, Observation space O, Budget N

Initialize empty history table $\mathcal{H}$
Initialize goal space representation R
Initialize goal space $\mathcal{T}$ and goal generator module $\mathcal{G}$
Initialize goal-conditioned optimization policy Π

IMGEP EXPLORATION
for i=1 *to* N **do**
  if $i < N_{init}$ **then** `// Initial random iterations`
    Sample random parameter $\theta \sim \mathcal{U}(\Theta)$
  else `// Goal-directed IM iterations`
    Sample target goal $g \sim \mathcal{G}(\mathcal{H})$
    Infer experiment parameters to achieve goal $\theta \sim \Pi(g, \mathcal{H})$
  Execute experiment with θ and observe system outcome o
  Characterize behavior $z = R(o)$
  Write (θ, o, z) to history $\mathcal{H}$

  IMGEP LEARNING
    if $i \mod K == 0$ **then** `// Periodically train network`
      for E *epochs* **do**
        Train R **on observations in** $\mathcal{H}$
      for $(\theta, o, z) \in \mathcal{H}$ **do** `// Hindsight learning`
        Relabel previous discoveries $H[z] \leftarrow R(o)$

return $\mathcal{H}$

Exploration starts with a *randomly initialized* representation R which is then *updated incrementally* during the exploration process. We propose a relatively straightforward training strategy which consists in periodically training the representation R for E epochs on the observation collected in the history $\mathcal{H}$. After each training phase, the set of previous discoveries is re-encoded with the updated representation R, which can be seen as a form of *hindsight learning* [190]. We also suggest that a form of *importance sampling* should be employed to assign greater weight to recently discovered patterns. The idea is that focusing training on the recently discovered patterns should help the representation to adapt to changing dynamics in the IMGEP discovery process. This should in turn increase the likelihood of the IMGEP agent to discover novel and potentially-interesting structures, akin to how human scientists tend to adapt their focus to uncover new insights.

What makes a good representation R? Representation learning, and more specifically unsupervised feature learning, is a broad and active area of research [211]. What makes a "good" representation is an open question, and generally depends on the target application. In image generation contexts, deep generative models like VAEs [137] or GANs [212] are generally evaluated by their ability to produce high-quality visual samples that closely resemble actual data distributions [213]. For control tasks or exploration, one is not so much interested by the high-dimensional generated samples but rather by the structure of the learned latent space. Notably, properties like disentanglement [214], linearity of prediction [215], independent controllability [216] or hierarchical organization [217] of features are posited to facilitate effective exploration and control. Similarly, Laversanne-Finot et al. [168] argue that a good IMGEP representation should be factorized: latent variables within the goal space should respond to "ground truth" factors of variation in the environment.

In the context of open-ended discovery, where recovering all factors of emergent behavioral variations is not possible, we argue that a good representation R should capture degrees of behavioral variations that facilitate the discovery of novel and potentially-interesting patterns. In this chapter, we propose that representations trained in an unsupervised and online manner on the discovered data could learn to extract the "relevant" emergent degrees behavioral variation, while adapting the representation throughout exploration. Here, following recent works in developmental robotics and reinforcement learning, we propose to apply standard VAE-based unsupervised feature learning approaches. As we will see in the next chapter, the choice of the training architecture and loss can strongly influence the final discoveries, and more flexible strategies might be needed.

4.3. Proposed implementation: IMGEP-VAE with CPPN primitives

Figure 4.1 provides an overview of the proposed approach in [•1], with the novel algorithmic ingredients depicted in light green.

4.3.1. VAE for online goal space learning

First, following previous works, we proposed to use the β-VAE training loss [209] to learn latent representations of Lenia patterns.

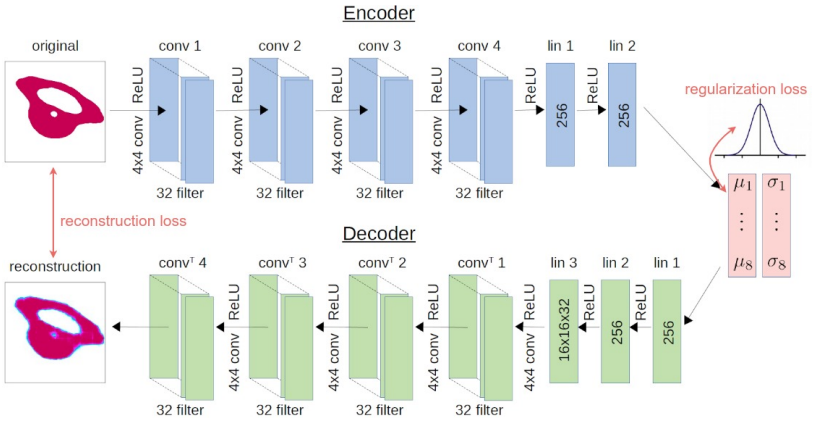


Figure 4.2.: Illustration of the VAE model architecture and training loss [137]

As illustrated in Figure 4.2, VAEs have two components: a neural *encoder* and *decoder*. The VAE is trained to reconstruct input images after compressing them into a compact latent representation. Note that, at the difference of classical auto-encoders (AEs), VAEs do not map images to a fixed vector but to a latent distribution. The training criterion is a combination of the pixel-wise reconstruction error between the input image and the reconstructed output, and a regularization loss encouraging the latent distribution to be close to a prior gaussian distribution with a diagonal covariance structure $\mathcal{N}(0, I)$. β-VAE is a simple augmentation of the VAE loss which gives more weight (β>1) to the regularization loss, with the aim to enhance disentanglement of the latent variables.

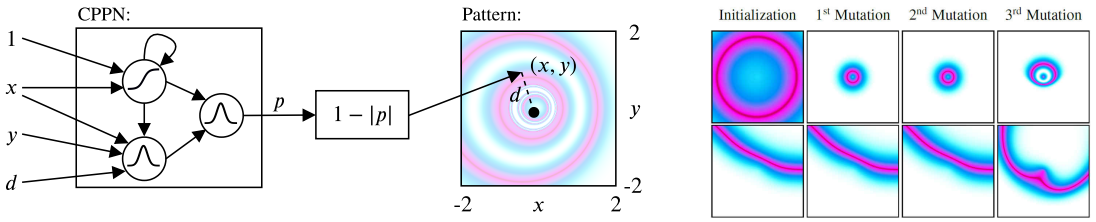
During exploration, the decoder network D is used for training of the VAE but only the encoder R is used to map Lenia observations o to a compact embedding $z = R(o)$. We use a 8-dimensional latent space Z and the VAE is trained periodically every $K = 100$ experiment rollouts. For the importance sampling we simply use a weighted random sampler which, for each training batch, selects 50% of patterns that come from the last K iterations and 50% that come from all previous iterations. At the difference of classical AEs, VAEs latent space is *continuous* which is particularly useful for the IMGEP as it allows easy random sampling and mutation of goals in Z. For goal sampling, we use a uniform sampling strategy within a hyperrectangle defined in Z, where the hyperrectangle is chosen large enough to allow sampling of a large goal diversity.

Related Work Nair et al. [169] and Pong et al. [170] also employed online training of VAEs to learn significant goal space features, similar to this paper. However, their focus was on sequential decision problems in robotics, impacting sample efficiency. These approaches, termed RL-IMGEPs, are distinct from the POP-IMGEPs methods presented here, which leverage independent experiments allowing the use of memory-based sample efficient methods. In evolutionary robotics, parallel work by Cully [218] and Grillotti and Cully [219] combined Quality-Diversity (QD) algorithms with a periodically trained deep auto-encoder (AE) to serve as a behavioral descriptor for QD. However it was applied in

simple simulated visual environments with robotic arm and few objects, as in Péré et al. [74] and Laversanne-Finot et al. [168].

4.3.2. CPPNs for effective parametrization of the initial state

The second key ingredient that we proposed to integrate into the IMGEP pipeline was the use of Compositional Pattern Producing Networks (CPPNs) [194] to parametrize Lenia initial state $A^{t=1}$.



(a) CPPNs for the generation of the initial state

(b) Mutations of CPPN-generated patterns

Figure 4.3.: CPPNs [194] are used for the random generation and mutation of Lenia's initial state.

For the exploration of systems with high-dimensional parameter spaces, the ability to effectively encode, initialize and update the initial state is critical to produce structured effects. In Lenia, directly randomizing the initial state $A^{t=1}$ (256×256 grid cells) leads to patterns resembling white noise, which tend to evolve into dead or global patterns, complicating the search of other types of patterns (Figure 4.4).

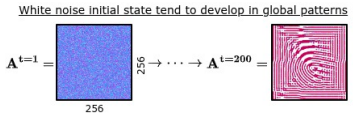


Figure 4.4.: Random sampling of the 256×256 initial state tend to develop into global patterns in Lenia.

CPPNs are small recurrent neural networks, proposed in [194], that were shown to generate patterns with interesting higher-level properties such as symmetry and repetition (Figure 4.3a), and can be "evolved" using random mutations of the network weights and structure (number of neurons, connections between neurons), enabling preservation and elaboration of existing regularities throughout exploration (Figure 4.3b). We integrate CPPNs as part of the experiment parameters θ such that, instead of having the IMGEP directly controlling the initial state $A^{t=1}$, it controls parameters θ of the CPPN used to generate the initial state. Therefore, instead of sampling and mutating directly an initial state, the weights and structure of the CPPN are randomly generated and mutated. In summary, CPPNs provide an efficient way to produce structured patterns and to smoothly evolve already explored configurations, which would be very hard to obtain with random sampling/mutation at the pixel level. Note that, whereas the number of parameters in θ is much smaller than when controlling the low-level 256×256 initial state, it is not fixed as the structure and the number of weights of the randomly sampled and mutated CPPNs can grow throughout exploration.

Related Work The need to effectively encode and initialize the initial state is analogous to a similar problem in the exploration of robot controllers. Direct sampling of robot actions at the lowest actuator level of actuators (and microscopic time scales) is a usually very inefficient. A key ingredient in developmental robotics for sample efficient exploration has been the use of structured primitives (dynamic motion primitives -

DMPs) to encode the space of possible body motions, which enabled to efficiently explore the high-dimensionality of many-dofs body [220]. In the Lenia system, the use of CPPNs to produce structured initial patterns is analogous to the use of DMPs in robotic systems.

4.4. Experimental Methods

Algorithms We compare the results of several IMGEP variants[1] (equipped with different types of goal spaces) and a random exploration baseline:

- **Random exploration:** The random exploration baseline samples parameters uniformly in Θ, for each of the N exploration iterations.
- **IMGEP-HGS:** The first IMGEP variant uses a hand-defined goal space that is composed of 5 features proposed in the original Lenia's paper [40], measuring properties of the final pattern $A^{t=200}$: 1) the sum over the activity of all cells, 2) the number of activated cells, 3) the density of the activity center, 4) an asymmetry measure of the pattern and 5) a distribution measure of the pattern[2].
- **IMGEP-PGL:** The second IMGEP variant uses a goal space that was pretrained on a database of patterns collected before the exploration process started. The training set consisted of 558 Lenia patterns: half were SLPs that have been manually identified by [40]; the other half were randomly generated with CPPNs. It uses the same β-VAE architecture and training loss than the IMGEP-OGL variant.
- **IMGEP-OGL:** The third IMGEP variant is the proposed goal exploration with online learning of the goal space (Algorithm. 3).
- **IMGEP-RGS:** an ablated IMGEP variant that uses, as goal space representation, a randomly-initialized neural embedding network (with the same architecture than IMGEP-OGL's encoder network).

1: implementation details of the IMGEP variants are given in Section D.3

2: See Section B.2.3 for details on these statistical measurements

Testbed environment We use the original Lenia system as testbed environment (see Subsection 3.2.2). Lenia's lattice resolution is fixed to 256×256, pattern evolution is stopped after $T = 200$ time steps, and the shape of Lenia's kernel is fixed to $b = 3$ concentric rings. All exploration algorithms, including the random exploration, interact with the Lenia system via the same parameter space (Θ) which includes a 7-dimensional set of parameters controlling Lenia's update rule $(R, T, \mu, \sigma, \beta_1, \beta_2, \beta_3)$ and a set of CPPN parameters governing the generation of the initial state $A^{t=1}$. For the observation space (O), only the system final state is observed ($o = A^{t=200}$) and fed into the different representation variants.

Experimental procedure Each algorithm underwent 10 independent repetitions of the exploration experiment. Each experiment encompassed $N = 5000$ exploration iterations[3]. For IMGEP variants, the initial $N_{\text{init}} = 1000$ iterations used random parameter sampling[4] to initialize their histories $\mathcal{H}$. Subsequently, during the following 4000 iterations, each IMGEP approach sampled a goal g from a uniform distribution across their respective goal spaces. Sampling ranges for the hand-defined goal space (HGS) are specified in Table B.4, and β-VAE-based goal spaces (PGL, OGL) had latent variable ranges set at $[-3, 3]$. The algorithm then selected the parameter θ_k from prior exploration in $\mathcal{H}$ whose reached

3: we used a experimental budget compatible with the application of the algorithms in physical experimental setups similar to [64]

4: identical initial random explorations were adopted for all exploration algorithms for fair comparison

goal $\hat{g}_k$ had the minimum Euclidean distance to the current goal g in the goal space. This selected parameter θ_k was then mutated to generate the explored parameter θ. For CPPN parameters, the random initialization and mutation followed the process used by CPPN-NEAT in [194] to define the CPPN structure and connection weights. Lenia parameters were initialized randomly through uniform distribution and mutated using a Gaussian distribution around the original values[5].

5: hyper-parameters used for parameter initialization and mutation are detailed in Section B.2.4

Evaluation procedure Our main motivation is to find a high *diversity* of patterns. To evaluate the diversity of discovered patterns, we propose to measure diversity by the spread of the exploration in an *analytic behavior space*. This space is externally defined by the experimenter and the covered area is measured by a binning-based metric (see Subsection 2.1.2). For the analytic behavior space, as we do not have access to an easily interpretable low-dimensional space, we constructed it by concatenating (i) 8 features learned by a VAE trained on a very large dataset of 42500 Lenia patterns identified during the many experiments over all evaluated algorithms (allowing to cover order of magnitude more patterns that what could be found in any single algorithm experiment); and (ii) the 5 hand-defined features from the original Lenia's paper (same than ones used in IMGEP-RGS). We used 7 bins per dimension of the analytic behavior space, resulting in a total of 7^{13} bins.

We also measured the diversity in the space of parameters Θ by constructing an *analytic parameter space*. This space is 15-dimensional and the concatenation of Lenia's 7 dynamical parameters (R, T, μ, σ, β_1, β_2, β_3) with the 8-dimensional latent representation of a β-VAE, but where the β-VAE was this time trained on the initial states of the 42500 Lenia rollouts. This diversity measure also used 7 bins per dimension.

Additionally, we categorized the patterns into 3 families: *dead* patterns (the activity of all grid cells being either 0 or 1), *SLP* patterns (finite and connected pattern of activity) and *TLP* patterns (remaining - usually spread over the whole state space)[6]. This categorization follows the identification of spatially localized patterns (SLPs) in Lenia as shown in Figure 4.5 (also called solitons in Conway's Game of Life), versus other global Turing-like patterns (TLPs). These categories allow us to analyze the exploration behaviors of the different IMGEP variants. In particular, as SLPs are a key reason scientists have used GOL models to study theories of the origins of life [221, 222], we are particularly interested in studying the performance of IMGEPs, and the impact of the goal space representation, on finding a diversity of *spatially-localized* patterns.

6: See Section B.2.2 for details on the implementation of the classifiers.

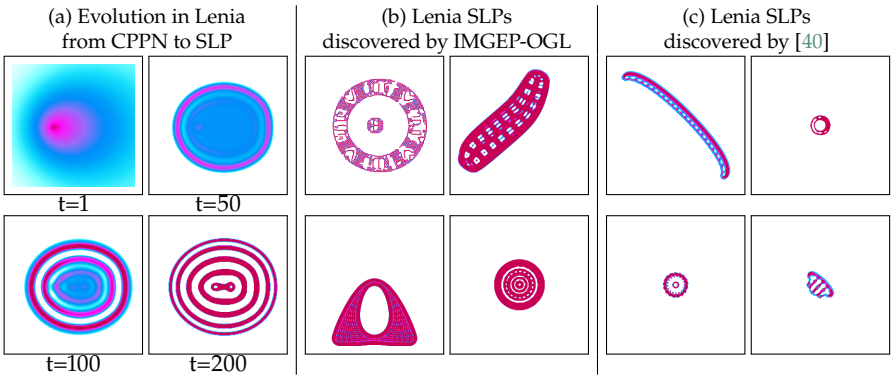
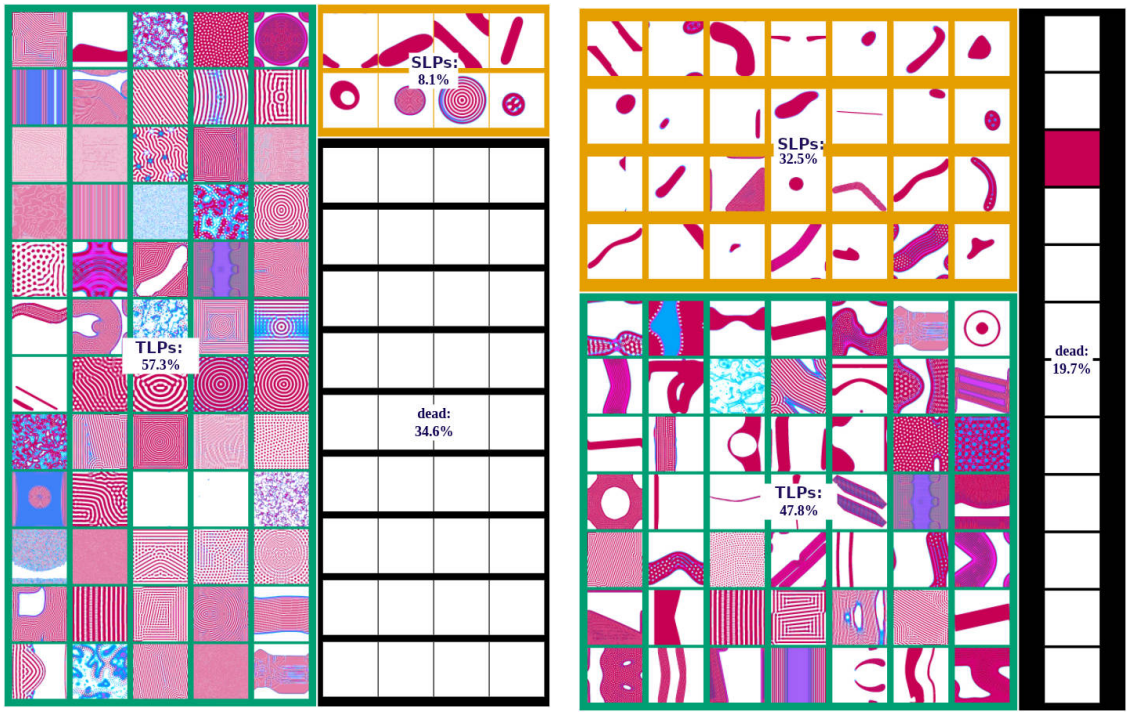


Figure 4.5.: Example patterns produced in the Lenia system. Illustration of the dynamical morphing from an initial CPPN image to a SLP (a). The automated discovery (b) is able to find SLPs as a human-expert manual search (c) by [40].

4.5. Results

In this section, we address several questions evaluating the ability of IMGEP algorithms to identify a diverse set of patterns.

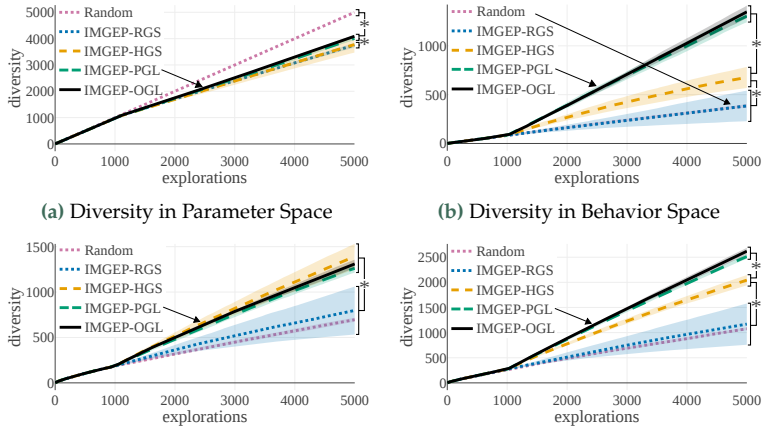
Does goal exploration outperform random parameter exploration? In robotics contexts where scenes are populated with rigid objects, various forms of goal exploration algorithms outperform random parameter exploration [168]. We checked whether this still holds in the Lenia system which have very different properties. Measures of the diversity



(a) Random Exploration

(b) IMGEP-OGL

Figure 4.6.: Example Lenia patterns discovered by random exploration (a) and IMGEP-OGL algorithm (b). Random exploration finds a majority of global patterns that spread over the whole grid (57%), a lot of dead patterns which is a strong attractor of the system (34%), and only a small proportion of spatially-localized patterns (8%). IMGEP-OGL finds less often in the "dead" attractor (still 20%, but the IMGEP also starts with a random exploration exploration phase) and much more SLPs than random exploration (32.5%).



(a) Diversity in Parameter Space

(b) Diversity in Behavior Space

(c) Behavior Space Diversity for SLPs

(d) Behavior Space Diversity for TLPs

Figure 4.7.: Discovered diversity: (a) While random explorations exhibit the highest diversity in the analytic parameter space, (b) IMGEPs achieve greater diversity in the analytic behavior space (except with random representations). (c) IMGEPs with a learned goal space uncover a larger diversity of SLPs compared to a hand-defined space. (d) Learned goal spaces are as effective as hand-defined ones in discovering diverse TLPs. IMGEPs with unsupervised goal feature learning efficiently discover both SLP and TLP pattern diversities. Shown is the average diversity ($n = 10$) with shaded areas indicating standard deviation (for some not visible because too small). Final diversity significantly differs (Welch's t-test, $p < 0.01$) among algorithms within the braces on the right.

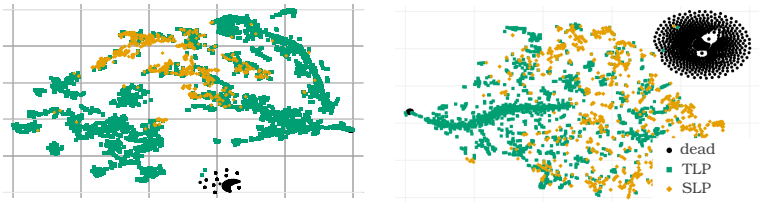
in the analytic behavior space confirmed the advantage of IMGEPs with hand-designed (HGS) or learned goal spaces (PGL/OGL) over random explorations (Figure B.12b). The organization resulting from goal exploration is also visible through the diversity in the space of parameters. IMGEPs focus their exploration on subspaces that are useful for targeting new goals. In contrast, random parameter exploration is unguided, resulting in a higher diversity in the parameter space (Figure B.12a). Figure 4.6 provides a qualitative visual illustration of these results[7].

7: See Figure B.2 to Figure B.6 for similar visualizations for other IMGEP variants

What is the impact of learning a goal space vs. using random or hand-defined features? We compared also the performance of random VAE goal spaces (RGS) to learned goal spaces (PGL/OGL). For reinforcement learning problems, using intrinsic reward functions based on random features of the observations can result in a high or boosted performance [223, 224]. In our context however, using random features (RGS) collapsed the performance of goal exploration, and did not even outperform random parameter exploration for all kinds of behavioral diversity (Figure B.12b). Results also show that hand-defined features (HGS) produced significantly less global diversity and less SLP diversity than using learned features (PGL/OGL). However, HGS found an equal diversity of TLPs. These results show that in this domain, the goal-space has a critical influence on the type and diversity of patterns discovered. Unsupervised learning seems efficient to discover both a diversity of SLPs and TLPs.

Is pretraining on a database of expert patterns necessary for efficient discovery of diverse SLPs? A possibility to bias exploration towards patterns of interest, such as SLPs, is to pretrain a goal space with a pattern dataset hand-made by an expert. Here PGL is pretrained with a dataset containing 50% of SLPs and is able to discover a high diversity of SLPs. Although being trained on many SLPs might help the discovery of diverse SLPs, the direct causation is not clear and might also be due to biases of the VAE training loss. Indeed, the new online approach (OGL) is as efficient as PGL to discover diverse patterns (Figure B.12b, Figure B.12c, Figure B.12d). Taken together, these results uncover an interesting

bias of using learned features with a VAE architecture, which strongly incentivizes efficient discovery of diverse spatially localized patterns.



(a) IMGEP-HGS Goal Space **(b)** IMGEP-OGL Goal Space

Figure 4.8.: (a) Hand-defined and (b) learned goal spaces have major differences shown here by a t-SNE visualization. The different number and size of clusters of SLPs or TLPs can explain the differences in their resulting diversity between the algorithms (Figure 4.7).

How do goal space representations differ? We analyzed the goal spaces of the different IMGEP variants to understand their behavior by visualizing their reached goals in a two-dimensional space. T-SNE [225] was used to reduce the high-dimensional goal spaces[8]. The hand-defined (HGS) and learned (OGL) goal spaces show strong differences between each other (Figure 4.8). We believe this explains their different abilities to find either a high diversity of TLPs (Figure B.12d) ot SLPs (Figure B.12c). The goal space of the HGS shows large areas and several clusters for TLP patterns (Figure 4.8a). SLPs form only few and nearby clusters. Thus, the hand-defined features seem poor to discriminate and describe SLP patterns in Lenia. As a consequence, when goals are uniformly sampled within this goal space during the exploration process, then more goals are generated in regions that describe TLPs. This can explain why HGS explored a higher diversity of TLP patterns but only a low diversity of SLP patterns. In contrast, features learned by OGL capture factors that better differentiate SLP patterns. This is indicated by the several clusters of SLPs that span a wide area in its goal space (Figure 4.8b). We attribute this effect to the difficulty of VAEs to capture sharp details [226]. They therefore represent mainly the general form of the patterns but not their fine-grained structures (as can be seen in Figure 4.9). SLPs differ often in their form whereas TLPs occupy often the whole cell grid and differ in their fine-grained details. The goal spaces learned by VAEs seem therefore better suited for exploring diverse sets of SLP patterns.

8: t-SNE is a dimensionality reduction technique where nearby points in the high-dimensional space stay close to each other in the 2-dimensional visualization. See Figure B.7 for t-SNE and PCA visualizations of other goal spaces variants.

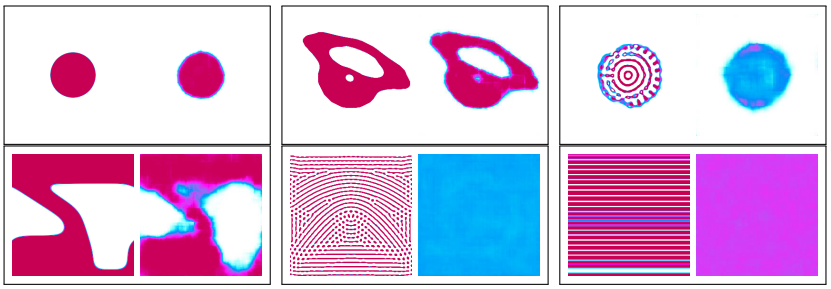


Figure 4.9.: Examples of patterns (left) and their reconstruction (right) by the β-VAE network used for the IMGEP-OGL. The dataset it is composed of SLP patterns (row 1) and TLP patterns (row 2) discovered throughout exploration.

4.6. Discussion and Future Work

In this chapter, we formulated the problem of automated discovery of diverse self-organized patterns in complex dynamical systems with high-dimensions both in the parameter space and in the observation space. We proposed that this problem calls for advanced methods requiring the dynamic interaction between sample efficient autonomous exploration and unsupervised representation learning. We showed that population-based IMGEPs are a promising algorithmic framework to address this challenge, by showing how it can discover diverse self-organized patterns in a continuous model of cellular automata called Lenia.

In particular, we introduced a new approach for learning a goal space representation online via data collected during the exploration process. It enables sample efficient discovery of diverse spatially-localized patterns without relying on prior human expert knowledge. We also analyzed the impact of goal space representations on the diversity and types of discovered patterns, and these analyses hinted that the choice of the representation can strongly bias the final discoveries.

As the Lenia system, while an abstract experimental testbed, shares properties with other artificial or natural complex systems, this work opens interesting perspectives for applying the IMGEP-OGL approach for efficiently mapping the space of behaviors of systems for which no appropriate model exists, *e.g.* systems often encountered in physics or chemistry. Once such a map has been discovered, it could yields transferable insights to scientists for solving various problems by leveraging the diversity of patterns found through the unsupervised discovery process. In fact, Falk et al. [139] recently transposed the proposed IMGEP-VAE approach for the exploration of another class of dynamical systems, known as the Kuramoto model [227, 228], which models synchronization of a set of coupled oscillators. These models are often used in physics and chemistry to describe synchronization phenomena observed in non-equilibrium many-body systems. Interestingly, they demonstrated the utility of the IMGEP-VAE approach for efficiently revealing diverse behaviors in these systems, including previously un-characterized behavior and corresponding new order parameters which can be re-used to reveal behaviors for other related models (Figure 4.11).

(a) Project Website

(b) Blog Post

(c) Interactive visualizations.

Figure 4.10.: Click (or scan) the above QR codes to access the project website, blogpost or for interactive visualizations of the IMGEP goal spaces and discoveries

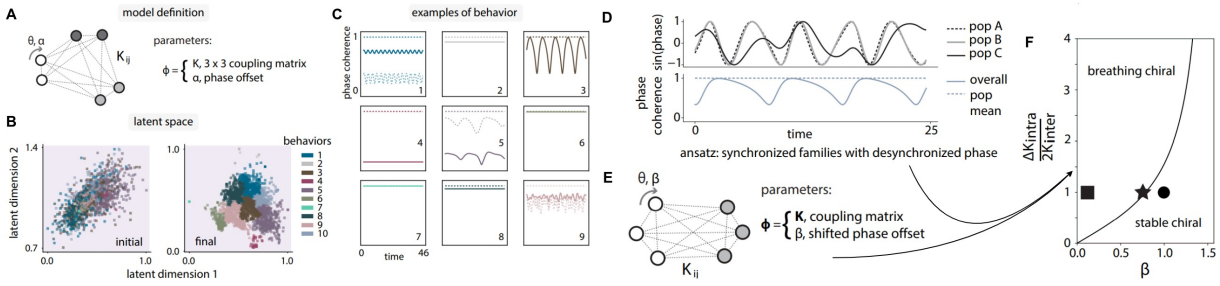


Figure 4.11.: An IMGEP-VAE approach applied to a 3-population Kuramoto model was shown to reveal a previously unknown phase and order parameters in the model, as well as to yield transferable insights for other models (Falk et al. [139]). (A) 3-population Kuramoto model. (B) VAE latent space at the start and end of IMGEP exploration. (C) Phase coherence examples from each of the states identified through latent space clustering. (D) One of the behaviors found in the IMGEP-VAE search shows complete synchronization within populations, but with one population desynchronized from the other two: a "chiral breather" state. This discovered behavior yields transferable insights for solving a simpler two-population Kuramoto model (E), notably to compute the steady-state behavior of the oscillators as a function of the model parameters (F). Figure is adapted from [139].

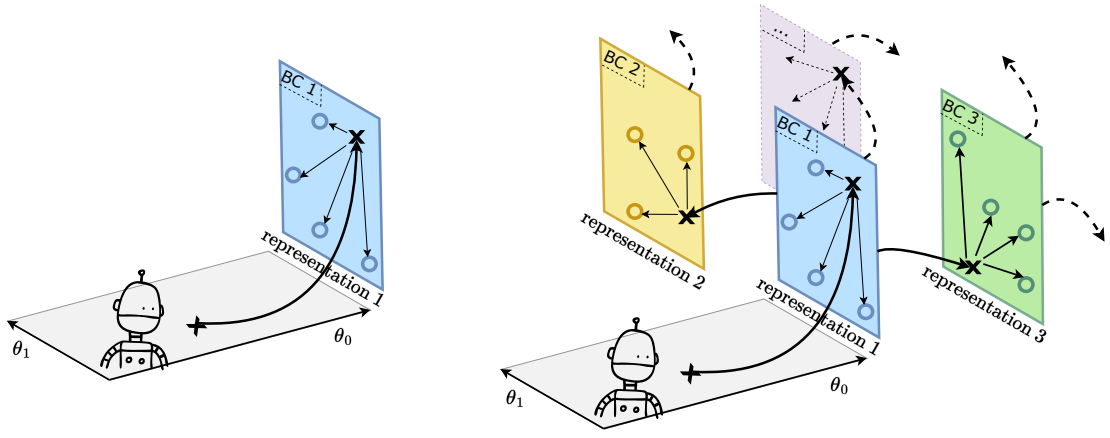
A major limitation of that work, however, is that we assume that using a single (monolithic) architecture and unsupervised training loss (VAE) for the learning of the goal space representation is sufficient to uncover the "relevant" factors of variations of the emergent patterns. As we will see in the next chapter, there is not such thing as a unique or "relevant" set of factors of variations, but it rather depends on what ones decides (and wishes) to observe.

Meta-Diversity Search: Learning and Exploration of Diverse Representation Spaces

5.

What is the aim of this chapter? Within this chapter, we delve into the concept of *Meta-diversity* search, driven by the understanding that a unique, definitive notion of *interesting* diversity doesn't exist. Instead, diverse possibilities emerge depending on the macrostates than one decides to observe. The core objective of Meta-diversity search is to enable continuous seeking of novel source of diversities while being able to quickly adapt the search toward a new unknown type of diversity, *i.e.* search a diversity of diversity (hence the term "meta-diversity").

How is this chapter organized? In Section 5.1, we explore the limitations of conventional diversity-driven methodologies that rely on monolithic behavioral descriptions, whether pre-designed or learned. In Section 5.2, we propose a shift from standard *diversity* search to *meta-diversity* search, where an agent incrementally learns diverse behavioral characterization spaces and discovers diverse patterns within each of them. In Section 5.3, we introduce one possible approach for unsupervised learning of *diverse* representations, leveraging a novel dynamic and modular model architecture called HOLMES. To effectively navigate the meta-diversity landscape, we suggest combining this approach with intrinsically motivated goal exploration processes, referred to as IMGEP-HOLMES. Moving to Section 5.4, we compare the use of monolithic and modular architectures for unsupervised learning of goal space representations and discuss their impact on characterizing and exploring diverse patterns in the environment. Finally, in Section 5.5, we present our contribution to the Minecraft Open-Endedness Challenge, where we leverage the concept of meta-diversity search and the proposed IMGEP-HOLMES architecture for innovative creations in Minecraft.



(a) Standard Diversity Search

(b) Meta Diversity Search

Figure 5.1.: Meta-diversity extends the standard notion of *diversity* to what we call *meta-diversity*, where an agent incrementally learns diverse behavioral characterization spaces and discovers diverse patterns within each of them. The objective of this process is to enable continuous seeking of novel source of diversities while being able to quickly adapt the search toward a new unknown type of diversity.

5.1. Motivation: Limits of Monolithic Goal Spaces

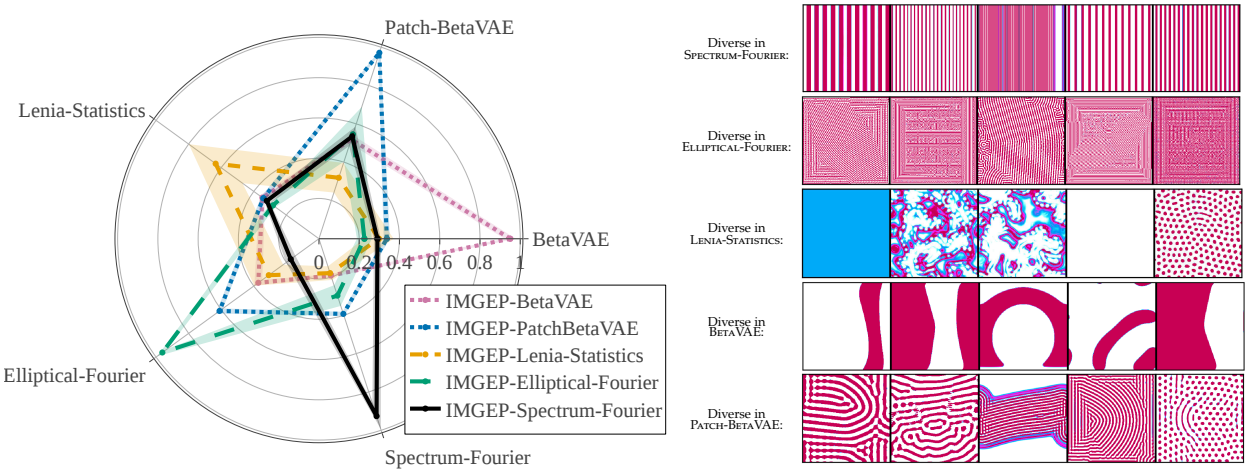
As discussed in Chapter 2, *diversity-driven* approaches can be powerful discovery tools in Science [•1, 64, 136], and can potentially be coupled with objective-driven searches to optimize complex tasks with deceptive reward [129]. These works leveraged recent families of machine learning algorithms that were shown to be effective at creating *behavioral diversity*, namely Novelty Search (NS) [126, 127] and Quality-Diversity (QD) [131, 132] coming from the field of evolutionary robotics; and intrinsically-motivated goal-directed exploration processes (IMGEP) [57, 58] coming from developmental robotics. A known critical part of these algorithms, is that they require the definition of a *behavioral characterization* (BC)[1] feature space which formalizes high-level degrees of behavioral variation in the environment [72]. Similarly, the automated discovery problem defined in the previous chapters assumed that the intuitive notion of diversity can be captured within a single BC space (what we call a *monolithic* representation). So far, this representation has either been hand-defined by the scientist, as in Grizou et al. [64] or unsupervisedly learned with deep auto-encoders directly from raw observations as proposed in Reinke et al. [•1]. While deep auto-encoders have shown to recover the "ground truth" factor of variations in simple generative datasets [210] or in simple visual robotic experiments [168, 218], it is impossible (and not desirable) to recover *all* the degrees of variations in the targeted self-organizing systems. Rather, such monolithic architectures are more likely to recover only a subset of the possibly-emergent behavioral variation, thus revealing a diversity that might match these particular dimensions but potentially overlook other interesting emergent aspects of diversity.

1: Throughout this chapter, we use the term *BC space(s)* to denote the space(s) in which diversity search operates as in [72]. It is equivalent to the *behavior space* of NS and QD and *goal space* of IMGEPs.

In evolutionary robotics, Pugh et al. [72] investigated the consequences for different approaches to diversity search (NS and QD) of varying the degree of *alignment* between the notion of *novelty* (choice of BC space) and the notion of *quality*. Here, they used a relatively simple Maze domain such that "quality" was simply measured as the robot performance to solve the maze task. In that case, quality-alignment of the BC space is intuitive: a BC encoding the final $x - y$ position of the robot strongly aligns with the robot ability to reach the maze target position, whereas a BC encoding the robot's successive facing directions would poorly align, as the robot could spin forever without progressing through the maze. In this domain, as expected, they showed that if the BC was sufficiently aligned with the notion of quality, then searching for novelty *alone* was sufficient to find not only diverse but also high-quality solutions. In the context of self-organizing systems, where the notions of "quality" (or *interestingness*) and "novelty" (or *diversity*) are much more challenging to apprehend, we wondered whether similar empirical investigations could be made to evaluate the impact of the BC space on the final discoveries.

To that end, we constructed 5 BC spaces with 8 dimensions each[2], and the exploration discoveries of an IMGEP equipped with the different BCs as (fixed) goal space are evaluated and compared in figure Figure 5.2. Here, instead on evaluating the impact of the BC space on a predefined external task, we evaluate the impact of the BC space in the IMGEP's ability to achieve high diversity in other (possibly-interesting) BC spaces. In line with automated discovery problem formulated in Chapter 3, the idea

2: a full description of the BC spaces construction and diversity metric can be found in Subsection C.1.1



(a) Diversities of discovered patterns in each BC

(b) Example representative patterns in each BC

Figure 5.2.: Although IMGEPs succeed to reach a high-diversity in their respective BC space, they are poorly-diverse in all the others. (a) Diversity for all IMGEP variants measured in each analytic BC space. For better visualization the resulting diversities are divided by the maximum along each axis. Mean and std-deviation shaded area curves are depicted. (b) Examples of patterns discovered by the IMGEPs that are consider diverse, *i.e.* spread apart, in their respective BC space. See Appendix Subsection C.1.2 for details.

is that producing discoveries *of quality* should mean generating diverse interesting patterns in the eyes of a future (yet unknown) end-user: its evaluation can hardly be predefined and should rather be adaptive to the user and acquired knowledge. The 5 BC spaces shown in Figure 5.2, were constructed to characterize different *types* of diversity based either on:

- Fourier-based image descriptors commonly used in the literature to characterize the frequencies of textures (Spectrum-Fourier) or shape of closed contours (Elliptical-Fourier [229]), notably in cellular-automata [230] and biology [231, 232].
- statistical features that were proposed in the original Lenia paper [39] to describe the activation of patterns (Lenia-Statistics).
- representations unsupervisedly learned on a large database of Lenia patterns, as proposed in [•1] (BetaVAE). Because β-VAE was found to poorly represent high-frequency patterns [•1], another variant trained on cropped patches is proposed (Patch-BetaVAE).

As can be seen in Figure 5.2a, although IMGEPs succeed to reach a high-diversity in their respective BC space, their discoveries are poorly-diverse in all the other BC spaces. These empirical results suggest that, if we could have a theoretical BC model under the form of a goal space that aligns with what a user considers as "diverse", *i.e.* diverse in an interesting or meaningful way, the IMGEP should be efficient in finding a diversity of patterns in that space. In practice however, constructing such a BC is very challenging, if not infeasible. Here, each BC was carefully crafted or unsupervisedly learned to represent what could be "relevant" factors of variations and yet, the IMGEP seems to exploit degrees of variations that might not be aligned with what we had in mind when constructing such BCs. Spectrum-Fourier is a clear example that was constructed to describe textures (in a general sense) but where the discoveries exhibit only vertical-stripe patterns with all kind of frequencies. More generally, when we look at representatives patterns of the discovered diversity per BC space in Figure 5.2b, those do not really align with what a human might consider "diverse" in Lenia.

In conclusion, relying on monolithic (engineered or learned) BC spaces draws several limitations when it comes to applying diversity search as a tool for assisting discovery in self-organizing systems, as the suggested discoveries are unlikely to align with the interests of a end-user.

5.2. Problem reformulation: Meta-Diversity Search

Humans have the remarkable ability to continually conceive, visualize and pursue goals of many types. Since childhood, humans develop the ability to mentally manipulate words, grammatical structures, and narratives, alongside the skill of envisioning visual elements, shapes, and structures for potential artistic creations or objects. As people transition into adulthood, their cognitive capabilities expand, allowing them to envision goals within increasingly complex and abstract domains. In the fields of science, mathematics, and music, for example, humans actively pursue complex goals across various state spaces, ranging from mathematical proofs and theorems to musical harmonies, melodies, and rhythms. This multidimensional creative process empowers humans to transcend the boundaries of established expressive domains, and plays a central role in driving innovation and scientific discovery.

In contrast, almost of all today's "creative" computers, including AI-driven generative programs, are concerned with exploring predefined conceptual spaces [233]. As discussed in the previous section, this is the case of standard novelty search (NS) and intrinsically motivated goal exploration processes (IMGEP) algorithms which generally target the exploration of a predefined BC space. Even in the approach presented in the previous chapter, where goal space representations are incrementally learned as discoveries are made, the BC space is slowly evolving and tend to strongly bias the final discoveries. While these approaches can effectively expand the horizons of discovered behaviors within a given BC space, they fall short in enabling the continuous invention of entirely new BC spaces where previously unimaginable discoveries could occur.

To address these limitations, we propose a natural extension of the standard notion of *diversity* search to we call the *meta-diversity* search (Figure 5.1): in an outer loop, one aims to learn a *diverse* set of behavioral characterizations (called the *meta-diversity*); then in an inner loop, one aims to discover a set of maximally diverse patterns in each of the BC spaces (corresponding to the standard notion of *diversity* in previous work). The objective of this process is to enable continuous seeking of novel source of diversities while being able to quickly adapt the search toward a new unknown type of diversity.

Meta-diversity

Meta-diversity extends the standard notion of *diversity*, where an agent discovers diverse patterns in a monolithic BC space, to what we call *meta-diversity*, where an agent incrementally learns diverse behavioral characterization spaces and discovers diverse patterns within each of them.

5.2.1. Related concepts

Various related concepts of "meta"-level changes have been explored in the literature, describing phenomena observed within naturally occurring open-ended processes.

For instance, when describing human creativity, Boden [233] distinguishes *exploratory* creativity, involving the generation of novel ideas within structured conceptual spaces, and *transformational* creativity, involving the transformation of some (one or more) dimension of the space, so that new structures can be generated which could not have arisen before. Examples of exploratory creativity encompass scenarios such as professional jazz-musician inheriting an accepted style of musical conventions and jazz-specific motifs, and vary them in a number of ways to originate novel jazz performances; or a chemist working with a rudimentary periodic table (prior to its modern completion) encouraged to uncover the potential existence of previously unknown elements. On the other hand, examples of transformational creativity encompass scenarios such as the creation of entirely new musical genres or scientific theories, such as Albert Einstein's theory of relativity or Charles Darwin's theory of evolution, that have revolutionized our conceptual frameworks and transformed our understanding of the world in ways that were previously inconceivable.

The concept of "meta"-level changes also finds resonance with the concepts of *variation* and *major transitions* in biology [234]. Variation typically refers to observed differences within well-defined genetic or phenotypic traits, such as variations in vertebral count or differences in limb size. On the other hand, major transitions refer to qualitatively important changes that opened entirely new state spaces for biological evolution. Examples of major transitions include the emergence of eukaryotic cells, sexual reproduction, multicellular organisms, cooperation in animal societies, and the development of language abilities in humans.

A related distinction in the context of emergent phenomena is that of *weak emergence* and *strong emergence*, as proposed by Bedau [235] and Chalmers [236]. Weak emergence pertains to phenomena that could theoretically be predicted from the known behaviors of all individual components. In contrast, strong emergence relates to a specific form of innovation in open-ended systems: the creation of entirely new state spaces upon which the system performs computations [237]. Transformational creativity shares similarities with strong emergence, as it leads to the emergence of entirely new structures and frameworks that cannot be trivially derived from the existing ones.

Over the last decades, a number of studies in artificial intelligence (AI) and artificial life (ALife) have aimed to better measure and model the concept of open-endedness. Their ultimate goal is to recreate open-ended processes within artificial systems from scratch. Interestingly, the notion of "meta" levels of open-endedness plays a central role in these endeavors [238–240]. For instance, Banzhaf et al. [238] define *novelty* with respect to both a model (the concepts used to describe the entities in a particular system of instances) and meta-model (the concepts used to build the model). They define three types of novelty: type-0 novelty, as novelty within the model (such as new instances of a given type); type-1 novelty, as novelty that changes the model (new types of instances or behaviors); and type-2 novelty, as novelty that changes the meta-model (new concepts needed to define the model). To draw parallels with the works presented in the first part of this manuscript, one could say that 1) diversity search within predefined BC spaces is likely to result in type-0 novelty, 2) diversity search within online learned BC spaces is likely to

result in type-1 novelty and 3) successful meta-diversity search should result in at least type-2 novelty, if not higher-order novelty (if one extends their definition toward $n-th$ order novelty). While there is still a lack of consensus on what makes a system open-ended or not, Soros et al. [76] postulates that the strongest form of open-endedness is one that not only solves problems, but creates new problems for it to solve. In the words of Adams [237], "the strongest form of open-endedness would not only be able to compute solutions in a given state space, but invent new state spaces for it to perform computations in", which is what we aim to achieve with meta-diversity search.

5.3. Proposed implementation: IMGEP-HOLMES

The objective of meta-diversity search is to continuously learn novel and divergent characterizations of the system outcomes while searching to discover diverse patterns within each of them. While they are various mechanisms that could address this objective, several key challenges need to be addressed. A first challenge is to unsupervisedly learn a diverse set of representations to characterize behaviors. A second challenge is to find a diverse set of patterns in each of those BC spaces. In this section, we propose a possible modular and dynamic representation learning approach, called HOLMES, to solve the first challenge (Subsection 5.3.1). Then we propose to use HOLMES representation to progressively grow the goal-space capacity of the IMGEP agent into an organized hierarchical representation, which we refer to as the IMGEP-HOLMES approach (Subsection 5.3.2), enabling to address the second challenge.

5.3.1. HOLMES

HOLMES is a dynamic architecture that "starts small" both on the task data distribution and on the network memory capacity, following the intuition of Elman (1993) [241]. A hierarchy of embedding networks $R = \{\mathcal{R}_i\}$ is then actively expanded to accommodate to the different niches of patterns discovered in the environment, resulting in a progressively deeper hierarchy of specialized BC spaces.

The HOLMES architecture has 4 main components:

1. a base *module* embedding neural network
2. a *saturation* signal that triggers the instantiating of new nodes in the hierarchy
3. a *boundary* redirection criteria that unsupervisedly clusters the incoming patterns into the different modules
4. a *connection* scheme that allows to instantiate new children modules in the hierarchy with feature-wise transfer from their parent module

Figure 5.3 summarizes the overall HOLMES architecture as used in the work presented in Etcheverry et al. [•3][3]. For each module we use a variational auto-encoder (VAE) [137] as base architecture, where only the encoder $\mathcal{R}_i$ is used for exploration (and decoder is used for training). The hierarchy starts with a single VAE $\mathcal{R}_0$ that is incrementally trained

3: Additional intuitions and implementation details are provide in Section C.2

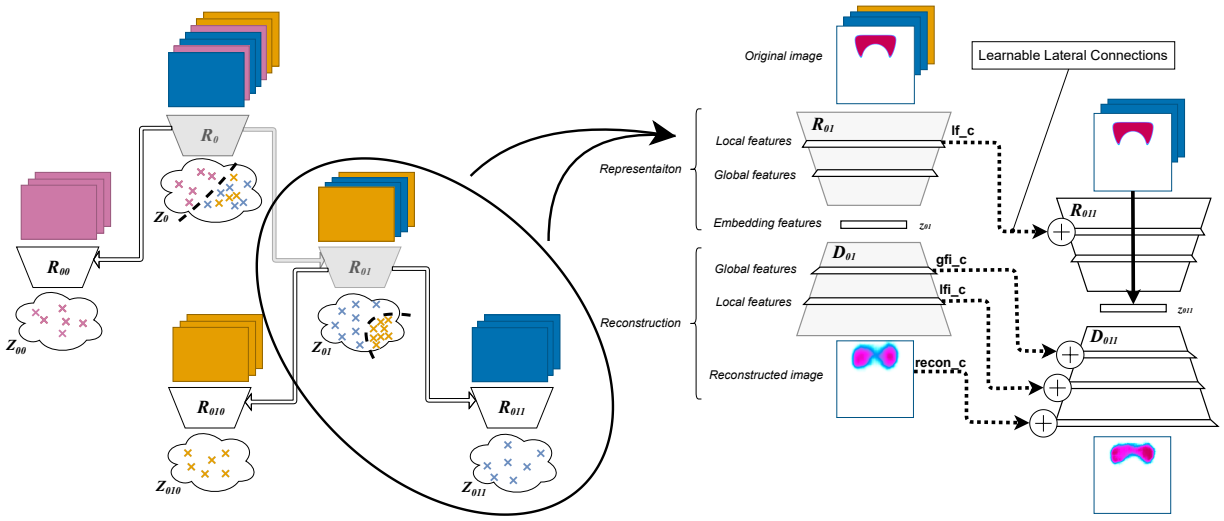


Figure 5.3.: Overview of HOLMES architecture used in the work presented in Etcheverry et al. [•3]. (Left) Each module uses a VAE [137] as the base architecture, where the embedding $\mathcal{R}_i$ is coupled to a decoder $\mathcal{D}_i$ ($\mathcal{D}_i$ is not shown on the left panel for readability). All non-leaf node VAEs are frozen as well as their incoming lateral connections (light grey). The leaf nodes are incrementally trained on their own niches of patterns (represented as colored squares above the embeddings) defined by the boundaries fitted at each node split (curved dotted lines in each BC space, represented as clouds). (Right) $\mathcal{R}_{011}$ is trained to encode new information in a latent representation z_{011} (plain vertical arrow) by learning to reuse its parent knowledge via the lateral connections (dotted arrows).

on the incoming data, encoding it into its latent characterization space Z_0. A *split* is triggered when a node of the hierarchy *saturates i.e.* here when the reconstruction loss of its VAE reaches a plateau, with additional conditions to prevent premature splitting (minimal node population and a minimal number of training steps). Each time a node gets *saturated*, the split procedure is executed as follows. First, the parameters of the parent $\mathcal{R}_p$ are frozen and two child networks $\mathcal{R}_{p0}$ and $\mathcal{R}_{p1}$ are instantiated with their own neural capacity. Besides, additional learnable layers called *lateral connections* are instantiated between the parent and child VAE feature-maps, drawing inspiration from *Progressive Neural Networks* (PNN) [242]. Here, these layers allow the child VAE to reuse its parent knowledge while learning to characterize novel dissimilar features in its BC space (see Section 5.4). Finally, a *boundary* is fitted in the parent frozen embedding space Z_p and kept fixed for the rest of the exploration. In this paper, the boundary is unsupervisedly fitted with K-means by assigning two clusters from the points that are currently in the node's BC space. The boundary serves to *redirect* future incoming observations forwarding through $\mathcal{R}_p$ either to the left child $\mathcal{R}_{p0}$ or to the right child $\mathcal{R}_{p1}$. After the split, training continues: leaf node VAEs as well as their incoming lateral connections are incrementally trained on their own niches of patterns while non-leaf nodes are frozen and serve to redirect the incoming patterns.

In HOLMES, the clustering of the different patterns is central to learn *diverse* BC spaces: it helps divide and relate observations of the world. Here, the clustering is influenced by the choice of the module and connections training strategy, that determines the latent distribution of patterns in the latent space, and by the clustering algorithm itself.

Related Work HOLMES represents one approach we have proposed for learning diverse behavioral characterization spaces in a continual manner, yet various alternative architectures could be considered to facilitate the meta-diversity search. Notably, recent studies have also introduced the idea of dynamically expanding the network capacity of a VAE for *continual* unsupervised representation learning [243–245]. However, these approaches were geared towards passive data characterization rather than active data collection, focusing on tasks such as unsupervised clustering of sequentially-arriving MNIST image classes [243, 244] or disentangling factors of variation in generative datasets [245]. In Appendix Section C.6, we delve deeper into the similarities and distinctions between these models and the HOLMES architecture. Another relevant architecture is the *divergent discriminative feature accumulation* (DDFA) proposed by Szerlip et al. [246]. DDFA is, to our knowledge, the only architecture that explicitly seeks to characterize *divergent* features *i.e.* features that can maximally discriminate among the training examples in novel ways. However, DDFA was also evaluated using passive image datasets (MNIST) and auxiliary classification tasks [246]. In HOLMES, the divergent characterizations from one BC space to another is primordial but not explicitly trained for (yet implicitly encouraged via the lateral connections, see Appendix Section C.5). HOLMES stands somewhere in between the classical unsupervised representation learning objective of auto-encoders, with the intuition that learning to reconstruct the inputs should yield features capturing *meaningful* dimensions of variations; and the purely divergent objective of DDFA. Although DDFA lacks the *modularity* of HOLMES, which we believe is pivotal for efficient diversity search in low-dimensional, stable state spaces, the central concept of DDFA, explicitly learning diverse *distinctions* in incoming data, could be a promising alternative for driving the meta-diversity search in a continuous, open-ended fashion.

5.3.2. IMGEP-HOLMES

The goal space of an IMGEP is usually defined as the BC space of interest, with a representation based on a monolithic architecture R. In this paper, we propose a variant where the IMGEP operates in a hierarchy of goal spaces $\{Z_i\}$, where observations and hence goals are encoded at different levels or granularity, as defined by HOLMES embedding hierarchy $R = \{\mathcal{R}_i\}$. The exploration process iterates N times through steps 1-to-5, as illustrated in Figure 5.4. Novelties in comparison to the IMGEP-VAE procedure proposed in previous chapter are depicted in light green. In this section we detail the implementation choices for each IMGEP step, and refer to Section C.4 for full implementation details.

1) The goal-sampling strategy of the IMGEP is divided in two sub-steps. **(a)** Sample a target BC space Z_k with a goal space sampling distribution $\mathcal{G}_s$. This is the first novelty of the IMGEP-HOLMES approach, where the agent is given an additional degree of control allowing to prioritize exploration in certain nodes of the hierarchy (and therefore on a subset population of patterns). We consider a simple variant where the target BC space is sampled uniformly over all the leaf nodes[4]. **(b)** Sample a target goal g in the selected BC space with a goal sampling distribution $\mathcal{G}$. In this paper, we use an adaptive uniform sampling strategy: the goal is

4: While not leveraged at the moment, as we will see in the next chapter modularity of the goal space enables intuitive guidance during the exploration process

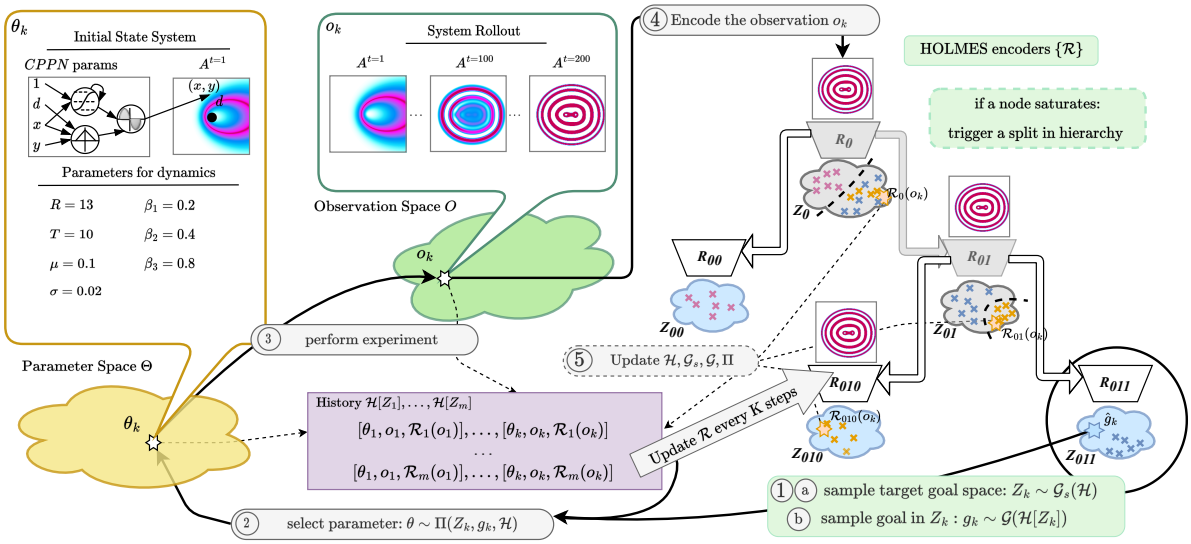


Figure 5.4.: Overview of the IMGEP-HOLMES approach proposed in Etcheverry et al. [•8]. The main novelties with respect to the IMGEP-VAE approach presented in the previous chapter (Figure 4.1) are shown in light green. They include the use of a modular architecture where a hierarchy of behavioral characterization spaces is progressively constructed (HOLMES), and the division of the IMGEP goal sampling strategy into two substeps: (a) sampling of a goal space and (b) sampling of a goal in the selected space.

uniformly sampled in the hyperrectangle envelope of currently-reached goals. Because the volume of the hyperrectangle is larger than the cloud of currently-reached goals, uniform sampling in the hyperrectangle incentivizes to search in unexplored area outside this cloud[5].

5: Again, this can be seen as a form of novelty search in the selected BC space

2) As in previous chapter, the parameter-sampling strategy Π generates the CPPN-generated initial state and Lenia's update rule in 2 steps: (a) given the goal $g \in Z_k$, select parameters $\hat{\theta}$ in $\mathcal{H}$ whose corresponding outcome $R_k(o)$ is closest to g in Z_k; (b) mutate the selected parameters by a random process $\theta = \text{MUTATION}(\hat{\theta})$.

3) Rollout experiment with parameters θ and observe outcome o.

4) Forward o top-down through the hierarchy and retrieve respective embeddings $\{R_k(o)\}$ along the path.

5) Append respective triplets $\{(\theta, o, R_k(o))\}$ to the history $\mathcal{H}$.

HOLMES online training The data distribution collected by the IMGEP agent directly influences HOLMES splitting procedure and training procedure by determining which nodes get populated and when. In this paper, we incrementally train the goal space hierarchy every $K = 100$ exploration step for $E = 100$ epochs on the observations collected in the history $\mathcal{H}$. During training, leaf node VAEs as well as their incoming lateral connections are incrementally trained on their own niches of patterns. As in the previous chapter, importance sampling is used at each training stage, giving more weight to recently discovered patterns.

Related Work The study by Pugh et al. [247] explored how the alignment between the behavioral characterization (BC) space and an external notion of *quality* in the environment (as discussed in Section 5.1) impacts novelty search approaches. In this context, they introduced a basic modular novelty search approach as a proof of concept for the potential

benefits of searching diversity across different BC spaces. Their investigation focused on a robotic maze environment equipped with two sets of predefined BCs (one for agent position and one for agent direction). The results demonstrated that driving the exploration using both sets of BCs simultaneously led to a higher likelihood of discovering *quality* behavior, indicative of task-solving abilities. However, at the difference of HOLMES, BCs were predefined and fixed, limiting the generalization to complex environments. In parallel, numerous techniques have emerged for state representation learning in reinforcement learning scenarios (for an overview, see Lesort et al. [248]). Those typically rely on auto-encoder models such as VAEs [169, 170], tuning the loss for targeted properties at the feature level such as disentanglement [214] or linear predictability [215], or coupling the encoder with predictive forward and inverse models [120, 214, 249–252] or priors of independent controllability [168, 216]. However, these studies all relied on a single embedding space, where all the observed instances are mapped to the same set of features and differs from HOLMES, which progressively grows the capacity of the agent's visual world model into an organized hierarchical representation.

5.4. Results

In this section we compare two exploration variants:

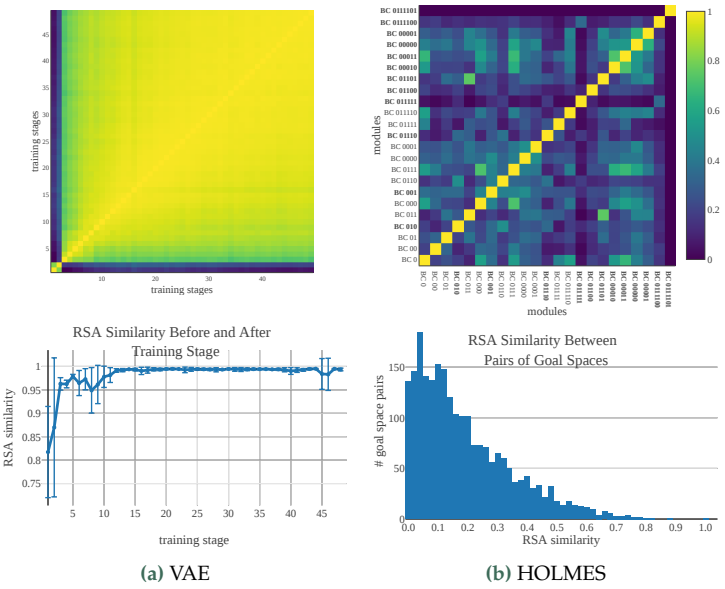
- **IMGEP-VAE**, an IMGEP equipped with a monolithic VAE as goal space representation which is learned online throughout exploration akin to Reinke et al. [•1]
- **IMGEP-HOLMES** which is defined in Section 5.3. HOLMES is also updated incrementally throughout exploration. HOLMES expansion is stopped after 11 splits (resulting in a hierarchy of 23 VAEs) and uses small-capacity modules and connections, such that its final capacity is still smaller than the monolithic VAE[6].

6: VAE and HOLMES architectures are detailed in Appendix Subsection C.4.2

Both IMGEP-VAE and IMGEP-HOLMES were given a budget of $N = 5000$ exploration runs, starting with $N_{init} = 1000$ initial random runs followed by 4000 goal-directed runs. For all algorithms, we conduct 10 repetitions of the exploration experiment with different random seeds. Please refer to Appendix Section C.4 for the full experimental settings.

5.4.1. Learning to characterize different niches

First, we are interested to investigate (1) whether and how much does the VAE learned representations evolve throughout exploration, and (2) whether and how much does the different VAE representations learned by IMGEP-HOLMES differ from one another. To quantify the (dis-)similarity of the learned representations, we propose to use representational similarity analysis (RSA), a technique coming from systems neuroscience [253]. We use RSA metric to quantify how much the representations embeddings (encoded behaviors) evolve throughout exploration for the IMGEP-VAE learned representation (Figure 5.5a) and for the IMGEP-HOLMES learned modules (Figure 5.6). We also use it to compare, at the end of exploration, the representations learned by the different HOLMES modules (Figure



(a) VAE **(b)** HOLMES

Figure 5.5.: RSA similarity index between 0 (dark blue, not similar at all) and 1 (yellow, identical). (a) Representations learned by IMGEP-VAE are compared in time between the different training stages. On the top, RSA matrix is shown for one IMGEP-VAE seed. On the bottom, mean-std curves (for the 10 seeds) are provided showing the RSA index similarity of the learned representation between consecutive training stages. (b) Representations learned by IMGEP-HOLMES are compared here at the *end* of exploration between the different HOLMES modules. On the top, RSA matrix is shown for one IMGEP-HOLMES seed. Leaf BC nodes are depicted in bold and BC spaces are ordered by their creation time (left-to-right in x-axis). On the bottom, the histogram of RSA index similarity, between all pairs of modules and aggregated over the 10 seeds, is shown for IMGEP-HOLMES.

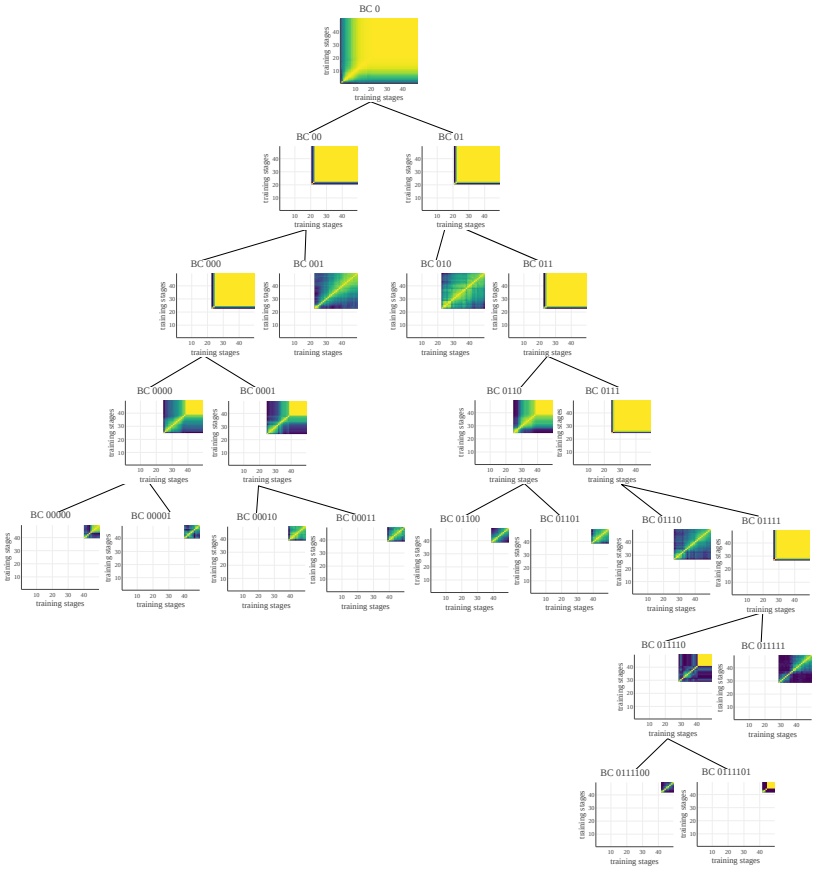


Figure 5.6.: Example of a hierarchy of behavioral characterization spaces learned by HOLMES. In each node, we display the RSA similarity index between 0 (dark blue, not similar at all) and 1 (yellow, identical), where representations are compared in time between the different training stages. HOLMES starts with a root node (BC 0, top) and iteratively splits the learned latent spaces, resulting in a tree structure (with leaf nodes at the bottom). When a node is split, the parent node is frozen and learning only continues in leaf nodes. For instance the root node BC 0 is split at training stage 21, which is why RSA indexes of BC 0 are always 1 (yellow) after stage 21 and reversely RSA indexes of the children BC 00 and BC 01 only start at stage 21.

5.5b). Different metrics have been proposed to compute the representational similarity, here we use the linear centered kernel alignment (CKA) index [254][7]. As can be seen in Figure 5.5a, after only 2 training stages the RSA similarity index between consecutive training stages is already very high (RSA>0.95) and after 15 training stages the VAE representation is completely saturated (RSA≈1). This means that, after only 200 explored patterns[8], the learned features of the monolithic VAE stop to evolve (or at least evolve very slowly). Therefore, even though the VAE is *incrementally trained* during exploration, the monolithic representation fails to adapt to

7: We refer to Appendix Section C.3 for CKA computation details

8: VAE representations are trained for 50 epochs every 100 exploration runs

the newly discovered niches of patterns, which in turn limits the scope of the IMGEP discoveries (similarly that for the pretrained goal spaces presented in Section 5.1).

On the other hand, as can be seen in Figure 5.5b, HOLMES succeeds to learn representations that are highly dissimilar from one module to another (RSA indexes closer to 0 than 1), which allows to target the discovery of diverse "types" of diversity. Note that here, RSA analysis is not shown for one VAE module throughout exploration but at the *end* of exploration and between the *different* VAE modules. An ablation study (see Section C.5 of appendix) highlighted the importance of HOLMES *lateral connections* to learn divergent features from one module to another, which is essential to enable the meta-diversity search. Without these connections, all VAE modules tend to learn similar features despite being trained on different niches of patterns (Figure C.5).

The full temporal analysis of HOLMES VAE modules is given in Figure 5.6. Here, we can see that node splits occurred at training stages where the RSA similarity index is high (yellow). For example, we see on the figure that the root node BC 0 saturates after approximately 15-20 training stages, which results in the instantiation of two child nodes BC 00 and BC 01[9]. Interestingly we can observe that some nodes (like BC 000) saturate much more quickly than others (like its sibling BC 001), which can be seen as a form of *learning progress*: some nodes represent state spaces where progress (finding more representative examples) is still to be made, whereas other nodes are already well covered and further splitting is needed to reveal new (potentially-interesting) state spaces to explore.

9: indicated by the fact that RSA plots for these nodes starts at training stage 21

Finally, as can be seen in Figure 5.7, the learning of divergent features in HOLMES can sometimes help the VAE modules to progressively capture more and more fine-grained details of the input patterns. This coarse-to-fine reconstruction ability, which is usually absent and a well-known limitation of VAEs [226] (though not necessary to learn divergent features), was however found to vary from one seed to another and depend on the results of HOLMES clustering. For instance, if the clustering results in one node having a majority of "stripes" patterns in its training set, its VAE is likely to successfully learn to reconstruct them (and vice versa).

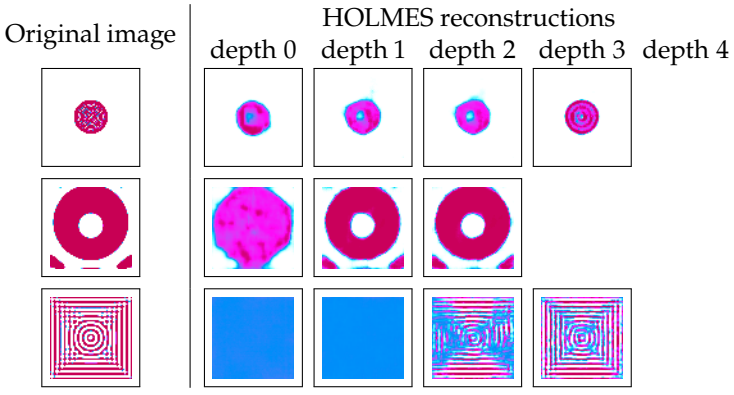


Figure 5.7.: Examples of patterns and their reconstructions along HOLMES tree. Figure adapted from [•2].

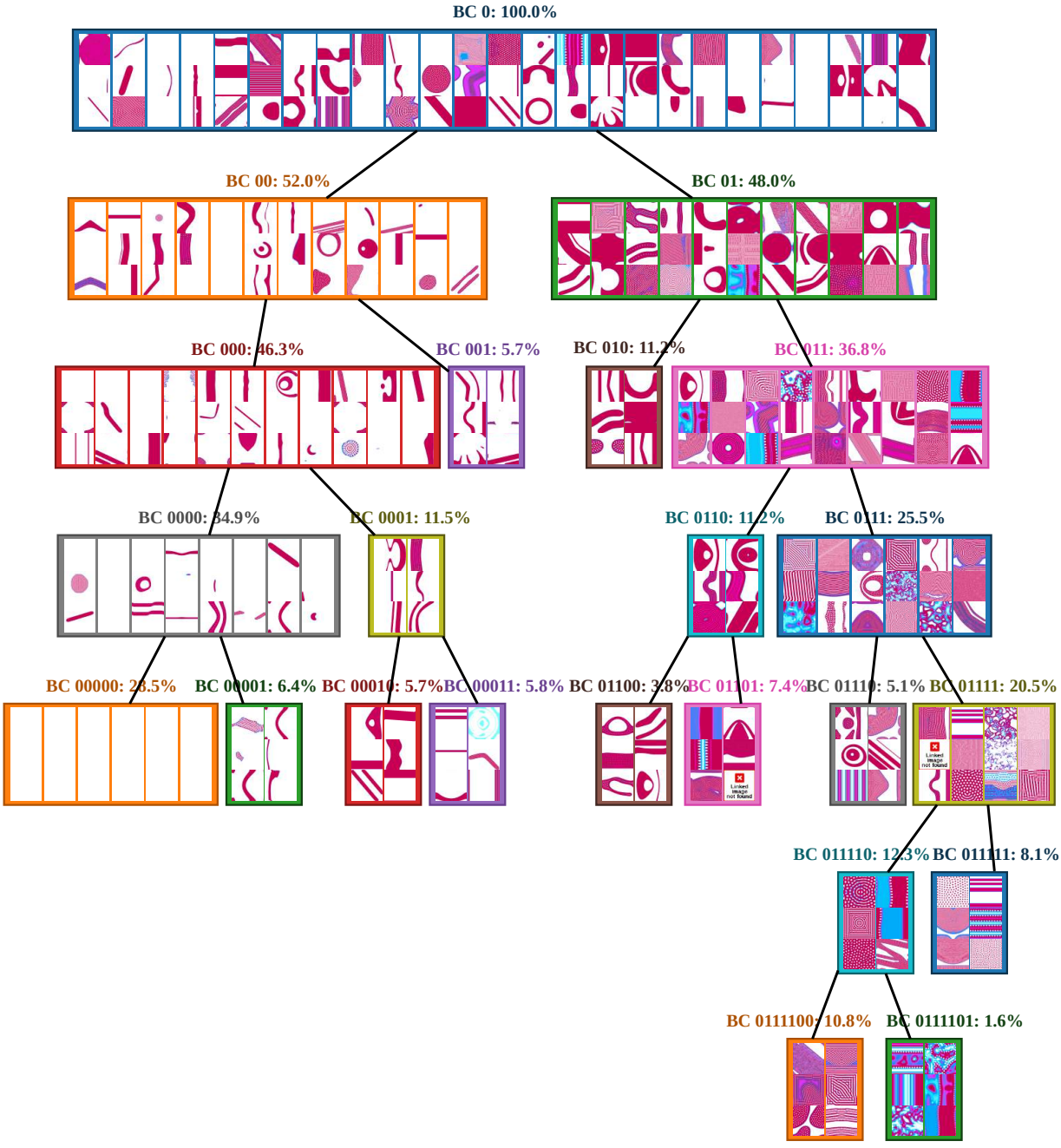


Figure 5.8.: Examples of patterns discovered by IMGEP-HOLMES within the learned tree hierarchy. The hierarchy is the same as in Figure 5.6. In each node we display the percentage of discovered patterns within that node, as well as a set of representative patterns from that node (*i.e.* representatives of the diversity, see procedure Subsection C.1.2). Size of the rectangles (and number of patterns per node) reflects the indicated percentage. For instance, patterns shown in the root node are representative of the diversity of all the discovered patterns in that particular run (100% of the patterns). The boundaries fitted when splitting each non-leaf node makes each pattern follow a particular path in the hierarchy, from the root node to a leaf node. The full database of discovered patterns and an interactive visualization of HOLMES BC spaces (and associated niche) can be found on the project website.

5.4.2. Learning to explore different niches

Secondly, we were interested to investigate the effectiveness of using the divergent learned BC spaces by IMGEP-HOLMES to uncover various *types* of diversity. However, as highlighted in Section 5.1, the quantitative evaluation of diversity in a complex system like Lenia poses important challenges. Opting for one or multiple arbitrary analytic BC spaces may not yield valuable insights, as being diverse based on such BC spaces may not align with our conception of diversity[10]. Therefore, in this section, our focus remains on *qualitative* assessment of the identified patterns[11].

Examples of patterns discovered by the IMGEP-HOLMES exploration within the established hierarchy of goal spaces (as depicted in Figure 5.6) are presented in Figure 5.8. These patterns exemplify the kind of diversity uncovered in their respective nodes. One can qualitatively observe that the splitting procedure tends to separate the discovered patterns into what seems to be visually distinct niches, enabling the meta-diversity search. Notably, BC 01 (right branch of the tree) exhibits a significantly higher proportion of Turing-like patterns (TLPs) compared to BC 00 (left branch), while the reverse is observed for Spatially-Localized Patterns (SLPs). Note how, despite IMGEP sampling goals uniformly across existing leaf BC spaces, the percentage indicated in each node does not reflect this uniformity (for example, only 5.7% of the patterns fall in the leaf BC 001). The interpretation is that leafs with low percentages correspond to unstable niches: when a goal is sampled in such a leaf, the small mutation applied in the parameter-sampling policy is sufficient to produce a pattern which is different enough to fall in another leaf.

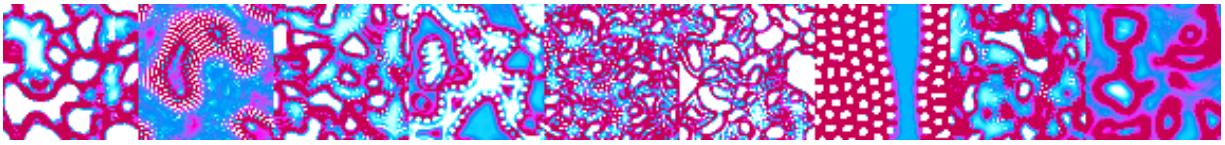
As we used a small budget of experiments ($N = 5000$), only a small number of patterns resided in the leaf nodes at the end of exploration. To further investigate the diverse "types" of patterns that could emerge in the discovered niche, the tree structure was fixed (preventing additional splits) and exploration was prolonged for 10000 additional runs. Surprisingly, the most interesting discoveries happened in the bottom-right niche of the hierarchy (BC 0111101). Whereas the patterns first discovered in that niche (N<5000) were fluid-like TLP patterns spreading over the whole grid (Figure 5.9a), pushing diversity-search within that state space revealed many interesting behaviors (Figure 5.9). Those not only included the discovery of stable and/or moving solitons similar to the "lifeform" patterns that were manually-identified in Chan [39] but also the discovery of behaviors whose existence remained unknown in the original Lenia variant. Those included forms of *pattern-emission*, where highly dynamical fluid-like structures where observed to emit SLPs[12], including gliders (Figure 5.9d). We could also witness the co-existence of several behavior "modes" within the same grid (*e.g.* oscillating, stable and shape-shifting behaviors in Figure 5.9h), which is quite rare in Lenia. These results confirm the interest of *divergent* search for finding interesting behaviors in *unexpected* niches of patterns. Here, the solitons didn't emerge as anticipated within the left hierarchy branch, abundant with SLP, but instead arose from evolving seemingly unrelated patterns. However, there is still a big part of "luck" given that these discoveries were facilitated by HOLMES effectively clustering these fluid-like patterns together and learned visual features[13] which, when varied, led to such discoveries.

10: In Chapter 6, we will explore how integrating human perspectives in the evaluation process can aid in selecting BC spaces that better align with human-intuitive notions of diversity

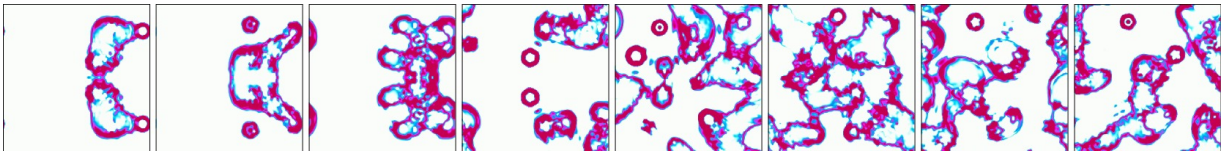
11: On the project website, visualization tools and interactive web-interfaces are proposed to facilitate such evaluation

12: While Lenia is an abstract model, this shares intriguing resemblances with the hypothesized emergence of protocells on Earth from a primordial "soup"

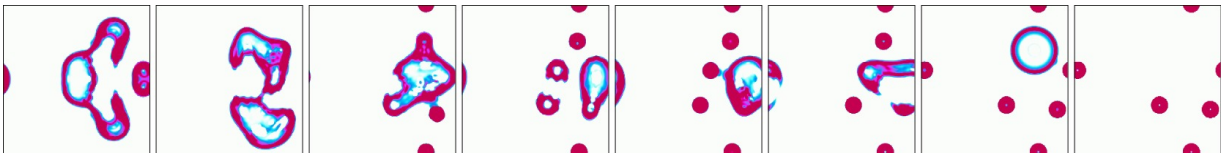
13: HOLMES only encodes the final states ($A^{t=200}$), such the intriguing dynamics seen in Figure 5.9 are a "side effect" of IMGEP-HOLMES searching for diverse morphological traits



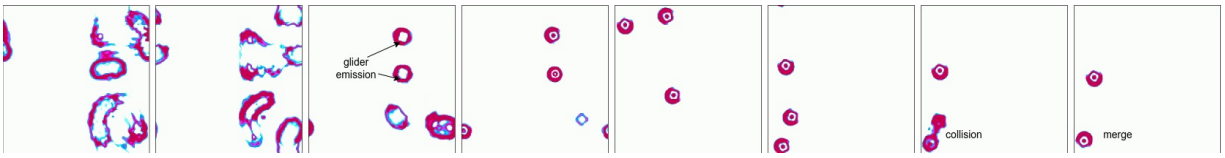
(a) Patterns originally discovered in the bottom-right node of the hierarchy (BC 0111101)



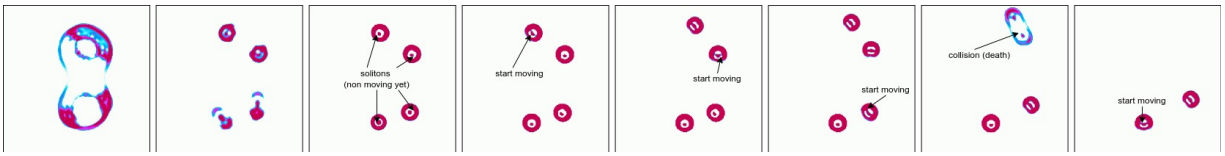
(b) Here, solitons are emitted from the highly-dynamical fluid-like patterns but quickly absorbed back into the moving matter



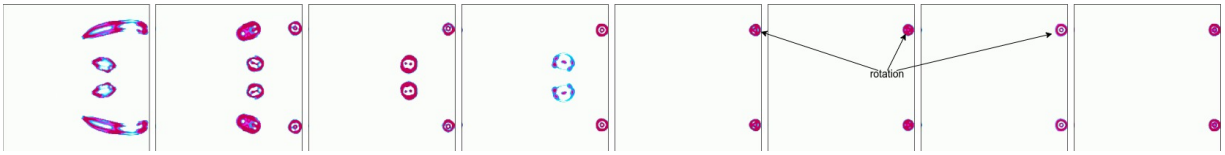
(c) Here, solitons that were emitted from the fluid-like patterns succeed remain stable (as the fluid-like dynamics disappear)



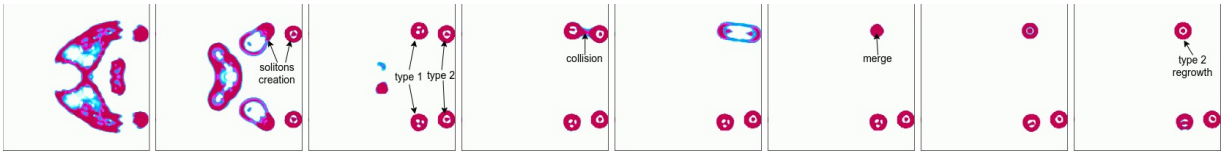
(d) Here, patterns emitted are *gliders* showing individuality and directional movement, although not robust to collision (merge)



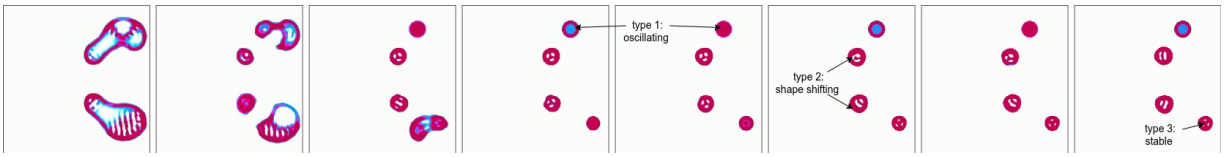
(e) Here, patterns emitted seem at first stable (radially-symmetrical) but slowly start to move one after the other (as symmetry breaks)



(f) Here, the self-organized solitons show *rotational* movement



(g) Here, solitons seem to co-exist into two different *shape modes*, with collision leading to merge and regrow in the second mode



(h) Here, solitons seem to co-exist into different *behavioral modes* (oscillation, shape-shifting, and stable)

Figure 5.9.: Examples patterns and behaviors taken from the niche of discoveries made by IMGEP-HOLMES in BC space 0111101. (a) Example of patterns that are initially discovered in that niche by IMGEP-HOLMES (N<5000). (b-h) Individual examples (one per row) that are then discovered in that niche when prolonging search (5000<N<15000), with the BC space being selecting by the IMGEP ≈ 8% of the time (as there are 12 leaf nodes in the hierarchy). The displayed examples were manually identified by us (human evaluator) as "interesting" among the niche of discoveries. Corresponding videos can be found on the project website.

5.5. Minecraft Open-Endedness Challenge

The Minecraft Open-Endedness Challenge[14] is an annual competition that prompts participants to consider how to achieve "open-endedness" in the Minecraft environment. This challenge essentially asks whether it's possible to create an artificial system that can generate endless novel surprises within Minecraft. Our submission, which participated in the first edition of this competition in 2021[15], proposed a strategy combining the use of a *self-organizing system* and a *meta-diversity search* as a potential approach for achieving that purpose. In our implementation, we simulated an artificial "chemistry" system which is based on the Lenia model of continuous cellular automata but adapted to the Minecraft environment (LeniaChem, see Section 3.2.3). This system functions as a mechanism for "growing" artifacts[16] within Minecraft. We then proposed that integrating a "discovery assistant" that employs the concept of *meta-diversity search* within the LeniaChem system could lead to an open-ended system capable of continuously generating diverse types of Minecraft creations. To that end, we again propose to leverage the IMGEP-HOLMES strategy for driving the artifact generation process.

14: https://evocraft.life/

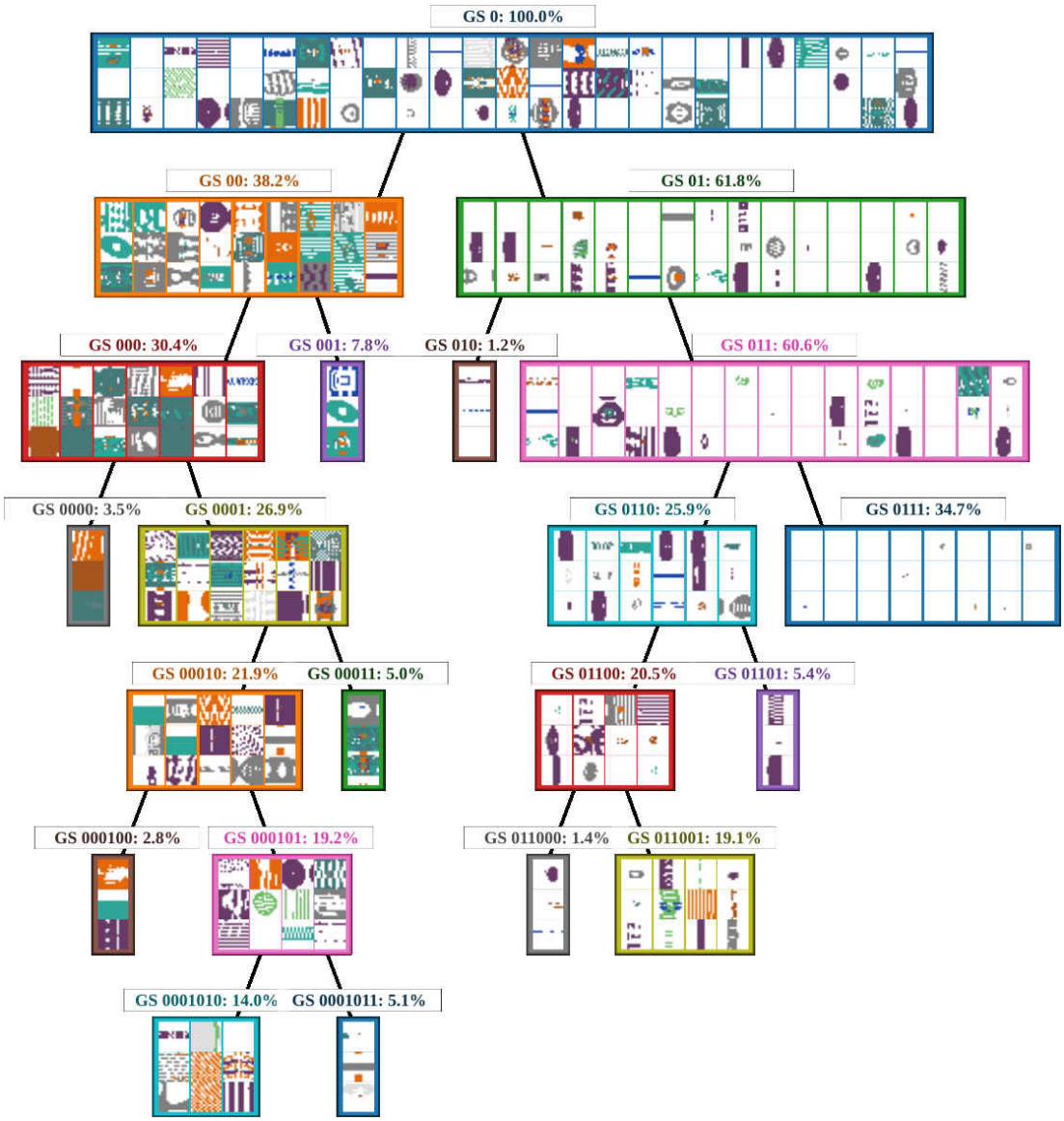
15: the competition was part of the Gecco 2021 conference competition track

16: In this section the word "artifact" is equivalent to "pattern"

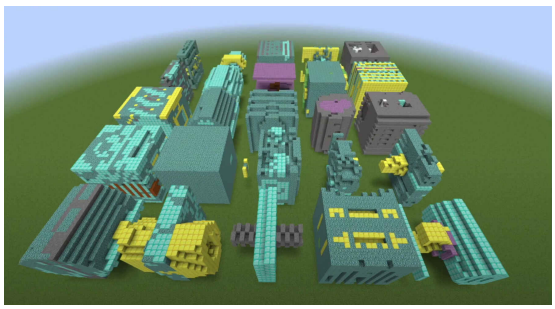
Altogether, the operation of our "discovery assistant" in Minecraft can be summarized intuitively as follows: Starting with a set of basic chemical blocks (e.g., water, wood, air, etc.), the discovery assistant assembles an initial "mixture" within a specific area of the Minecraft world (resembling a "petri-dish"). This assistant can also control certain "physics" parameters that influence the development of the final artifact (similar to factors like temperature or pH). Initially uncertain about the spectrum of artifacts that can self-assemble from these "mixtures", the assistant begins with random exploration, which proves inefficient as many mixtures tend to vanish out (dead patterns in LeniaChem). After this initial random exploration phase, the assistant transitions to learning how to characterize discovered artifacts in a latent behavior characterization (BC) space. This knowledge aids the assistant in targeting novel goals and selecting parameters that are more likely to achieve those goals. Over time, as the assistant successfully uncovers diverse artifacts within its current BC space, its representational capacity becomes saturated, limiting its ability to identify new sources of diversity in the environment. To address this, the assistant can generate new representations with additional capacity to capture novel features of the emergent artifacts. It can also decide how to cluster its findings into distinct niches, enabling specialization and exploration of divergent search spaces, all the while using accumulated knowledge from its various BC spaces. Each time a node reaches its capacity, it can expand by splitting in two novel child nodes, leading to increasingly complex representations that drive the discovery of novel artifacts in Minecraft.

Figure 5.10.: The video of our submission as well the full database of discoveries can be visualized on this link.

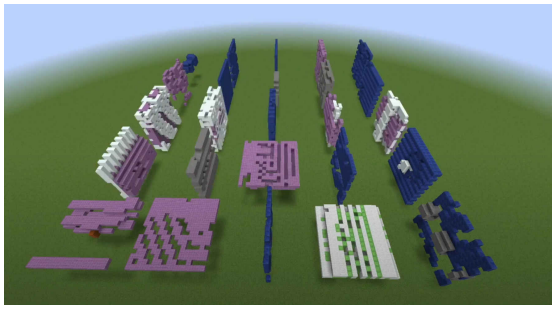
Our experiments encompassed both 2D (64^2 and 32^2) and 3D (16^3) scenarios, enabling the assistant to create diverse buildings in Minecraft. Figure 5.11 shows the resulting hierarchy and representative artifacts that have been actively constructed by the artificial discovery assistant during exploration in the 3D LeniaChem system. Once again, we can qualitatively observe that the splitting procedure tends to separate the discovered artifacts into what seems to be visually distinct niches, enabling the generation of *diverse types of diverse artifacts* in Minecraft.



(a) Visualization of IMGEP-HOLMES learned tree hierarchy



(b) Example artifacts "grown" in GS 000



(c) Example artifacts "grown" in GS 010

Figure 5.11.: Examples of 3D structures "grown" in Minecraft by IMGEP-HOLMES. (a) Visualization of the learned HOLMES tree hierarchy as well as a set of representative artifacts from each node (same than in Figure 5.8). As the artifacts are in 3D (16^3), we only show the central slice (z=8) but (b-c) gives an overview of how the structures look like in Minecraft. "GS" stands for goal space and is equivalent to "BC". (b-c) Example of Minecraft artifacts "grown" when sampling goals in GS 000 (b) and GS 010 (c).

5.6. Discussion and Future Work

In this chapter, we have defined the concept of *meta-diversity search*, in which an artificial "discovery assistant" progressively acquires a diverse range of representations to define behaviors and then explores these representations to uncover varied patterns within each of them. We proposed that this problem calls for advanced methods requiring the dynamic interaction between unsupervised learning of *diverse* representations and autonomous *sample efficient* discovery of diverse behaviors in the learned representation spaces.

To that end, we have introduced HOLMES, a *dynamic* and *modular* model architecture that systematically expands the agent's world model into a well-structured hierarchical representation. Through the fusion of this architecture with intrinsically motivated goal exploration processes, referred to as IMGEP-HOLMES, we have demonstrated its effectiveness as a discovery assistant. This assistant learns to characterize and explore a multitude of pattern niches within the continuous model of cellular automata, known as Lenia. In some cases, this even led to the discovery of gliders from unexpected niches of patterns, including unseen pattern-emitting lifeforms when their existence in the original Lenia variant remained an open question [40]. Furthermore, we suggested that combining meta-diversity search and self-organizing systems open up promising possibilities in the pursuit of "open-endedness" in artificial systems, which we leveraged in our submission to the Minecraft Open-Endedness Challenge.

(a) Website accompanying [•3]

(b) Blog post accompanying [•4]

Figure 5.12.: Click (or scan) the above QR codes to access the project websites accompanying our (a) NeurIPS 2020 paper and (b) submission to the Minecraft open-endedness challenge.

While IMGEP-HOLMES is one possible implementation of the proposed concept of *meta-diversity search*, many other implementations could be envisaged. For instance, since publication of our work, an approach called MC-AURORA has been proposed by Cazenille [255] to combine a quality-diversity search approach with modular unsupervisedly-learned auto-encoders (AEs) in complex reinforcement learning scenarios (Figure 5.13). At the difference of IMGEP-HOLMES, MC-AURORA uses a flat ensemble with a predefined number of AEs which are all trained on the same dataset (all collected discoveries). In the context of meta-diversity search in self-organizing systems, we believe that adaptive capacity and organization of the discoveries into distinct niches as proposed in IMGEP-HOLMES might play an important role to enable the continuous discovery of new sources of incoming variations. An interesting feature of MC-AURORA is that it introduces a diversity component into the AE training loss, explicitly encouraging the learning of diverse BC spaces.

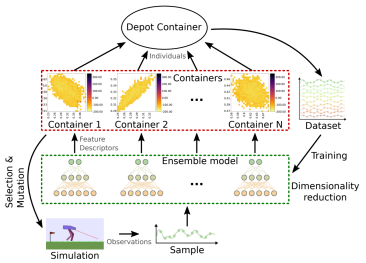


Figure 5.13.: MC-AURORA. Figure is taken from [255].

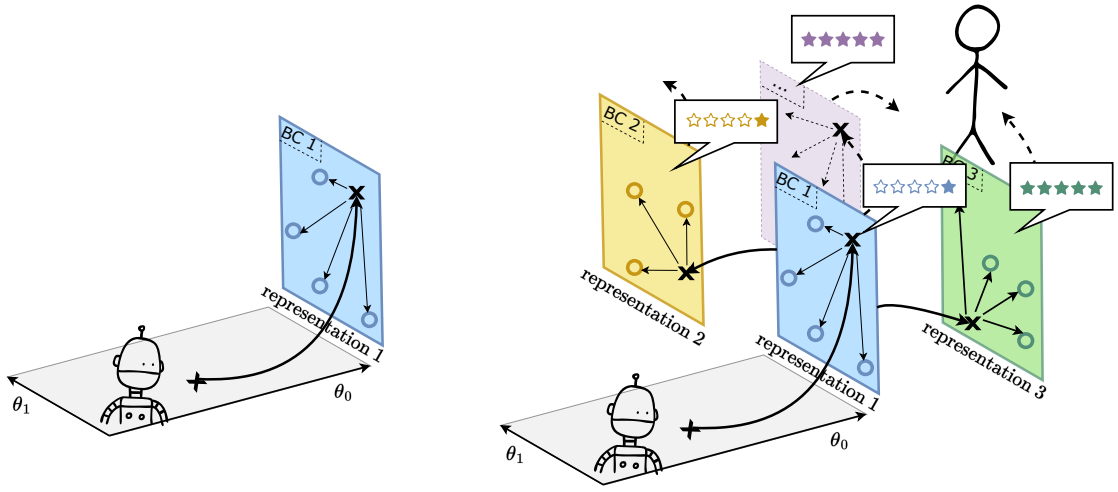
Several avenues for future research aim to enhance the IMGEP-HOLMES approach. First, the HOLMES architecture currently maintains a degree of rigidity in the way it isolates different pattern niches, employing a binary tree with unsupervisedly determined boundaries that remain fixed throughout exploration. More flexible approaches, *e.g.* leveraging human guidance, could be envisaged. Second, it employs the same architecture for all modules, with all VAEs being trained to encode features of the final Lenia patterns at t=200. It would be interesting to explore the simultaneous use of different architectures and training losses, especially those encoding dynamic characteristics of the discovered behaviors, to further increase diversity of the representations.

Lastly, a central limitation of the work presented in this chapter, which we deem central to open-endedness and meta-diversity search, is that the successful achievement of new *types* of diversity can only be subjectively evaluated. The definition of novelties and novel types of novelty depends on the interests and motivations of the final observer. In the next chapter, we propose to augment the objective of meta-diversity search as seeking to leverage the diverse BC spaces to efficiently adapt to a new *interesting* type of diversity, which corresponds to the initially unknown preferences of an external end-user, and expressed through simple and sparse feedback.

6. Human in the Loop to Guide Exploration

What is the aim of this chapter? In the first part of this manuscript, we presented the problem of automated *diversity-driven* discovery in self-organizing systems (Chapter 3) and suggested that *online* learning (Chapter 4) of *diverse* reprepresentation spaces (Chapter 5) were key ingredients to enable the discovery of diverse types of diversity. In this chapter, we go a natural step further and argue that what constitutes an "interesting" type of diversity strongly depends on the final end-user and its motives. We therefore propose to transition from an entirely automated discovery process to a more assisted approach, where external guidance is used to steer the search toward directions aligning with the end-user preferences. To that end, we propose to combine the *divergent* meta-diversity search approach, focusing on continuous exploration of new diversity sources, with external source of human *guidance*, representing a dynamic blend of automated and human-driven exploration.

How is this chapter organized? In Section 6.1, we discuss the limitations of purely divergent diversity search for achieving discovery of both novel and *valuable* outcomes in self-organizing systems. In Section 6.2, we propose to introduce *human guidance* in the IMGEP exploration process. After discussing various forms that such guidance could take, we introduce one specific framework where a curious AI "discovery assistant" seeks to adapt the meta-diversity search process to the preferences of an external end-user. To that end, Section 6.3 introduces a variant of IMGEP-HOLMES leveraging its modular architecture to integrate human preferences through sparse feedback. Finally, in Section 6.4 we consider two end-user models respectively interested in different diversities, demonstrating how standard IMGEP biases discoveries, while IMGEP-HOLMES can adapt to these user preferences with minimal feedback.



(a) Standard Diversity Search

(b) The AI Discovery Assistant

Figure 6.1.: The AI "discovery assistant" can leverage its diverse BC spaces to specialize efficiently towards a particular type of diversity, corresponding to the initially unknown preferences of an end-user, and expressed through simple and sparse feedback.

6.1. Motivation: Limits of Purely Divergent Diversity Search

In the preceding chapter, we introduced the concept of *meta-diversity*, which represents an extension of the conventional notion of *diversity* in AI-guided exploration. We discussed its connection with human concepts of *exploratory* and *transformational* creativity [233]. Yet *creativity*, a central feature of human intelligence and engine for innovation and scientific discovery, involves more than just generating *new* goals or acquiring fresh skills. True creativity hinges on formulating objectives that are simultaneously *novel* (surprising) and *valuable* (intriguing, useful, aesthetically pleasing, etc.). The subjective assessment of the *value* of these goals is closely linked to the *cultural context* as what we find interesting, or not, is significantly shaped by the norms and structures accepted within our culture [233]. Similarly, the process of scientific discovery involves not only formulating new experimental designs but also evaluating their significance and relevance within the scientific community. What is perceived as groundbreaking discovery in science will depend of various political factors and societal needs, all of which contribute to the dynamic cultural context of science. Therefore, just as creativity encompasses the evaluation of novelty and value within a cultural context, scientific discovery relies on the assessment of new ideas and findings within the context of existing scientific knowledge and the broader societal landscape.

The majority of contemporary "creative" computers or AI programs tend to be exploratory. They navigate predefined conceptual search spaces crafted by engineers or experts, with the assumption that novel discoveries within these spaces will be of interest or value to end-users (the conventional approach to diversity search). However, when it comes to aiding the discovery of self-organizing systems, we have seen how furnishing a program with a representation of an "interesting" conceptual space requires substantial domain expertise from the engineer. Even then, it is improbable that this space will fully align with the interests of end-users (see Section 5.1). To overcome this limitation, we have proposed a shift towards what we have termed *meta-diversity search*. This approach can be likened to a form of "transformational" creativity, as the dimensions of the conceptual space (referred to as BC space) are progressively updated and dynamically expanded throughout exploration. However, a fundamental limitation of purely divergent meta-diversity search, and likely a significant reason why most current AI-driven "creative" search primarily emphasize exploration rather than transformation of the search space, is that as the space is arbitrarily transformed, the resulting structures may lack interest or value. Although we believe that the *divergent* aspect of meta-diversity search is pivotal for uncovering unanticipated pathways of innovation (Figure 5.8) and for generating diverse potentially valuable representations (Figure 5.6), one would naturally except the search strategy to be able to quickly *adapt* to the preferences of an external end-user. Adapting the search is particularly crucial in the context of automated discovery where experiments often demand significant investments in terms of time and resources, and where one does not want to waste resources on uninteresting or irrelevant outcomes.

6.2. Problem reformulation: The "AI Discovery Assistant" framework

To address these limitations, a natural extension is to integrate *human guidance* into the discovery process, shifting from fully "automated" discovery to what we might instead call "assisted" scientific discovery. As proposed in Chapter 2, in that view one could consider the AI agent as a *developmental learner* aiming to make novel discoveries in order to augment its understanding of the natural world; and the human end-user as the equivalent of *cultural processes* in the sense of shaping the AI developmental trajectories with guidance and instructions that are *subjective* and *adaptive* to the current value system (Figure 2.21). Within that view, the discoveries that will be considered as "valuable" could vary from end-user to end-user, and from time to time.

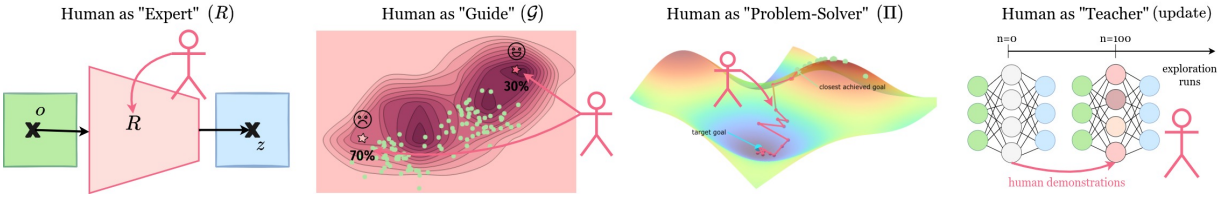


Figure 6.2.: Four roles that the human could play in the IMGEP exploration loop: the *expert* for outcome characterization (R), the *guide* for goal selection ($\mathcal{G}$), the *problem-solver* for goal achievement (Π) and the *teacher* for learning (update).

Human-AI collaborations is a broad and active area of research in many scientific domains, and they are obviously many ways in which human feedback could be integrated into the AI decision-making process [207]. In the IMGEP computational framework, a natural extension would be to have the human influencing or even replacing one of the IMGEP main components: the goal space representation (R), the goal sampling strategy($\mathcal{G}$), goal-achievement strategy (Π) or learning strategy (model updates). As illustrated in Figure 6.2, this leads to four possible "roles" for the human. The first role is the *expert* role: part of the outcome characterization (R), the human could participate in describing the system observations. For instance, one could imagine asking the human to provide labels or to score (predefined or novel) aspects of the experimental outcomes. A second role would be the one of the *guide*: part of the goal generation process, the human could influence the IMGEP goal selection mechanism by expressing preferences in certain regions of the goal space $\mathcal{T}$. One could for instance imagine the human browsing some representative discoveries in the goal space and select the preferred regions, such that the IMGEP could prioritize goal sampling in those areas. A third role for the human would be the one of the *problem-solver*: here the human would directly be in charge of proposing experiment parameters θ in order to achieve (or not) a target goal g. Obviously, if the human is to take that role, it should remain sparse, without which the "automated" aspect of discovery would completely disappear. Here, one could imagine a human scientist performing some experiments on its own (prior or parallel to the AI exploration) and providing the collected observations to assist the AI goal-conditioned decision making. Finally the human could play the role of the *teacher* as part of the internal model updates. For instance, the human could provide a set of (labeled) demonstrations to help the AI's internal models learning, as

was proposed by Nguyen and Oudeyer [164] to demonstrate motor skills to an intrinsically-motivated robot learner.

In this thesis, we focus on the human as *guide* scenario which is believe the less demanding to the human, as it does not require some sort of expertise to characterize the system outcomes nor to collect experimental data, and still provides valuable information to shape the AI discovery process. In fact, they are two levels of guidance with which we believe the human can play an important role. The first one, that we focus on in this chapter, is what we call *preference-based* guidance. Here, the AI is seen as a *discovery assistant*: it cannot predict what a future (unknown) end-user will find interesting in Lenia (as there is so much to be done with it), but it aims to find diverse discoveries enabling, on the one hand, the human end-user to better express its preferences, and on the other hand, the agent to quickly adapt its discoveries as soon as such preferences are expressed. In the context of meta-diversity search for instance, a successful discovery assistant agent is one which can leverage its diverse BCs to specialize efficiently towards a new *interesting* type of diversity, corresponding to the initially unknown preferences of an end-user, and expressed through simple and very sparse feedback (Figure 6.1).

The second type of human guidance, which we begin to explore in the applicative Chapter 7 and Chapter 8 (in the second part of this manuscript), is what we call *environment-based* guidance. Here, the human directly intervenes in the environment to provide cues (or constraints) in order to assist (or obstruct) self-organization toward the goal in the system[1]. While we believe the generation of environmental elements could play a crucial role in assisting self-organization in the system, choosing how to generate these elements - automatically or interactively - remains an open question[2].

Whether the human is integrated as an expert, guide, problem-solver or teacher within the exploration process they are two important interface components for human-in-the-loop systems: the choice of a *visualization* to show the human end-user the important (and interpretable) AI-collected information that it needs to provide feedback; and the choice of a *feedback form* to collect the human-provided feedback in an exploitable format for the AI. In this chapter, we present initial experiments that were made with *simulated* end-users, sidestepping the technical challenges of integrating actual human users directly into the loop, though hopefully experiments will real humans could be done in future work. [3].

1: For instance, in Chapter 7 we will see how humans can integrate various elements in the environment such as obstacles, food or attractor objects - all defined within the CA paradigm

2: In Chapter 8, we propose that classical experiments from behavioral sciences, originally testing various navigation competencies that living agents exhibit, can serve as a source of inspiration for generating these cues/constraints

3: In Chapter 10, we will present the ADTOOL software package which aims, among other things, to facilitate human interaction with the exploration process

6.3. Proposed implementation: preference-guided IMGEP-HOLMES

In the previous chapter, we introduced an algorithm wherein an intrinsically motivated goal exploration process (IMGEP) was augmented with a modular architecture, HOLMES, which dynamically expands to accommodate novel environmental variations. This chapter delves into the notion that an effective "discovery assistant" should not merely strive to represent these new variations continually; it should also efficiently specialize in a form of diversity that aligns with an end-user's initially unknown preferences, expressed through simple, sparse feedback. To this

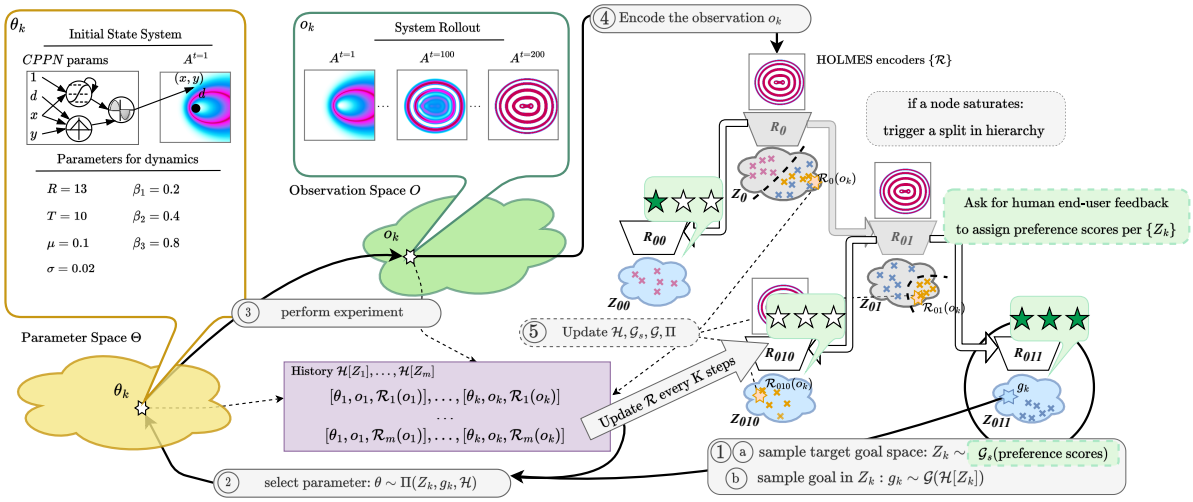


Figure 6.3.: Overview of the *preference-guided* IMGEP-HOLMES approach proposed in Etcheverry et al. [•8]. The main novelties with respect to the non-guided IMGEP-HOLMES approach presented in the previous chapter (Figure 5.4) are shown in light green. They include the request for human end-user (sparse) feedback throughout exploration to assign preference scores to the different BC spaces (leaf nodes) of the hierarchy, as well as the integration of these scores in the goal space sampling strategy $\mathcal{G}_s$.

end, we propose an extension of the IMGEP-HOLMES algorithm, termed *preference-guided* IMGEP-HOLMES, outlined in Figure 1. Novelties with respect to the non-guided IMGEP-HOLMES approach are highlighted in light green. They include two novel, simple ingredients.

The first ingredient is on the human end-user's side and enables to provide feedback throughout exploration. Various decisions surround this aspect, including when to give feedback, where within the discoveries (with respect to the HOLMES hierarchy), and what type of feedback to request. For this work, we simply propose to "pause" exploration each time a split occurs in the hierarchy. As a reminder, a split occurs when a node in the hierarchy saturates, resulting in the instantiation of two child representations. Each child inherits its niche of patterns from the parent, learns to characterize specialized features, and strives to discover more representative patterns along those features. It seems natural, therefore, to ask for human feedback each time new children are created. The human, by visually examining a few representative images per module, can select the leaf nodes with their preferred discoveries and assign preference scores (here scores between 0 and 1).

The second ingredient is on the AI agent's side and enables to integrate feedback into the exploration decision-making process. Here, we focus on a straightforward approach within the IMGEP goal-sampling strategy. As a reminder, this strategy consists of two steps in IMGEP-HOLMES: (a) selecting a target BC space from all current leaf nodes and (b) selecting a target goal within that BC space. Since the human is asked to provide preference scores for all current leaf BC spaces each time a new BC space is created, we integrate these scores as follows: (a) we employ softmax sampling based on the assigned score probabilities to select the target BC space, and (b) we use uniform sampling within the selected BC space, as was done previously within the hyperrectangle envelope of currently reached goals. Intuitively, this corresponds to prioritizing goal sampling in the BC spaces of interest, allocating them more experimental budget.

Related work The field of Interactive Evolutionary Computation (IEC) integrates human feedback with evolutionary algorithms to explore complex pattern spaces, especially in aesthetic domains like art and design [256]. Early examples include Sims' work on evolving cellular automata rules using genetic algorithms and human judgment, which enabled the discovery of interesting emergent behaviors [257]. Another interesting work is the one proposed by Langdon [258] which used interactive evolutionary algorithm to evolve snowflake patterns via L-system grammars. The approach leveraged an internet-based platform that engaged a multitude of end-users, resulting in its designation as an "open-ended" evolution system [258]. Another well-known application of IEC include the PicBreeder web platform [259, 260], where users can evolve CPPN-generated images by selecting ones that appeal to them to produce a new generation. What sets PicBreeder apart is its creation of an online community that not only facilitates users' generation of their own images but also permits the ongoing evolution of images initiated by others. This collaborative and ongoing process continually increases image complexity, which hence qualifies even more as a form of open-ended system [261]. It's worth noting that in the context of IMGEP-HOLMES, our feedback mechanism differs from traditional IEC works. We require feedback only at each split of the hierarchy, in contrast to systems where users provide feedback at every generation, individually selecting interesting patterns for the subsequent generation. In our work, each experiment is independent but it will be interesting for future work to enable the reuse of discoveries via collaborative platforms. While Picbreeder was applied to static image generators, one could imagine similar interfaces being deployed to guide exploration in developmental self-organizing systems like Lenia[4].

4: This is actually a typical use-case that we aim to facilitate with the developed ADTOOL software presented in Chapter 10

6.4. Experiments

In Lenia and cellular automata in general, they are two categories of patterns have been extensively studied, and that we refer to as Spatially-Localized Patterns (SLP) and Turing-Like Patterns (TLP) (Figure 3.14). For our experiments, we assume the presence of two end-user models, each with an interest in discovering a diversity of either SLP or TLP patterns. To guide this process, we employ the classifiers[5] for SLP and TLP proposed in Chapter 4 to simulate human end-users. Whenever a split occurs in the HOLMES hierarchy, we use these classifiers to score the various leaf nodes based on the number of SLP (or TLP) patterns within each node at the time of the split. The resulting scores then influence the goal sampling strategy of the preference-guided IMGEP-HOLMES algorithm, as described in Section 6.3. Please note that this approach provides a practical mean for simulating an external user, but has limitations as a higher number of instances of a pattern category does not necessarily equate to "more interesting" instances. The algorithms guided using this procedure are respectively denoted as **IMGEP-HOLMES (SLP)** and **IMGEP-HOLMES (TLP)**.

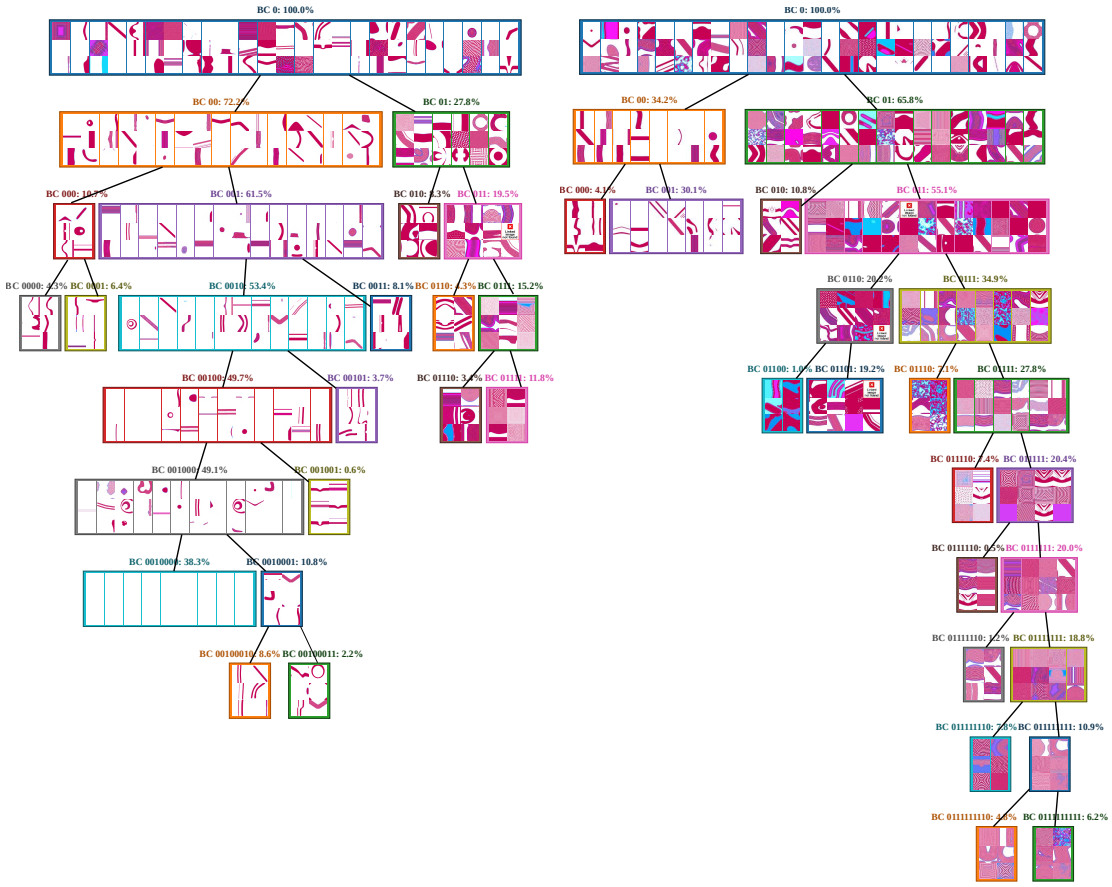
5: Please refer to Appendix Section B.2.2 for a full description of the classifiers

We are primarily interested to test whether our *discovery assistant* search can efficiently produce a diversity of "interesting" patterns, but once again this is far from trivial to evaluate. To that end, in Subsection 6.4.1,

we initially present a qualitative assessment of the identified patterns, similar to the approach in Chapter 5. Subsequently, in Subsection 6.4.2, we introduce a *proxy* evaluation of the discovered diversity. This proxy metric is designed to closely align with the evaluator's judgment regarding what constitutes a diverse set of SLP and TLP patterns.

6.4.1. Qualitative Evaluation

Figure 6.4a and Figure 6.4b show discovered patterns when IMGEP-HOLMES is guided towards SLPs or TLPs, respectively. We observe that the (simulated) user guidance is able to significantly affect both the structure of the hierarchy and the types of discovered patterns. When guided towards SLPs, as shown in Figure 6.4a, most of the discovered patterns are SLPs (TLPs are concentrated in the leafs BC 01110 and BC 01111 with approximately 15% of the discovered patterns). On the contrary, when guided towards TLPs, as shown in Figure 6.4b, most of the discovered patterns are TLPs (SLPs are concentrated in the leafs BC 000 and BC 001 with approximately 34% of the discovered patterns). As a consequence of this bias toward either SLPs or TLPs, we observe that HOLMES has created more branches in the direction of the desired patterns - either SLPs on the left part of the tree, or TLPs on the right - in order to enrich their corresponding representations.



(a) IMGEP-HOLMES (SLP) **(b)** IMGEP-HOLMES (TLP)

Figure 6.4.: Examples of patterns discovered by IMGEP-HOLMES within the learned tree hierarchy, when guided towards SLPs (a) or TLPs (b) through simulated user feedback as described in Section 6.3. The figures are generated with the same procedure than Figure 5.8.

6.4.2. Quantitative Evaluation

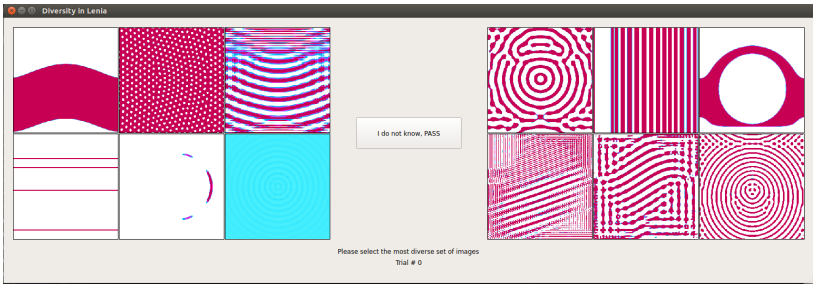


Figure 6.5.: Interface used for selecting of the best proxy-BC analytic space that correlates with human judgement of what represents a *diversity of SLP* and a *diversity of TLP*.

Selection of a *proxy*-BC for evaluation of SLP and TLP diversity For evaluation, an experiment with a human evaluator has been conducted for selecting the BC space, among the 5 proposed in Section 5.1, that correlates the most with the evaluator judgement of what represents a diversity of SLP and a diversity of TLP. The experiment consisted in repeatedly showing the human with two sets of patterns (as shown in Figure 6.5) and asking the human to click on the set that he considers is the more *diverse*, according to its intuitive notion of diversity. If the human cannot choose between the two sets, he can click on the "pass" button. In background, the procedure to generate the sets is the following:

1. Randomly select a (BC, category) pair, where BC ∈ {Spectrum-Fourier, Elliptical-Fourier, Lenia-Statistics, BetaVAE, Patch-BetaVAE} and category ∈ {SLP,TLP}.
2. Randomly draw 750 candidate sets of 6 images among a database of 7500 patterns of the current category[6].
3. Select the most *similar* and the most *dissimilar* (*i.e.* diverse) sets among those 750 sets. The (dis-)similarity of a set is measured as a function of the distances between pairs of images in the current BC space[7].
4. The sets are displayed to the human in random presentation order.

One human evaluator participated in this experiment and performed a total of 500 clicks, i.e., 50 evaluations for each (BC, category) pair. The agreement score was 1 if the human selected the set considered diverse by the BC, 0 if it selected the other set and 0.5 if the human chose the "pass" option. Table 6.1 reports the mean and standard deviation of the agreement scores for each (BC, category) pair. Notably, the human evaluator identified $\text{BC}_{\text{Elliptical-Fourier}}$ as the most suitable proxy space for evaluating the diversity of SLP patterns, achieving a 98% agreement score. For assessing the diversity of TLP patterns, $\text{BC}_{\text{Lenia-Statistics}}$ emerged as the top choice, with a 92% agreement score.

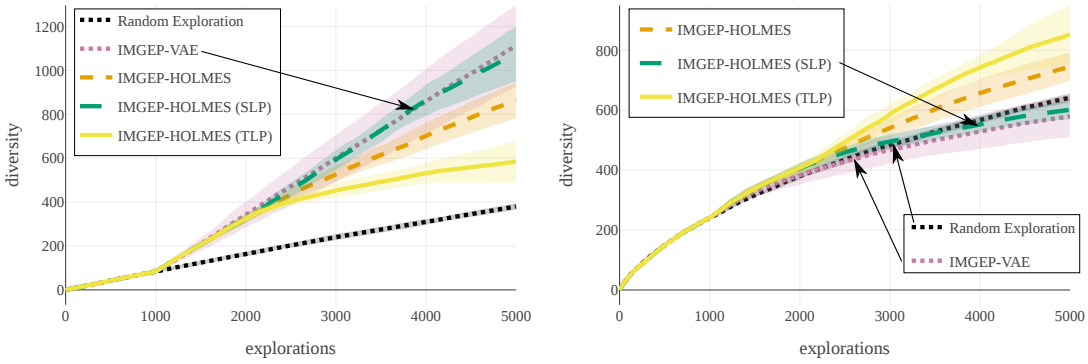
6: As explained in Appendix, a database of 7500 SLP patterns and 7500 TLP patterns was gathered from various exploration variants conducted in Lenia

7: Distance function is provided in Appendix Subsection C.1.2

	Spectrum-Fourier	Elliptical-Fourier	Lenia-Statistics	BetaVAE	Patch-BetaVAE
SLP	0.5 ± 0.18	**0.98 ± 0.04**	0.59 ± 0.12	0.1 ± 0.06	0.89 ± 0.08
TLP	0.2 ± 0.13	0.47 ± 0.1	**0.92 ± 0.07**	0.75 ± 0.08	0.38 ± 0.08

Table 6.1.: Human-evaluator agreement scores (mean ± std). Best scores are shown in bold. The agreement score is significant at level $\alpha = 5\%$ if it is above $0.64 = 0.5 + 1.96 \times \sqrt{\frac{0.25}{50}}$.

Results Non-guided IMGEP-VAE and IMGEP-HOLMES variants used in the previous chapter (see Section 5.4) are compared with the guided IMGEP-HOLMES variants. The results are also compared to Random



(a) Diversity of SLP in $\text{BC}_{\text{Elliptical-Fourier}}$

(b) Diversity of TLP in $\text{BC}_{\text{Lenia-Statistics}}$

Figure 6.6.: Diversity discovered by the algorithms during exploration. The discovered patterns classified as SLP in (a) (resp TLP in (b)) are projected in $\text{BC}_{\text{Elliptical-Fourier}}$ space in (a) (resp $\text{BC}_{\text{Lenia-Statistics}}$ in (b)), where the binning-based measure is used. Mean and std-deviation shaded area curves are depicted.

Exploration baseline (where parameters θ are randomly sampled for the 5000 explorations runs) which serves as reference for the *default* diversity found in Lenia (and by all algorithms during the first 1000 explorations). The diversity discovered by the different algorithms, measured within the selected analytic BC spaces, is shown in Figure 6.6. In line with the results of Chapter 4, we see that the reconstruction bias of the monolithic VAE leads IMGEP-VAE to find a high diversity of SLPs but leads to poor diversity of TLPs (lower than random exploration). On the other hand, when non-guided, IMGEP-HOLMES finds a higher diversity than Random Exploration both for SLPs and TLPs. Interestingly, when guided, IMGEP-HOLMES can even further increase the diversity in the category of interest which confirms the qualitative insights from Figure 6.4.

Additional baselines In Appendix Section C.7, we consider additional IMGEP baselines with unsupervised learned (monolithic) BC spaces. Baselines differ in the encoder architecture and training strategy, including several variants of variational auto-encoders as well as contrastive approaches[8]. The findings indicate that all goal space representations learned through monolithic architectures tend to suffer from the "plasticity-stability" dilemma. This dilemma, well known in continual learning [262], also hampers the potential for effectively guiding the discovery process (see Figure C.6). Indeed, a lack of plasticity was observed for all VAE variants, indicating that not only do the learned features fail to adapt to new sources of environmental variation, but they also pose challenges for introducing human guidance into the process, as these learned features are unlikely to adaptively align with human preferences. On the other hand, a complete lack of stability was observed in contrastive approaches. In this case, the representation is highly sensitive to the patterns discovered during training, leading to drastic shifts in learned features from one training stage to another. This is also something that it is not desirable as a human end-user may be more interested, at least for a time, in exploring a given space rather than in transforming it in unpredictable ways. Furthermore, the absence of stability makes it challenging to control the types of features learned, preventing effective alignment with human preferences. In contrast, the HOLMES architecture seems to provide an appropriate "plasticity-stability" tradeoff enabling flexible representations and efficient guidance.

8: Contrastive approaches, detailed in Section C.7, basically encourage the representation R to maximize similarity in encodings between differently augmented versions of the same observation

6.5. Discussion and Future Work

In this chapter, we explored ways to incorporate *human guidance* into the standard IMGEP framework. We introduced a straightforward adaptation of the IMGEP-HOLMES algorithm, utilizing HOLMES modular representations to incorporate human preferences and guide exploration with minimal feedback. Considering two end-user models respectively interested in two types of diversities (diverse spatially localized and diverse turing-like patterns), we showed that a monolithic IMGEP will make discoveries that are strongly uneven in relation to these user preferences, while IMGEP-HOLMES is better suited to escape this bias by learning divergent feature representations. We also show how the guided IMGEP-HOLMES variant can efficiently shape the search toward those two types of *interesting* diversities with little amount of user feedback.

Once again, we believe that similar approaches could be deployed in other types of systems, *e.g.* systems encountered in physics or chemistry for which it is difficult to computationally express *a priori* what a human end-user might find interesting or not. Falk et al. [139], which recently transposed the IMGEP-VAE approach presented in Chapter 4 to explore behaviors of so-called Kuramoto models[9], also proposed what they called a "human-aligned curiosity search" which was directly inspired from the IMGEP-HOLMES algorithm (Figure 6.7-A). Interestingly this approach enabled the discovery of several behaviors with non-trivial multi-population dynamics, in accordance with the human intuition curiosity search was aligned with (Figure 6.7-D).

9: Kuramoto models [227, 228] are widely-used models to describe synchronization phenomena in physics and chemistry

A major limitation of the experimental results presented in this chapter is that we rely on *simulated* end-user feedback. In Chapter 10, we present the ADTOOL software package developed in collaboration with engineers at the FLOWERS team which, among other things, aims to facilitate flexible interaction with real end-users throughout the exploration process.

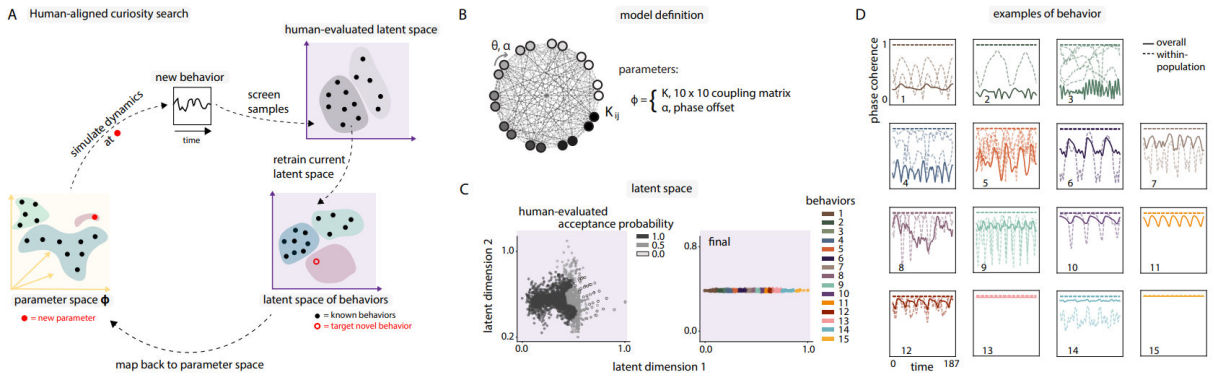


Figure 6.7.: Human-aligned curiosity search applies to a 100-dimensional Kuramoto model discovers various chaotic multi-population behaviors (Falk et al. [139]). (A) The proposed approach, called human-aligned curiosity search, proceeds as follows: 1) an initial VAE latent space is constructed and explored, 2) at some point the latent space is frozen and subsequent behaviors are redirected through the initial, frozen latent space, where samples are accepted or rejected based on probabilities assigned to different parts of latent space by a human observer, 3) Following screening through this human-evaluated latent space, sampling proceeds with a separate newly-instantiated VAE for the new, human-aligned latent space. (B) A 10-population Kuramoto model with a total of N = 100 oscillators. (C) (left) VAE latent space following initial curiosity search without human alignment. Latent space clusters identified in initial search are human-evaluated for interest and assigned acceptance probabilities. (right) Final latent space at the end of human-aligned curiosity search. (D) Phase coherence examples from each of the states identified through latent space clustering. Discovered behaviors include breathing chimeras (7, 14), nearly synchronized (13, 15), and chiral phases (11). Figure is adapted from [139].

Use cases of the curious discovery assistant

Exploring the Self-organization of Robust forms of Agency in continuous CA models

7.

What is the aim of this chapter? As a first applicative use case, this chapter investigates the following scientific question: can we find environmental rules in the Lenia system that can lead to the self-organization of *robust* forms of *sensorimotor agency*? We propose a set of methodological tools as well as an extensive analysis of the discoveries, and discuss their various implications for research in AI and biology.

How is this chapter organized? In Section 7.1, we provide some background on the *mechanistic* and *enactivist* views on cognition, and motivate the search for "robust sensorimotor agents" in continuous CA models. Then, in Section 7.2, we present our methodological approach to tackle the *automated discovery* and *systematic characterization* of these "robust sensorimotor agents" in Lenia. This combines the use of tailored curiosity-driven exploration approaches together with a battery of quantitative and qualitative tests to assess the generalization capabilities of the discovered agents. In Section 7.3, we show how the proposed search method progressively leads to the emergence of individuality, locomotion, and sensorimotor abilities, while these behaviors are very hard to obtain with random or handmade search methods. Moving to Section 7.4, we demonstrate that the discovered sensorimotor agents not only exhibit individuality and locomotion, but are also capable of strong adaptivity and generalization to out-of-distribution perturbations, reminiscent of properties observed in biological organisms. Finally, in Section 7.5, we present a more recent contribution called Flow Lenia, which is extension of the original Lenia system. We discuss its several benefits for studying artificial lifeforms, defining physical constraints in the environment, and promoting the emergence of diversity and evolutionary activity *within* the CA dynamics, which are all central challenges in artificial life.

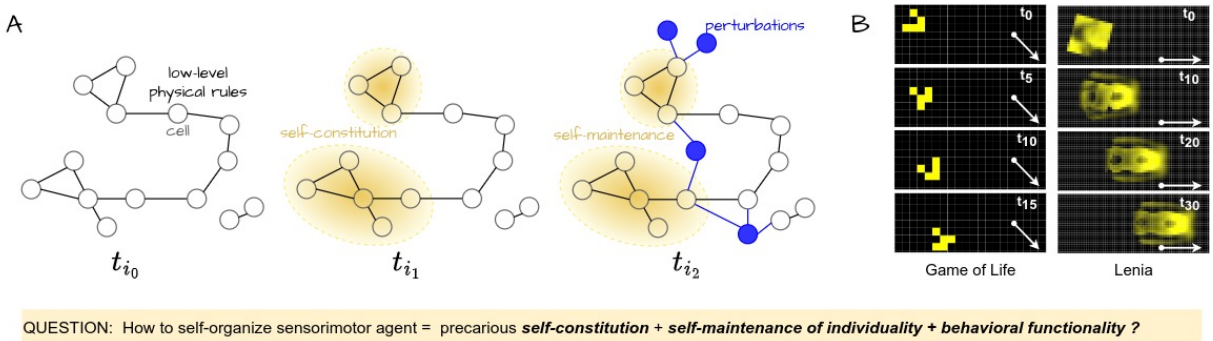


Figure 7.1.: Overview of the scientific question. (A) The enactivist framework: (t_{i_0}) In the beginning there is only an environment made of low-level elements (cells) and physical laws (local rules). There is no prior notion of agency, no body, no sensor. (t_{i_1}) Agents can come to existence through the coordination of the low-level elements (self-constitution of individuality). (t_{i_2}) To maintain their integrity, agents must sense and react to perturbations using only local update rules (self-maintenance of individuality). (B) In cellular automata models like the Game of Life and a more complex continuous extension called Lenia, it was shown that it is possible to self-organize so-called *gliders i.e.* spatially-localized patterns with directional movement (timesteps and arrows for direction are displayed). (Question) In this work, following the enactivist modeling framework, we try to answer the following scientific question: is it possible to find environments in which a subpart could self-organize and be called a "sensorimotor agent"? This would require the existence and emergence of gliders-like structures that not only self-constitute and show motility, but are robust to external perturbations necessitating to develop some form of sensorimotor apparatus enabling them to make "decision" and "sense" at the macro scale through local interactions only.

7.1. Introduction

Understanding what has led to the emergence of life, cognition and natural agency as we observe in living organisms has been a central debate across many sectors of life sciences. Biological organisms are made of collections of cells that follow low-level distributed rules and yet they constitute a coherent unitary whole, displaying strong *individuality*[1] and *self-maintenance*[2] in their environment, what was described to be an *autopoietic system*[3]. While a central concept in theoretical biology, the characterization of an autopoietic system and the understanding of the processes underlying its self-organization remain a live issue. Further demystifying how those processes do not just give rise to organic individuation[4] but also to sensorimotor[5] and even intersubjective[6] agency, is at the center of the debate [265]. In fact, recent advances in biology and basal cognition suggest that many autopoietic systems that we find in nature, including plants and brainless animals, are robust sensorimotor agents capable of using a body for sensing opportunities, computing decisions and acting in their environment [138]. The pragmatic and complementary question to the debate, central in ALife and AI research, is: can we engineer the necessary ingredients leading to the emergence of functional forms of life and sensorimotor agency in an artificial substrata in which initially there is literally no body (and thus no sensing, no acting, no agent)? Although there is already a large body of work that proposes to study the emergence of life and cognition in agents-as-they-could-be, it is generally done either by jumping over the biological processes that enable organisms to survive (the *mechanistic view*, as *e.g.* in reinforcement learning which considers pre-existing agents with predefined sensors and actuators) or inconclusive so-far in showcasing higher-level forms of sensorimotor agency (the *enactivist view*, as in *e.g.* artificial chemistry which studies how forms of agency can emerge from low-level chemical reactions). Herein, after giving some background on the mechanistic and enactivist views on cognition and on their current limitations, we suggest that modern tools from machine learning (ML) can help us bridge the gap between those two views. Whereas ML tools have mainly been deployed within the mechanistic framework, we show that they can efficiently assist the discovery of environments that self-organize relatively-advanced forms of sensorimotor agency whose existence and understanding is fundamental within the enactivist framework for supporting theories about the origins of life and cognition.

1: ability of a self-organizing structure (subpart of the environment) to preserve and propagate some spatiotemporal unity [263], making it a distinguishable coherent entity in the domain in which it exists

2: ability of a self-organizing structure to modify its interactions with the rest of the environment for maintaining its integrity

3: Introduced by Maturana and Varela [264], the concept of autopoiesis refers to a system capable of producing and maintaining itself by creating its own parts

4: regulation at the metabolic, transcriptional and morphological level to maintain organic integrity [265]

5: active engagement in loops of actions and perceptions in the external environment [265]

6: active engagement in communicative interactions and structural coupling with other agents [265]

(a) Demo and Videos **(b)** Paper Codebase **(c)** Notebook **(d)** Blogpost

Figure 7.2.: (a) Sensorimotor Lenia companion website with interactive web-demo and several videos, referred as **movie SX** in the chapter. (b) Notebook with Sensorimotor Lenia experiments and demo implementation. (c) Seensorimotor Lenia blogpost.

In the mechanistic view, one assumes the existence of agents that have well defined physical body and information processing brain allowing them to interact with the rest of the environment through predefined sensors and actuators (Figure 7.3). Robots for instance are referred as embodied agents: their individuality is clear, as they can easily be distinguished from the rest of the environment, and their self-maintenance is often not a problem, as their body does not change over time except for rare cases of real world or artificially-induced degradation. Hence it is not questioned what makes an agent an agent or even what makes a body a body [265]. Rather, a more central question is to understand how higher-level cognitive processes and sensorimotor adaptivity can arise in the agent through its interactions with the environment. A common methodology is the generation of a distribution of environments (tasks and rewards) and the use of learning approaches, such as deep reinforcement learning, to train the agent's brain to master and generalize those tasks. Within that framework, it was shown that it is possible to engineer agents capable of repertoires of advanced sensorimotor skills such as precise locomotion [266], object manipulation [267], tool use [268] and even capable of adapting the learned behaviors to unseen environmental conditions [167]. Interestingly, they show that the use of curriculum learning[7] is crucial to generate generally capable agents. However, the clear body/brain/environment distinction of the mechanistic framework bears little resemblance with the way information seems to be processed by biological systems. Notably it goes against the concept of morphological computation [269], which argues that all physical processes of the body, not only electrical circuitry in the brain but also morphological growth and body reconfiguration, are integral parts of cognition and can achieve advanced forms of computation.

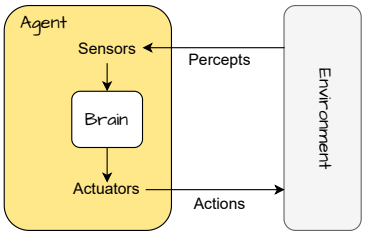


Figure 7.3.: Mechanistic view. One assumes the pre-existence of a body with sensors and actuators, though which an agent can interact with its environment.

7: family of mechanisms that adapt the distribution of training environments to the learner capabilities

The enactive view on embodiment however is rooted in the bottom-up organizational principles of living organisms in the biological world. The modeling framework typically uses tools from dynamical and complex systems theory where an artificial system (the environment) is made of low-level elements of matter (called atoms, molecules or cells) described by their inner states (*e.g.* energy level) and locally interacting via physics-like rules (flow of matter and energy within the elements) (Figure 7.1A-t_{i_0}). There is no predefined notion of agent embodiment, instead it is considered that the body of the agent must come to existence through the coordination of the low-level elements (Figure 7.1A-t_{i_1}) and must operate under environmental perturbations and precarious conditions[8] (Figure 7.1A-t_{i_2}). Hence, the self-constitution and self-maintenance of individuality are prior conditions for any agency to emerge as it determines the agent's own existence and survival [265]. This shifts the problem of "building agents as-they-could-be"to a problem of engineering second-order emergence [270]: how to design environments that can give rise to self-constituting agents that, coupled with the rest of environment, give rise to sensorimotor behaviors? Previous work has shown that the realisation of autopoietic entities in computational media is possible [222, 271–273]. For instance, fully emergent structures showing spatial localization and movement have been discovered, such as the well-known gliders in the game of life up to richer life-like patterns in continuous models of cellular automata (Figure 7.1B). So far however, two major challenges remain poorly addressed in the enactivist literature. First, autopoietic

8: the idea that bodies are constantly subjected to disruptions and breakdowns [265]

structures have so far mainly been discovered by human eye and as the result of time-consuming manual search, limiting their discovery and analysis. While some recent works, based on information theory tools, have proposed quantitative measures of individuality in order to facilitate their identification [263, 274], their algorithmic implementation remains difficult in practice. Second, among the very few works that proposed a deeper analysis of the robustness capabilities of the discovered patterns (based on the enumeration of all possible perturbations that a structure can receive from its immediate environment) [191, 192, 273, 275], findings suggest that glider-like structures typically remain quite fragile to external perturbations such as collision with other patterns [191].

In this work, following the enactivist framework and considering a class of continuous cellular automata called Lenia [39, 40] as our artificial "world", we show that modern tools from machine learning can help addressing the problem of engineering second-order emergence. We propose a method based on curriculum learning, diversity search and gradient descent, enabling to efficiently shape the search process and to successfully navigate the chaotic outcome landscape of the Lenia system. In particular, we use a family of algorithmic processes called intrinsically-motivated goal exploration processes (IMGEP), an efficient form of diversity search algorithm [57]. While mainly deployed in the fields of developmental robotics [58] and developmental AI [172, 173], recent works have shown how IMGEP can also form useful scientific discovery assistants for revealing the range of possible behaviors in unfamiliar systems such as chemical oil-droplet systems [64], physical non-equilibrium systems [139] and models of continuous cellular automata systems as the one considered here [•1, •3]. At the difference of previous papers, we introduce two novel elements: the use of gradient descent for local optimization and the use of stochastic perturbations within the curriculum. With this method, we are able to find environmental rules leading to the emergence of patterns that self-constitute, self-maintain and move forward under various obstacle configurations, *i.e.* autopoietic entities displaying robust forms of sensorimotor agency. We then propose a battery of quantitative and qualitative tests, all formulated within the continuous CA paradigm, to further assess the robustness and generalization capabilities of the discovered self-organized patterns. Interestingly, the agents also show strong robustness to several out-of-distribution perturbations ranging from perturbing the agent structure in various ways not seen during training (including by a collision with another agent) to changing the scale of the agent. Furthermore, when tested in a multi-entity initialization and despite having been trained alone, not only the agents are able to preserve their individuality but they show forms of coordinated interactions (attractiveness and reproduction), which could be interpreted as a form of intersubjective communication [191]. Those results illustrate the achievable generalization capabilities of artificial self-organizing agents, with respect to their mechanistic counterpart, opening interesting avenues for AI. At the same time, they provide interesting models about the way information might be processed by (brainless) biological agents to ensure robust maintenance of sensorimotor functions despite environmental and body perturbations [276].

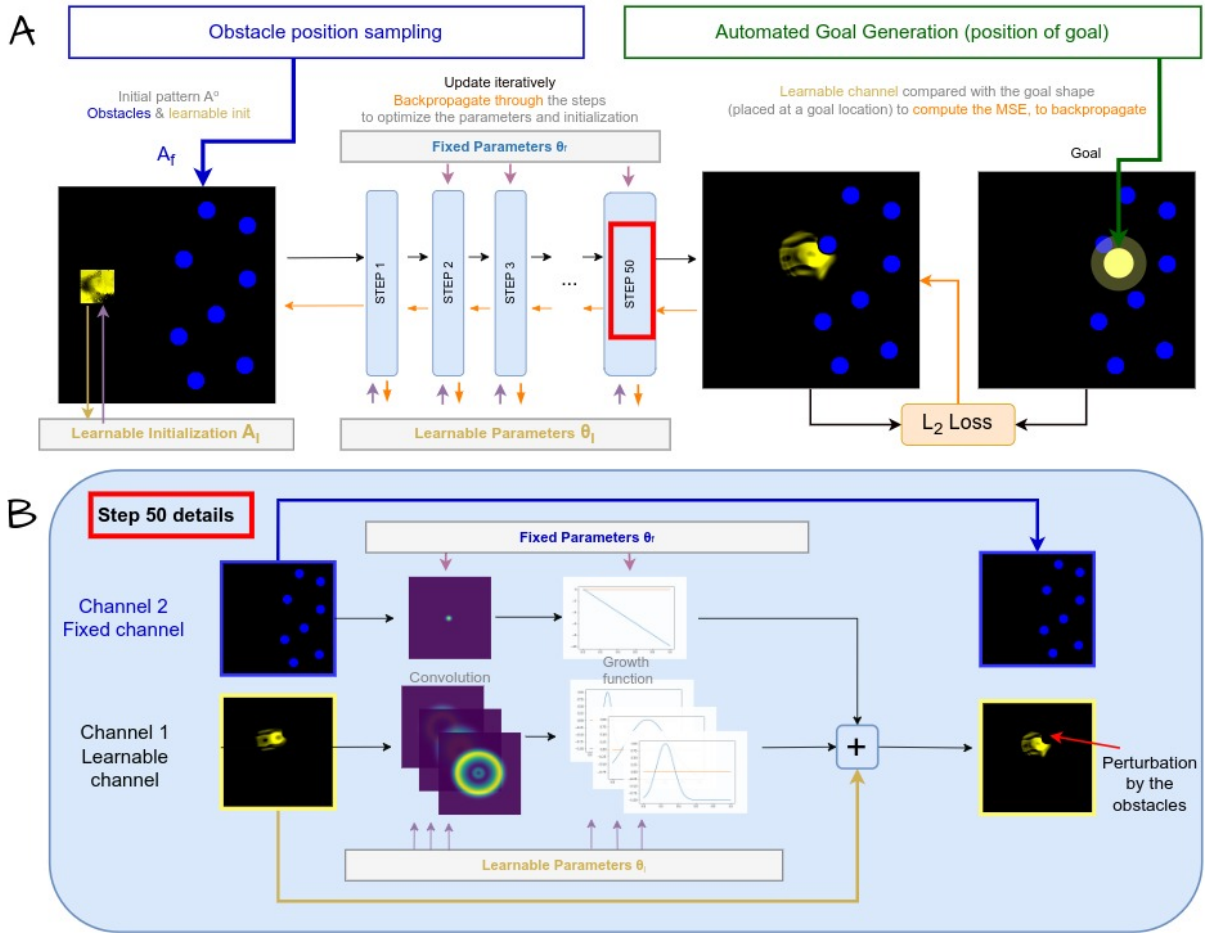


Figure 7.4.: (A) Illustration of one experimental rollout with automated (i) generation of target goal (green), (ii) generation of environmental obstacles (blue) and (iii) optimization of learnable parameters toward goal (backpropagation shown in orange). The initial state is iteratively updated by the parametrized rule, we then compute the goal conditioned loss from the last state of the rollout and propagate gradient across the steps to the learnable parameters and initialization. (B) Detailed view of a step in Lenia with obstacles. A convolution followed by a non-linear growth function is applied on both channels, resulting in a growth update which is added to the current state of the learnable channel. Both the convolution and growth functions are parametrized (see Appendix Subsection D.2.1).

7.2. Study of Sensorimotor Agency in continuous CA models

Cellular automata (CA) are, in their classic form, a grid of "cells" $A = \{a_x\}$ that evolve through time $A^{t=1} \longrightarrow \ldots \longrightarrow A^{t=T}$ via the same local "physics-like" laws. More precisely, the cells sequentially update their state based on the states of their neighbors: $a_x^{t+1} = f(\mathcal{N}(a_x^t))$, where $x \in \mathcal{X}$ is the position of the cell on the grid, a_x is the state of the cell, and $\mathcal{N}(a_x^t)$ is the neighborhood of the cell (including itself). The dynamic of the CA is thus entirely defined by the initialization $A^{t=1}$ (initial state of the cells in the grid) and the update rule f (how a cell updates based on its neighbors).

In this work we use Lenia, a class of continuous CA which is a recently-proposed generalization of Conway's Game of Life [39, 40], as our self-organizing system[9]. Previous works in Lenia have shown that there exist local update rules f, that can lead to the self-organization of long-term stable complex patterns that display interesting life-like behaviors [•3, 39, 40]. Those include forms of individuality (spatially-localized organi-

9: The Lenia system is detail in Section D.2

sation), locomotion (directional movement) and even basic sensorimotor capabilities (change of direction in response to interaction with other patterns in the grid). However, in previous work, self-maintenance of those behaviors in discovered spatially-localised patterns were typically quite fragile to external perturbations (for example collision with other agents **Movie S1**), and properties of robustness and generalization were not specifically studied and tested. Furthermore, these findings have so far relied on handmade exploration, which can be very hard and time-consuming as random rules rarely result in the emergence of localized patterns and even less moving ones (**Movie S2**).

In this work, we propose to use automated experimentation to organize the exploration of Lenia without any human intervention. More particularly, the automated experimentation aims to find local update rules f leading to the self-organization of stable (and if possible diverse) agents with sensorimotor capabilities. We also provide tests in order to assess the sensorimotor capabilities of the obtained patterns.

7.2.1. The Lenia environment

Lenia is a class of continuous cellular automata where each CA instance is defined by a set of parameters θ that conditions the CA rule f_θ. Once the parameters θ conditioning the update rule have been chosen, the system is a classical CA where the initial grid pattern $A^{t=1}$ is iteratively updated. In the multi channel version of Lenia [40], the system is composed of several communicating grids which we call channels. Intuitively, we can see channels as the domain of existence of a certain type of cell. Each type of cell has its own physics : it has its own way to interact with other cells of its type (intra-channel influence) and also its own way to interact with cells of other types (cross-channel influence).

In this work, we are interested in finding parameters (θ,$A^{t=1}$) leading to the self organization of moving agents robust to external perturbations from the environment. For this aim, we need to introduce perturbations in the system in a controlled systematic way, both for testing the robustness and as criteria during the search. However, due to the dynamical nature of the system, controlled perturbations over several steps in the CA system are often hard to introduce. To help solve this issue, we propose to take advantage of the multi channel version of Lenia and separate the low level elements of the system in two types: the first "fixed"channel, which is hand-engineered, introduce elements that act as our stable controlled obstacles (blue in Figure 7.4); the second "learnable"channel, where parameters of the physics are learned, is where the agent has to emerge (yellow in Figure 7.4). In practice, the environment parameters (θ, $A^{t=1}$) are separated in two. The first part, denoted (θ_f, $A_f^{t=1}$) in a hand engineered part where θ_f gives the rule on how obstacles block matter from going in while $A_f^{t=1}$ gives the obstacle placement and shape. [10]. The second part however, denoted (θ_l, $A_l^{t=1}$), is free and is learned in order to self-organize an agent with sensorimotor capabilities.

10: Details on how we implement obstacles as part of the CA rule can be found in Appendix Subsection D.2.2

What we are searching for is thus learnable parameters (θ_l, $A_l^{t=1}$) that will induce a physic leading to the self-organization of agents that are able to move and survive in a grid where obstacles perturb their structure and therefore may break their integrity. Note that finding

pattern with such capabilities is not trivial, for example moving patterns found by hand in [39, 40] (as the Lenia glider), which are stable without perturbations, often die from the collision with our engineered obstacles (**Movie S3**). Note that in our system, if an agent is to emerge, the only way it can "sense"previously-introduced obstacles is from the perturbations obstacles induce on its structure, it cannot "see"the obstacle.

7.2.2. Intrinsically Motivated Goal Exploration Process (IMGEP)

Formally, a set of parameters $(A_f, \theta_f, A_l, \theta_l)$ in Lenia maps to a certain sequence of states (trajectory o). This trajectory can then be mapped to a vector $R(o)$, through a defined characterization function R. This vector provides a behavioral description of the trajectory, and the image of R represents the space of possible behaviors that can emerge in the system. A curious automated discovery assistant can then be applied to explore this behavioral space to discover a diversity of behavior. In this work, we use Intrisically Motivated Goal Exploration Process (IMGEP) to organize the exploration of the behavior space. IMGEP relies on *goal-directed* and *diversity-driven* search, which we leverage to drive the system toward the emergence of desired (sensorimotor) behaviors. More precisely, given a goal-sampling strategy G, IMGEP automatically samples target *goals* $g \sim G$ which are points in the behavioral space. For each goal g, the objective is then to optimize toward parameters (θ_l, A_l) leading to a sequence of state which is mapped as closely as possible to this goal. To score the trajectory according to a goal, a loss function $\mathcal{L}(g, o)$ taking as input the trajectory and the goal is used.

The behavioral descriptor R we choose in this work is the position of the center of mass at the last timestep of a simulation. The behavioral space then consists of all possible (x,y) coordinates in the grid. The objective for a given goal $g = (x, y)$ is thus to find parameters (A_f, θ_f) leading to the emergence of a spatially localized pattern attaining the goal position at the last timestep under several perturbation by obstacles. In this work, we choose to define the (goal-conditionned) loss as the mean squared error (MSE) between the state at the last timestep of the trajectory and a disk centered at the goal position. In addition to closeness to the goal position, the loss function we use incentivizes localization of the mass to prevent pattern explosion and collapse, which is a very common outcome of Lenia parameters. We then use gradient descent to optimize the learnable parameters $(\theta_l, A_l^{t=1})$ by backpropagating the loss through the steps and make progress toward the the goal (Figure 7.4).

Gradient descent optimization has already been successfully applied with cellular automata [37] on learning CA parameters leading to the growth of a target pattern [104] or texture [106], or enabling (static) cellular collectives capable to perceive their large scale structure [107], proving the effectiveness of such method (with some additional component for training for long term stability) in complex chaotic self-organizing dynamic. However, in this work, we consider moving agents which are a fragile type of pattern in Lenia as moving forward in such system means to grow new cells at the front while the ones at the back die. This equilibrium between growth and death is also challenged by the random perturbations we introduce in the system. This means that changes

of parameters, because of the chaotic nature of the system, can easily break the equilibrium between growth and death of cells making the optimization harder.

To help with this difficult optimization landscape we propose to introduce a *curriculum* for making small improvements iteratively. Curriculum learning has already been applied for optimizing cellular automata rule with gradient descent, as a solution for getting out of a trivial local optima in Variengien et al. [277][11].

11: In our case the curriculum also solves technical gradient flowing problem, detailed in Appendix Subsection D.3.6

Here, the intuitive idea behind our curriculum is to first learn rules leading to moving (spatially localized) agents which we train to go further and further (in the same amount of timesteps, hence faster) and at some point train them to go further while dealing with obstacles. To do so, the fixed environment $A_f^{t=1}$ we sample for training has a certain structure: the left half of the grid is free from obstacles while the right part contains obstacles that will be randomly placed at every rollout (blue in Figure 7.5.a). The sampling strategy G we chose in the IMGEP also participate to the curriculum as it is biased to randomly sample goals that are a little bit further than previously attained positions[12]. Putting target goals in the obstacle area means that during training, the potentially emerging agents will have to go to a specific location while its structure is perturbed by obstacles randomly placed. The gradient descent optimization will incentivize recovery from perturbation and to keep moving despite being damaged. In addition, the fact that the obstacles are randomly placed should incentivize generalization to different perturbations.

12: More information on the sampling strategy can be found in Appendix Subsection D.3.4

To sum up, the IMGEP iteratively (and automatically) generates increasingly difficult goals for which we will try to find, and optimize using gradient descent, learnable parameters $(\theta_l, A_l^{t=1})$ that will lead to the self organization of agents achieving these goals. For each goal (position), the optimization steps are done under several obstacle configurations $\{A_f\}$ in order to learn to resist to different perturbations. After each optimization, we then test the optimized parameters under various environmental configurations $\{A_f\}$ to assess the reached position (and distance to target goal). We store this (parameters, reached position) couple in history $\mathcal{H}$ in order to be able to use it as a starting point for subsequent goals.

7.2.3. Evaluation of the discovered patterns

Whereas the notion of *agency* is closely tied to the ability of an organism to maintain its own organization despite encountering novel circumstances, the robustness of current artificial autopoietic systems is lagging far behind the robustness of their biological counterparts. We believe that this limitation, together with the difficulty of engineering such autopoietic systems, is a major reason why we have not assisted yet to a wider adoption of the enactivist framework by the AI community. The IMGEP search, precisely intended to facilitate the search of such autopoietic systems, should provide us with a database of parameters $\{(A_f, \theta_f)\} \in \mathcal{H}$ that (when successful) lead to the self-organization of patterns that are robust (at least) to the different obstacle configurations seen during training.

To go further and characterize *agency* and the degree of *robustness* of the discovered parameters/patterns, we propose an empirical evaluation procedure in two stages. First an "agency filter"is used on the database of

discoveries to discard parameters that do not lead to the self-organization of what we call "agents"in Lenia. More precisely, our filter implements several classifiers, inspired from ones proposed by Reinke et al. [•1], to detect whether the emergent matter does not disintegrate (vanishes or explodes), forms a coherent entity (single soliton), and does so during a long-enough time window (longer than training). In addition to the agency filter, we also introduce a "moving filter" which tells if an agent is moving (travels a minimum distance) or not (examples of discovered "agents" that are considered not moving are shown in **Movie S4**).

Then, to assess the capabilities of selected agents to withstand perturbation by obstacles we perform a "basic obstacle test": testing them on obstacle configurations similar to the ones seen during training; and various "generalization tests": running them through a battery of tests with several *out-of-distribution* perturbations that were not seen during training. In particular, we test the discovered sensorimotor agents to harder obstacle configurations, stochastic cell updates, changes of initialisation and changes of scale that were not experienced during training. Given a distribution of perturbations, we measure *robustness* as the average performance over sampled perturbations, where performance is a binary success metric that determines whether the agent "survived" the perturbation or not. As "survival"metric, we simply apply our agency filter to detect whether the (perturbed) emergent entity is able to self-maintain despite the introduced variations (ie is still an agent at the end of the test). Note that this metric closely follows the definition of *cognitive domain* of an autopoietic system, which was introduced by Maturana and Varela [264] and later defined by R. Beer as the percentage of *non-destructive*[13] perturbations, out of all possible perturbations, that the autopoietic system can tolerate [192]. Because measuring the cognitive domain "as such"would require an exhaustive enumeration of all possible perturbations and all possible "valid"states that the entity can take, which is not tractable in the Lenia environment, we instead rely on a proxy metric and on a set of chosen empirical tests. We refer the reader to Appendix Subsection D.4.1 for more details on our evaluation procedure.

13: a perturbation is said to be *destructive* if it fundamentally disrupts the entity's organization leading to its disintegration [192]

In addition, we provide an interactive web-demo[14] where one can replay the discovered agents and test them to all sorts of freely-drawn perturbations including custom obstacle shapes, addition and/or removal of mass, interactions with other agents in the grid and control of environmental cues (attractive elements) in the Lenia grid.

14: the demo can be found at `https://developmentalsystems.org/sensorimotor-lenia-companion/`

Altogether, we hope that those (quantitative and qualitative) tests, which were all implemented within the continuous CA paradigm, can serve as a good baseline to evaluate the generalization capabilities (and hence the degree of agency) of autopoietic systems in enactivist research, akin to commonly deployed benchmarks in AI for evaluating mechanistic forms of agency [167].

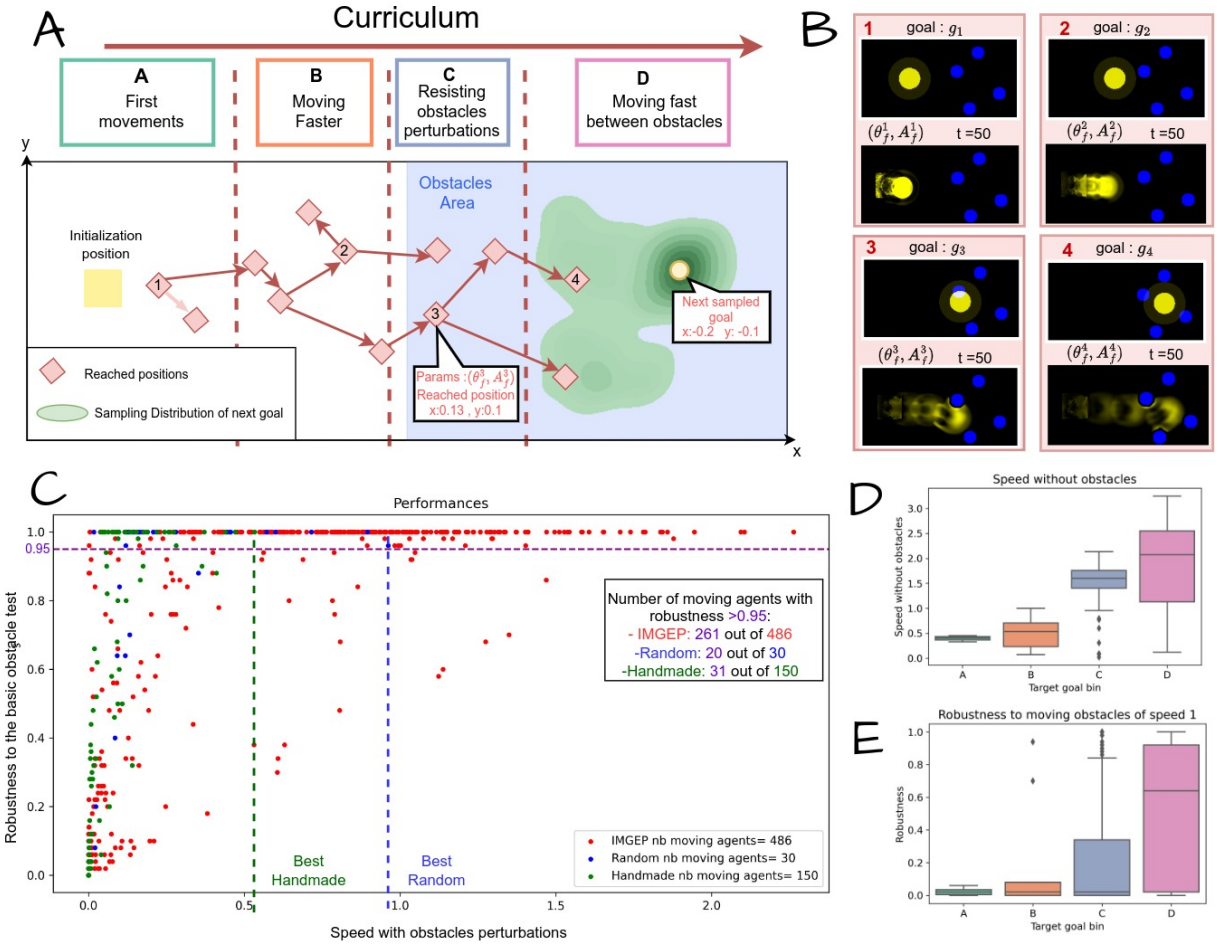


Figure 7.5.: Overview of the proposed approach and results. (a) Illustration of the training curriculum. The curriculum iteratively sample goal position (yellow disk), further in the grid, starting from very close to the initialization (A) to further away without obstacles (B) to further away in the obstacle area (C, then D). Arrow between reached positions (red square) represent that the parameters leading to a pattern attaining the tip of the arrow position was initialized before training by the parameters reaching the back of the arrow position. (b) Examples of patterns obtained along the curriculum as well as their associated goal. We observe patterns going further and further in the same amount of steps (50 steps) and for the latter dealing with obstacles in their way. To display the trajectory of the agent in the learnable channel (yellow) we superposed the frames over all timesteps putting more transparency in earlier timesteps. (c) Performances in term of robustness to the basic obstacle test and speed with obstacle perturbations of the moving agent produced by: IMGEP (red), random search (blue) and handmade search (green). (d-e) Distribution of the speed without obstacles perturbation (d), and of the robustness to moving obstacles (e), of the moving agents discovered by the IMGEP along the curriculum. Details on the speed measure can be found in Appendix Subsection D.4.3. We observe that the curriculum translates in an improvement in the 2 presented quantities.

7.3. Curiosity-driven Search Reveals Environmental Rules leading to the Systematic Emergence of Sensorimotor Entities

In this section, we analyze the discoveries made by the proposed approach (IMGEP) and compare it with two other exploration baselines: a *random search*, where parameters are sampled uniformly in the parameter space (same ranges than for the IMGEP, given in Appendix Subsection D.2.3); and a *handmade search*, where we collected the discoveries made by semi-automatic search and expert selection as presented in the original Lenia papers [39, 40]. Each IMGEP experiment has 160 goal-sampling steps, hence output 160 parameter sets (A_l, θ_l), but performs in average 11700

Lenia rollouts, due to stochasticity in the method (see Appendix Section D.3). For IMGEP and random search, 10 independent repetitions are performed (where random search is given the same experimental budget of 11700 rollouts per seed). Note that the comparison with handmade search, while interesting, is challenging in practice as it is the result of tedious search for which the total experimental budget is unknown, and which was conducted over some Lenia hyper-parameters that are not all included in the automated search (*e.g.* various number of channels or kernels). Moreover we use a slightly different parametrization of the rule to allow for differentiability (details in Appendix Subsection D.2.1).

For the three baselines (IMGEP, random search and handmade search), we filter the obtained parameters to select only the moving agents (passing the agency and moving test) and measure their speed and robustness under the basic obstacle tests as described in Subsection 7.2.3.

Individuality, locomotion and sensorimotor capabilities

As illustrated in Figure 7.5, the IMGEP search enabled to evolve agents along a curriculum which progressively led to the emergence of individuality, locomotion and sensorimotor capabilities.

At first, the IMGEP samples goals that are not too far from initialization (zone A in Figure 7.5.a) enabling to find rules leading to the self organization of spatially localized area which starts to move a little bit from initialization (as shown in Figure 7.5.b.1). Then, from these newly learned rules the IMGEP samples further goals (zone B in Figure 7.5.a) which lead to spatially localized patterns that are able to move further in the grid in the same amount of time (Figure 7.5.b.2). At this point, some obtained parameters already lead to the self-organization of *moving agents*, *i.e.* passing our empirical agency and moving tests (long-term stable solitons capable of moving while self-maintaining). Moving agents patterns are in fact already not trivial to find through random search in the parameter space as only 30 moving agents were found through the 10 seeds of random search out of a total of 117 000 trials of parameters.

The IMGEP pursues the curriculum, taking advantage on the previous learned parameters that are already able to emerge moving agents, and puts further target goals within the obstacle area (zone C in Figure 7.5.a), leading to moving agents entering the obstacle area (Figure 7.5.b.3). As expected, the parameters resulting from those goals sampled in the obstacle area have a higher robustness to obstacles (Figure 7.5.e) and agents trained with further goals move in average at faster speeds in environment without obstacles (Figure 7.5.d).

At the end of the curriculum loop, the obtained rules often lead to the self organization of moving agents that are able to navigate fast in an area with obstacles while still maintaining their integrity (Figure 7.5.b.4, **Movie S5**). The emerging agents are capable of changing direction and recover in response to perturbations induced by the obstacles, *i.e.* have sensorimotor capabilities, and this only through the global coordination of the identical low level parts (without having any central unit computing decision).

In total, 9 out of the 10 seeds led to at least one sensorimotor agent, *i.e.* moving agent with a measured robustness >0.95 in our basic obstacle

test. Note however that the performance in term of speed with obstacles varies from one seed to another (see Appendix Table D.1), which depends on the initialization on the method. Over the 10 seeds a great part of the obtained emerging moving agents are sensorimotor agents. In fact, over 10 seeds, 486 of the 1600 parameters (10 seeds x160 parameters) led to moving agent according to our agency and moving filter, from which 261 have a robustness to obstacles>0.95.

As a comparison, out of the 117 000 parameters generated by the 10 seeds of random search, only 30 led to moving agents from which 20 have a robustness to obstacles >0.95. Our method surpasses random search in term of speed with obstacles and robustness of the obtained agents, as well as the total number of long term stable moving agents obtained as can be seen in Figure 7.5.c (486 for IMGEP and 30 for random search in total over 10 seeds and with the same Lenia rollout budget). We also get emerging agents with better robustness and speed than the ones found in the original Lenia papers [39, 40] (Figure 7.5.c).

7.4. The Discovered Entities showcase strong Generalization Abilities

Biological organisms are able to maintain phenotypic stability in the face of diverse environmental perturbations arising from external stresses, intracellular noise, and even quite drastic changes during morphogenesis such as perturbations to the embryo structure [278] or to the substrate cellular size [279]. It has long been recognized that robustness is an inherent property of all biological systems that has been strongly favored by evolution [280]. In this section, we are interested to see if similar robustness capabilities can be achieved by the artificial self-organizing agents that have been discovered by our artificial evolution workflow (Figure 7.6). To do so we evaluate the generalization capabilities, over the proposed battery of tests, of the 10 best agents discovered by the IMGEP, random and handmade search variants, as well as on the agents that have a speed within obstacles greater than one (91, all discovered by IMGEP). "Best" here is computed according to the speed-robustness criteria presented in Figure 7.5-c, *i.e.* the fastest with obstacle that also have a robustness in the basic obstacle test>0.95. The performances are fully reported and compared in Appendix Table D.2. As we will see, the discovered agents showcased quite impressive generalization capabilities at the organic, sensorimotor and inter-subjective levels [265]. We group the observed generalization capabilities into six categories: harder obstacle configurations (external stresses), stochastic cell updates (per-cell noise), changes of initialisation ("embryo" variation), changes of scale (compute capacity variation), interactions with other agents in the grid (inter-agents regulation) as well as with human-controlled environmental cues (observer-agent regulation).

Harder obstacles We first tested the agents generalization capabilities to a larger and more challenging set of obstacle configurations. The test set includes controlled configuration with varying number, size and speed of obstacles (Figure 7.6-A-a), as well as human-drawn obstacles such as

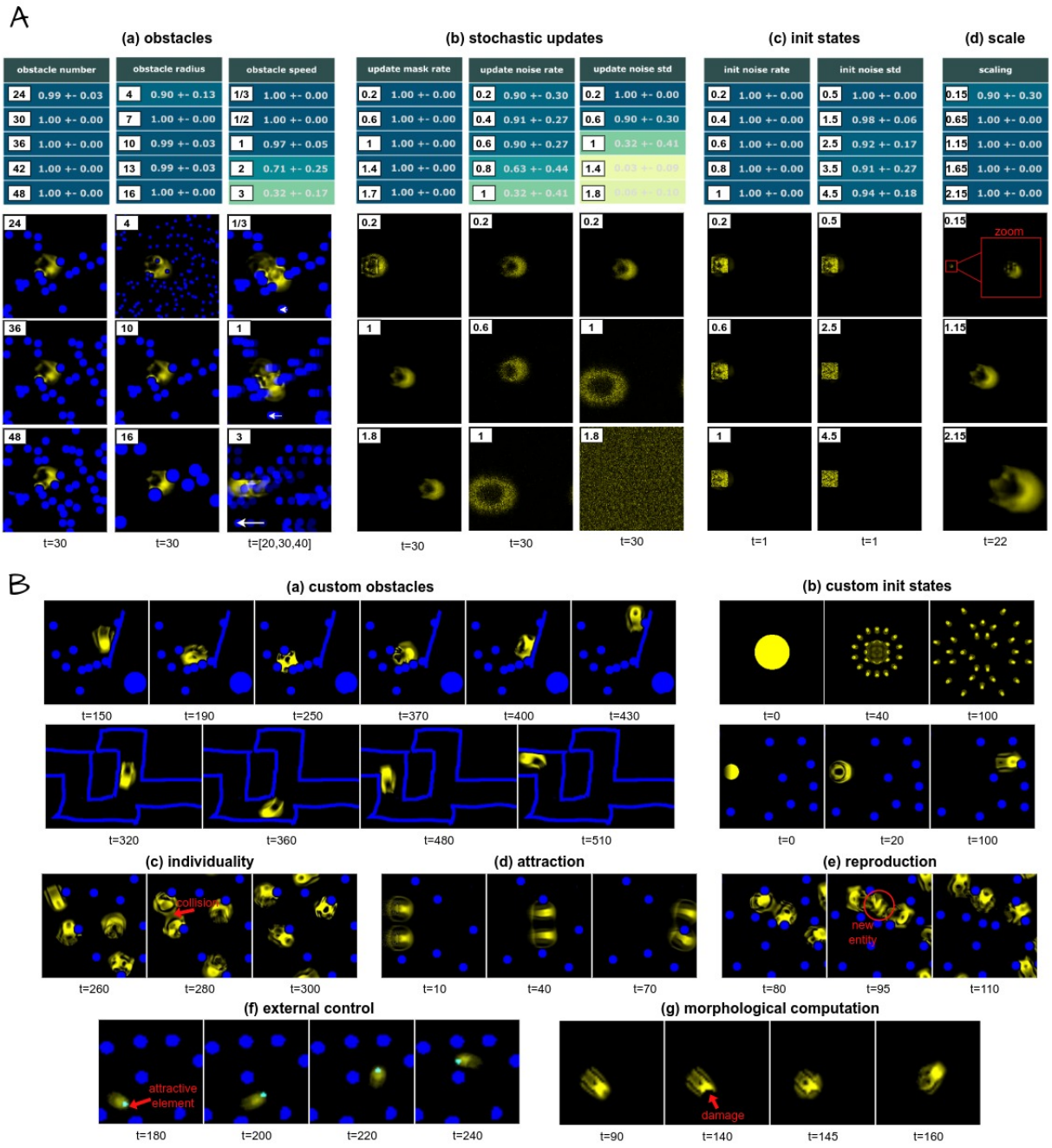


Figure 7.6.: Generalization of the discovered sensorimotor agents. (A) We conduct a battery of quantitative tests which we organize in 9 families of parameterized perturbations that test for various (a) obstacle number, size and speed, (b) rate of cell updates, as well as rate and magnitude of noise added to the updates, but also (c) rate and magnitude of noise added to the initial state and (d) scaling factors. For each family, we test for 5 different parameter values, *i.e.* perturbation strength, resulting in a total of $9 \times 5 = 45$ tests. For each test, the performance of an agent is computed as the average score of survival over 10 random seeds. A score of 1 (dark blue) means that the agent survived all 10 tests whereas a score of 0 (light yellow) means that the agent survived none of the tests. The table reports the mean and standard-deviation performances, over the 10 best agents discovered by our goal-directed curriculum, for all of the 45 tests (one table cell per test), where "best"is determined by the speed/robustness criteria introduced in Figure 7.5-c. Below each column, we show snapshots of system rollout at test time given the newly introduced perturbations. The shown snapshots are all taken from rollouts of the "best"agents, and from the first seed (out of the 10 tested random seeds). Timesteps are specified under the images, for instance snapshots of the perturbations applied on the initial state are shown at t=1. (B) We also conduct a battery of qualitative tests, where we tested the (best) discovered agents to all sorts of difficult perturbations including (a) freely-drawn obstacles such as walls, mazes or dead-ends (b) freely-drawn initial states such as very big disks (resulting in the emergence of multiple entities) or small disks with gradient asymmetry, (c-d-e) introduction of other agents in the grid (resulting in the emergence of inter-agent interactions such as individuality maintenance, attraction and reproduction), (f) the introduction of novel low-level elements that have an "attractive"effect on the agents (allowing external user to guide the agent trajectory in the grid); and (g) custom mass removal (pixel erasing).

vertical walls and dead ends (Figure 7.6-B-a). Interestingly, whereas some well-placed perturbations can lead to death or explosion, the discovered agents show strong robustness and generalization to most of the test set configurations. They showed quasi-perfect survival to grids with up to 48 obstacles, to grids with smalls (but dense) or big (but sparser) obstacles, and to obstacles with moderate speed. High-speed obstacles however (speed greater than two), seem to challenge agent's survival (Figure 7.6-A-a), even though the IMGEP-discovered agents are still much more robust to moving obstacles than the ones discovered by random and handmade search (appendix Table D.2). Those results suggest that, by training for fast-moving and obstacle-resisting behaviors, our goal-directed curriculum favored the self-organization of agents that are able to *quickly* recover from perturbations induced by the environment, even ones not seen during training. For instance, qualitative tests also showed that the discovered agents are able to successfully navigate forward while coming across tightly-packed obstacles, walls of various inclinations, corners, dead ends and even bullet-like types of obstacles (**Movie S6**).

Stochastic updates We then tested the agents generalization capabilities to asynchronous and noisy cell updates (Figure 7.6-A-b). As proposed in Mordvintsev et al. [104], relaxing the traditional assumption of synchronous update in cellular automata (which assumes a global clock) is closer to what you would expect from a self-organized system, and can be done by applying a random update mask on each cell (parametrized by the update mask rate). Despite the update mask enforcing asynchronous and less (or more) frequent cell updates at test time, the discovered parameters still give rise to self-organized agents that perfectly self-maintain (survival scores of one) and that showcase very similar morphology and behavior than the agents with synchronous updates (**Movie S7**). The agents are slowed (or fasten) a little bit but this is what we can expect as each cell is updated in average only a fraction of the time (or several times per timestep). We also relax the assumption of exact update by adding random noise (parametrized by noise rate and magnitude), to the cell states during the system rollout. Here, we observe that the agents can resist quite consequent quantities of noise but passed a certain level, as expected, the collective looses its integrity and desintegrates.

Changes of initialization While the initialization pattern has been learned with a lot of degree of freedom (pattern in $[0, 1]^{40\times40}$), we can look if similar patterns (phenotypes) can self-organize from other (maybe simpler) initialization patterns (Figure 7.6-A-c). This capacity to converge to the desired anatomy in spite of a different initialization ("embryo"), is something that can be found in biological organisms [278], and that we can expect in our system as well. Interestingly we indeed observed a quasi-perfect robustness to noise-altered initial states (survival scores close to one), and this even for quite high amounts of noise (except for few configurations that led to death). These results suggest that the final phenotype forms a strong attractor towards which the different initial mass pattern tend to converge under the learned CA rule. The learned rule is hence prone to encode, grow and maintain a specific target morphology (and its associated functionality), which is consistent with the agent ability to recover from obstacle-induced perturbed morphology.

As shown in Figure 7.6-B-b, we also tested for handmade initial patterns such as bigger disks and same-size asymmetrical disks. Interestingly, the large disk initialization led to multiple entities forming and separating from each other. The same-size disk, which is much simpler than the trained initial states but preserves some form of asymmetry (gradient activation), also converged toward the same morphology. However the robustness to initialization is not perfect as many initializations, such as smaller size and/or symmetrical disk, easily lead to death (**Movie S8**).

Changes of scale Similarly, while the initialization and update parameters have been learned at a certain spatial resolution during training resulting in agents of a certain size (in term of number of cells), we can artificially change the scale at test time by approximate resizing of parameters (see Appendix Subsection D.4.5). As shown in Figure 7.6-A-d, we tested for different down-scaling (and up-scaling) factors that surprisingly resulted for most of them in fully functional agents with the overall same structure but smaller (or larger) size in terms of number of cells. For agents which are down-scaled, and hence have much less cells to do the computation, it is particularly surprising that they are still able to sense and react to their environment and still show relatively-advanced levels of robustness (**Movie S9**). This scale reduction has a limit (a scaling of 0.15 already leads to some death) but we can go quite far down and still obtain functional phenotypes. For the bigger agents, which therefore have more space to compute (but also more cells to organize), we observe similar results where agents still self-organize to functional phenotype. Once again, this resonates with findings in biology suggesting that organisms are able to accommodate cell-size differences by adjusting cell number in order to maintain roughly constant body size and structure [279].

Interactions We were then interested to test how the discovered agents would react when interacting with other agents in the grid. Given the set of parameters (A_l, θ_l), we can trigger the forming of several macro-entities at test time by replicating the initialization square pattern ($A_l \in [0,1]^{40\times40}$) at different locations within a larger grid ($A^{t=1} \in [0,1]^{512\times256}$) and letting the system unroll. Doing so leads to the development of several entities of the same "specie" (governed by the same update rule/physic θ_l). As illustrated in Figure 7.6-B, we did that for several of the discovered sensorimotor agents, and qualitatively observed several interesting emergent interactions.

The first thing that we observed is that, several of the discovered agents show strong *individuality* preservation (**Movie S10**). The fact that the individual agents do not merge nor enter in destructive interactions despite being all made from identical cells is an intriguing example of how the boundary of a "self" [281] can emerge and maintain in self-organizing systems. In particular results suggest that, in the Lenia system, individuality can be obtained as a byproduct of training an agent alone. Our intuition is that by trying to prevent too much growth during training, it learned to prevent any living cell that would make it "too big", including living cells from other entities here.

A second type of interaction that can be observed with certain parameters/environments is *attraction*. As illustrated in **Movie S11**, two agents placed in the same grid can show attraction when coming close enough from one another, leading them to stay together and move in the same direction. Interestingly, when they encounter an obstacle, they are able to separate briefly and then to reassemble together. Similarly, even when they stay together, we can still qualitatively observe two distinct entities that are interacting with one another while maintaining their overall shape and integrity. This type of behavior has been studied in the game of life under the concept of *consensual domain* [191].

A third type interaction that has been observed in some of the discovered agents is a form of *reproduction* where collision between two agents give rise to the birth of a third entity (**Movie S12**). This kind of interaction seems to happen when one of the two colliding entities is in a certain "mode", like when it just hit a wall. Our intuition is that when it hits a wall, the self-organizing agent produces a growth response in order to recover. During this growth response if there is extra additional mass coming from another entity then the self-organizing agent might split off from the created mass while the separated mass, from robust self-organization (see "Changes of initialization" above), grows into a complete individual.

External control A central challenge in synthetic biology, when faced with unconventional forms of agency such as collective of cells, is to find new ways to communicate with the cells to induce desired behaviors at the collective level without having to physically "rewire"the structure of the agent (*e.g.* via genome editing) but rather by introducing externally-controlled cues in the environment [45]. Here, we are interested to see whether we can induce (novel) target behaviors in the discovered agents without having to modify the learned parameters θ_l. In particular, we investigate whether the agents can show *attraction* to some novel elements in their environment (like in nature organisms being attracted to certain chemicals, lights or temperatures) and if we could use those elements to guide the macro-entity. To do so, we introduce a new type of "attractive"low-level elements within the Lenia CA paradigm. More precisely, given the set of learned parameters θ_l, we introduce a novel local rule with parameters θ_a that determine the physical influence of the attractive elements onto the agent cells. To find parameters θ_a triggering the desired attraction effect at the agent behavioral level, a simple random search with final human assessment was performed[15]. **Movie S13** is an example of obtained behavior where we can observe the sensorimotor agent getting attracted to the newly-introduced environmental element (disk of cyan particles) which allows the external user to "control"the agent trajectory by moving the disk in the grid. Interestingly, in spite of this novel behavior, agents are capable to maintain their normal sensorimotor capabilities showing robustness to collision with obstacles and other agents in the grid. Besides, once the attractive element is removed the agents return to their normal behavior. However adding extra rules also fragilize equilibrium that existed in the agent rules as it creates perturbations that the agent has not been trained to withstand, leading sometimes to death or explosion (or to other behaviors such as reproduction due to extra boost of growth). Once again parallels can be drawn with findings in biological organisms, for instance [282] show that

15: see Appendix Subsection D.4.5 for details on the procedure

controlled UV light beam can be used to externally guide the trajectory of micro-swimmers to perform on-demand drug discovery. While we only tested for attraction-type of generalization behaviors, we believe that more sophisticated types of environmental guidance could be induced.

Morphological computation This section has provided several empirical evidences of how adaptive high-level functionality can emerge from a collective of low-level, decentralized elements. In order to withstand the tested perturbations, the cellular collective first needed to "sense" the induced perturbations through a deformation of the macro structure. After this deformation it had to "communicate" the information and make a collective "decision" on where to grow next. Then it had to move and regrow its shape, altogether giving rise to the observed robustness of the macro structure. In order to better visualize the physical manifestation of decision-making within the cellular collective, we manually suppressed a part of the agent (Figure 7.6-B-g). We observe that perturbation of the macro-structure is what leads to the direct change of direction, supporting the fact that computation of the decision is made at the morphological level. Whereas the idea that morphology, decision-making and motricity are highly entangled phenomena is a long standing idea in biology in biology [269], providing a human-interpretable account of how the decision is made remains a difficult problem.

7.5. Flow Lenia: Towards Open-Ended Evolution in CA through Mass Conservation and Parameter Localization

The discoveries presented in this chapter have shown the existence of environmental rules in Lenia leading to the emergence of autopoietic (*i.e.* self-produced) spatially localized patterns (SLPs), not only resembling microscopic life-forms but displaying various life-like behaviors like individuality motility, and self-replication. These observations confirm that Lenia is a particularly interesting testbed system for studying the emergence of life-like phenomena and even sensorimotor capabilities, which is central in enactive theories of cognition and artificial intelligence [270].

However, as was shown in this chapter, SLPs and moving SLPs are quite difficult to find in Lenia, necessitating advanced search algorithms. Another important challenge in ALife and AI, and in the current Lenia system, is about the design of systems displaying open-ended intrinsic evolution (*i.e.* unbounded growth of complexity through intrinsic evolutionary processes) [77]. Such a process is called *intrinsic* since no final objective (*i.e.* fixed fitness function) is set by the experimenter, the fitness landscape is intrinsic to the system and depends only on its current state, as in natural evolution where there is no final goal [127]. In Lenia, different "genomes" (update rule parameters) cannot coexist in the same grid. This means that, even though we were able to discover a great diversity of "creatures"[16] in Lenia, they cannot exist in the same world (*i.e.* the same simulation) and thus cannot interact[17]. Obtaining such an evolutionary process in the Lenia system, and in a CA in general, could be achieved by embedding information in the system locally modifying the

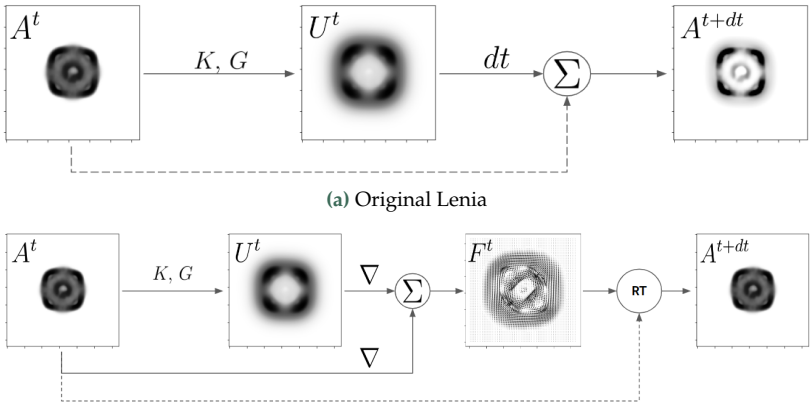
(a) Website

(b) Notebook

Figure 7.7.: (a) Flow Lenia website with several videos. (b) Notebook with Flow Lenia implementation and demo.

16: In this section we use the term "creature" as a general term to refer to "life-like" moving SLP patterns in Lenia

17: The interactions shown in Figure 7.6 happened between creatures of the same type (same "genome")



(a) Original Lenia

(b) Flow Lenia

Figure 7.8.: (a) Lenia update rule. The growth U^t is computed with kernels K and growth functions G defined by a specific parameter configuration sampled in Lenia's parameter space. A small portion of the growth is then added to activations A^t to give the next state A^{t+dt}. (b) Flow Lenia update rule. Affinity map U^t is computed as in Lenia. The flow F^t is given by combining the affinity map and concentration (*i. e.* activations) gradients. Finally, the next state is obtained by "moving" matter in the CA space according to the flow F^t using the reintegration tracking method [195].

update rule and so the properties of emerging creatures, which should in turns enable multi-species simulations. Such simulations might set the stage for evolution to occur in populations of patterns each with their own update rule. However, achieving it in CA like Lenia is still an open-problem.

In this section, we present Flow Lenia, a variant of the original Lenia system which we briefly introduced in Section 3.2.3 of this manuscript. Flow Lenia stems from the idea that adding *mass conservation* to the Lenia system, could solve many of the aforementioned challenges. Indeed, such a constraint could (i) constrain emerging creatures to spatially localized ones, (ii) allow for the design of multi-species simulations and (iii) provide an important evolutionary pressure. After briefly explaining how Flow Lenia integrates mass conservation within the Lenia dynamics, we will see why we believe it opens promising perspectives toward achieving open-ended evolution (OEE) in such artificial substrata.

Flow Lenia Model As illustrated in Figure 7.8, the main idea of Flow Lenia is to reinterpret the output of the *growth update* $U^t = G(K * A^t)$, which is originally used to *create* new mass (or remove existing one), as an *affinity field* $F^t \approx \nabla U^t$ which is instead used to *move* the distribution of "matter" toward high affinity regions in space. In practice, Flow Lenia uses the reintegration tracking method [195], which is fully described in Appendix Subsection E.1.2. Intuitively, one can see Flow Lenia as a grid-based approximation of a particle system (with infinite number of particles) and which has the property of conserving the total mass. Thus, Flow lenia can be seen as a new kind of model at the frontier between continuous CA and particle systems[18]. Flow Lenia is implemented in JAX, and we refer to the notebook for playing with the system.

18: A particle model directly inspired by the Flow Lenia formulation has been recently proposed in Mordvintsev et al. [199]

The role of mass conservation in the emergence and exploration of artificial creatures The first interesting result in Flow Lenia is that it is much easier to discovery SLPs that exhibit interesting and complex behaviors than in the original system. In fact, by performing a simple random exploration of the system[19], we can observe that most of the patterns generated in Flow Lenia are SLPs (see Figure 7.9). This suggest that mass conservation constrain (almost all) emerging patterns to spatially localized ones, though some configurations still lead to scattered

19: Explored parameter ranges and hyper-parameters are given in Table E.1

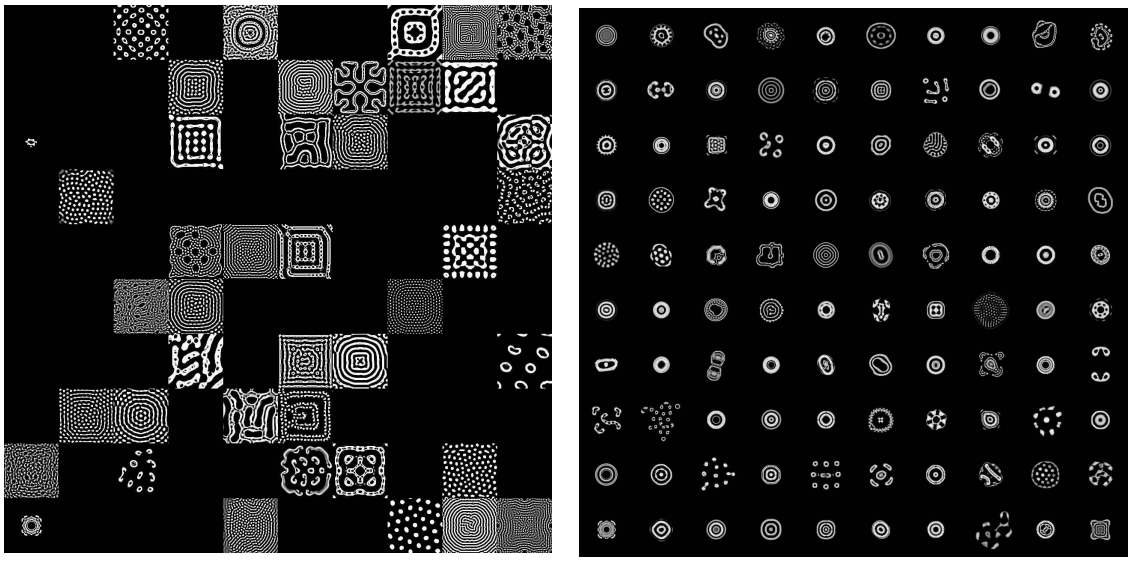
matter. While some of the emerging SLPs tend to be static ones, dynamic patterns are also quite common in Flow Lenia and several interesting dynamics can be found, as illustrated in Figure 7.10-a. We also found that multi-channel creatures often show more complex dynamics and patterns with very modular shapes where each channel seems to occupy a different role (Figure 7.10-a bottom). We also found that Flow Lenia update rule parameters can be easily optimized to generate SLPs with desired behaviors. This is a difficult task in the original Lenia as SLPs are in fragile equilibrium (can easily vanish or explode), requiring advanced optimization methods like curriculum learning and gradient descent (as proposed in Section 7.2). In Flow Lenia, the spatial localization constraint is intrinsic to the system making the emergent behaviors more stable and much easier to optimize. Using simple evolutionary strategies [283] to optimize the update rule parameters[20] and the initial configuration ($A^{t=1}$), we have been able to successfully find good solutions for 4 different tasks[21]: directed motion, angular motion, navigation through obstacles and chemotaxis. Obtained creatures are shown in Figure 7.10-b.

20: we optimized the Flow Lenia update rule with different number of kernels and either 1 or 2 channel

21: Implementation details can be found in Appendix Subsection E.2.2

Parameter localization: unlocking possibilities for multi-species simulation One very interesting feature of Flow Lenia is that it enables the integration of the parameters of the CA update rules within the CA dynamics, making them dynamic and localized. Intuitively, this because we can now "attach" a vector of parameters to the matter locally modifying how it behaves and let it flow with it[22]. This allows for multi-species simulations, with locally coherent update rules that define properties of the emerging creatures as shown in Figure 7.11-a (where all creatures patches are initialized with different parameters). We can see that the creatures, although in the same CA, display different morphologies even though they have been initialized with the same pattern. We also ran larger scale simulations (*e.g.* 1024×1024 worlds)

22: Additional details on how this is implemented in practice are given in Appendix Subsection E.1.3



(a) Original Lenia

(b) Flow Lenia

Figure 7.9.: Patterns generated from randomly sampled parameters sets (no cherry picking) in Lenia (a) and Flow Lenia (b). Parameters used for both systems are the same (*i.e.* each 10×10 cell corresponds to one parameters set). Initial patterns $A^{t=1}$ are set with a 40×40 patch with matter drawn from uniform distribution in the center of the grid and no matter everywhere else. Parameters of the update rule are sampled uniformly within the ranges detailed in Appendix Table E.1.

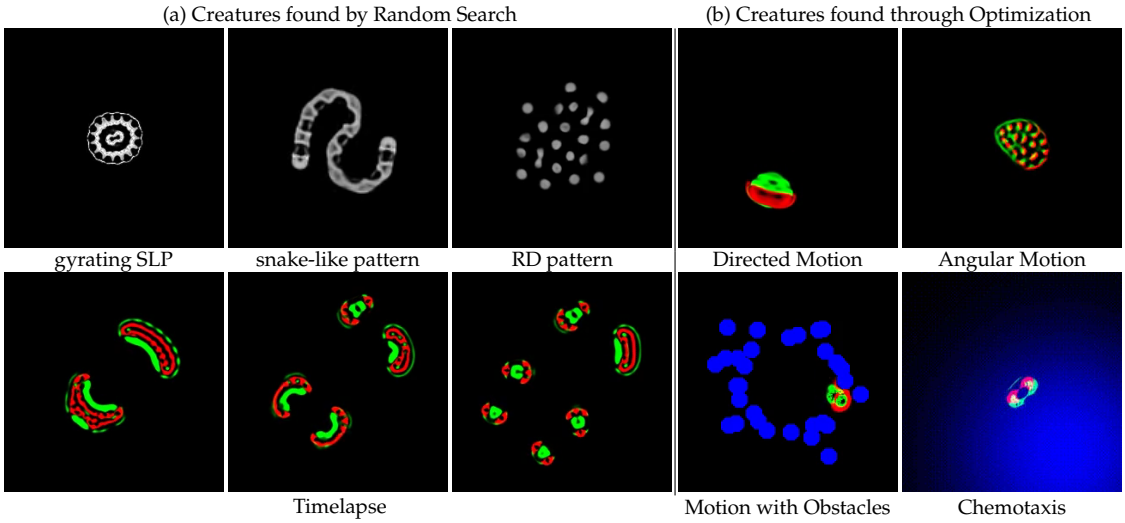
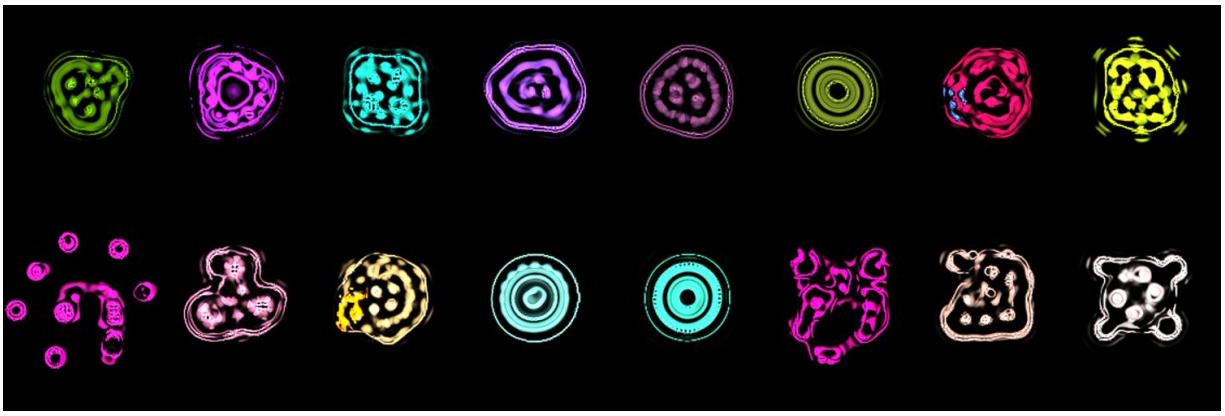


Figure 7.10.: Example patterns produced in Flow Lenia with simple search strategies. (a) Examples of dynamical behaviors discovered with random search. (a-top) Gyrating SLPs, snake like patterns with complex motion emerging from attraction/repulsion dynamics, and dividing and merging dots resembling reaction-diffusion patterns are frequently observed. (a-bottom) Timelapse shows a 2-channels creature displaying complex division patterns and interesting modular creatures whose characteristics change depending on their total mass while being of the same "kind". (b) Best performing creatures discovered found by the 4 optimization tasks detailed in Appendix Subsection E.2.2. The first creature, found in 2-channel Flow Lenia, shows directional movement which results from attraction/repulsion dynamics between the 2 channels. The second creature exhibit internal dynamics leading it to periodically make 180° turns while moving in straight line the rest of the time. The third creature, was found to display directional movement despite encountering obstacles in its way, akin to what was done in [•5] but where obstacles are implemented here as adding a strong repelling flow. Finally, the last creatures was successfully trained for sensing a "chemical" gradient and climbing it towards its maximum.

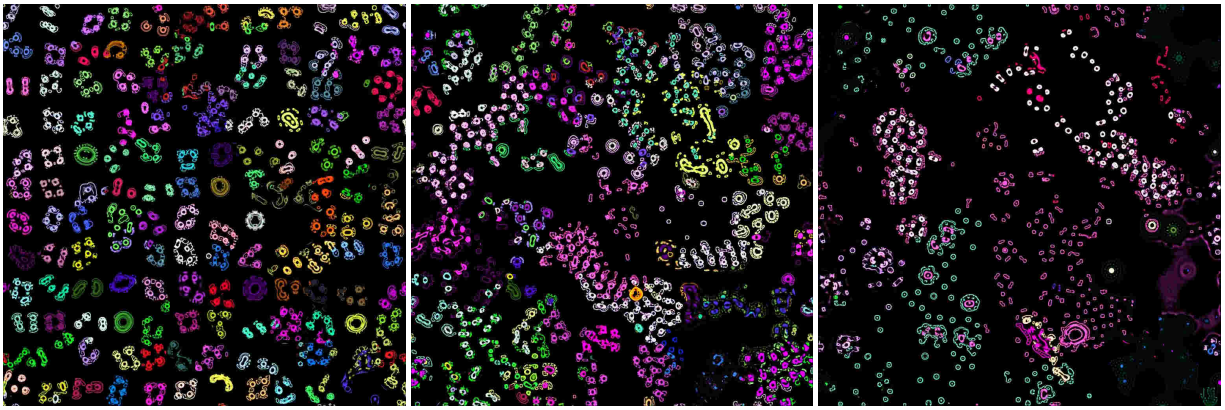
using parameter embedding as shown in Figure 7.11-b. Such a simulation display very interesting dynamics on the large scale where some "species" can take over large parts of the world by contaminating other species before reaching more stable states. Also, we can see the emergence of coherent creatures composed of different parameters where one of those will compose a sort of membrane around a nuclei composed by different parameters (see green and pink creature on bottom left at the end of simulation). Such a pattern can be seen as a form of symbiosis.

Physical constraints in the environment: towards open-ended evolution? Finally, while parameter embedding could already lead to the emergence of intrinsic evolutionary processes where parameters would compete for available matter, we can believe that adding intrinsic selective pressures in the form of environmental constraints (or opportunities) in the environment could bootstrap evolution. Whereas environment design is poorly addressed and quite challenging in cellular automata systems, we believe that it is crucial to study the emergence of agency and cognition in those systems as argued in Godfrey-Smith [284]. To that end, we explored two types of environmental constraints[23]: changing "temperature" (Figure 7.12a) or introduction of "food" resources that creatures need to collect in order to replenish their own constantly decaying pool of resources (Figure 7.12b). In Figure 7.12a, we show an example (single specie) simulation in Flow Lenia where temperature is linearly increasing from left to right, showing very different phases of the systems. More interestingly, patterns at the frontier between the Turing-like phase (center) and the equilibrium phase (right) are much more dynamic and display unpredictable dynamics suggesting a critical regime. In Figure 7.12b, we show an example multi-species simulation with random set of

23: Implementations details are provided in Appendix Subsection E.1.4



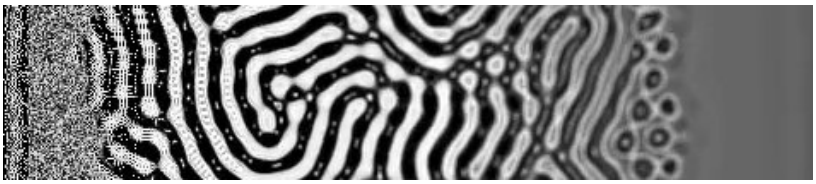
(a) Example multi-species simulation



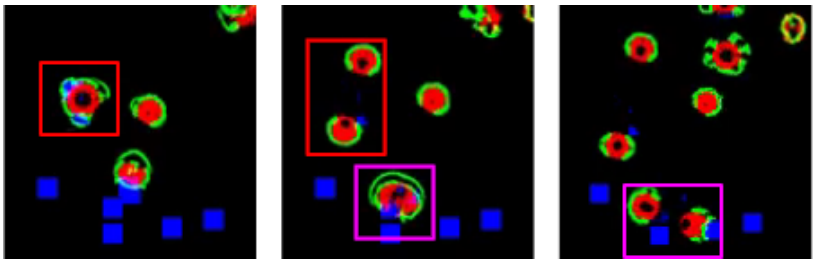
(b) Larger-scale multi-species simulation

Figure 7.11.: Multi-species simulations. (a) Sample of a multi-species simulation with random set of parameters and initialization patterns, where color code for parameters. (b) Timelapse of larger scale multi-species simulation. World is a 1024×1024 grid initialized with 144 creatures with distinct parameters represented by colors. We simulate 200k timesteps and use softmax sampling as the mixing rule and random mutations every 500 steps (see website for videos).

parameters, parameter embedding and food, leading to the emergence of several interesting patterns. First, we observe that some creatures, while not having trained for it, are able to go towards nearby food sources and consume it. We can hypothesize that creatures with such a capability will survive (and grow) while other will not leading to intrinsic evolution. Quite interestingly, complex patterns can emerge from the change of mass induced by decay or food consumption. For instance, when growing after eating, some creatures will divide in two identical creatures, which is crucial for evolution to occur. On the other hand, mass decay also leads to interesting dynamics where creatures undergo phase transitions, changing their shape and behavior, when their mass falls below a certain threshold which can lead them to adopt foraging behavior while being initially static.



(a) Varying temperature from "cold" (left) to "hot" (right)



(b) Adding "food" resources (blue) in the environment

Figure 7.12.: Environmental constraints in the Flow Lenia system. (a) Effect of changing temperature in Flow Lenia, where temperature is linearly increasing from left to right. (b) Timelapse of simulation with parameter embedding and food (in blue) showing division events (highlighted with boxes).

7.6. Discussion and Future Work

In this chapter, we proposed a method combining IMGEP diversity search with gradient descent and curriculum-driven exploration (Section 7.2) and demonstrated its efficiency for learning the update rule and initialization state, from scratch in high dimensional parameters space, leading to the systematic emergence of different robust agents with sensorimotor capabilities (Section 7.3). More broadly, we believe that the set of tools presented here can be useful in general to discover parameters that lead to complex self-organized behaviors. Moreover, we have extensively analyzed the discovered self-organized agents by proposing a battery of quantitative and qualitative tests to characterize the robustness of the discovered agents. These tests demonstrated that the discovered sensorimotor agents not only exhibit individuality and locomotion, but they are also capable of strong adaptivity and generalization to out-of-distribution perturbations (Section 7.4). Finally, we have introduced Flow Lenia, an extension of the original Lenia system which opens many interesting perspectives toward open-ended evolution (OEE) both for defining physical constraints in the environment (hence opportunities for emergence) and for having diversity and evolutionary activity emerge *within* the CA (Section 7.5).

Something noteworthy to reiterate in closing this chapter, is that for all the discovered sensorimotor behaviors presented in the Lenia system the computation of decision is done at the macro (group) level, showing how a group of simple identical entities can make "decision" and "sense" at the macro scale through local interactions only, and without a clear pre-existing notion of body/sensor/actuator. Seeing the discovered agents, it can even be hard to believe that they are in fact made of tiny parts all behaving under the same rules.

Lot of exciting roads remain to be taken in order to fully capture the value of complex self-organized systems such as (Flow) Lenia, and for the development of automated (AI-based) tools to assist us on that road.

A major path to explore is about the detection of agency and cognition in such systems where everything is emergent with no predefined notion of

individuals. Several of the analyses we make in this work are empirical estimations or subjective. Information theoretical measures of concepts like individuality, autonomy and agency [263, 274, 285] might offer useful tools to that end, although they remain difficultly applicable in practice.

Another interesting path to explore is about the engineering of subparts of the environmental dynamics with functional constraints. Environment design is poorly addressed and quite challenging in cellular automata systems, and more advanced environment design strategies might be crucial to study the emergence of agency and cognition in those systems [268, 284]. For instance, introducing the need to develop some kind of memory to anticipate future perturbations might enable the search for more advanced agent behaviors such as basic forms of learning. Beyond individual capabilities, we could even wonder under what conditions one could observe the emergence of an open-ended evolutionary process [76] directly in the environment, without any outer algorithm, resulting in the emergence of agents with increasingly complex behaviors. This would be like building the physical rules of an "Universe" and letting agency and evolution emerge from the interactions between parts. With the Flow Lenia formulation, we have shown how the integration of the update rule parameters within the CA dynamics enable the coexistence of multiple species within the same simulation. Such a feature represent an important step towards the design of emergent microcosms [286] in which could emerge intrinsic, maybe open-ended, evolutionary processes through inter-species interactions. However, demonstrating (and evaluating) the emergence of intrinsic evolutionary processes within such system appears as a difficult task. To achieve this, we might need to use an optimization process to evolve all the environmental rules instead of pre-specifying some of them by hand. More broadly, having more systematic ways to generate environmental rules could take us closer to the fundamental scientific quest of designing open-ended artificial systems with forms of functional life and agency "as it could be".

Despite those fundamental scientific questions, future work might also consider broader applications of this work for biology and AI.

In biology, inferring low-level rules to control complex system-level behaviors is a key problem in regenerative medicine and synthetic bio-engineering[24] [13, 14]. In this regard, cellular automata offer an interesting framework to model, understand and control the emergence of growth, form and function in self-organizing systems. However, they remain abstract models: entities in the CA exist on a predefined grid topology whereas physical entities have continuous position and speed ; states in the CA are well-defined whereas it is not clear where and how information is processed in living organisms; rules in the CA operate at a predetermined scale whereas real-world processes operate at nested and interconnected scales.

24: In the next chapter, we investigate these questions in biological models of gene regulatory networks

In AI, with the recent rise of web-deployed machine-learning models including large language models [287, 288], we are also faced with an increasing blurring of boundaries between the AI and the rest of the "environment" (human end-users and the web itself). It is hence central to understand how agency and cognition might arise in those systems, how we can detect it and how we can interact with them despite the extremely large input and behavioral spaces involved. In this regard we

believe that abstract environments like the one considered in this work be useful to better inform the debate in much bigger models, as they are rich enough to support emergent agential behaviors while simple enough to study those questions explicitely. Far from trivial, transferring insights from the considered artificial systems to real biological systems or to very large AI systems is an exciting area of research with a potential broad range of medical and societal applications [16, 61].

Revealing Diverse Behavioral Competencies of Gene Regulatory Networks

8.

What is the aim of this chapter? As a second applicative use case, this chapter demonstrates how the curiosity-driven exploration algorithms, even in their simplest version, can also be used to assist biologists mapping the space of possible behaviors of gene regulatory networks (GRNs). We discuss the several implications that the discovered "behavioral catalogs" can in turn have for fundamental research questions in basal cognition, and for practical applications in biomedicine and bioengineering.

How is this chapter organized? In Section 8.1, we motivate and provide some background on the problem. In Section 8.2 we formalize a theoretical perspective on GRNs as *agents* navigating a *problem space*. In Section 8.3, we introduce and showcase the effectiveness of the proposed curiosity-driven exploration algorithms at revealing the spectrum of reachable states that GRNs can exhibit via *minimal* and *non-genetic* interventions. Then, in Section 8.4, we further characterize the *robustness* of the discovered behaviors via a battery of empirical tests. Finally, in Section 8.5, we present preliminary experiments regarding potential applications and *reuses* of these discoveries for biological research.

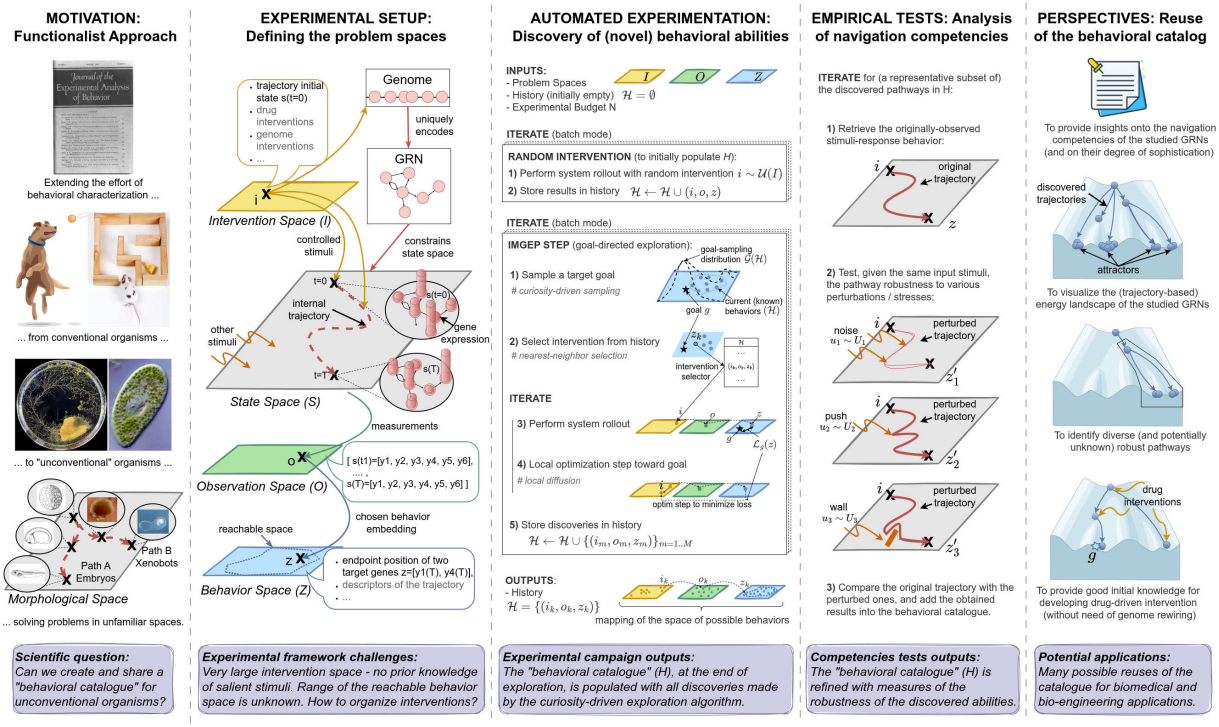


Figure 8.1.: Overview of the proposed framework. (a) MOTIVATION: To better understand navigation competencies in unconventional organisms solving problems in unconventional spaces, it is essential to construct comprehensive "behavioral catalogs" for these novel entities, which in turn requires sophisticated exploration methods to discover the extent of possible behaviors. Images are taken from [66, 289–293]. (b) EXPERIMENTAL DESIGNS: We formalize GRN behavior as a navigation task and propose to investigate it by defining abstract and observer-dependent "problem spaces" that we use to organize the observed biological behaviors and their exploration. (c) AUTOMATED EXPERIMENTATION: Pseudo-code of the curiosity-driven goal exploration process that we use to automate the discovery of behavioral abilities that GRNs can exhibit in behavior space. (d) EMPIRICAL TESTS: We use a battery of empirical tests to identify the robust goal states of the systems, i.e. the one that can be attained under a wide variety of perturbations. (e) PERSPECTIVES: We explore several potential reuses of the discovered "behavioral catalogs" across evolutionary biology, biomedicine and bioengineering contexts.

8.1. Introduction

Developing methods to recognize, map, predict, and control the complex, context-sensitive behavior of chemical and genetic networks is an essential frontier of research in science and engineering. These systems, such as gene regulatory networks and protein pathways, are known to be instructive drivers of embryogenesis, cell behavior, and complex physiology [294–296]. Understanding the control properties of these systems is critical not only for the study of evolutionary developmental biology [297–301], but also for comprehending and intervening in various disease states, including cancer [302–304], and for the construction of novel synthetic biologicals in bioengineering contexts [305–309].

Thus, much work has gone into mathematical modeling and computational inference of both protein pathways and gene regulatory network models [310–313], resulting in the development of large collections of publicly-available models such as the Biomodels database [314, 315]. Yet, despite the wealth of available models, scientists still largely lack an effective understanding of the range of possible behaviors that these models can exhibit under different initial conditions and environmental stimuli, and are in search of systematic methods to reveal and optimize those behaviors via external interventions. The full extent of the computational and control properties of such networks are not yet well-understood; while dynamical systems theory has been extensively used to characterize their behavior [316, 317], it is not known what other sets of tools might reveal and exploit interesting properties of this ubiquitous biological substrate. The field of diverse intelligence[1] has suggested that strong functional symmetries between pathway networks and neural networks could imply the existence of learning and other kinds of behavior in this unconventional substrate [318–322]. Specifically, it has been hypothesized that gene regulatory networks (GRNs) and other molecular networks could be endowed with surprising navigation competencies allowing them to robustly reach diverse homeostatic or allostatic states despite a wide range of perturbations [323–326]. Exploiting these innate competencies could provide a promising roadmap for the design of interventions in regenerative medicine and bioengineering contexts [327, 328].

1: also known as *basal cognition*

However, significant challenges remain in practice for the exploration and behavior-shaping of these innate competencies, which presents a barrier to the use of these ideas in regenerative medicine and bioengineering. Because of the non-linearity and redundancy in pathway dynamics, passive exploration strategies such as random screening are likely to either fail in uncovering the full range of potential behaviors or require time and energy beyond the available resources. Here, we formalize and investigate a view of gene regulatory networks as agents navigating a problem space. We propose a framework and automated tools, leveraging (1) curiosity-driven goal-directed exploration algorithms coming from recent advances in machine learning and (2) a battery of empirical tests inspired from behaviorist approaches, for mapping the repertoire of robust goal states that GRNs can reach within this problem space despite various perturbations. A key novelty of this work is the use of AI-based exploration tools to map the space of possible behaviors in biological networks, which opens interesting avenues for efficient mapping of

unfamiliar system behaviors, yielding transferable insights for diverse problem-solving once such a map is discovered.

The challenge of exploring and mapping spaces of complex and self-organized behaviors appears in many fields such as diverse intelligence in biological systems, minimal active matter, and robotics: many systems in these areas provide a rich space of evolved, engineered, and hybrid systems that offer many of the same fundamental problems of behavior and control regardless of specific composition or provenance [145]. These span many orders of spatio-temporal scale, from molecular assemblies to swarms of complex organisms [321, 329–331]. One set of approaches seeks to develop tools to identify the optimal level of control, ranging from physical rewiring to various methods from cybernetics and behavioral sciences, to reveal and exploit the native competencies and computational capacities of these systems [305]. Specifically, it is increasingly realized that the level of competency (and thus the appropriate level of control) often cannot be guessed by inspection of a system's components, and that its position on a spectrum ranging from passive matter to complex metacognition must be determined empirically [45, 145, 332, 333]. This is critical not only for fundamental understanding of evolution of bodies and minds [319, 334–338], but also for the design of interventions in biomedicine and synthetic morphology contexts [13, 14]. Yet, a common property in many of these systems is that it is expensive in time and energy to conduct experiments: empirical exploration needs to be made under limited resources. Thus, methods for automating *efficient* exploration and discovery of a diversity of behaviors in these spaces may be widely useful. As explained below, we will here leverage methods from developmental artificial intelligence initially designed for the specific purpose of exploring a diversity of behaviors using a limited budget of experiments.

One especially fascinating set of systems concerns cellular molecular pathways, or gene regulatory networks (GRNs). In the lab or clinic, these pathways are usually treated as simple machines, with intervention strategies focusing on rewiring their structure to achieve a desired outcome: adding or removing nodes (gene therapy), or changing connection weights (by targeting promoter sequences or protein structures) [339–342]. However, the emergent, generative nature of development and physiology ensure that it is often very hard to know which genes/proteins to modify, and how, in order to reach a complex desired system-level outcome [343]. Moreover, the responses of cells and tissues to drugs changes over time, making it even more difficult to infer specific interventions that will induce a stable improvement in pathway state in vivo. Indeed, with the exception of antibiotics and surgery, most available treatment modalities do not solve the underlying problem – they seek to mitigate symptoms, which recur (or expand) once the drug is withdrawn. Next-generation solutions, which would offer true healing (stable correction), require an understanding of the homeostatic and allostatic properties of networks with respect to how they traverse the space of transcriptional, physiological and anatomical states. An understanding of the behavior policies of networks as they dynamically navigate these problem spaces is essential for predicting what stimuli can be used to re-set their setpoints and guide them to autonomously maintain a healthy state. In the language of behavioral neuroscience, this strategy

corresponds to exploiting their native robustness, decision-making, and navigational competencies to induce predictable, long-lasting changes in functionality.

Significant challenges remain in revealing and controlling the range of behaviors that can self-organize in these cellular and molecular pathways. To characterize steady-state concentrations and responses to small perturbations, conventional methods rely on piecewise-linear approximation of the system behavior [344–348], but struggle with higher-dimensional systems or wider parameter ranges which limits their applicability [349]. Other works have proposed the porting of tools from network control theory to identify sets of control nodes that can drive the network behavior toward target steady states [350]. These methods typically exploit the network topology [350–354] or regulatory structure [355–357] to identify control strategies based on permanent knockout/activation of genes or temporary perturbations, the latter being preferable in clinical context.

However, these approaches often require prior knowledge of target attractor states or are limited to Boolean network models. Other works have explored the use of machine learning tools, such as evolutionary search [358–360] and gradient-descent optimization [361, 362], for controlling continuous ODE biomolecular networks with high-dimensional parameter spaces, mainly in the context of synthetic circuit engineering [363, 364]. While providing powerful optimization tools, these approaches tend to focus on rewiring network structure and connectivity. Moreover, the choice of a predefined fitness function and parameter range initialization is not only critical to the success of optimization [359] but largely restricts exploration of the behavior space [362].

In contrast, an alternative line of research proposes exploring and leveraging the inherent molecular mechanisms of adaptivity and robustness in cellular pathways as a promising approach for drug interventions that do not rely on genomic editing or gene therapy [323, 365]. Recently, a broad, substrate-independent behavior science perspective suggests novel properties of gene regulatory networks (GRNs) and other biological networks [203, 318]. This perspective views GRNs as agents that convert activation levels of specific genes (inputs) to those of effector genes (outputs), with intermediate nodes in between, leading to strategies for controlling network behavior based on a specific history of inputs (experience) rather than through network rewiring. Notably, the concept of training a chemical pathway using pulsed input stimuli (node activation or suppression drugs) has been formalized, and several networks have been analyzed to establish a taxonomy of memory types found in biological GRNs and pathways [366, 367].

Here, building upon recent research [325, 366, 367], we take the next step and investigate a view of gene regulatory networks as agents navigating a problem space toward target goal states with varying degrees of competency (Figure 8.1-a). We seek to implement a definition of goal that abstracts it from conventional associations with human or other advanced brains and facilitates the use of tools from cybernetics, behavior science, and control theory to understand broader aspects of biological regulation. Here we use the term "goal" state to refer to a system's steady state, which it expends effort to reach despite interventions or barriers - a definition appropriate to the study of basal (or minimal) proto-cognitive regulatory systems.. Our definition of goal does not imply "purpose"

(high-level goals where an agent has the meta-cognition to think about having goals and what they might be), and we do not attribute high-level competencies (such as re-setting one's own goals) to GRNs.

Our particular focus lies in investigating two types of navigation competencies: *versatility*, which refers to the capacity to reach diverse goal states under different interventions, and *robustness*, which refers to the ability to reach a goal state despite various perturbations. The primary scientific question of this work is: What is the repertoire of robust goal states that a GRN can actively reach through *minimal* and *non-genetic* interventions within a *navigation task* context, and can we develop systematic methods and automated tools to aid scientists in discovering this repertoire?

To address this question in practice, our experimental framework revolves around the definition of "problem spaces", which we use as tractable components of the GRN's overall state space (Figure 8.1-b), and on a set of methodological contributions which we organize around three sub-questions:

1. *Automated discovery of diverse behavioral abilities with autotelic curiosity search* (Figure 8.1-c): What is the range of possible goal states that GRNs can exhibit and how can we devise efficient exploration strategies to automatically identify these goal states? Defining goal states as attractor states of the underlying gene regulatory network, we show that traditional screening methods can be very inefficient in discovering the range of possible goal states. To address this, we propose to use intrinsically-motivated goal exploration processes (IMGEP) [57, 368], a recent family of diversity-driven machine learning approaches also known as *autotelic curiosity search* which was recently shown to form a useful discovery assistant for revealing the behavioral diversity of unfamiliar systems such as chemical oil-droplet systems[64], physical non-equilibrium systems [139] and models of continuous cellular automata [•1, •3, •5].
2. *Evaluation of the navigation competencies* (Figure 8.1-d): How competent is the GRN, in terms of robustness to perturbations, in attaining the diverse previously-identified goal states? Prior studies have offered definitions of robustness in biological networks, characterized as the degree of variation in functionality [369] or phenotypic trait [370] under specific environmental or genetic changes. However, these studies often consider a predefined functionality and random perturbations in network parameters [360, 371, 372] or specific gene knockouts [373]. Environmental perturbations on the other hand are often limited to random variations in initial conditions within a predefined range [349, 374]. Here, inspired from behaviorist approaches, we test hypotheses about non-genetic resistance with respect to various navigation competencies that living agents often exhibit, and that do not require structural changes of network properties or connectivity. Those tests assess the system's ability to maintain robustness despite various perturbations encountered during traversal, including developmental noise in gene expression levels, sudden "pushes" within transcriptional space, and the energy barriers or "walls" acting as force fields in the environment.
3. *Potential reuses of the discovered "behavioral catalog" and framework* (Figure 8.1-e): Can the constructed behavioral catalogs be useful for fundamental research and practical therapeutic applications,

and can the framework be easily applied to other systems and problem spaces? We propose that the discovered competencies may provide valuable insights for understanding evolvability and developmental robustness, and provide a fertile source for the design of interventions in biomedicine and synthetic morphology contexts. We also suggest that the framework and automated tools, which are observer-focused and substrate-independent, could be transposed to other systems and problem spaces.

The overall framework is summarized in Figure 8.1. Applying it on a database of 30 continuous (ODE) models from the Biomodels website, consisting of a total of 432 systems defined as GRN model-behavior space tuples, revealed several interesting insights. First, results suggested that most of the surveyed systems are capable of reaching a surprisingly wide spectrum of steady states depending on their initial state. Interestingly, random screening strategies were not able to reveal this diversity of reachable states (or at least not in a sample efficient way), confirming the need for more advanced exploration strategies like curiosity search. Secondly, among the discovered steady states, we were able to identify several robust goal states i.e. ones that the system consistently reaches despite various perturbations during traversal of transcriptional space. Altogether, these findings seem to suggest that cell phenotype and functionality could be the result of a multi-step program [351] that could be flexibly and robustly reprogrammed by appropriate stimuli [45]. Finally, we demonstrate possible reuses of this "behavioral catalog" for comparing the network's competencies across different classes of organisms, as well as for the design of non-genetic drug interventions. We also explore an alternative reuse of the framework to reveal new kinds of reachable "goals" in synthetic gene networks, suggesting alternative strategies for the design of gene networks in a bioengineering context.

An interactive executable version of the paper, as well as step-by-step tutorials and notebooks can be found on the project website. The full codebase of the proposed automated experimentation pipeline is written end-to-end in JAX, a high-performance numerical computing library that we leverage for parallel experimentation and computational speedups of the ODE models time-course simulations.

(a) Interactive Paper and Tutorials

(b) Codebase

Figure 8.2.: Scan (or click on) the above QR codes for accessing the paper's companion website and github repository

8.2. Generalizing GRN Behavior as a Navigation Task

The GRNs analyzed in this study are biological pathway networks taken from the BioModels repository [314, 315]. The term "GRN" is used broadly to include protein interaction, gene regulatory, and metabolic networks. In these mathematical models, the dynamic interactions between nodes of the network (molecular species) are modeled with a system of ordinary differential equations, enabling to quantitatively simulate time-course behavior (model rollouts) and observe the dynamics of node activities over time (Figure 8.3-a). Here, following a terminology which aims to integrate concepts from dynamical complex systems with concepts from behavioral sciences, we propose to conceptualize GRN behavior as a *navigation task* (Table 8.1). Model rollouts are viewed as "trajectories"

Table 8.1.: Glossary of terms used in this paper, with the proposed isomorphism which generalizes concepts from dynamical complex systems and behavioral sciences under a common navigation task perspective.

Dynamical Systems Terminology	**Behavioral Science Terminology**	**Proposed Isomorphism**	**Navigation Task Terminology**
system: a set of interconnected elements that interact to produce emergent behavior	**organism**: a living being that responds to stimuli and adapts to its environment	Both are collections of lower-level elements that interact to produce emergent behavior and can adapt at the system level	**agent** or **GRN**
phase-space trajectory: set of states taken by the system when starting from one particular initial condition	**behavioral trajectory**: the sequence of states that an organism exhibits in response to stimuli	Both represent the sequence of states or behaviors that a system or individual experiences over time	**trajectory**
initial condition: initial state of a system's variables and parameters that condition its dynamics	**stimuli**: events that might (or might not) trigger a response in an organism	Both represent incoming variations that set a system or organism in motion	**intervention** or **perturbation**
critical parameter: a parameter or condition that, if changed, can cause a system to undergo a qualitative change or phase transition	**salient stimuli:** stimuli that are particularly relevant or meaningful to an organism, either because they are associated with reward or punishment or because they are novel or unexpected	Both represent the incoming variations that have a significant impact on a system's steady-state or organism's response	**effective intervention**
steady-state (or attractor): a stable state (or set of states), towards which the system tends to evolve over time	**observed response**: outcome or endpoint of a behavioral trajectory towards which an organism converges	Both represent the endpoint that a system or organism is moving towards	**reached endpoint** or **goal**
robust attractor: stable attractor toward which the system tends to evolve under various initial conditions and perturbations	**target goal:** it is assumed that an organism engages in a goal-directed manner when it exhibits new ways or actions to achieve a similar outcome when faced with novel circumstances	Both represent a stable endpoint or goal that the system successfully attains under various perturbations	**robust goal**
controllability: degree to which the system's dynamics (and resulting steady states) can be controlled or manipulated	**trainability:** degree to which an organism's behavior can be modified or shaped by experience or conditioning	Both represent the capacity of a system or individual to be influenced or changed by controlled interventions	**versatility**

in transcriptional space where network steady states are "goal states" (endpoints) that the "agent" (GRN) can reach with varying levels of competencies. As for living agents, these competencies may range from unstable locomotion patterns to more advanced forms of goal-directed behavior like path following, obstacle avoidance, or even forms of spatial memory and foresight. In this paper, we are particularly interested in investigating two forms of navigation competencies that we refer to as *versatility*, the capacity to reach diverse goal states under various interventions, and *robustness*, the capacity to reach a goal state despite various perturbations. Note that versatility and robustness are studied with respect to different sources of incoming environmental variation, respectively interventions and perturbations.

To investigate these competencies in practice, our experimental framework is based on the definition of "problem spaces", which include the observation space (O), behavior space (Z), intervention space (I) and perturbation space (U) as defined in Table 8.2. To be consistent with our navigation task terminology introduced in Table 8.1, we refer to a behavior z as the reached "goal state" of a GRN trajectory. However these "goals" may lie on a continuum between complete robustness and high sensitivity, and our primary interest lies in identifying *robust* goals of the system. Whereas several choices could be made for the intervention space I and perturbation space U, we intentionally consider *minimal* and *non-genetic* interventions to investigate the "native" goal states of the GRN, and *environmental obstacles* to investigate for navigation

Table 8.2.: Problem spaces used in this study

Problem Space	Generic definition	Specific definition in this study
Observation Space (O)	Space of raw observations made during the GRN model rollout to measure its state or behavior	Records node activities over time as $o = (y(0), \ldots, y(T))$, where y(t) is an n-dimensional vector (n = number of nodes) and T is the measured reaction time
Behavior Space (Z)	A projection of the observation space used by the experimenter to encode the "goal states" of a model rollout into a tractable (lower-dimensional) space	Encodes the trajectory endpoint of a model rollout. Represents a cell phenotype defined by the state values of some nodes (relevant biological markers), such that $z = (y_{i1}(T), \cdots y_{im}(T))$ (we use m=2 in this study for simplicity and visualization)
Intervention Space (I)[a]	A space where interventions represent controlled sources of incoming variation that the experimenter can exert on the GRN model rollout to drive it toward novel or targeted states	Sets the initial state $i = (y_1(0), \ldots, y_n(0))$ of a model rollout. Defined as a hyper-rectangle $I \subseteq \mathbb{R}^n$ where the boundaries are 20 times larger than the min and max values taken by the respective nodes from default initial conditions
Perturbation Space (U)	A space where perturbations represent external sources of incoming variation, used by the experimenter to characterize the robustness of a given goal state	Includes three classes of (stochastic) perturbations including noise perturbation U_n, push perturbation U_p, and wall perturbation U_w

[a] The *intervention space* I is equivalent to what we referred to as the *parameter space* Θ in the previous chapters

competencies classically observed in other living agents. Examples of simulations, interventions, and perturbations are shown in Figure 8.3.

Then, a typical analysis using our framework relies on a 2-step procedure, detailed in the subsequent sections. First, to assess the versatility of the GRN, we define an exploration strategy which is in charge of organizing the sequence of interventions $i_1, \ldots, i_N$ to drive the system toward a maximally diverse set of reachable endpoints $\{z_k \in Z\}_{k=1,N}$, while using a limited budget of experiments N. Secondly, to assess the robustness of the discovered goal states $\{z_k \in Z\}$, we conduct a battery of empirical tests to characterize their degree of sensitivity to novel perturbations, with a fixed experimental budget of P perturbations per selected behavior z. At the end of this 2-step procedure, we obtain the "behavioral catalog" (H) of the studied GRN, which includes the history of experiments $H = \{\left(i_k, o_k, z_k, \ \{\left(u_p, o_p, z_p\right), \ p = 1...P\}\right), \ \ k = 1 \ldots N\}$.

Following this framework, the behavioral catalog is constructed for a database of 30 biological networks consisting of a total of 432 systems, where a system is defined as a (GRN model, intervention space (I), behavior space (Z) tuple, as described in Appendix Section F.2 and Table F.1. These catalogs provide valuable empirical observations and insights into the navigation competencies of the studied GRNs, particularly in their ability to consistently achieve diverse goal states under various tested perturbations. Statistical analyses of the results are presented in Figure 8.6, Figure 8.8, and Figure 8.4, and specific results for the RKIP-ERK signaling pathway [375] are shown in Figure 8.3, Figure 8.5, Figure 8.7, and Figure 8.9.

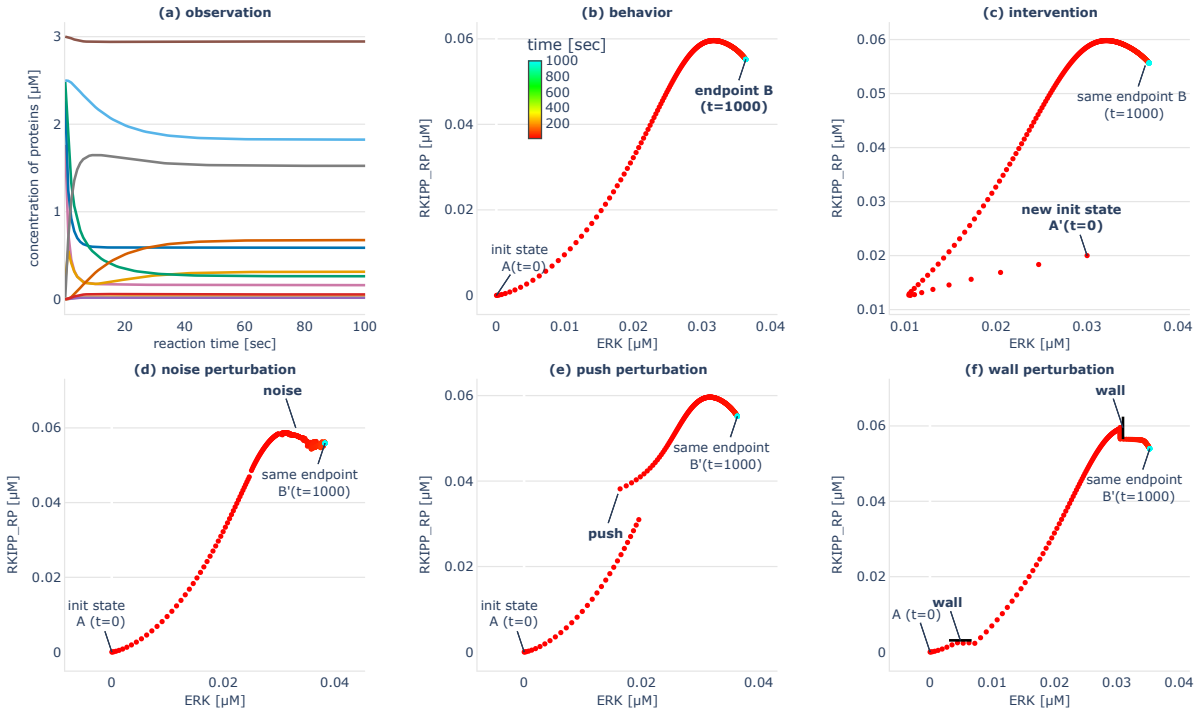


Figure 8.3.: Illustration of the experimental setup and chosen problem spaces on an example GRN model which has 10 nodes and models the influence of RKIP on the ERK Signaling Pathway [375]. (a) Time-course evolution of the different nodes $y_1, \ldots, y_{10}$ (one color per node) when starting from the default initial conditions (as provided in [375]). The observation captures the states taken through time $o = [y(t=0), \ldots, y(t=T)]$ where $y = [y_1, \ldots, y_{10}]$. (b) Corresponding trajectory in transcriptional space (phase space), for two target nodes (ERK, RKIPP_RP), from $t = 0$ (A, in red) to $T = 1000$ seconds (B, in cyan). We can see that the trajectory converges to endpoint B in less than 100 seconds, and then stay there. The behavior (or reached goal state) is the endpoint $B = [y_{ERK}(T), y_{RKIPRP}(T)]$, where T is chosen big enough to ensure convergence. (c) The intervention is setting the initial state of the system trajectory (for all nodes): $i = [y_1(t=0), ..., y_{10}(t=0)]$. (d-e) Example of perturbations used in this paper. (d) Noise perturbation, here applied to all 10 nodes every 5 secs until t=80 secs. (e) Push perturbation, here applied to the two target nodes (ERK, RKIPP_RP) at t=3 seconds. (f) Wall perturbation, also applied to the two target nodes (ERK, RKIPP_RP), here at 10% and 90% of the total distance traveled. Supplementary Figure F.2 shows examples of other possible "drug" or "genome" interventions that can be implemented in the accompanying software, as well as the possibility to perform interventions (or perturbations) in parallel using batched computations.

8.3. Curiosity Search Uncovers a Diversity of Reachable Goal States

One advantage of modeling GRN behavior within a tractable behavior space Z is that we can then deploy strategies to efficiently discover and map that space. Notably, recent diversity-driven machine learning techniques such as Novelty Search [126, 127], Quality Diversity [131, 132] and Intrinsically-Motivated Goal Exploration Processes (IMGEP) [57, 368] are explicitly designed to efficiently explore a so-called "behavior space" or "goal space" which is basically a (predefined or learned) model of the overall state space. In particular IMGEPs, which were originally developed for the learning of inverse models of highly-redundant mapping in robotics context [57], were recently shown to successfully assist discovery in complex self-organizing systems [•1, •3, 64, 139].

Here, we propose to use an IMGEP to control GRN initial states and maximize the diversity of discovered endpoints $\{z \in Z\}$ within a limited budget of N experiments. The IMGEP operates in two phases: initially, N_{int} interventions are sampled randomly from I to populate history H, then remaining interventions are generated through a goal-directed process which relies on several key internal models. Those including

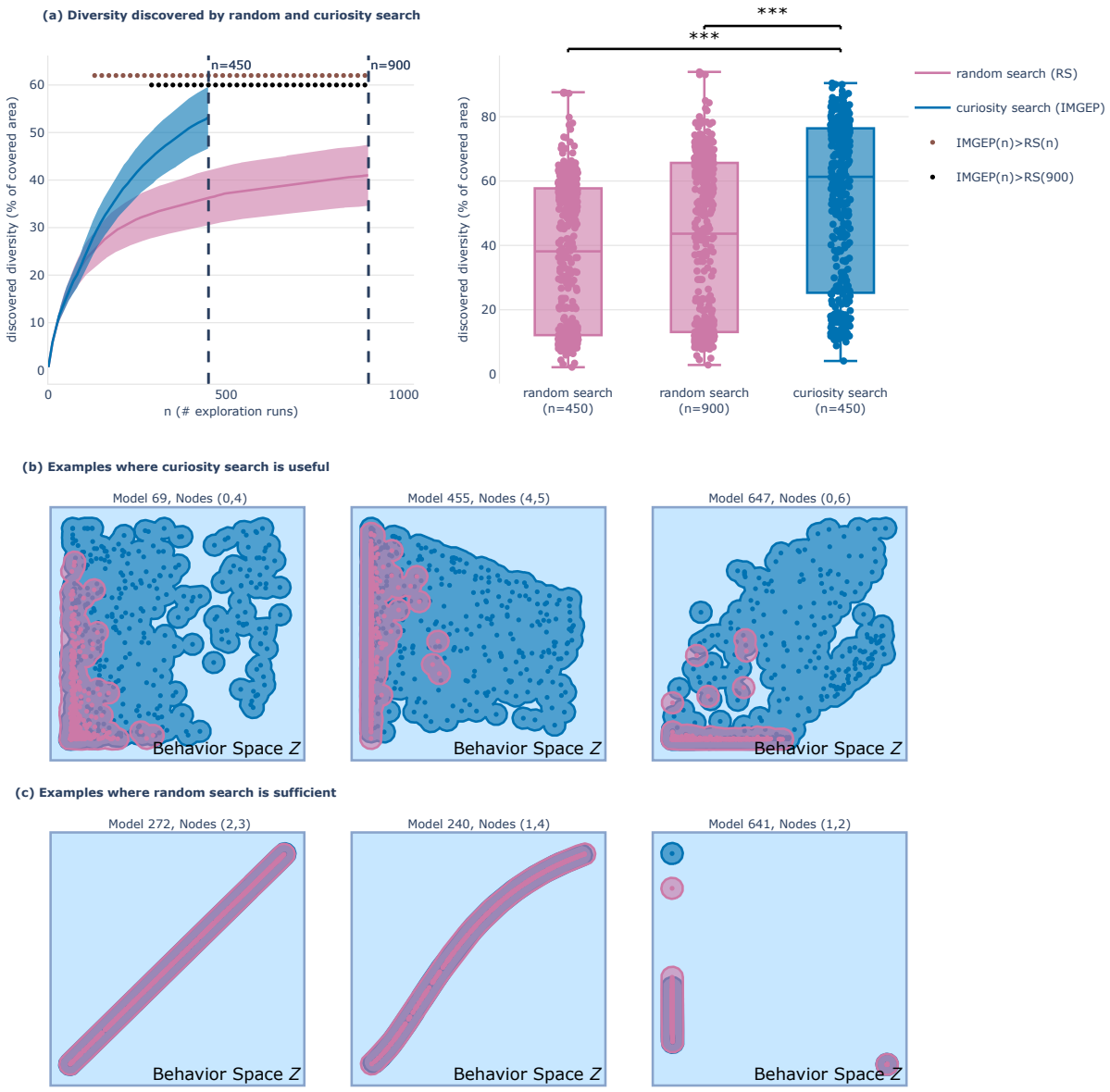


Figure 8.4.: Curiosity search uncovers a wide spectrum of reachable states in behavior space Z. (a) Diversity of endpoints discovered by random search (pink) and curiosity search (blue) for the 432 systems. Diversity is measured as the volume of the union of the set of hyperballs of radius ϵ that have for centers the discovered endpoints $\{z \in Z\}$ as depicted by the shaded area in (b-c) with $\epsilon = 0.05$. (a-left) Mean and standard deviation curves of the diversity of behaviors discovered throughout exploration (with random search having twice more experiments n=900). Dots indicate significance (p<0.05) when testing curiosity search (n) against random search (n) in brown, and against random search (n=900) in black, with a Welch's t-test. Standard deviation is divided by 4 for visibility. (a-right) Detail of the diversity obtained in the left plot for all 432 systems at n=450 and n=900, where *** indicate significance (p<0.001). (b-c) Discovered endpoints at the end of exploration (n=450) by random search (pink) and curiosity search (blue) for 6 example systems of our database. (b) Examples of systems for which curiosity search is much more sample-efficient than random search in finding a diversity of reachable states in behavior space Z. (c) Examples of systems with low-redundancy mapping $I \to Z$ such that random search in I is already quite efficient in covering behavior space Z, and curiosity search performs equivalently.

a goal-embedding module (R) that encodes observations (o) into the IMGEP goal space ($\mathcal{T}$), a goal generator module (G) that samples goals from the goal space based on intrinsic motivation incentives (e.g. to promote novelty or learning progress), and a goal-conditioned optimization policy (Π) that generates candidate intervention parameters to achieve the current goal. Given those internal models, the goal-directed phase iterates through 1) sample a target goal $g \sim G(H)$, 2) infer intervention parameters to achieve that goal $i \sim \Pi(g, H)$, 3) conduct an experiment

with the intervention i, observe the outcome o, and compute the reached goal $z = R(o)$, and 4) store the tuple (i, o, z) in history H. Here, we use the GRN behavior space Z as the IMGEP goal space $\mathcal{T} = Z$. Hence "target goal" refers to a goal sampled by IMGEP while "reached goal" refers to an actual endpoint of the GRN trajectory, discovered by IMGEP while targeting a potentially different point in Z. Throughout exploration, the IMGEP dynamically refines its Z-traversal strategy based on the knowledge acquired by its discoveries. Here we opt for a simple IMGEP variant such that the exploration process can be seen as performing novelty search in behavior space Z [134]. The pseudocode of our IMGEP pipeline is shown in Figure 8.1-c and details about the internal models are provided in Appendix Section F.2. The final outcome is a "behavioral catalog" of the GRN, containing the diverse goal states discovered by IMGEP: $H = \{(i_k, o_k, z_k)\,,\ \ k = 1 \dots N\}$.

We deploy the IMGEP, also known as "curiosity search," on all 432 systems in the biological network database. Our evaluation focuses on two related competencies: the IMGEP agent's ability to empirically reveal a diversity of reachable goal states in the (GRN, I, Z) system, referred to as "discovered diversity," and the GRN agent's competency to naturally reach diverse goal states, referred to as "versatility." The true versatility of the GRN is unknown and can only be inferred through empirical exploration and proxy metrics.

For evaluating diversity, we measure the area covered in Z by the IMGEP discoveries using the threshold-coverage metric [71] and compare it with the area covered by the diversity of a naive random screening strategy (which uniformly samples interventions in I). In Figure 8.4, the diversity discovered by the two exploration variants is shown for the 432 (GRN,I,Z) systems, where random search is given a budget of experiments (N) which is twice bigger (N=900) as the one given to the curiosity-search algorithm (N=450). Interestingly we see that, on average, at n=290 the curiosity search already significantly outperforms the final diversity achieved by random search, while only utilizing one-third of its experimental budget (N=900). Whereas we are dealing with numerical systems and our codebase allow for efficient and parallel execution, each experiment still consists of $\frac{T}{\Delta T} = 25000$ model steps, where each step integrates the ODE system. Repeating that N times for each of the 432 systems starts to be very costly, which is why having efficient exploration strategies is very valuable (and would be even more valuable when scaling the framework to larger databases). Even more critical, as illustrated in Figure 8.4-b, it seems that, for some systems, random search is not able to discover the "latent" regions revealed by the IMGEP in Z, or it would need an extremely large budget of experiments. On the other hand, as illustrated in Figure 8.4-c, there are some systems for which random search is already quite efficient in revealing diverse behaviors in Z, and for which IMGEP performs equivalently.

In fact, the goal-directed strategy of the IMGEP is particularly beneficial for (GRN,I,Z) systems with high nonlinearity or redundancy in their ($I \rightarrow Z$) mapping, as seen in Figure 8.5 and studied in robotics contexts [71]. Redundancy implies that many interventions in I lead to similar effects in Z, as illustrated in Figure 8.3 where various interventions and perturbations converge to the same endpoint. In these systems, random search will preferentially discover points in areas of high redundancy in Z

whereas the IMGEP, whose exploration is directed uniformly in goal space, should cover different levels of redundancy equally. In general, when dealing with large intervention spaces and limited experimental budgets, curiosity search can be particularly useful for efficiently navigating Z-space.

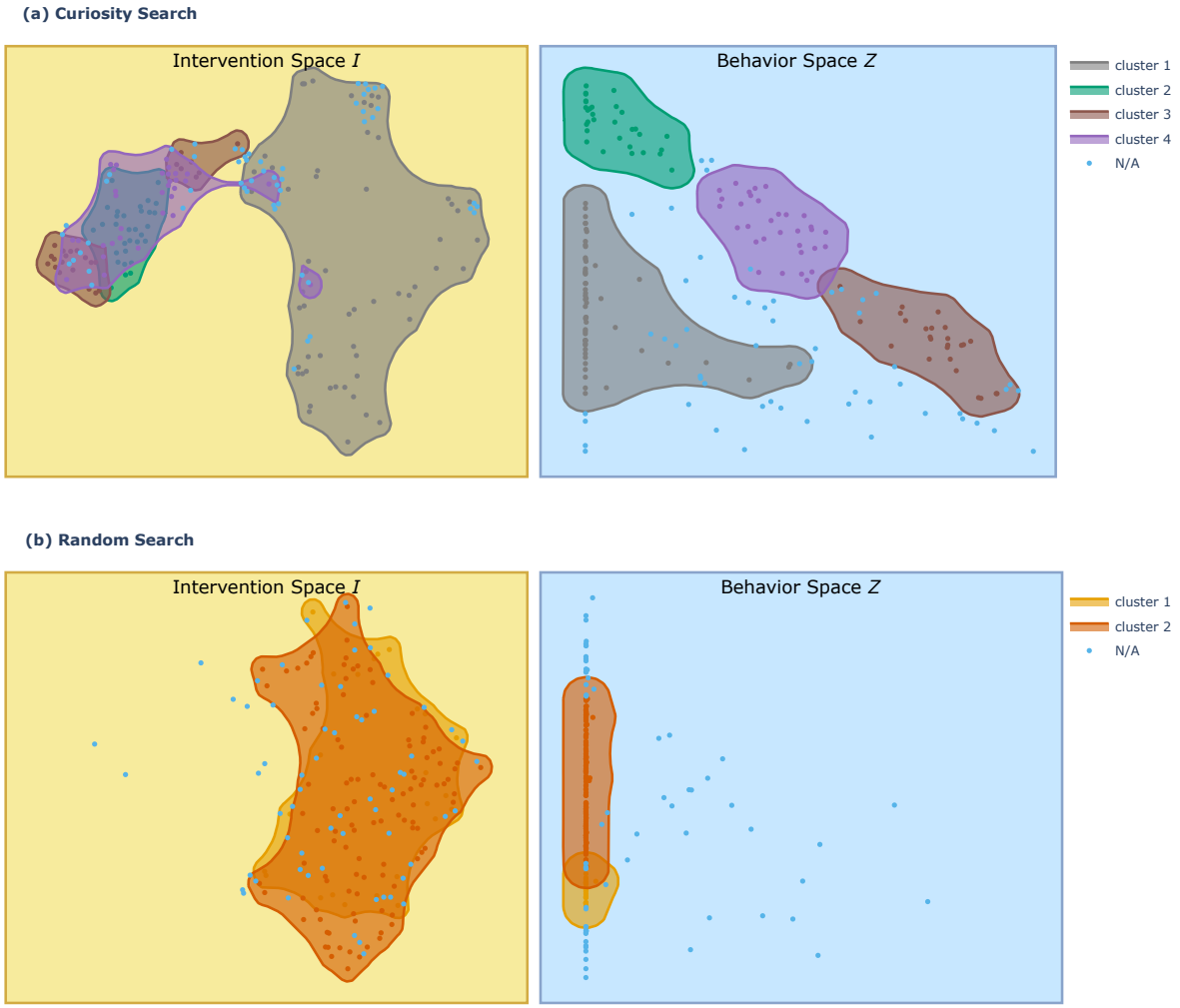


Figure 8.5.: Illustration of the non linearity and redundancy of the $I \rightarrow Z$ mapping, and of the interest of using goal-directed exploration strategies. Plot shows the reachable points discovered by curiosity search (a) and by random search (b) in the behavior space Z and their corresponding starting points in the intervention space I, for the RKIP-ERK signaling pathway system [375]. The intervention space is 10-dimensional, and here we show the TSNE reduction in 2D. We apply HDBSCAN clustering [376] on the points discovered in Z, which produced the 4 clusters for curiosity search (displayed in gray, green, purple and orange; non-assigned points are displayed in light blue) and 2 clusters for random search (displayed in light and dark orange). We then visualize where those regions in behavior space mapped back in the intervention space, by applying the same coloring. (a) Looking at the curiosity search discoveries, we can see the non-linearity of the $I \rightarrow Z$ mapping, where small regions of intervention space can map to large regions of the behavior space (like the orange area) and reversely (gray area). We can also see the redundancy of the behavior space which is clearly concentrated in the left border of the space (ERK close to zero) which can seemingly be reached from very large portions of the intervention space (gray area). (b) Looking at random search discoveries, we can understand that it is very inefficient as it spends most of its exploration budget in the region of intervention space that converges to the left border in Z, and fails to explore the orange, purple and green regions discovered by curiosity search which seemingly lead to the more novelty in Z.

Finally, as the IMGEP efficiently drives the GRN into diverse goal states with minimal interventions, we propose that the diversity achieved by the IMGEP can serve as a good proxy metric of the GRN versatility. Notably, analysis of example systems in Figure 8.4 reveals that many GRNs can reach a broad spectrum of steady states. Whereas our database is limited to certain systems (see Appendix Section F.2) and might not be

representative of all biological pathways, this observation underlines the existence of various phenotypes that can be realized. It also highlights the critical importance of identifying salient interventions that can effectively control cellular states within this spectrum of possibilities, notably as many cancer types are due to epigenetically non-identical cells [377].

8.4. Empirical Tests Reveal Robust Navigation Competencies

We are then interested in characterizing the degree of robustness of the previously-discovered "goal states" in order to identify the ones that the GRN can consistently be reached by the GRN despite encountering various perturbations. Whereas many studies have proposed rigorous analysis of the "robustness" of biological networks [369, 370], the generated perturbations often target variations in the regulatory rules (i.e. variations at the hardware level) and variations are often sampled independently (and prior) to observations of the GRN dynamical behaviors [349, 360, 371, 372, 374, 378]. Here instead, we propose to conduct a battery of empirical tests that draw inspiration from classical "displacement experiments" [379, 380] and "barrier experiments" [381] commonly used in behavioral sciences to assess the navigation competencies of various animals. As illustrated in Figure 8.3, we consider *environmental* perturbations that perturb the GRN trajectory with 1) various degrees of noise in the gene expression levels, 2) sudden "pushes" during the GRN traversal of transcriptional space, and 3) energy barriers or "walls" acting as force fields that constrain the GRN traversal. Importantly, those perturbations are *conditioned* on the observed behavior of the GRN. The magnitude of the noise and of the pushes is scaled proportionally to the extent of the observed trajectories, and the walls are generated in locations that the GRN would "naturally" visit without the induced perturbation. While intuitive from a behaviorist point of view, where one would adapt experimentation when testing animals in different contexts (e.g. to study homing behavior of an ant and of a sea turtle, or of an ant in food deprivation and in reproduction phase) [382], robustness studies in systems biology often neglect those aspects. We propose that a behaviorist lens on robustness can help understanding forms of non-genetic resistance in transcriptional space, which is crucial for the development of therapeutic strategies [377].

To assess the degree of robustness of the discovered goal states, our evaluation procedure is the following. For each (GRN, I,Z) system of the database, we retrieve a representative set of trajectories previously discovered using the curiosity-search algorithm and subject these trajectories to $P = s \times r$ perturbations conditioned on the GRN goal-reaching trajectory $i \rightarrow z$ prior perturbation. Here, s represents the different perturbation distributions which correspond to various "tests" and "levels of difficulty" (e.g. noise magnitude and frequency, number of walls, etc.) and r is the number of (stochastic) perturbations sampled per family. The pseudocode is illustrated in Figure 8.1-c and details about the different family of perturbations are provided in Appendix Section F.2. At the end of this process, the behavioral catalog is augmented with the perturbed trajectories $H = \{\big(i_k, o_k, z_k,\ \{(u_p, o_p, z_p),\ p = 1 \dots P\}\big),\ \ k = 1 \dots K\}$.

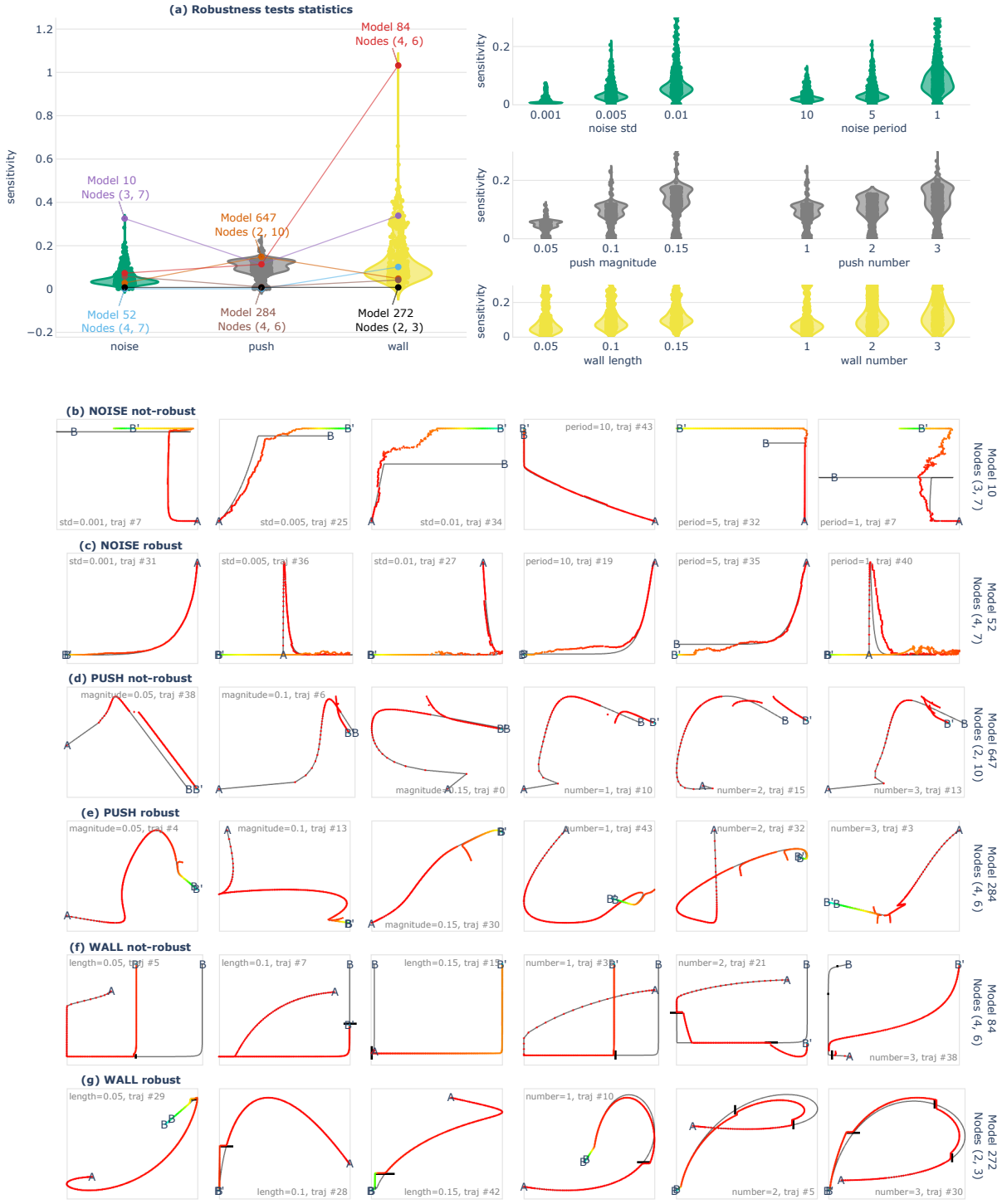


Figure 8.6.: Identification of robust traversal strategies in transcriptional space. (a) Violin plots show, for each of the 432 systems (one point per system), the median sensitivity (over the K representative goal states) to the noise (green), push (gray) and wall (yellow) perturbation families. Violin plots on the right detail the median sensitivity for the 18 sub-families. (b-g) Each row provides examples of perturbed trajectories of either extremely-robust or extremely-sensitive example (GRN, Z) system (on average over the K goal states) for the three families of perturbations, as shown by annotations in (a). For instance, the first row (b) shows perturbed trajectories of the (model #10, nodes (3,7)) system which has the highest sensitivity to noise whereas the last row (g) shows trajectories of the (model #272, nodes (2,3)) system which has a nearly perfect robustness to walls. Each image contains an example trajectory for a given (i, u), and one u per sub-family is shown per column. For instance in the first row (b), the trajectories are perturbed with the different sub-families of noise ($\sigma_n \in [0.001, 0.005, 0.1]$, $p_n \in [10, 5, 1]$) which can be seen as various levels of difficulty. For each trajectory we annotate the starting position (A), endpoint prior perturbation (B), and endpoint after perturbation (B'), and show the original trajectory in black. The perturbed trajectory is shown in colorscale (from red at t=0 to cyan at t=3000 secs).

Figure 8.6a: (b) Except for few cases (trajectory #43), the system (model #10, nodes (3,7)) system is not robust to noise as its trajectories are easily deviated from the original endpoint. (c) The (model #52, nodes (4,7)) system however, except for rare cases (trajectory #35), consistently reaches its original target despite encountering various amounts of noise. Interestingly, trajectories #36 and #40 consistently follows a complex up->right-down->left path, despite the induced noise. (d) The (model #647, nodes (2,10)) system, except for few cases (trajectory #0), is typically deviated from its original trajectory when being pushed away. Interestingly though, it seems to follow similar (parallel) trajectories. (e) The (model #284, nodes (4,6)) system, is an example of an extremely robust system which, despite many push configurations (in magnitude and number), consistently returns to its original trajectory. Interestingly, the trajectories of this system are relatively complex with several loops and detours. (f) The (model #84, nodes (4,6)) system is not very robust to walls, and typically deviates or blocked when it encounters a wall. (g) The (model #272, nodes (2,3)) system is another example of an extremely robust system which, despite many wall configurations (in length and number), consistently returns to its original path. Once again interestingly, the trajectories of this system are relatively complex with several loops and detours.

As the use of "spaces" comes with the notion of similarity and distance, we can then easily evaluate the *sensitivity* of a goal state z with respect to a set of perturbation $\{u_p, p = 1 \dots P\}$ as the average distance in behavior space Z between the original trajectory endpoint z and the perturbed trajectories endpoints $\{z_p\}$. Here our distance is simply the Euclidean distance, normalized by the extent of the trajectory prior perturbation in Z. We can then identify the so-called "robust goals" of the systems as the ones that have the lower sensitivity to perturbations. These sensitivity analyses can be useful in two important ways. On the one hand, they allow us to quickly identify the "extreme" examples of robustness, both at the system-level and at the goal-level, providing several insights into the degree of "competencies" that some biological networks might exhibit in their relative space (Figure 8.6). On the other hand, these analyses also allow us to map the *heterogeneity* of cellular responses and to better understand how non-genetic perturbations might modulate the *landscape* of reachable cell phenotypes (Figure 8.7).

Figure 8.6 shows the median sensitivity, over the representative goal states, for the 432 systems of our database and for the noise, push and wall perturbations families (as well as for the s=18 sub-families which correspond to varying degrees of perturbations). Overall, even though we observe varying degrees of sensitivity between systems (and between magnitudes of perturbations, which is expected), one first and interesting observation is that the median sensitivity remains relatively low, suggesting that GRNs could not only exhibit versatility (with respect to the considered interventions) but also robustness (with respect to the considered perturbations). In fact, looking at the "extreme" examples, we can identify quite impressive examples of complex and yet highly-robust space traversal strategies, with non-linear trajectories exhibiting many "detours" and "loops" but yet consistently reaching the same endpoint despite several pushes (Figure 8.6-e) or walls (Figure 8.6-g) on the way.

Figure 8.7 shows how the constructed catalog H can be used to generate the energy landscape of the studied system. In biology, landscape formalisms have been used to comprehend the underlying dynamics of several systems, such as cell cycles and cell differentiation [384, 385]. It is believed that such system-level visualizations could be particularly useful to apprehend non-genetic heterogeneity in the context of cancer treatment and stem cell differentiation [377, 383]. A recent landscape-generation method only proposes to approximate the pseudopotential energy through simulation trajectories obtained throughout exploration of the system [383], making it a widely applicable method which we can directly apply here. However, the paper relied on Monte Carlo simulation to generate the trajectories. Due to the previously mentioned non-linearity and redundancy of the $I \rightarrow Z$ mapping, this can lead to poor estimation

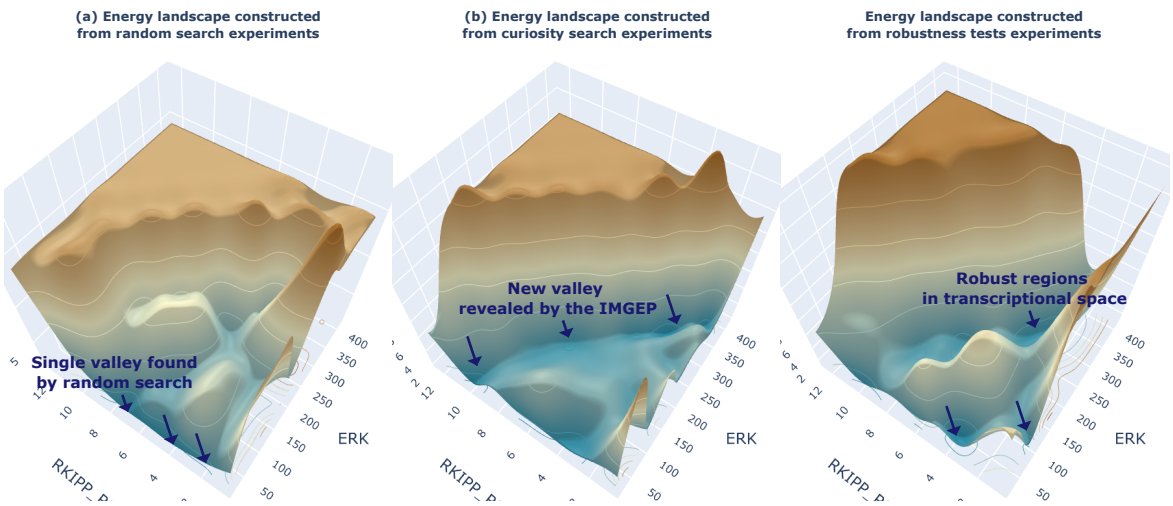


Figure 8.7.: Energy landscape visualization based on the trajectory-based landscape generation method [383], and constructed from different set of GRN trajectories, respectively trajectories generated (a) by the random search exploration, (b) by the curiosity-driven exploration, and (c) by the robustness tests experiments.

of the overall energy landscape (Figure 8.7-a). Instead, when generating the landscape from the trajectories discovered by our curiosity search exploration, we are able to reveal a new and wide "valley" of reachable states (Figure 8.7-b). Interestingly, the landscape-generation method can also be used to better comprehend the effect of external cues on the gene regulatory network, by visualizing how much they deform the energy landscape for instance leading to new shaped valleys (Figure 8.7-c). For the example system RKIP-ERK pathway [375], results highlighted a specific region of behavior space (with low RKIP and high ERK activation levels) that seems to be particularly robust, i.e. consistently reached by the GRN from certain initial conditions, and that might be associated with tumor development [386].

8.5. Possible Reuses of the Discoveries in Biology

Our framework generated a catalog of stimuli, responses, and navigation test situations for the different GRN models contained in our database. Creating and sharing such a "behavioral catalog" with the scientific community is possibly one of the more exciting aspects of the work with new organisms. Furnished with such an empirically based dataset and detailed observations, one can 1) conduct statistical analysis across the population of studied organisms to inform fundamental research questions and 2) reuse the acquired knowledge to design specific behavior-shaping experiments in organisms of interest. As our framework focuses on observable behavior and is agnostic about the internal construction of the organism, another exciting perspective is to deploy it to different problem spaces and other classes of natural, chimeric or synthetic organisms. This section illustrates preliminary experiments along those three types of reuse.

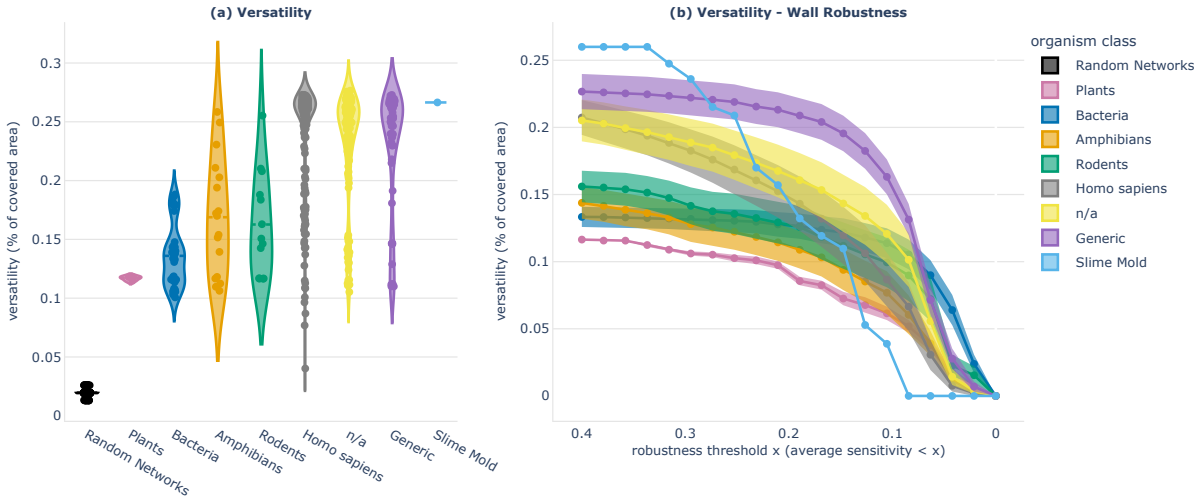


Figure 8.8.: Analysis and comparison of the degree of sophistication, in terms of versatility and robustness, between different classes of GRN. We categorize the GRNs by class of organism they belong to: plant, bacteria, slime mold, amphibian, rodent, homo sapiens, or generic. "n/a" refers to networks for which this information is not available. (a) Violin plots show the versatility of the 432 systems (one point per system) for each class. Versatility of one system is measured as the area covered by all the goal states discovered by curiosity search (equivalent to what we call diversity in Figure 8.4). (b) Trade-off (aka Pareto) mean and standard deviation curves that represent the trade-off among versatility and wall robustness performances as taken by the different classes of GRNs (standard deviation is divided by 4 for visibility). For each system, versatility (y-value) is measured as the area covered by the set of robustly achieved goal states, where the criterion of goal-achievement is a binary which tests whether the goal-reaching sensitivity (on average overall wall perturbations) is below a certain threshold (x-values). Violin plots in (a) are ordered in ascending order according to the class mean y-value at x=0.4 in (b).

8.5.1. For the study of developmental robustness

A first use-case we explore is to conduct statistical analysis to categorize versatility and robustness in the surveyed networks on the basis of species in evolutionary strata. We consider seven categories, namely, plant, bacteria, slime mold, amphibian, rodent, homo sapiens, or generic. Here, generic corresponds to the networks not associated with any species but related to generalized biological processes. Please note that the surveyed database is relatively small with respect to the wealth of available models and biological pathways, so we can hardly claim that these results represent the true distribution of competencies across these organism categories. Still, as shown in Figure 8.8, results suggested interesting patterns.

First, on average, generic and Homo sapiens GRNs exhibit higher versatility (mean 0.228 and 0.238) compared to rodent and amphibian GRNs (mean 0.163 and 0.169), which in turn show higher versatility than bacteria and plant GRNs (mean 0.136 and 0.117). These findings are particularly intriguing in the context of the recently-formulated hypothesis of multi-scale competency architecture [45]: could the observed variation in versatility among different classes of GRNs contribute to the degree of versatility observed at higher-level scales? Collecting such experimental data for broader classes of organisms and GRNs will be crucial to understand how competencies at the molecular scale can impact the overall functionality and adaptability of organisms at higher scales, and to understand how evolution might have exploited this modular architecture for shaping the observed adaptivity and reprogrammability of biological systems.

Secondly, when comparing with the versatility of random networks

(in black), generated to follow the same distributions of network size and connectivity as biological networks (as proposed in [367], see Appendix Section F.2), we observe that random network versatility is much lower (<0.026) than the versatility observed in biological networks. Once again, it is difficult to draw strong conclusions as the gene circuit model used for the random networks is relatively limited, whilst generic and studied across a range of biological contexts [387–390], and it will be interesting to scale the comparison to a broader and more complex range of ODE-based random models. Still, these findings hint that versatility prevalence might be a strong invariant of biological intelligence shaped by evolutionary processes.

Finally, we categorize the versatility-robustness tradeoff in the different categories of organisms. The idea is to compare the GRN competencies to robustly achieve diverse goal states, for different robustness thresholds. In Figure 8.8b, we plot the mean and standard deviation pareto curves for the different categories of organisms and observe that, in average, the pareto-optimal solutions are mostly achieved by generic cell GRNs, even though bacteria GRNs can robustly reach more goal states for exigent robustness criteria (high x-values). The slime mold GRN can reach highly diverse goal states but the tradeoff quickly drops with wall perturbations, and there is only one system in our database belonging to this category so results might be not representative. Once again, those results are very interesting as generic cells GRNs are a building block that has been extensively reused by evolution across several organisms and contexts, bacteria have evolved to be very resistant (e.g. to antibiotics), and slime molds are a unicellular organism well known for its diverse capabilities, especially navigational ones [391–394].

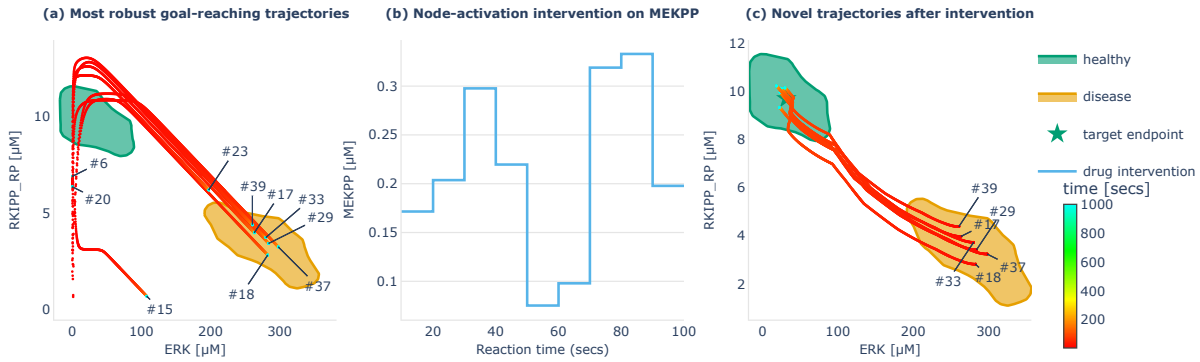


Figure 8.9.: Identification of stimuli-based stepwise intervention triggering robust re-set of disease states into healthy physiological states. (a) The 10 most robust identified goal states (average sensitivity <0.05) and the corresponding reaching trajectories are displayed for the example RKIP-ERK signaling pathway [375]. We can see that most of them converge toward attractors in the "disease" region (orange). (b) Discovered stepwise stimuli intervention on MEKPP which we apply on states stuck in the "disease" region for 100 seconds. (c) The discovered intervention successfully brings back all points from the "disease" region closer to the target endpoint in the "healthy" region, and this under various tested perturbations (as shown in Supplementary Figure F.4). The optimization procedure that led to the discovery of this intervention is described in the main text.

8.5.2. For the development of therapeutic interventions

Understanding forms of non-genetic resistance and non-genetic heterogeneity is crucial across a wide range of cancer and treatment contexts [377]. Here, we illustrate how the constructed behavioral catalog can provide a fertile source for the design of therapeutic strategies, notably in

the context of network control, using again the example of the RKIP-ERK signaling pathway [375]. In Figure 8.5, we saw that curiosity search revealed four clusters of reachable steady states for this system. From a clinical perspective, one might denote the green cluster as "healthy" region of behavior space and the orange cluster as "disease" region of the behavior space, as high levels of ERK and low-levels of RKIP are often linked to tumor development [386]. In Figure 8.9-a, we plot those two clusters as well as the 10 more robust goal-reaching behaviors from the behavioral catalog of this system, i.e. the goal states with the lower average sensitivity to the induced perturbations. We see that 6 out of the 10 more robust trajectories end up in the "disease" region, suggesting that certain configurations of initial state are very likely to reach that region despite chemical blockers (here pushes, walls, and noise), which was also visible on the system's energy landscape in Figure 8.7-c. Looking at the six trajectories, it seems that they all follow similar patterns where RKIP activation level increases past a certain threshold, and only then converge toward the disease region. This might already provide an interesting biomarker for prediction of tumor development, but what we are really interested here is to build upon that knowledge to develop stimuli-based interventions allowing to re-set the GRN setpoints from the identified "disease" steady states back to steady states within the identified "healthy" region . To do so, we define a parameterized stimuli-based intervention and a performance function, and search for parameters that optimize this performance. For the intervention function, we use a piecewise constant function that determines which nodes to intervene on (here MEKPP), when to apply the intervention (here every 10 seconds for 100 seconds), and with what amplitude (which are the parameters that we are seeking to optimize). The choice of the intervention function, which is arbitrary in this example, would typically depend on the experimental constraints, *e.g.* which nodes can be targeted with drugs and at which precision. For the performance function, we define the centroid of the "healthy" region as the target setpoint and compute performance of the stepwise intervention as the average distance of the novel setpoints (after intervention when starting from the 6 disease setpoints) to the target setpoint, and under a distribution of stochastic walls, pushes and noise perturbations. Hence a successful intervention should re-set the disease setpoints to healthy setpoints for all discovered disease states and robustly across the various tested perturbations. For optimization, we simply perform random search as this was sufficient here to discover one intervention (as shown in Figure 8.9-b) that successfully reset the setpoints (as shown in Figure 8.9-c) under various tested perturbations (as shown in Supplementary Figure F.4). Here random search was sufficient to find a successful intervention, but more advanced optimization strategies like evolutionary algorithms or stochastic gradient descent could be envisaged for harder problems. Overall, mapping the "latent" behavioral abilities of GRNs in healthy physiology and disease states may have important implications for the identification of robust stimuli-based interventions that focus on behavior shaping instead of micromanaging all molecular states, and that can be exploited in therapeutic contexts.

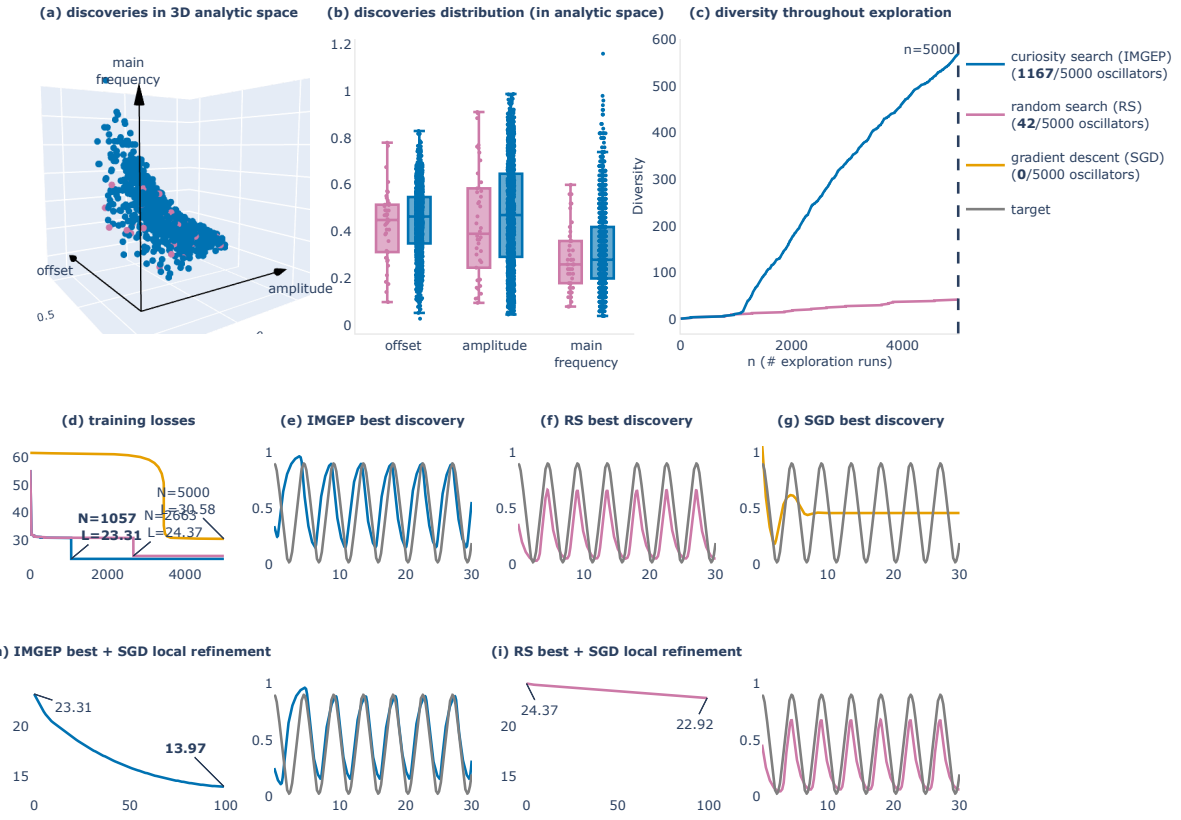


Figure 8.10.: Comparison of three alternative strategies for the design of oscillator circuits: curiosity search (blue), random search (pink), and gradient descent (orange). (a-c) Given a budget of 5000 experiments, curiosity search is able to find 1167 oscillator circuits (ones showing sustained oscillations), whereas random search only finds 42 oscillators and gradient descent does not discover any (starting from a single random initialization). (a) 3D scatter plot of the 42 random search discoveries (pink) and 1167 curiosity search ones (blue) in the (amplitude, main frequency, offset) analytic behavior space. (b) Box plots projecting points from the 3D scatter plot into the respective (amplitude, main frequency, offset) axes. (c) Diversity discovered throughout exploration, where diversity is measured with a binning-based space coverage metric (20 bins per dimension). (d) Evolution of the training loss L for the three exploration strategies. (e-f-g) Corresponding best discoveries (for which L is minimal) for the three exploration strategies. (h-i) Local training loss and resulting finetuning of the best discoveries with gradient descent.

8.5.3. As alternative strategy to gene circuit engineering

The final type of reuse we explore is not a direct reuse of the constructed behavioral catalogs, but rather a reuse of the proposed automated tools to reveal different kinds of behaviors in a bioengineering context. A common problem in synthetic biology is to optimize the configuration and parameters of a gene model network to optimally perform a desired functionality, also known as gene circuit engineering [364]. Recent approaches rely on optimization-driven machine learning strategies, such as evolutionary algorithms and stochastic gradient descent. However, choosing the right loss function and parameter initialization for these optimization methods is a well-known problem in machine learning. These issues can lead to optimization algorithms getting trapped in local minima within the complex landscape of possibilities. In response to these challenges, we propose to investigate whether the curiosity-driven exploration strategy can be employed as an alternative (diversity-driven) strategy. Whereas traditionally-employed for exploratory purposes, these exploration strategies were also shown to facilitate the resolution of external, pre-defined tasks characterized by sparse or deceptive rewards [135], by effectively exploring solution space.

Here, we consider the target application of oscillator circuit engineering followed in [361], where parameters of a gene circuit model are optimized to produce oscillation patterns with target amplitude A, frequency w and offset b. This time, the intervention space includes both genetic interventions (setting kinematic parameters of regulatory rules) and environmental interventions (setting the initial state y_0). We then compare three alternative exploration strategies: curiosity search, random search and a global optimization strategy using gradient descent as proposed in [361], all given the same experimental budget ($N = 5000$). For curiosity search, the behavior space Z is defined as the image space of the discrete Fourier transform of the observation. We then use the exact same IMGEP algorithm as before, but operating within the novel problem spaces (I, Z). For gradient descent, we follow the procedure proposed in [361]. We define a loss function which measures the mean square error between the observed node activation levels y and the target oscillation (represented as a cosine wave). We then randomly initialize the parameters $i \sim U(I)$ and use Adam optimizer for N=5000 optimization steps. In addition, we also use gradient descent for local refinement of the best discoveries made by the other exploration strategies (curiosity search and random search), this time with a limited budget of $N = 100$ optimization steps.

In Figure 8.10, we show that curiosity search is again significantly more efficient than random search in revealing a diversity of possible oscillator behaviors. Out of 5000 trials, random search was able to find only 42 configurations leading to sustained oscillations whereas curiosity search was able to find 1167 (and gradient descent did not find any). Without focusing on the target objective, curiosity search is able to efficiently cover the analytic $(A, \omega, b$ space (Figure 8.10, a-c), thus discovering oscillators close to the target one (Figure 8.10-e). Instead, when starting from a random initial condition, gradient descent is very likely to get trapped in a local minimum where it converges to the target offset b but fails to produce oscillations (Figure 8.10-d and Figure 8.10-g). However, whereas the global optimization is unsuccessful in this example, gradient descent seems to be useful to locally refine close-enough solutions, as can be seen here when refining the best discoveries made by curiosity search and random search (Figure 8.10-h, 9-i). These results suggest that a diversity-driven exploration strategy, eventually combined with a more advanced local optimization strategy, can offer promising and cost-effective alternatives for the design of synthetic gene networks. More generally, as our framework only relies on empirical investigation for inferring the mapping between interventions and behaviors (treating them as abstract variables in observable problem spaces), we believe it offers an exciting perspective to be deployed across various problem spaces and classes of organisms.

8.6. Discussion

This paper presents a novel framework aimed at uncovering the navigation competencies of gene regulatory networks (GRNs). The framework conceptualizes GRNs as agents actively navigating the transcriptional space and provides a set of tools, leveraging computational models of curiosity-driven learning and exploration, with a battery of empirical

tests inspired from behaviorist tradition, for automated experimentation and behavioral characterization. The proposed framework is novel in two central ways. First, it introduces a novel AI-based toolbox to the field of biological network analysis. We show how this toolbox, leveraging the successful ingredients of recent intrinsically motivated learning algorithms - originally developed to enable robotic AI agents to explore and learn diverse skills in novel and unstructured environments [57, 368] - can be transposed to assist efficient discovery of behavioral abilities within biological pathway models like GRNs. Secondly, rather than merely mapping the attractor states [316, 317, 348] or analyzing their sensitivity to model parameter changes [345, 346] as extensively proposed in conventional GRN analysis methods, our framework investigates the dynamic adaptability of these networks' navigation competencies in response to various changing environmental conditions. With this approach, our aim is to uncover whether diverse competencies, analogous to the ones exhibited by living agents, can be found within physiological network dynamics. Notably, these competencies are discovered without necessitating structural alterations to network properties or connectivity. Importantly, our framework and its associated tools do not make any assumptions about the structure or origin of the biological network, making it in theory adaptable to the study of diverse unconventional intelligences across various domains.

By applying this framework to a curated database of GRN models, we discovered a diverse range of behavioral responses that GRN can exhibit under different initial conditions, and characterized their robustness to various perturbations. Our analysis revealed a number of interesting aspects of GRN navigation of the state space, which can be leveraged in several contexts. These automated tools form the first step towards cost-effective *in silico* simulation and interrogation platforms: the "behavioral catalogs" produced by this process can be a first stepping stone for better understanding the GRN functionalities as well as for designing drug-driven interventions in a biomedical or bioengineering context.

There are several limitations and avenues for future work to this study. First, these networks are studied as a model in isolation and it is possible that some of the ODE models (or solvers) provide spurious behaviors within certain parameter ranges which might not map to observable phenotypes *in vitro*. Interestingly, this limitation also suggests an interesting further direction to this work: using the automated discovery toolbox to assist model inference, allowing to efficiently identify the rare or unexpected behaviors of the ODE model and suggest whether further refinement is needed or not. Another interesting direction for future work, as our framework considers the GRN model as a black-box and works with limited experimental budget, would be to directly apply it to *in vitro* GRN models at the bench. One could for instance integrate experimental constraints to the search by defining families of empirically-testable interventions and perturbations, as well as specify clinically-relevant goal spaces and perturbations. Even if in a biological setting versatility and robustness phenomena may be harder to detect, or harder to alter, these results can be used to (1) design synthetic biology circuits with advanced capabilities [395], and (2) conduct studies of subcellular proto-cognitive phylogenetics, to help understand the evolutionary pressures for and against reprogrammability in cell regulatory machinery. Another limita-

tion of our work is that we consider predefined problem spaces, here the space of GRN steady states (or Fourier descriptors of the dynamics in the bioengineering example). The dynamics of gene regulatory networks are relatively simple (usually converge to stable points or periodic orbits) allowing such hand-defined descriptors. To scale the framework to higher-dimensional and more complex problem spaces, recent works from the IMGEP literature suggest using unsupervised learning of goal space representations [•1, •3]. Whereas these works were applied to abstract models of multicellular patterning, similar works could be envisaged in more realistic systems, such as sophisticated model of multicellular morphogen and/or bioelectrical patterning which were used to suggest in-vitro experimental manipulations [396, 397].

The tools presented here, and the behavioral repertoire we identified, are just the beginning, and much work remains. Future efforts must test additional competencies across the spectrum of cognition (memory, creative problem-solving, valence, etc.) and extend the tools we presented here to explore them . The predictions made by our computational tools can now be tested in real cells, using emerging tools for physiological profiling in the living state and a diverse set of biochemical, biomechanical, and bioelectrical perturbations. We anticipate a tight and productive feedback loop between computational theory that suggests new experiments, and results in living cells that greatly extend our computational perspective on what they can do [398–402]. Such interdisciplinary work, pulling together concepts and techniques across fields, is likely to have major implications for fundamental understanding of evolution, intelligence, and dynamical control, as well as drive novel kinds of therapeutics that leverage the innate behavioral competencies of living matter [338, 403].

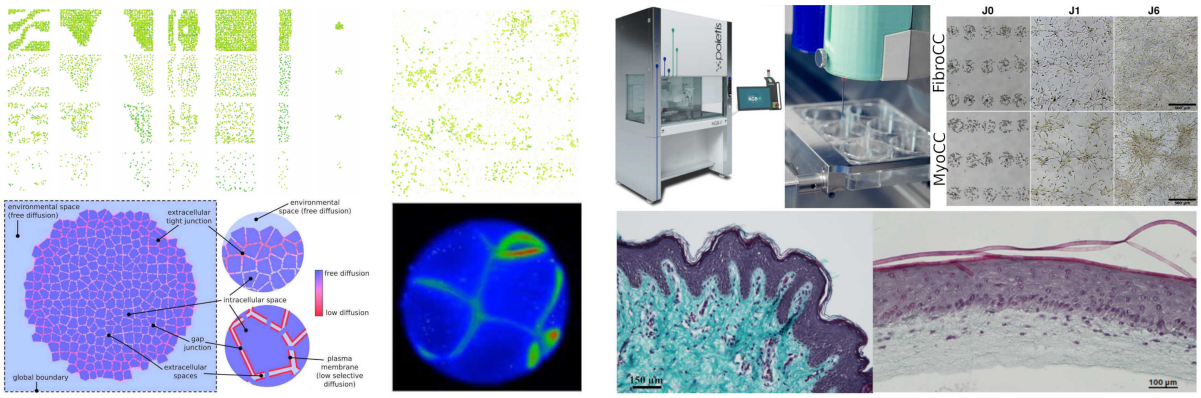
Targeted Reuses of the curious discovery assistant

Towards Applications in Biological Morphogenetic Systems

9.

What is the aim of this chapter? This chapter can be read as a *perspective* chapter where we elaborate on a series of experiments, some of which are *preliminary*, while others are simply *envisaged* for future work. The objective of these experiments is to probe the potential applications of curiosity-driven algorithms in actual *biological morphogenetic systems*.

How is this chapter organized? In Section 9.1, we motivate the long-standing challenge in tissue engineering research, of particular importance for the Poietis biotechnology company, to understand and eventually govern the mechanisms underlying the *morphogenesis* of cellular collectives into functional structures at the tissue level. In Section 9.2, we present *preliminary* experiments conducted in numerical systems tailored to simulate biological behaviors at the tissue level within morphogenetic systems. Those include experiments within a sophisticated model developed and used by biologists at the Levin Lab to simulate and predict bioelectrical patterning at the tissue level (Subsection 9.2.1), as well as experiments conducted in a model specifically designed to simulate the behaviors of fibroblast-populated collagen matrices manipulated experimentally at Poietis (Subsection 9.2.2). Finally, in Section 9.3, we present few experimental campaigns that could be envisaged as proof of concept of automated AI-driven exploration in a bioprinter-controlled morphogenetic system, via the leveraging of a laser-assisted *bioprinting* technology that is developed at the Poietis company. We discuss how these experiments could benefit the exploration and shaping of cellular self-organization toward desired morphological or functional outcomes at the tissue-level, as well as the challenges associated with conducting such real-world experiments.



(a) Numerical Models of Biological Morphogenetic Systems **(b)** Bioprinter-Controlled Biological Morphogenetic Systems

Figure 9.1.: Targeted reuses of the curiosity-driven exploration algorithms for applications in biological morphogenetic systems. (a) Examples of sophisticated numerical models of tissue-level cellular behaviors targeted in this research. Those include the BETSE simulator presented in Subsection 9.2.1 (bottom) as well as the SimCells simulator presented in Subsection 9.2.2 (top). (b) Overview of the bioprinting technology developed at Poietis, as well as its use for investigating the complex mechanisms of skin morphogenesis *in vitro*.

9.1. Motivation

Whereas all the applications presented so far were conducted in abstract or relatively simple numerical systems, something that we have been really interested to pursue, within the context of my PhD and beyond, is to transpose the developed algorithms to real-world wet physico-chemical or biological systems, particularly those manipulated at the bench.

In particular, my PhD is supported by the French Poietis company, with whom I have maintained a collaborative partnership during the entirety of my three-year doctoral program. Poietis is developing a laser-assisted bio-printing device that enables the precise positioning of cells and bio-materials in space, and that can be employed in pair with a computer-aided design software to fabricate biological tissue architectures with a micro-scale resolution (Figure 9.2). With such device, the company targets both various research applications [404] as well clinical applications [20, 405], such as the design of functional personalized tissues for patients requiring skin grafts where cells would be taken from the patient itself [406].

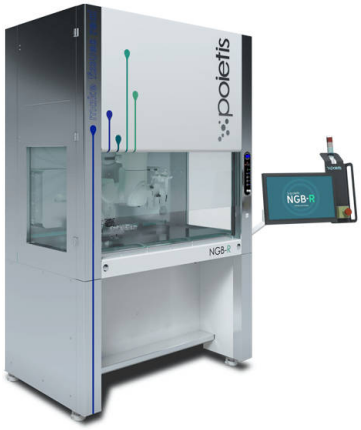


Figure 9.2.: Poietis NGB device. NGB integrates (i) computer-assited design (CAD software), (ii) automated robotic bioprinting with laser-assisted technology, (iii) multimodal extrusion and micro-valve heads to print the biomaterial, and (iv) motinoring of the tissue maturation via cell microscopy and image processing tools. More details in Section 9.3.

However, in addition to the numerous technological challenges associated with achieving accuracy and reproducibility in the printing process (*i.e.* at t=0), it remains very challenging for the company (and for biologists in general) to fully understand and control the maturation of the tissue through time (t>0). Indeed, the process of *morphogenesis* by which cellular collectives self-organize into functional structures at the tissue level is a highly-complex phenomena that involves living entities (cells) as well as sophisticated cellular communication protocols and feedback loops. Understanding, predicting and controlling the mechanisms underlying collective behavior at the tissue level is a major challenge for the company and more broadly for tissue engineering and research in biology.

Within the company, while the experimental procedures is now nearly fully automated thanks to the development of the NGB device (Figure 9.2), the planning of experiments remains primarily a manual task undertaken by biologists. These biologists rely on trial and error as well as personal intuition to process the complex data and explore the high-dimensional parameter space. This is a typical use case where scientists could greatly benefit from a "curious AI discovery assistant" for mapping and navigating the space of possible outcomes.

Although, for various reasons, we have not been able to conduct experiments with actual morphogenetic systems utilizing the bioprinting device throughout the duration of my doctoral program, we are highly enthusiastic about the prospects of developing "curious" artificial intelligence systems that can aid biologists in the exploration and control of morphogenetic processes within these systems. In this chapter, we present preliminary experiments in that direction that were conducted in more-sophisticated and realistic numerical models of tissue-level cellular behaviors (Section 9.2), or that are envisaged for future work using Poietis bioprinting device (Section 9.3).

9.2. Numerical Morphogenetic Systems for Biological Tissue Simulation

In this section, we present preliminary experiments that were made in more complex and "realistic" numerical systems that have been devised for, and in collaboration with, biologists within the context of simulating cellular behaviors at the tissue level during morphogenesis.

In Subsection 9.2.1, we first present the BioElectric Tissue Simulation Engine (BETSE): a python numerical system which was developed in the team of Michael Levin to simulate cellular tissue bioelectrics [397, 407]. Then, in Subsection 9.2.2, we present preliminary experiments conducted within a numerical model of fibroblast-collagen tissue formation [408] implemented within the SimCells software [409], which was developed by a previous doctoral student at Poietis and focuses on mimicking the behavior of fibroblast-populated collagen matrices[1].

1: Fibroblast-populated collagen matrices are a commonly utilized model in tissue engineering for investigating skin tissues, an area of extensive exploration by the Poietis company (see Section 9.3)

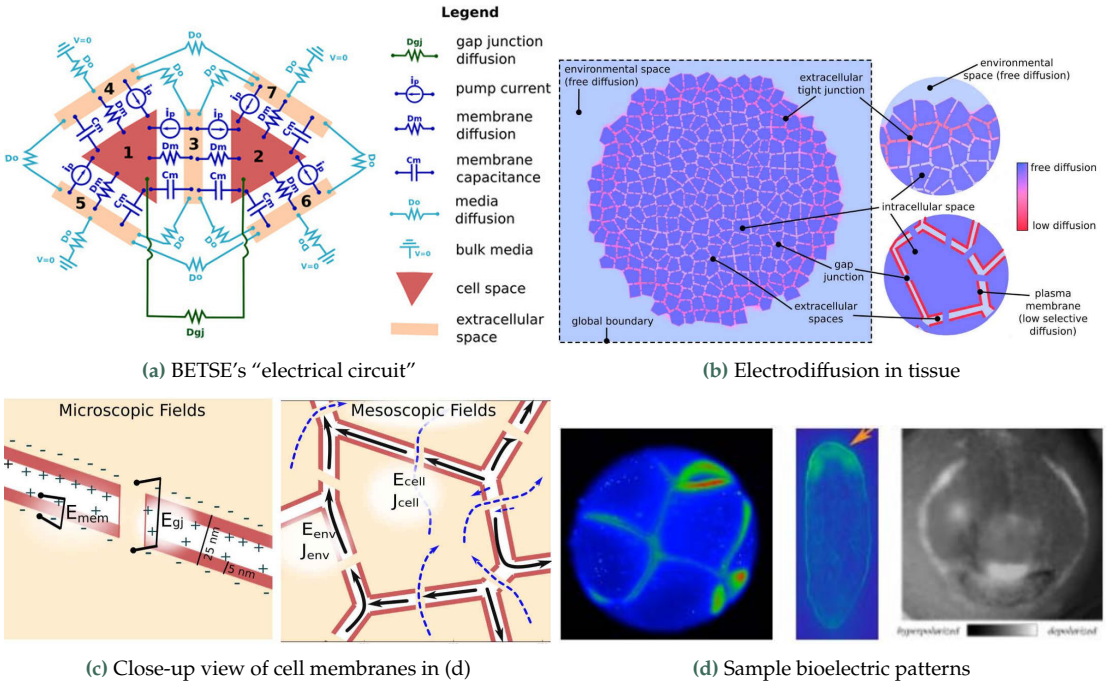
9.2.1. BETSE: BioElectric Tissue Simulation Engine

Overview of BETSE[2] BETSE is a computational model designed to simulate *electrodiffusion* processes, which govern the movement and diffusion of ions (charged particles) in biological systems at the tissue level [397, 407]. Electrodiffusion is a combined phenomenon encompassing *electrophoresis*, which characterizes the motion of charged particles under the influence of an electric field (dictated by voltage gradients), and regular *diffusion*, which accounts for particle motion driven by concentration gradients.

2: BETSE source code can be found at https://github.com/betsee/betse

BETSE offers the capability to incorporate multiple types of ions and to model various phenomena governing electrodiffusion (Figure 9.3):

- *Passive electrodiffusion*: physical movement of ions occurring in the extracellular matrix (ECM) which is mathematically modeled with Nernst-Planck equations in BETSE.
- *Ion pumps*: enzymes located on cell membranes responsible for actively transporting ions, often against concentration gradients, and that are mathematically described by Michaelis-Menten enzyme kinetic relations in BETSE.
- *Ion channels*: specialized proteins situated within cell membranes, actively governing the passage of ions by functioning as selective gateways that permit specific ions to traverse the membrane under particular circumstances. In BETSE, a variety of ion channels are modeled, primarily utilizing the Hodgkin-Huxley formalism, with parameterization informed by scientific literature. These channels can be categorized as *leak channels* (constantly open), *voltage-gated channels* (open or close based on membrane voltage), *or ligand-gated channels* (open or close based on ligand concentration, *e.g.* calcium).
- *Gap junctions*: direct connections between cells facilitating the direct passage of ions and molecules, bypassing the extracellular space. Gap junctions in an open state are analogous to passive diffusion in the environment and are modeled using the Nernst-Planck equation. Some gap junctions in BETSE are voltage-gated, meaning their opening or closing is dependent on membrane voltage.



(a) BETSE's "electrical circuit"

(b) Electrodiffusion in tissue

(c) Close-up view of cell membranes in (d)

(d) Sample bioelectric patterns

Figure 9.3.: Overview of developmental bioelectricity modeling in BETSE. (a) The fundamental "bioelectric circuit" implemented in BETSE, shown on a simplified geometry of two triangular cells (1 and 2) surrounded by their respective extracellular spaces (3-7). Note that in BETSE, and in contrast to the simplified image shown, cells are defined from a Voronoi diagram and are polygonal with four or more membranes, and that a larger network of 10-1000 cells is considered in simulations. We refer to Pietak and Levin [397] for additional details. (b) Electro Diffusive mass transport in a gap-junction networked cell cluster is assumed to follow three pathways, detailed in (c): (1) transmembrane - between intra and extracellular spaces across the plasma membrane; (2) inter-cellular- between cellular spaces via GJ; and (3) environmental- between extracellular spaces and in the global environment. (d) Sample bioelectric patterns, as revealed by voltage reporter dye technique [410] of the cleavage-stage frog embryo, the planarian flatworm, and the developing frog face (left-to-right).

Why exploring BETSE? Michael Levin, as other biologists, thinks that *bioelectricity* plays a fundamental role in orchestrating cellular behaviors and tissue development, and that exploiting biolectring signals holds great promise for applications in tissue engineering, regenerative medicine and cancer reprogramming [412], offering a higher-level layer of developmental control than genetic engineering [45]. Among the very few frameworks modeling non-neural bioelectric activity involved in tissue patterning, BETSE stands out as a sophisticated and biorealistic framework [413]. On of the main advantages of BETSE is that the set of input parameters is not far from being a "recipe" for biologists: most variables are "writable" in real systems manipulated at the bench (*e.g.* via specific ion channel-targeting drugs, injecting mRNA in the cells to produce channel misexpression [414], or optogenetics) and some are "readable" as well (*e.g.* via voltage reporter dyes [410]). This alignment allows BETSE to serve as a practical guide for laboratory experimentation, and in fact several studies have already utilized BETSE's predictions to inform biological experiments. For instance, Pai et al. [414] used BETSE to predict interventions to restore the correct default membrane resting potential pattern after it has been altered by a nicotine teratogen (nicotine molecules are introduced in the simulation and experimentally), intervention which is needed for the embryo to develop correctly. Similarly, Pietak et al. [415] used BETSE to model anterior-posterior axis control in planaria regeneration, and shown it enabled accurate prediction of development from various altered body plans tested in the lab.

9.2.1.1. Scientific discovery in BETSE

Developing automated (AI-driven) tools to reveal diverse behaviors in BETSE could be very useful to assist scientists identifying novel bioelectric states with potentially desirable properties that could then be exploited at the bench. However, exploring the space of biolectric patterns is complex due to voltage-gating effects in channels and gap junctions, leading to feedback loops and dynamic interactions within tissues. Even in tissues made of identical cells (*i.e.* here cells with the same geometry and same channels), their dynamic opening and closing processes generate diverse and rich behaviors, contributing to symmetry breaking and spatial fluctuations. Moreover, the space of configurable parameters is extremely large in BETSE, due to all the physical phenomena involved and modeled (see an example configuration file). In addition to all the physical phenomena governing electrodiffusion, recent versions of BETSE also integrates genetic networks and models their interaction with bioelectric networks [407], adding an extra layer of complexity to the explorable parameter space. The recent work of Hazan and Levin [416], which was taking place approximately at the same time than my visit to the Levin Lab, proposes to use genetic algorithms to automatically explore the input parameters of BETSE and try to solve the (complex) inverse problem of predicting which kinds of ion channel properties in a field of cells can give rise to a desired transmembrane resting potential pattern (V_{mem}). Whereas this work serves as a first step toward the design of machine learning tools for improved bioelectric control of growth and form, two main limitations were identified by the authors. First, they acknowledge that defining a desired V_{mem} pattern can be a non-trivial task, as it's difficult to formalize a function that captures all the features that a human observer intuitively recognizes as correct. Secondly, genetic algorithms assume that the fitness function can guide the search to the target by following the highest fitness score, which is likely to be unsuccessful in practice due to the complexity of the dynamics involved [416]. As argued in Chapter 2, this is a major limitation of objective-driven approaches in general. Diversity-driven approaches like the ones presented in this thesis could provide an alternative strategy for mapping the space of possible behaviors, and eventually be coupled with local optimization.

Random Exploration Campaign In order to get a better idea of the the kinds of patterns that BETSE can produce, and to gather a first database, we performed a random exploration campaign of 2000 experiments in BETSE. A major limitation of BETSE is that it is very slow to compute: a single simulation of 250 seconds in BETSE with $\approx$ 2000 cells takes $\approx$ 50 minutes to run on a strong computer, resulting in a very time-consuming task overall[3]. We explored the same parameters than in Hazan and Levin [416], comprising of 17 parameters influencing BETSE "physics" and parameters influencing the initial state. Whereas in Hazan and Levin [416] the initial state is always composed of 5 disks parametrized by position, radius and intensity, which is a possible (but not very flexible) way to initially break the symmetry of the pattern, we instead used Compositional Pattern Producing Networks (CPPNs) [194] to produce structured initial states (as in Chapter 4). An overview of the kinds of discovered patterns is provided in Figure 9.4, as welll as a visual intuition about what seems to be easy versus hard-to-find patterns in BETSE.

3: For this campaign we leveraged the JeanZay supercomputer to run experiments in parallel using 26 threads per job (found to be the more efficient after small benchmark)

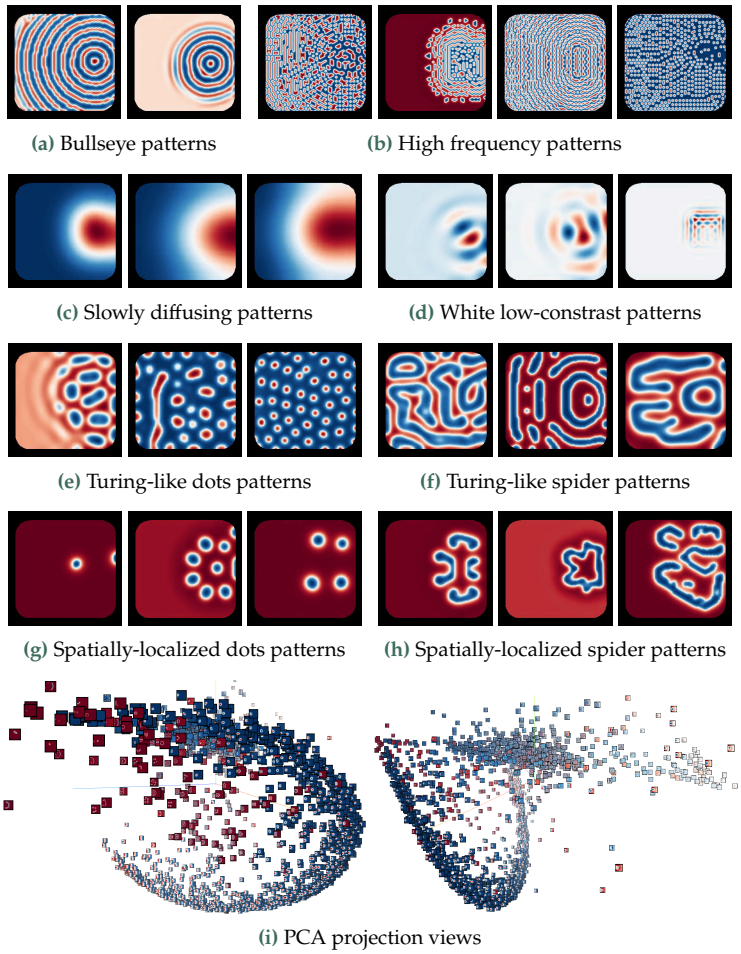


Figure 9.4.: Discoveries made with a simple random search in BETSE. We were able to identify (a) "bullseye" patterns made of concentric bands, which was optimized for in Hazan and Levin [416]; (b) high frequency pattern resembling game-of-life discrete structures; (c) slowly diffusing patterns; (d) "white" patterns *i.e.* with low V_{mem} and low variance; (e-f) turing-like patterns (TLPs); and (g-h) spatially-localized patterns (SLPs, non-moving). (i) Using tensorflow PROJECTOR tool we projected all the discoveries in a low-dimensional space using PCA dimensionality reduction. Screenshots of the resulting visualization in 3D space are shown. Full visualization can be accessed by clicking (or scanning) the below QR code. We can observe that slowly diffusing patterns are largely prevalent in BETSE (or at least in the explored parameter range), yet not very interesting. On the other hand, spatially localized patterns exists but are harder to find, and we did not find moving SLPs.

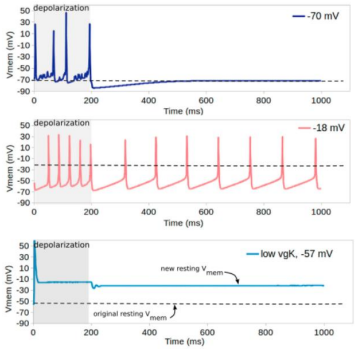


Figure 9.5.: Examples of identified V_{mem} profiles in BETSE. Note that these examples are in the case of cellular cluster with spatially-stable V_{mem}, such that the observable can be reduced to a 1D signal, but it is often not the case. Image reproduced from [397].

Future Directions While we decided to not pursue experimentation, mainly due to the short time of my visit at the Levin Lab (and decision to focus instead on the GRN project), there are several avenues for future work and automated discovery that could be interesting to pursue in BETSE. First, finding a diversity of V_{mem} spatio-temporal profiles in BETSE could be very interesting to biologists. For instance, Figure 9.5 identifies three temporal regimes in BETSE explaining observed behaviors in biology [397]: (top) cells can robustly go back to default V_{mem} after perturbations, (middle) cells can showcase periodic self-excitability akin to cardiac cells, and (bottom) cells can permanently alter their resting V_{mem} potential, which could explain the sustained depolarization observed in cancer cells. It would be interesting to deploy the approaches presented in the first part of this manuscript to try and find a diversity of such behaviors, although the definition of (spatio-)temporal descriptors might not be trivial. Secondly, searching for persistent dynamical patterns *i.e.* moving SLPs capable of preserving some *individuality* (akin to the patterns discovered in Lenia in Chapter 7) and eventually of robustly reaching various positions in physiological space (akin to the *navigation competencies* observed in GRNs in Chapter 8 but this time at the tissue level) would be particularly interesting as well. However, the existence of these patterns remains unknown as experiments made so far did not reveal any moving SLPs.

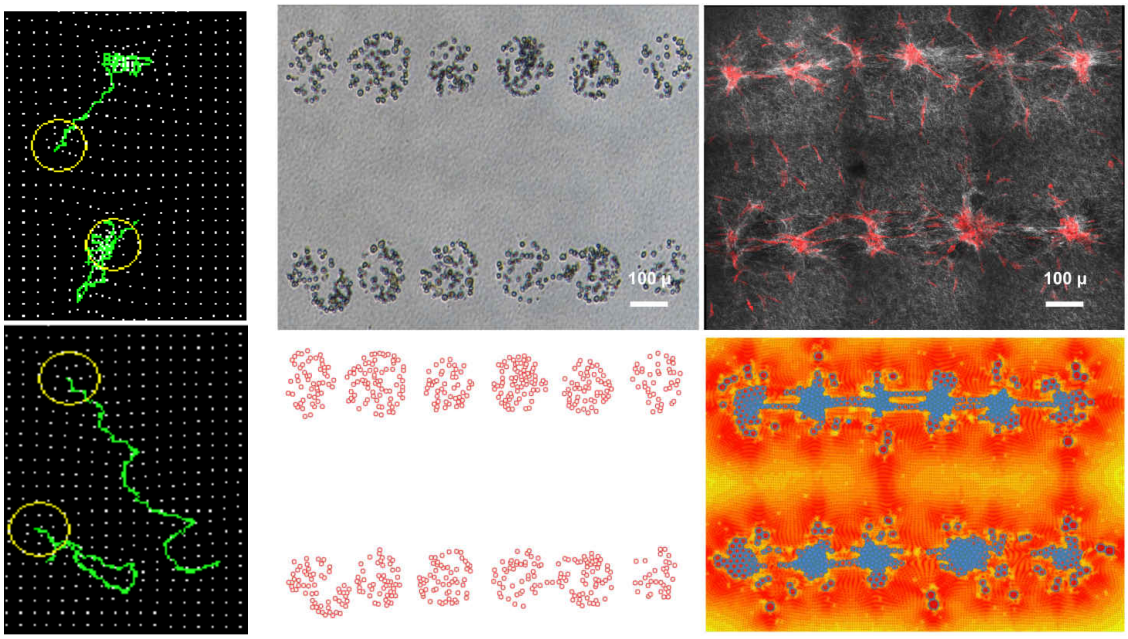
9.2.2. SimCells: Exploring Multicellular Behaviors with Agent-Based Models

Overview of SimCells[4] SimCells is a freely available software based on a multi-agent system that simulates deformable and interacting cells (*agents*) within *matrices* [409]. These matrices, akin to channels in Lenia, store various data types (*e.g.* cell orientation, age, and type) which are employed and updated by the simulation's update rules during each step. The update rule - aka "physics" - is defined by a sequence of *conditions* and *actions* associated to the different entities (cells, molecules or fields) present in the environment. Similar to BETSE, SimCells is a versatile tool designed for modeling various multicellular systems, enabling the implementation of different agent types, each with distinct characteristics such as size and adhesion force, as well as modeling various agent behaviors like migration, chemotaxis, division, differentiation, apoptosis, membrane deformation and adhesion[5]. The underlying physics engine is implemented in OpenCL and optimized for multicore devices with GPUs, accommodating simulations with a substantial number of cells and large environments, depending on the GPU's computational power (scaling up to environments sized at 2048x2028).

4: SimCells can be downloaded at https://virtulab.univ-brest.fr

5: BETSE was originally intended to be used by biologists without prior programming expertise such that most steps of modeling (definition of agent types, behaviors and initial state) can be done graphically in a Java-based graphical user interface [409]

A Fibroblast-Collagen Model in SimCells As detailed in Section 9.3, the Poietis company has been particularly interested in developing and studying *in vitro* models of fibroblast populated collagen matrices, which are commonly used in tissue engineering to develop dermo-epidermal models closely resembling physiological skin. Across various experimental studies conducted by biologists of the company, several critical factors affecting the self-organization of dermal fibroblasts have been identified. These factors are largely associated with the mechanical attributes of the collagen gel and the conditions of cell culture, as documented in Douillet et al. [417]. While bioprinting provides new powerful tools for precise control during the fabrication of in vitro tissue models, investigating the dynamic mechanisms of matrix remodeling by distinct fibroblastic populations or organizations in a laboratory setting can be resource-intensive in terms of time and materials. Leveraging computational modeling to predict and explore these interactions would be highly advantageous. To address this need, Arthur Douillet, a former doctoral student at Poietis, developed a specialized model of Fibroblast-Collagen interactions within the SimCells numerical environment [408]. Based on experimental findings, this model specifically focuses on modeling *mechanical interactions* between the fibroblast cells and their extra-cellular matrix environment (ECM). To that end, the model introduces a representation of the collagen gel using a *spring-mass system*. In this system, masses (referred to as "grains" in SimCells) act as adhesive points where cells can adhere, analogous to collagen fibers. These masses are connected by springs, and the orientation and alignment of these springs are parameterized according to the position and arrangement of the masses. Additionally, the mechanical properties of the springs, including spring length and stiffness, are also parameterized. This framework enables the modeling of localized mechanical properties within the ECM, allowing examination of deformations at the tissue level induced by traction and contraction forces exerted by cells, as illustrated in Figure 9.6.



(a) Cell trajectories

(b) Simulation of fibroblasts on collagen gel

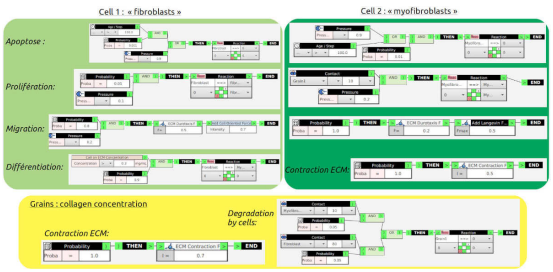
Figure 9.6.: Overview of the Fibroblast-Collagen model implemented in SimCells [408]. (a) Example trajectories (green) of cells (yellow) positioned on a spring mass system with low (top, k=0.05) or strong (bottom, k=0.4) stiffness, where masses are displayed in white. Cells are motile and contractile, provoking remodeling of the ECM. (b) *In vitro* (top) and *in silico* (bottom) simulation of fibroblasts on collagen gel. Snapshots were captured at two time points: t=0 (left) and after 48 hours for real cells (top right), and after 1000 simulation steps in SimCells (bottom right). The initial arrangement of cells (top left) resulted from the deposition of droplets using Poietis' laser-assisted bioprinter, and the same initial configuration was employed in the simulation (bottom left). In the top right image, acquired through confocal microscopy with SHG (Second Harmonic Generation) technique, the positioning of cells (in red) and collagen fibers (in white) is visible. Simulation outcomes (bottom right) depict the final positioning of cells (in blue) and the tension in the springs (indicated by a gradient from yellow, signifying low tension, to red, signifying high tension). Images are reproduced from [418].

9.2.2.1. Scientific discovery in SimCells

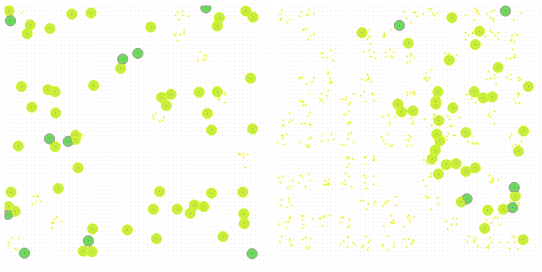
Similar to BETSE, SimCells has the advantage of aiming for biorealistic simulations, establishing a quasi direct correspondence between simulation parameters and experimental variables such as initial cell positioning and collagen concentration. Obviously modeling cellular morphogenesis is a very complex task and a big gap remains between simulation and laboratory experiments. In this section, we present preliminary experiments that were made for applying the IMGEP approach proposed in Chapter 4 and see whether it could be useful to reveal novel types of possible collagen-fibroblast interactions than the ones illustrated in Figure 9.6[6].

6: All experiments were implemented with the ADTOOL software that will be presented in Chapter 10

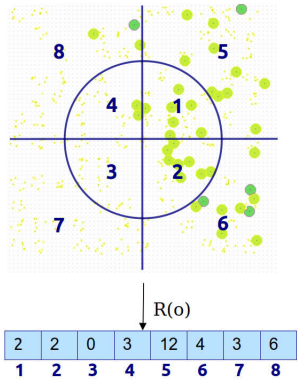
Software modifications Several updates of the SimCells software were required to facilitate automated discovery, and we would like to thank Pascal Ballet for dedicating time and providing valuable assistance in implementing these modifications. These encompassed: (1) creating a standalone application with all dependencies, enabling command-line execution; (2) optimizing the mass-spring model from Douillet et al. [408], resulting in a significant acceleration of simulations (approximately 10-fold) and reduced memory usage (approximately 50%); (3) introducing a headless mode option for running simulations on supercomputers; (4) abstracting input parameters (θ) in a configuration file; and (5) exporting output matrices (o) generated during simulations, thereby facilitating further data analysis (previously limited to graphical interfaces).



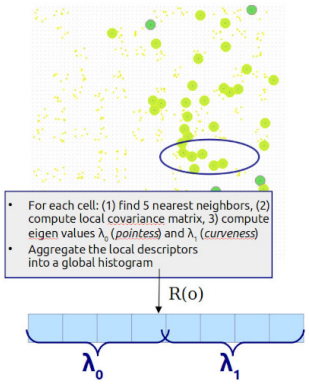
(a) Model Update Rule



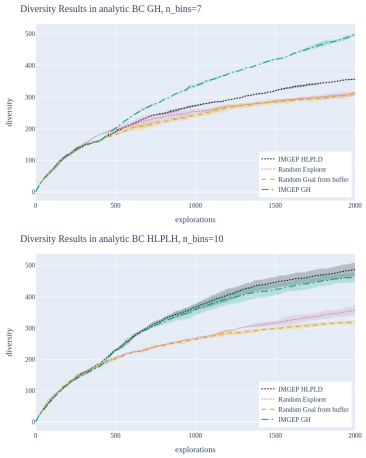
(b) Simulations with low (left) and high (right) collagen concentration



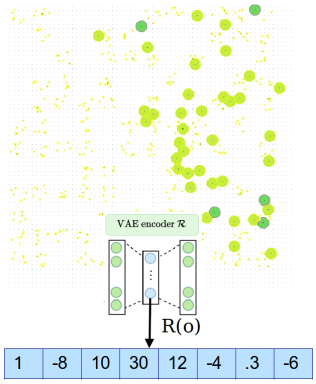
(c) Global histogram descriptor



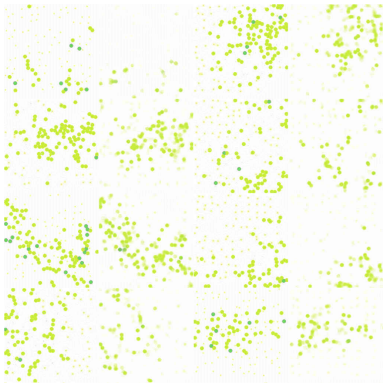
(d) Histogram of local point-level descriptor



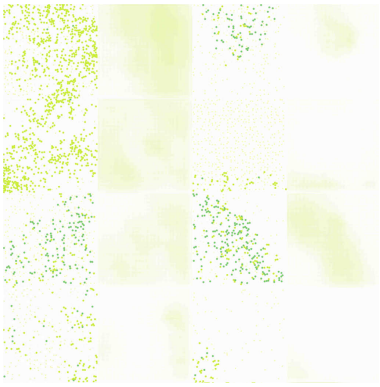
(e) Discovered Diversity



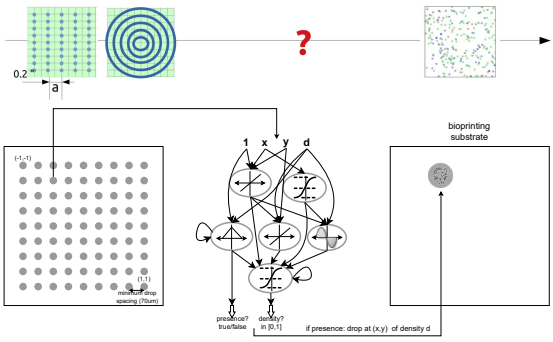
(f) Online-learned VAE encoder



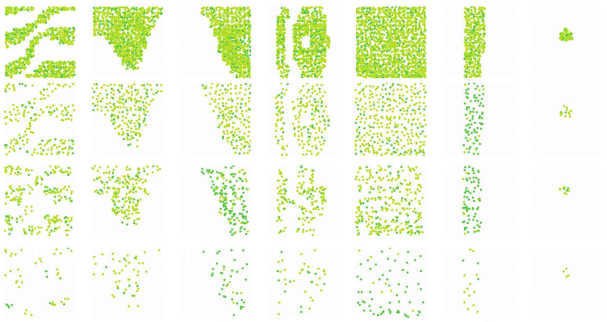
(g) Example VAE reconstruction (512 × 512)



(h) Example VAE reconstruction (1024×1024)



(i) CPPN parametrization of the printing pattern



(j) Example CPPN-generated patterns

Figure 9.7.: Overview of the experimental campaign made in SimCells. (a) Implemented behaviors for the fibroblast and myofibroblast cells, as well as collagen grains in the proposed model. (b) Example simulations in the resulting model. (c-d) Hand-defined representations used to characterize outcomes in SimCells: global histogram descriptor (c) and histogram of local point-level descriptor (d). (e) Discovered diversity by four exploration variants in SimCells. (f) VAE representation used to characterize outcomes in Simcells, incrementally trained during exploration. (g-h) Example reconstructions (from a held validation set) on a 512 × 512 (g) and a 1024 × 1024 (h) environment in SimCells. (i) Generation of the initial pattern with predefined shapes (top-left), random initialization (top-right) or CPPN (bottom). (h) Example CPPN-generated patterns with various droplet sizes and concentrations constraints per row.

Model implementation In light of recent experimental findings at Poietis, it was observed that cultured fibroblast cells exist in two distinct phenotypes: fibroblast and myofibroblast, with each playing distinct roles in matrix remodeling. These findings, detailed in Douillet et al. [417], also identified specific culture conditions for high and low myofibroblast differentiation rates in vitro. Consequently, we chose to incorporate these two distinct phenotypes into our simulation, assuming we could control their ratio experimentally. Three components were introduced into the SimCells environment: fibroblasts, myofibroblasts, and collagen grains, with distinct behaviors outlined in Figure 9.7a. In essence, in the proposed model[7], cells undergo apoptosis if they aged excessively or experienced excessive pressure, while they reproduce through oriented division under other circumstances and certain probability. Fibroblasts were programmed to exhibit high motility along rigidity gradients (durotaxis) but lacked contractile properties, though they were programmed to differentiate into myofibroblasts in regions with high collagen concentration. Conversely, myofibroblasts were not much mobile but highly contractile (which plays a crucial role in wound healing and lesion closure). Collagen grains were also programmed to be contractile and gradually degraded in the presence of cells. In Figure 9.7b, we illustrate example simulations featuring different populations of fibroblasts (light green), myofibroblasts (dark green), and collagen concentrations (yellow).

7: Note that the proposed model is based on my (non-expert) findings from the scientific literature but more precise and complex behaviors could be considered

Basic IMGEP Search Given this model, we then performed basic exploration strategies to implement a fully automated discovery loop. The controlled parameters (θ) included the ECM durotaxis force for fibroblasts, the myofibroblast contractile force, and the initial printing pattern $A^{t=1}$ (randomly initialized with a specified probability of presence and phenotype). The observation (o) returned the matrix of cell positions. For the behavioral characterization $z = R(o)$, we implemented two simple 8-dimensional descriptors: a global histogram descriptor (referred to as GH, Figure 9.7c) that counts the number of cells falling within 8 concentric zones at the end of maturation (after $T = 1000$ steps), and a histogram of local point-level descriptors (referred to as HLPLD, Figure 9.7d) where local point features express pointness (λ_0) and curveness (λ_1) based on the distribution of neighboring points, and are then aggregated into a global histogram. We then compare four algorithm variants: an IMGEP using GH as goal space (IMGEP GH), an IMGEP using HLPLH as goal space (IMGEP HLPLH), a random exploration (Random Explorer), and an exploration strategy that randomly selects a prior discovery from the buffer and applies random parameter mutations (Random Goal from buffer). Each variant had a budget of $N = 2000$ rollouts in SimCells. The IMGEPs employed uniform sampling within the hyperrectangle envelope of currently-reached goals in Z as the goal sampling strategy, along with nearest neighbor selection operator and random mutation expansion operator for the goal achievement strategy. The achieved diversity is displayed in Figure 9.7e, evaluated in the GH analytic space (top) and HLPLD analytic space (bottom) using a binning-based metric. As anticipated, IMGEP operating in the GH space exhibited the highest diversity in that space, while the variant operating in the HLPLD space achieved the highest diversity within that space. We observe that achieving diversity in HLPLD did not correspond to diversity in GH, whereas diversity in GH also led to diversity in HLPLD.

Online Learned Goal Spaces Representations We proceeded to incorporate the algorithm ingredients proposed in Chapter 4 into the IMGEP pipeline: online learning of goal space representation using a deep Variational Autoencoder (VAE) [137] and the generation of structured initial patterns utilizing Compositional Pattern Producing Networks (CPPNs) [194]. The VAE representation was initialized with an 8-dimensional latent space (as depicted in Figure 9.7f) and was fed the rendered image (at T=1000 steps) as input. Figure 9.7g illustrates that the VAE successfully learned to reconstruct "sharp" images for a grid size of 512×512 (Figure 9.7g), as compared to the blurry reconstructions learned for the 1024×1024 environment (Figure 9.7g)[8], although upon closer examination, we observed that this was due to overfitting of the model (generating images similar to those in the training set but not the actual images from the validation set displayed here).

8: blurry reconstructions is also what we obtained in Lenia (see Figure 4.9)

Structured Initial Pattern with Bioprinting Constraints Generating the initial cell-printing pattern poses a non-trivial, high-dimensional challenge. Biologists within the company often resort to using simple parametric shapes (*e.g.*, circles or lines, as depicted in top-left of Figure 9.7i), which are easy to define but lack flexibility. Conversely, employing completely random patterns is inefficient for producing optimized structured effects in the environment. Once again, we proposed to employ CPPNs but this time exploiting their flexibility of accommodating diverse output types to incorporate the bioprinting experimental constraints into the initial state generation process. Indeed, as detailed in the next section, the bioprinting device cannot precisely control individual cell positions, but it can generate droplets with controlled droplet sizes and cell densities, while maintaining minimum spacing requirements between droplets. While CPPN outputs were originally continuous and one-dimensional (*e.g.* image intensity for image generation [194] or connection weight for neural network generation [419]), they can be modularized and subdivided into binary or continuous outputs, which we take advantage of here to integrate the experimental constraints[9], as depicted in Figure 9.7i. Exemplary generated patterns are illustrated in Figure 9.7h, with various droplet sizes and concentrations across different rows.

9: Using modular and binary outputs is also what we used in LeniaChem (Section 5.5) and was also proposed by Cheney et al. [420] to evolve soft-robot morphologies

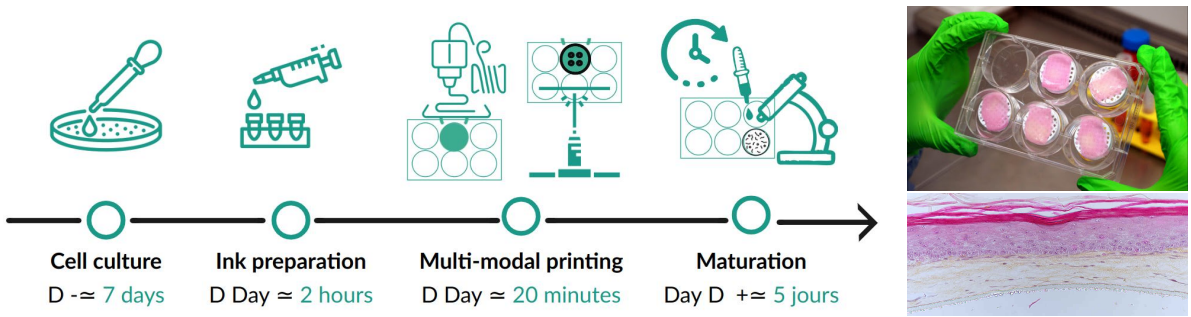
Limitations and Future Work While these experiments represented an initial step towards designing machine learning tools for the automated discovery of diverse cell-ECM interactions within an agent-based simulator like SimCells, several limitations remain to be addressed. First, the model itself turned out to be not really interesting either leading to predictable behaviors such as cells converging on the positions of collagen grains (in cases of strong durotaxis), differentiating into myofibroblasts, and eventually dying, or exhibiting random movements (in cases of low durotaxis) without much evolution, which is not very realistic. Although outside the scope of this thesis, it would be interesting to push the modeling aspect and integrate more complex and realistic cellular behaviors in the model. Secondly, for cellular-based systems like SimCells, it might be beneficial to directly learn representations from the unstructured point cloud of cell positions and attributes, instead images[10], for instance employing approaches as presented in Achlioptas et al. [421].

10: Which would also be useful for real systems, as tracking cell positions can easily be done and learning representations directly from the raw microscopy images might be more complicated

9.3. Bioprinter-controlled Morphogenetic Systems for Biological Tissue Engineering

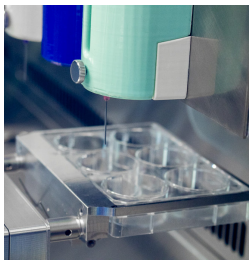
In this section, we present possible experiments that could be envisaged for future work with real morphogenetic systems at the laboratory, leveraging Poietis bioprinting technology and the proposed curious automated discovery algorithms. The presented experimental directions were devised in collaboration with biologists from the Poietis company.

Laser-assisted Bioprinting Before diving into the proposed experimental campaign, we briefly explain the core of Poietis experimental pipeline and bioprinting technology. Bio-printing is an additive manufacturing process similar to 3D printing, but transposed to the production of biological tissues. The process involves depositing living cells onto a biological substrate, assembling them in space, and stacking them layer by layer, in an automated and computer-assisted manner. Various bio-printing techniques exist, including inkjet-based, pressure-assisted (extrusion), and laser-assisted (LAB) bioprinting [423]. Poietis specializes in laser-assisted bioprinting (LAB) and has developed the NGB platform, combining LAB with other biofabrication methods for precise tissue engineering, as detailed in Figure 9.8. NGB is a platform enabling (nearly) fully automated bioprinting process, with enhanced 1) precision (ability to position cells very accurately in a 3D environment), 2) resolution (capability to print at the single-cell resolution), and 3) cell viability [404].

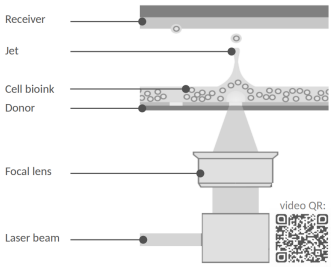


(a) Poietis Experimental Pipeline

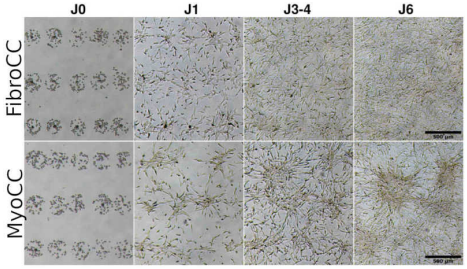
(b) Poieskin®



(c) Bio-material printing



(d) Laser-assited cellular bioprinting



(e) Tissue maturation

Figure 9.8.: Overview of Poietis bioprinting experimental pipeline. (a) Main experimental steps and estimated time for bioprinting of a simple fibroblast-collagen model (2D printing of fibroblast on $1cm^2$ layer of collagen). (b) Picture (top) and histology (bottom) of Poieskin®, a full bioprinted human skin model (derm+epiderm) produced with the NGB platform. (c) Bio-extrusion and micro-valve heads that can be used to print biomaterials. (d) LAB technology: laser pulses are focalized on a cartridge composed of an ink film spread on a glass plate (donor) which results in the formation of an ink jet towards a substrate on which cell microdroplets are collected (receiver). By controlling the physical conditions of the ejection (energy, viscosity, *etc.*), the volume, size and cell density of the droplets can be controlled. (e) Example of tissue maturation observed with microscopy timelapse for two culture conditions, adapted from [422].

Existing approaches for the exploration of cell-ECM interactions Recent advancements in tissue engineering and synthetic biology, including technologies like Poietis LAB and advanced microscopy imaging, have significantly enhanced our capabilities in manipulating, observing, and comprehending the complex processes involved in cellular morphogenesis. As illustrated in Figure 9.9, various factors exert influence over cellular interactions with both neighboring cells and the extracellular matrix. These factors encompass complex cellular communication mechanisms, environmental elements, and feedback loops. Below, we provide a (succint) review of recent studies in biology that have explored the effects of different factors on cellular behaviors and matrix remodeling. Our focus lies on the interactions between fibroblasts and collagen in the dermal layer, a subject of particular significance for our experimental campaigns and simulations within the SimCells framework. We categorize these research papers based on the factors they investigate:

- **Cellular phenotype** (culture conditions)
  - culture medium [424]
  - culture seeding density [417]
- **Physical and mechanical environment of the cells**
  - **Stiffness of the substrata**
    - biomaterial stiffness, degradability and ligand density [425]
    - collagen concentration [426]
  - **Mechanical loading**
    - floating vs restrained support [417, 427]
    - externally applied forces [428]
  - **Initial cell patterning** (disposition and density) [429]
  - **2D vs 3D environment** and geometry of adhesion sites [425]
  - **ECM micro-patterning**
    - micro-contact printing [430]
    - microfabricated post-array-detectors (mPADs) [431]
- **Chemical environment of the cells**
  - UV light to control the formation/degradation of chemical crosslink in hydrogels [432]
  - growth factor to reverse the myofibroblast differentiation [433]
  - growth factor and pro-contractile elements [434]
  - inhibitor of actin polymerization[11] [429]

11: Actin is a protein found in the cytoskeleton of fibroblast cells which is needed for the cell to exert contractile forces on the collagen fibers, central for motility and ECM remodeling

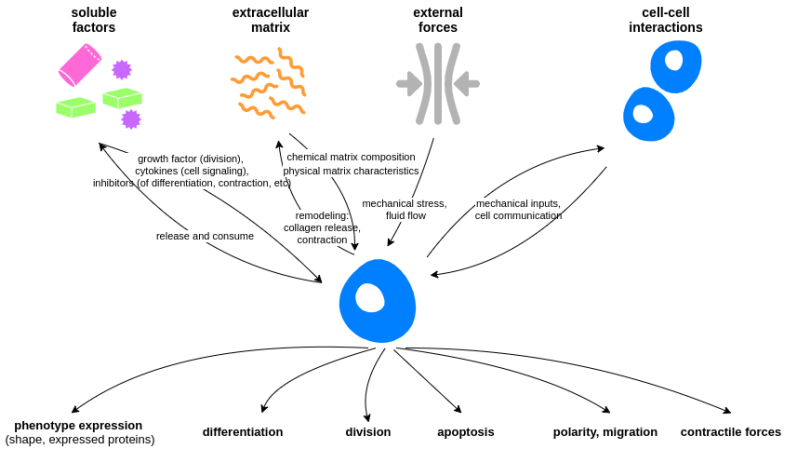


Figure 9.9.: Factors influencing cell-ECM interactions include (i) chemical interactions with soluble factors in the environment, (ii) mechanical and chemical interactions with the extracellular matrix (*e. g.* collagen gel), (iii) external forces applied at the tissue or organ level, and (iv) cell-cell communications. These factors in turn influence various cellular behavior such as phenotype expression, differentiation, division, apoptosis, migration and contraction of the local ECM.

While parameters like initial cell arrangement, culture conditions, and extracellular matrix composition are extensively investigated in laboratory experiments, these studies are typically conducted manually by biologists who isolate and analyze a limited number of parameters individually, relying on visual inspection for characterization. Although machine learning tools have not been widely applied for automated parameter exploration in this context[12], there is growing belief among researchers, including ourselves, that such tools have significant potential to enhance our understanding of the diverse outcomes in morphogenetic systems [437].

12: Shi et al. [435] and Ruberu et al. [436] introduced a Bayesian optimization method for the automatic optimization of bioprinting parameters, focusing on enhancing printing accuracy and stability at printing time ($t = 0$). However, as far as we are aware, there have been no prior studies using ML approaches and bioprinting for automatically exploring tissue-level self-organized dynamics.

Proposed experimental campaign(s): automated discovery of diverse cell-ECM interactions

A major interest of the bio-printing technology, is that it provides the opportunity to study the complex mechanisms of skin morphogenesis in vitro. However, the high-complexity of the processes at stake makes the study of the emerging structures and their exploration very difficult to apprehend with traditional approaches.

In this section, we present few experimental campaigns that could be envisaged as a proof of concept of automated AI-driven exploration in a bioprinter-controlled morphogenetic system, and using the IMGEP algorithms developed in this thesis. While various kind of morphogenetic processes could be envisaged, we decided to focus on fibroblast-collagen skin models which have been extensively studied by Poietis[13].

13: In 2018, Poietis produced the first human dermo-epidermal model, Poieskin® (Figure 9.8b). They are now working on clinical development for a bioprinted skin substitute using LAB technology [406].

Experimental parameters Based on insights from the literature, and discussions with bioligists that manipulated fibroblast-collagen models at Poietis, we suggest to explore the experimental parameters presented in Table 9.1, which are believed to have an effect on dermal fibroblasts behaviors and functionalities [422]. Those include parameters relative to cell culture, ECM composition and organization, and cell patterning.

Table 9.1.: Envisaged experimental parameters, for each step of the experimental pipeline (Figure 9.8a). Parameters shown in gray may be held constant rather than varied. For each parameter, we propose an experimental procedure to confirm precise parameter control (or assess margin of error), along with corresponding controllable variables in the SimCells simulator.

Pipeline	Experimental Parameters	Experimental Validation	SimCells Parameters
Preparation of the cellular ink	FibroCC or MyoCC seeding concentration (M/ml) which were found to produce high (50% in MyoCC) and low (5% in FibroCC) myofibroblast differentiation rates respectively (see [417])	Fluorescent microscopy to characterize the percentage fibro/myofibro and cell count	Vary cell phenotype (fibroblast versus myofibroblast)
Collagen printing and gelling	Gel height (mm), collagen concentration (mg/ml), glass or plastic support, gel patterning (CAD), gelling temperature and C02	Dynamic mechanical analyzer (DMA) to characterize matrix stiffness and porosity. Atomic force microscopy (AFM) with sillicium beads to characterize collagen fiber orientations	Vary ECM properties (springs stiffness and rest length, masses weights and arrangement)
Cells printing	Droplets patterning and density (themselves controlled by physical conditions of the laser-assisted cell ejection like energy, viscosity, *etc.*)	Bright field microscopy to characterize droplets size, position and cell count	Vary initial patterning (droplets size, position and cell count)

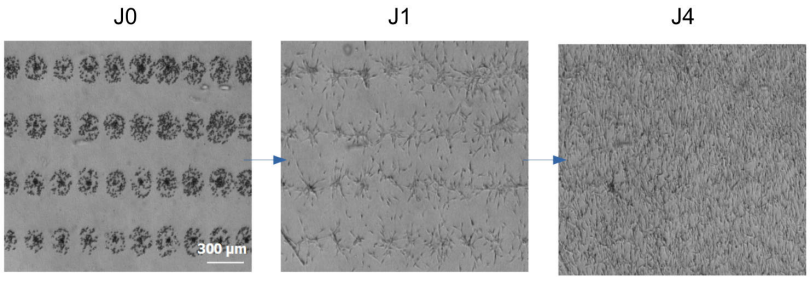
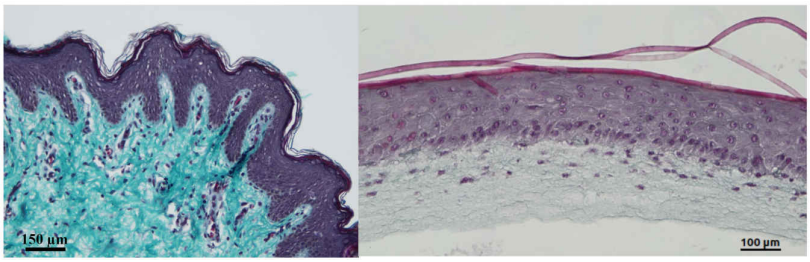
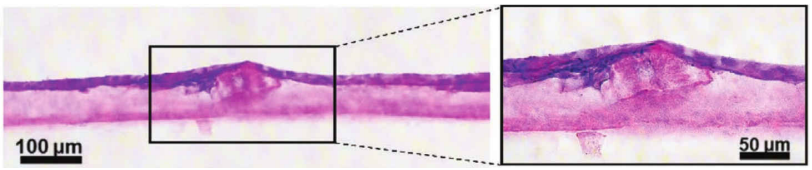


Figure 9.10.: Can we find parameters leading to diverse cell layer orientations? Example of fibroblast-collagen tissue maturation obtained experimentally at Poietis where we can observe that fibroblast cells align vertically at J4 *i.e.* perpendiculary to the printed lines at J0.



(a) Native (left) versus bioprinted (right) skin constructs



(b) Results by [429]

Figure 9.11.: Can we find parameters leading to diverse derm surface topographies? (a, left) Cross-section of in-situ skin with an invaginated junction between the dermis (light blue extracellular matrix and purple cells) and the epidermis (stratified layers of purple cells). (a, right) Cross-section of skin reconstructed through bioprinting. One can observe the flatness of the dermoepidermal junction (DEJ) with respect to the native skin. (b) Cross-section of a bi-layered skin model obtained with inkjet bioprinting in Park et al. [429]. We can see a small "bump" where fibroblasts aggregate within a condensed surrounding collagen matrix.

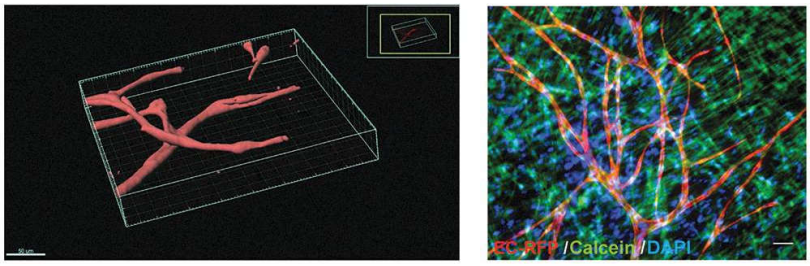


Figure 9.12.: Can we find parameters leading to diverse tube-like epithelial structures? Self-organized endothelial cell networks obtained by Baltazar et al. [438] through bioprinting in skin medium. (left) 3D self-assembled endothelial networks obtained at day 10, reconstructed from Z-stack confocal imaging of dermal constructs. (right) Confocal imaging of dermal constructs obtained at day 50.

Objective #1: exploration of diverse cell layer orientations Something that would be particularly interesting for biologists would be to control the orientations of cells and collagen fibers within the gel matrix, or at least to find diverse possible orientations (and ways to robustly achieve those orientations experimentally). Indeed, inducing collagen structure and mechanical properties to control lines of tension is something highly desirable in tissue engineering[14]. At the moment, achieving and consistently reproducing a diversity of such orientations experimentally is very hard. As an example, the Poietis team observed some years ago that under specific conditions, they could generate a cell layer oriented perpendicular to the printed lines, as depicted in Figure 9.10. This is an interesting result, because it is known that once cells in an upper layer establish a specific orientation, subsequent layers tend to adopt the same orientation [439]. This implies that controlling the macroscopic orientation of a single layer could potentially extend to influence the orientation of an entire tissue. However, those results were found "by luck" and in fact the company has not been able to reproduce those results *in vitro* since then. Rather, cell layers typically exhibit one of two

14: For instance *Langer's lines* on human skin, which play a central role for wound healing and scar formation, are parallel to the natural orientation of collagen fibers in the dermis and generally perpendicular to the underlying muscle fibers

configurations: cellular "tapestry" where cells homogeneously cover the layer but with random orientations, or cellular "clusters" where cells heterogeneously cover the layer and gather in contractile clusters, as illustrated in Figure 9.8e under the FibroCC and MyoCC condtions. This is a first use-case where the proposed diversity search AI algorithms could be very useful for exploring the space of possible behaviors.

Objective #2: discovery of diverse derm surface topographies Another interesting use case would be to aim for diverse possible surface topographies of the self-organized dermis layer. Indeed, the dermis is a key element in the mechanical properties of the skin and the proper differentiation of the epidermis: human dermoepidermal junction (DEJ) possesses papillary microstructures and inducing self-organization of similar 3D microstructures in the laboratory is also highly desirable[15] yet challenging. Currently, bioprinted skin equivalents tend to exhibit relatively flat DEJs compared to native human skin (Figure 9.11a) as replicating 3D elevated microstructures through layer-by-layer stacking of bioprinted cells and biomaterials is non trivial. Park et al. [429] recently demonstrated that it is possible to self-organize 3D elevated microstructures from initially printed flat surfaces. The achieved "bumps" were even shown to self-maintain in full skin tissue models when combined with the epidermis layer (Figure 9.11b). However, these preliminary findings were constrained to a few predefined printing patterns, limiting the extent of 3D remodeling achieved. We believe that using the developed diversity search AI algorithms for a bolder exploration of input parameter space could facilitate the discovery of a wider array of surface topographies.

15: For instance, the papillary microstructures on human DEJ play a crucial role in anchoring the epidermis to the dermis while facilitating nutrient and waste exchange between the two layers

Objective #3: discovery of diverse epithelial tube-like structures Finally, something that is quite challenging to achieve in bioprinted skin equivalents but that would be highly desirable from a tissue engineering perspective would be to introduce a dermal vasculature[16]. However, the printing of 3D endothelial networks is relatively recent and there are still major limitations in the generation of functional endothelial networks of physiologically relevant dimensions in vitro [438]. Notably, achieving to construct vessels of the same diameter as the capillaries found in the skin microvasculature ($< 50\mu m$) is very hard. Past attempts at creating capillary-like networks with 3D bioprinting technology typically involved pre-patterning channels in a hydrogel matrix and subsequently lining them with endothelial cells. Baltazar et al. [438] recently shown that it is possible to achieve functional endothelial networks solely through self-assembly of endothelial cells, enabling the generation of pre-vascularized dermal compartments before engraftment (Figure 9.12). We believe that leveraging the precision of the laser-assisted bioprinting technology developed at Poietis, together with the developed diversity search AI algorithms could assist biologists constructing diverse possible self-assembled tubular structures, notably of various diameters. Here, instead of printing fibroblast cells and controlling their phenotype via culture conditions as proposed in Table 9.1, our approach would involve culturing endothelial stem cells under specific conditions designed to achieve an 80% differentiation into epithelial cells while maintaining a 20% population of pericyte cells[17].

16: For instance, human vascular cells play a central role to promote healing and tissue maturation [440] and dermal vasculature might be necessary to achieve long-term stable engraftment of bioprinted tissues in human patients [438]

17: Epithelial cells self-organize in tubular structures forming protective vessel barriers, while pericyte cells contract capillaries for regulating blood flow

Table 9.2.: Experimental observations and low-dimensional characterizations envisaged for the different experimental campaign objectives. We also list the corresponding observable variables that, when implemented, could be investigated in the SimCells simulator.

Objective	Experimental Observations	Outcome Characterization	SimCells Observations
Objective #1	1) Timelapse measurements (×4 or ×10 every 16 minutes for 6-7 hours) with cell nucleus and/or cell membrane and/or microbeads tracking with fluorescence; 2) Confocal microscopy at the end of maturation (T=5 days) to characterize cell nucleus and orientation; 3) Second harmonic generator (SHG) at the end of maturation (T=5 days) to characterize collagen fiber orientations	Statistics of cell and/or collagen fiber orientations per regions	1) Cell nucleus and/or cell membrane and/or masses tracking; 2) Cell nucleus and orientation at the end of simulation; 3) Spring orientations at the end of simulation
Objective #2	Atomic force microscopy or Surface metrology at the end of maturation (T=5 days) to characterize ECM height profile	Surface height descriptors	ECM height profile (for 3D environment)
Objective #3	1) Confocal microscopy with immunostraining and 3D reconstruction; 2) Permeability test in ionic solution	1) Tubular structure (diameter, direction); 2) impermeability score	not implemented

Outcome characterizations Depending on the targeted biological objective, outcomes of tissue maturation might be characterized with the measurements proposed in Table 9.2.

IMGEP design choices Even though the experimental campaign would target small deliverables (*e.g.* $<1cm^3$ tissue), each experiment would require significant budget in terms of time, money and resources. The idea would therefore be to use an approach similar to the one presented in Chapter 8 for exploring the space of GRN behaviors with a very simple IMGEP operating in a predefined low-dimensional goal space (*e.g.* ones proposed in Table 9.2) and targeting a diversity of outcomes in that space using a low experimental budget $100 < N < 1000$. Depending on the choice of the goal space, it might be very useful to introduce some curriculum in the goal sampling as proposed in Chapter 7. For instance, progressively sampling goals toward increasingly elevated or rugged surfaces for the campaign #2 or toward progressively thiner or more permeable tubular structures in #3 might significantly increase sample efficiency in finding a diversity of such behaviors.

9.4. Summary

Overall, introducing a novel methodology utilizing the proposed diversity search AI algorithms to the biologist community, for exploring the spectrum of possible morphogenetic system behaviors, was a main objective of this thesis. With the work on gene regulatory networks and accompanying notebook tutorials (Chapter 8), targeted at a biological audience, we made a first step in that direction. Deploying the developed algorithms to guide experimentation in the bioprinting platform on real morphogenetic systems, would be a major step forward toward in that direction. Given the results obtained throughout this thesis highlighting the sample efficiency of IMGEP approaches in discovering diverse behaviors compared to less sophisticated strategies (*e.g.* grid search methods commonly used in tissue engineering and synthetic biology), one can expect similar results to be found in the systems presented in this chapter.

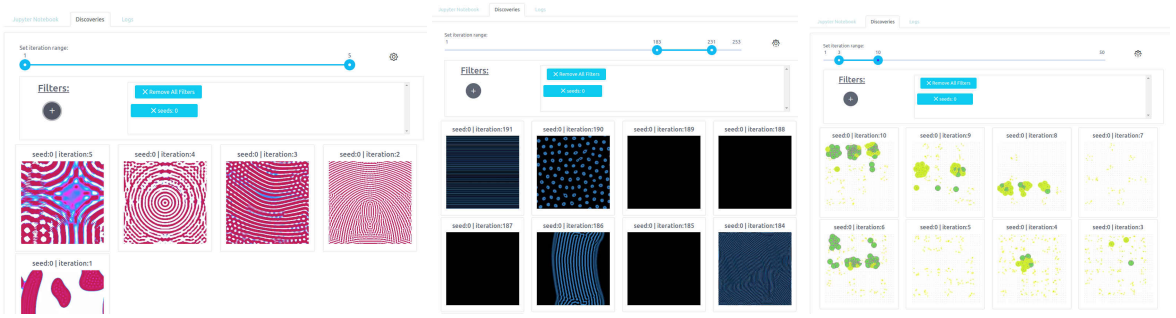
Whether we could discovery truly novel and surprinsing tissue-engineered dermal structures for biologists, remains an open question. Obviously,

real-world experimentation is distinct from digital simulation and they are many additional challenges that we should face as such as stochasticity, variability, and limited control over parameters due to noisy sensors and unknown variables, in addition to the expected challenge of operating under limited experimentatal budget.

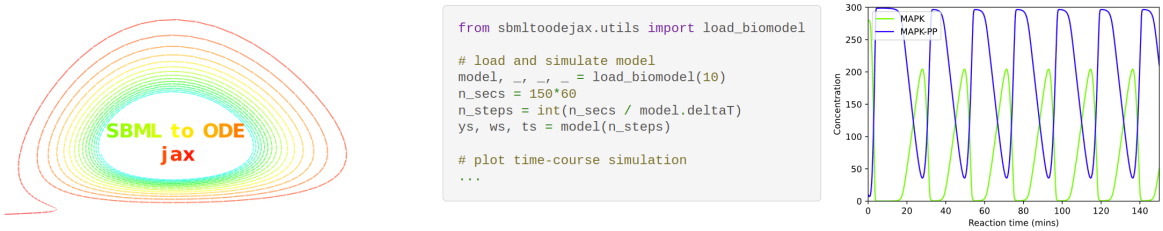
10. Software Ecosystem

What is the aim of this chapter? This final chapter gathers the *software* contributions that were made in this research, all of which are based on open-source code released under the MIT license. In particular we discuss two principal open-source projects: the ADTOOL and SBMLtoODEjax software packages, for which additional efforts were taken to encourage their broader reuse and development by the community.

How is this chapter organized? In Section 10.1, we present the ADTOOL software, which is still under active development at the FLOWERS lab, and whose principal objective is to facilitate automated and interactive exploration of complex systems based on the artificial curiosity algorithms presented in this research. In Section 10.2, we then present the SBMLtoODEjax software, a lightweight library written end-to-end in JAX which is designed to facilitate the efficient simulation and optimization of ODE SBML models, models that are extensively used for the analysis of biological networks (see Chapter 8). Finally, in Section 10.3 we gather useful links towards the various resources presented in this research, including github repositories to replicate the experiments presented in this manuscript, as well as additional interactive blog posts and tutorials that were made for dissemination purposes.



(a) ADTOOL



(b) SBMLtoODEjax

Figure 10.1.: Main *software* contributions on this research. (a) Examples of experiments and discoveries conducted in the ADTOOL software with the original Lenia system (left), pytorch-based differentiable Lenia system (middle) and SimCells system (right). (b) Logo (left) and example usage (middle-right) of the JAX-based SBMLtoODEjax software for biological network analysis.

10.1. ADTOOL: Assisted and Automated Discovery for Complex Systems

10.1.1. Context

ADTOOL is a software aimed at facilitating automated and interactive exploration of complex systems based on the artificial curiosity algorithms developed at the FLOWERS team, and with a particular focus on those developed during my doctoral thesis. The development of the ADTOOL software is an ambitious engineering project that has been led by three software development engineers at the FLOWERS lab: Clément Romac, Mathieu Perie, and Jesse Lin, and that is still under active development.

My specific role in this project primarily revolved around engaging in proactive discussions and providing supervision throughout the project's development. Additionally, I could contribute to testing the software as an "expert" beta user. This notably encompassed the integration of more advanced algorithmic variants such as IMGEP-VAE (Chapter 4) and IMGEP-HOLMES (Chapter 5), the integration of several complex systems (*e.g.* variants of the Lenia system and Simcells simulator as illustrated in Figure 10.1), and the launching of experiments.

(a) Github Repository

(b) Documentation and Tutorials

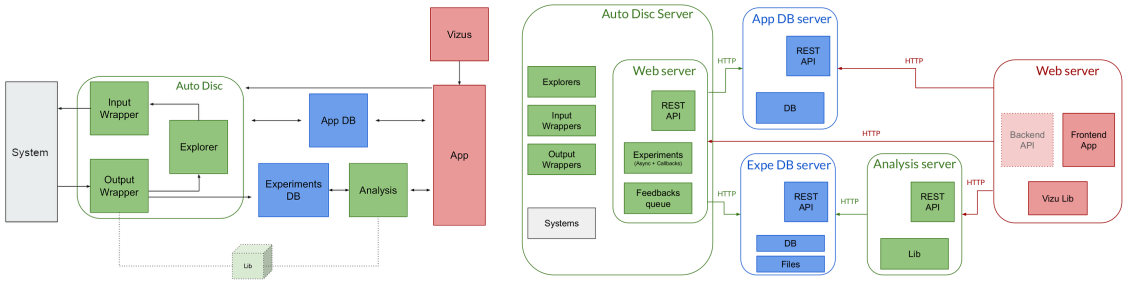
Figure 10.2.: Scan (or click on) the above QR codes for accessing ADTOOL (a) github repository, and (b) online documentation with user tutorials. The code is open-source and under MIT license.

10.1.2. Motivation

The conventional approach to experimentation in complex system often involves manual manipulation of numerous experimental parameters, resulting in slow and inefficient exploration. The primary objective of the ADTOOL software was to democratize the application of curiosity-driven exploration algorithms developed within the FLOWERS team, and in particular the ones developed during the present research, as efficient tools for assisting scientific discovery and for automating experimentation.

Indeed, as we had the opportunity to present our research work to various scientists from diverse fields, we observed a certain interest in integrating our algorithms into the systems that these scientists explore. These interactions, however, highlighted the need to enhance the accessibility of our exploration algorithms, especially for users lacking expertise in programming or machine learning. Consequently, a central motivation driving this project was to enable non-specialist users to independently conduct experiments using our algorithms, while providing them with a good degree of autonomy to tailor the exploration algorithms, visualization tools, and analyses to their specific requirements.

Finally, a key factor that motivated the development of this project, highlighted in Chapter 6, was the the aspiration to seamlessly incorporate, within the exploration algorithms, interactive exploration between the AI agent and actual human end-users. Enabling flexible interactions on one hand, and a user-friendly experience on the other, introduced quite a few engineering challenges which also motivated this project.



(a) ADTOOL Functional Architecture **(b)** ADTOOL Technical Architecture

Figure 10.3.: Overview of ADTOOL software (a) functional and (b) technical architecture.

10.1.3. Software Overview

The software's functional architecture, as illustrated in Figure 10.3a, is structured into several functional blocks, each serving specific functions and interacting with other blocks. First, there is a *modular python API* (Auto Disc, depicted in green) which allows to define and study various combinations of exploration and visualization algorithms, and to easily plug them with several target systems (depicted in gray). Secondly, there is a *user-friendly web application* (depicted in red) which enables scientists to perform, guide and visualize experiments by sending commands to the Python library. Between these components, two databases (depicted in blue) are responsible for storing structured data related to experiments and their configurations (App DB) and housing all data generated by the experiments (Expe DB[1]).

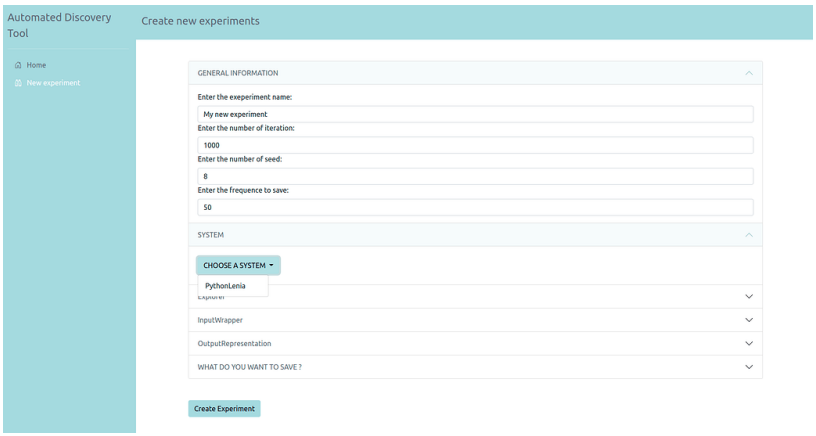
1: Expe DB is the equivalent to what we refered to as the "history" $\mathcal{H}$

Without delving into exhaustive technical details, the overarching technical architecture chosen for the implementation of these functional blocks, as depicted in Figure 10.3b, relies on two fundamental elements. First, it employs a *partitioned architecture* that breaks the software into various *microservices* utilizing REST APIs, and which interact through HTTP requests. Secondly, it employs a *client-server architecture*, wherein the client side is used for visualization and interaction with the experiments (within a Web Window, in JavaScript) and the server side is used for computation and running the experiments (in Python). Moreover, depending on the experimental needs, experiments can either be executed locally or on remote servers, which can be particularly useful for running computations on supercomputers when available.

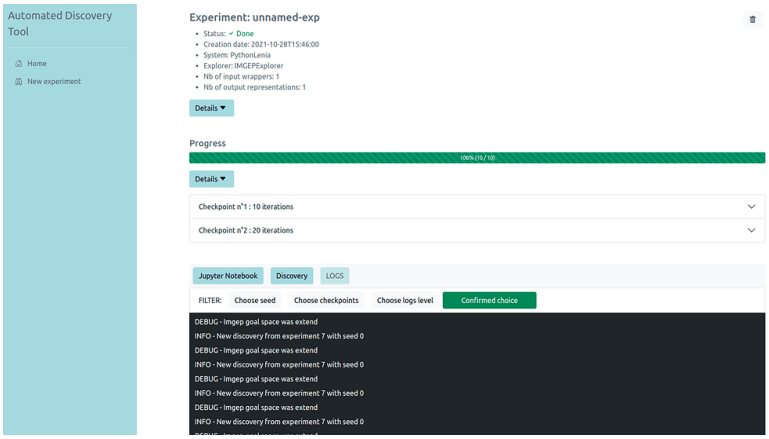
Finally, the front-end web application, designed to enable users to create, interact with, store, and analyze experiments, integrates sevaral components depicted and elaborated upon in Figure 10.4. It notably includes various visualization tabs and jupyter notebooks for flexible interactions with the discovery process.

Overall, the functional and technical architecture of ADTOOL accommodates two main modes of utilization. The first mode is intended for more "expert"[2] end-users who wish to implement novel systems, wrappers or explorer modules within the Python library. The second mode is rather intended for "mainstream" end-users who whish to explore the existing systems, *e.g.* by trying different explorer variants and comparing the resulting discoveries, and that simply need to engage though the web interface as depicted in Figure 10.4.

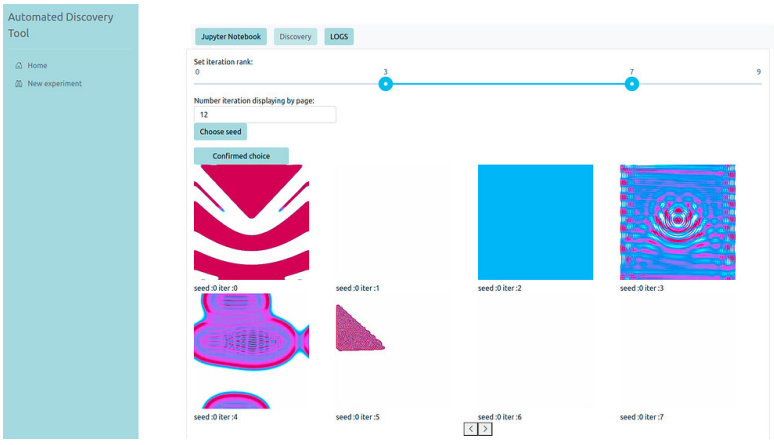
2: Note that the modular python API shoul make it relatively easy for the "expert" user to incorporate new systems or modules (*e.g.* new representations) without the need for extensive modifications



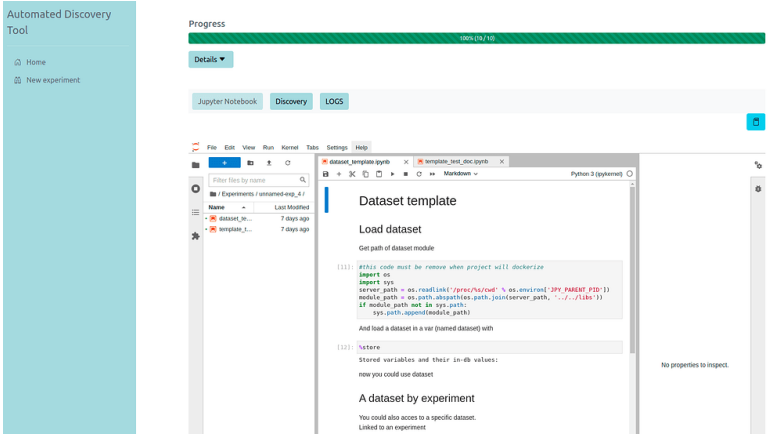
(a) Creating an Experiment



(b) Running and Monitoring Experiment



(c) Visualizing the Discoveries



(d) Analyzing and Interacting with the Discovery Process via Jupyter Notebooks

Figure 10.4.: Overview of the end-user experience in ADTOOL.

(a) The user can create new experiments by selecting the system, explorer, input wrappers and output representations, alongside their respective hyperparameters. It can also select whether it wants the experiment to be executed on its local machine or a remote server, as well as what should be saved in ExpeDB after each discovery (*e.g.* visual rendering of discovery, parameters θ, goal representation z, *etc.*).

(b) Monitoring page on which the user is redirected once the experiment is created and launched. Here, user can see details of the experiment, track its progress and see the different checkpoints generated during experiment.

(b-c-d) Below checkpoints, three tabs are accessible to monitor the experiment.

(b) The *LOGS* tab enables to review the experiment logs.

(c) The *Discoveries* tab provides a simple interface where the user can see the discoveries (updated throughout exploration as new discoveries are received), here retrieved from an exploration of the Lenia system.

(d) The *Jupyter Notebook* tab provides the end-user with a direct means of engaging with the discoveries archived in the Expe DB. This feature serves a dual purpose: firstly, it facilitates analytical endeavors, enabling user to access the discoveries and analyze them with their own pipeline, and secondly it facilitates human-in-the-loop interactions, although this feature is still under development. To elaborate on the second purpose, specific Jupyter notebooks can be associated with the Python modules of the Auto Disc library to request and collect feedback from end-users during exploration, which is then sent back to as input to the python modules which they can used to guide experimentation.

10.1.4. Why use ADTOOL?

While ADTOOL is still under active development, it has reached an advanced stage of development, offering several notable advantages.

To begin with, the Python library within ADTOOL incorporates modular implementations of curiosity-driven exploration algorithms. This modular design simplifies the integration of new systems or additional modules into the exploration pipeline. As a result, ADTOOL could serve as a solid foundation for external users who seek to access these algorithmic variants and potentially advance them into more sophisticated variants[3], or simply reuse them "as such" to explore their own systems[4].

Specifically, while existing variants of curiosity-driven algorithms (and autotelic learning approaches in a broader sense) primarily operated in an autonomous manner, this tool introduces several promising possibilities to orchestrate their activities in tandem with human end-users. In the context of assisted scientific discovery, notably, this tool holds promise for conducting experiments with real-time human feedback, while alleviating the technical complexities associated with AI-user collaboration.

Finally, the proposed software architecture, summarized in Figure 10.3, is designed to offer a high degree of flexibility to the end user. The use of a REST API enables seamless interaction with the provided algorithms, whether they are executed locally on the user's machine or on a remote server. The client libraries, on the other hand, provide generic functions for interacting with the server in several ways. The utilization of web technologies for the graphical interface should also facilitate the creation and customization of new interfaces to meet the specific needs of users.

3: As an example of the software's capabilities, I was myself able to integrate the IMGEP-VAE and IMGEP-HOLMES algorithms presented in Part I. These integrations represent already quite advanced applications, as they involve the real-time learning of deep learning models in PyTorch and the execution of experiments on remote servers.

4: As an example of basic usage, I could myself performed various exploration experiments on several complex systems as illustrated in Figure 10.1.

10.1.5. Future work

The software is currently in an alpha stage of development: it is functional and has been used internally at Inria FLOWERS, but may not have features which are convenient for different workflows.

While the engineers of the team have already implemented the technical ingredients for enabling interactions with end-users during the exploration process, via the jupyter notebook interfaces, these implementations have so far been limited to proof-of-concept demonstrations. A major direction for future work involves conducting more advanced experimental campaigns involving active human participation. For instance, replicating the IMGEP-HOLMES experiments presented in Chapter 6 with preference-based guidance of (simulated) end-users, but this time involving actual human participants, could be a promising direction.

Finally, providing demonstrations to potential new users through this integrated software, enabling them to control the algorithms and visualize exploration results in real-time, via several example tutorials, could be very useful. This should hopefully enable broader dissemination of the software, which has only be used internally at the moment.

10.2. SBMLtoODEjax: Efficient Simulation and Optimization of ODE SBML models in JAX

10.2.1. Context

SBMLtoODEjax is a software that was developed during my stay at the Levin Lab in Tufts University (Boston, USA), within the context of the research project centered on mapping the space of possible behaviors of gene regulatory networks, as detailed in Chapter 8. The motivation for developing this software stemmed from the observation that existing tools designed for simulation of biological network models were not very tailored to explore the space of possible interventions on those systems, nor to facilitate automated discovery.

While SBMLtoODEjax was first developed as a tool for the work presented in Chapter 8, we decided to package it in a standalone, lightweight and adaptable library, as we believe it has the potential to be reused in several contexts at the interesection of machine learning and biology. To encourage its broader reuse and development by the community, additional efforts were made to make it easily accessible, to provide a comprehensive documentation with various hands-on tutorials, and to propely package and share it with an open-source license.

(a) Github Repository

(b) Documentation and Tutorials

Figure 10.5.: Scan (or click on) the above QR codes for accessing SBMLtoODEjax (a) github repository, and (b) online documentation with various tutorials. The code is open-source and under MIT license.

10.2.2. Motivation

As discussed in Chapter 8, developing methods to explore, predict and control the dynamic behavior of biological systems, from protein pathways to complex cellular processes, is an essential frontier of research for bioengineering and biomedicine [441]. Significant effort has gone in computational inference and mathematical modeling of biological systems, which has resulted in the development of large collections of publicly-available models, typically stored and exchanged on online platforms (such as the BioModels Database [314, 315]) using the Systems Biology Markup Language (SBML), a standard format for representing mathematical models of biological systems [442, 443].

Despite the wealth of available SBML models, scientists still lack an in-depth understanding of the range of possible behaviors that these models can exhibit under different initial data and environmental stimuli, and lack effective ways to search and optimize. Except for a subset of simple networks where system behavior and response to stimuli can be well understood analytically (or with exhaustive enumeration methods), onerous sampling of the parameter space and time-consuming numerical simulations are needed. This remains a major roadblock for progress in biological network analysis.

On the other hand, recent progress in machine learning (ML) has led to the development of novel computational tools that leverage high-performance computation, parallel execution and differentiable programming and that promise to accelerate research across multiple areas of science, including biological network analysis [444] and applications in drug discovery and molecular medicine [363, 445]. However, to our

knowledge, there is no software tool that allows seamless integration of existing mathematical models of cellular molecular pathways (SBML files constructed by biologists) with ML-supported pipelines and programming frameworks. Whereas there exists many software tools for manipulation and numerical simulation of SBML models, they typically rely either on specialized simulation platforms limiting the flexibility for customization and scripting (such as COPASI [446, 447], Virtual Cell [448, 449] and Cell Designer [450, 451]) or provide scripting interfaces in Python or Matlab but rely on backend engines that do not support hardware acceleration or automatic differentiation (like Tellurium [452, 453] and SBMLtoODEpy [454] python packages, or the Systems Biology Format Converter (SBFC) which generates MATLAB and OCTAVE code [455]).

SBMLtoODEjax seeks to bridge that gap by bringing SBML simulation to the JAX ecosystem, a thriving community of JAX libraries that aim to accelerate research in machine learning and beyond, with diverse applications spanning molecular dynamics [109], protein engineering [456], quantum physics [457], cosmology [458], ocean modeling [459], photovoltaic research [460], acoustic simulations [461] and fluid dynamics [462]. SBMLtoODEjax aims to integrate this ecosystem and provide tools to accelerate research in biological network analysis.

10.2.3. Software Overview

SBMLtoODEjax is a lightweight library that allows to automatically parse and convert SBML models into python models written end-to-end in JAX, a high-performance numerical computing library with automatic differentiation capabilities [198]. SBMLtoODEjax is targeted at researchers that aim to incorporate SBML-specified ordinary differential equation (ODE) models into their python projects and machine learning pipelines, in order to perform efficient numerical simulation and optimization with only a few lines of code. Taking advantage of JAX's core transformation features, one can easily boost the speed of ODE models time-course simulations and perform efficient search and optimization by running simulations in parallel and/or using automatic differentiation to find derivatives. SBMLtoODEjax extends SBMLtoODEpy, a python library developed in 2019 for converting SBML files into python files written in Numpy/Scipy [454]. The chosen conventions for the generated variables and modules are slightly different from the standard SBML conventions with the aim to accommodate for more flexible manipulations while preserving JAX-like functional programming style.

Documentation We refer to the documentation for additional details on SBMLtoODEjax's design principles and chosen conventions, main advantages and limitations, and full API docs. Various hands-on tutorial notebooks for loading and simulating biomodels, running parallel executions and performing gradient descent optimization are also provided.

10.2.4. Why use SBMLtoODEjax?

Simplicity and extensibility SBMLtoODEjax retains the simplicity of the SBMLtoODEPy library to facilitate incorporation of the ODE models into one's own python projects. As shown in Figure 10.6, with only a few lines of python code, one can load and simulate existing SBML files. Moreover, one can easily refactor the models to its need.

```python
from sbmltoodejax.utils import load_biomodel

# load and simulate model
model, _, _, _ = load_biomodel(10)
n_secs = 150*60
n_steps = int(n_secs / model.deltaT)
ys, ws, ts = model(n_steps)

# plot time-course simulation
...
```

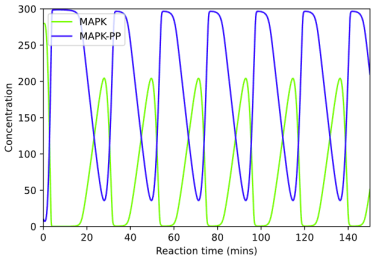


Figure 10.6.: Example code (left) and output snapshot (right) reproducing original simulation results of Kholodenko 2000's paper [463] hosted on BioModels.

JAX-friendly The generated python models are tailored to take advantage of JAX main features. Model rollouts use `jit` transformation and `scan` primitive to reduce compilation and execution time of the recursive ODE integration steps, which is particularly useful when running large numbers of steps (long reaction times). Models also inherit from the Equinox module abstraction [464] and are registered as PyTree containers, which facilitates the application of JAX core transformations to any SBMLtoODEjax object.

Efficiency simulation and optimization The application of JAX core transformations, such as just-in-time compilation (`jit`), automatic vectorization (`vmap`) and automatic differentiation (`grad`), to the generated models make it very easy (and seamless) to efficiently run simulations in parallel. For instance, as shown in Figure 10.7, with only a few lines of python code one can vectorize calls to model rollout and perform batched computations, which is especially efficient for large batch sizes.

```python
# vector of initial conditions
batched_y0 = ...

# batch model
batched_model = vmap(model,
                     in_axes=(None, 0),
                     out_axes=(0, 0, None))

# run simulation in batch mode
batched_ys, batched_ws, ts = batched_model(n_steps,
                                           batched_y0)
```

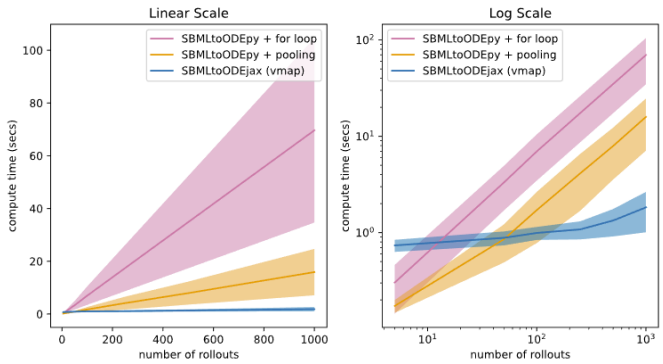


Figure 10.7.: (left) Example code to vectorize calls to model rollout (right) Results of a (rudimentary) benchmark comparing the average simulation time of models implemented with SBMLtoODEpy versus SBMLtoODEjax (for different number of rollouts i.e. batch size). For additional details on the comparison, please refer to our Benchmarking notebook.

As shown in Figure 10.8, SBMLtoODEjax models can also be integrated within Optax pipelines, a gradient processing and optimization library for JAX [465], allowing to optimize model parameters and/or external interventions with stochastic gradient descent.

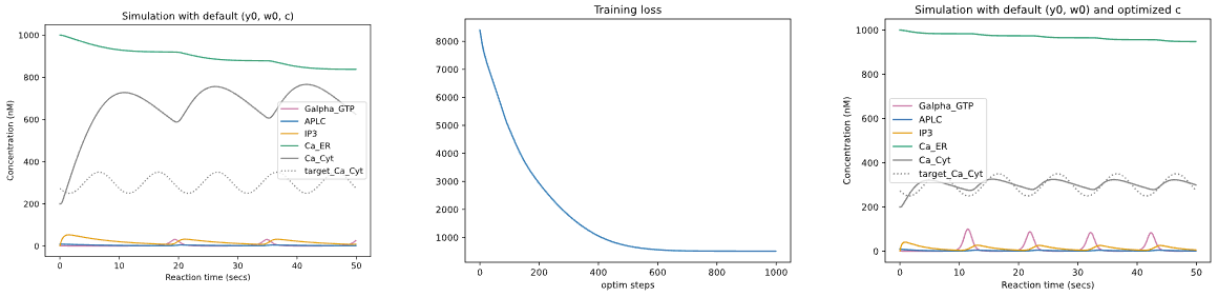


Figure 10.8.: (left) Default simulation results of biomodel #145 which models ATP-induced intracellular calcium oscillations, and (arbitrary) target sine-wave pattern for Ca_Cyt concentration. (middle) Training loss obtained when running the Optax optimization loop, with Adam optimizer, over the model kinematic parameters c. (right) Simulation results obtained after optimization. The full example is available at our Gradient Descent tutorial.

Altogether, the parallel execution capabilities and the differentiability of the generated models opens interesting possibilities to design and optimize intervention strategies.

10.2.5. Future Work

SBMLtoODEjax is still in its early phase and does not yet handle all possible cases of SBML files. We aim to handle more cases in future releases, and welcome contributions to that end.

Similarly, SBMLtoODEjax only integrates one ODE solver for now (`jax.experimental.odeint`), but could benefit from more [344].

Finally, whereas SBMLtoODEjax is to our knowledge the first software tool enabling gradient backpropagation through the SBML model rollout, applying it in practice can be hard[5]. Further optimizing the models to be efficiently differentiable, *e.g.* using simpler Euler methods as ODE solver, might benefit their broader usage.

5: GRN models are recurrent networks that are generally ran for many steps, with each step calling the ODE solver. This can lead to long backpropagation compute times and gradient issues.

10.3. Other Resources

For the other materials presented throughout this research, including github repositories, notebook tutorials, interactive web-demos, blogpost and presentations, we refer the interested reader to Section 1.4.2.

Discussion

11. Summary

As explained in our introductory chapter, the present research aims to contribute to the development of *digital discovery assistants* that can assist discovery in self-organizing systems, and in turn help to address various challenging problems in Science. Towards this, after formalizing the *automated discovery* problem and surveying standard AI paradigms in Chapter 2, we presented a series of works approaching this challenge through a novel *developmental AI* perspective. This paradigm shift proposes to equip AI agents with *autotelic learning processes*, which means enabling agents to autonomously generate and pursue their own goals driven by intrinsic motivations signals, which should in turn promote the autonomous discovery of diverse kinds of self-organized behaviors.

Our research's primary focus in Part I was directed towards the reuse and enhancement of *intrinsically-motivated goal exploration processes* (IMGEPs), a foundational family of *curiosity-driven* exploration algorithms, originally coming from developmental robotics, to guide scientific experiments in complex dynamical systems. In Chapter 3, after providing some background on the IMGEP goal-directed search formalism, we introduced the numerous challenges that arise for transposing IMGEPs to automatically find a *diversity* of *interesting* self-organized patterns in complex systems.

We then proposed a series of algorithmic ingredients aimed at creating more effective "curious discovery assistants" capable of automating the long-term discovery of novel and interesting structures in Lenia, a class of continuous cellular automata (CA) models, which was shown to be a particularly useful testbed with rich possibilities for emergence.

In Chapter 4, we proposed to integrate the *unsupervised* and *online* learning of goal space representations, via the use of deep variational auto-encoders (VAEs) directly trained on the collected raw observations. We showed how, not only this removed the need for human expertise in engineering these representations or relying on pre-collected datasets, but also proved quite effective at discovering a diverse range of spatially-localized patterns (SLPs). Although, the discovered SLPs' remained relatively "basic" compared to the various *gliders* known to exist in Lenia, this work formed the first step toward sample-efficient and fully automated discovery of diverse self-organizing patterns in continuous cellular automata models, outperforming random exploration baselines and other IMGEP baselines that employed hand-defined goal spaces.

Then, in Chapter 5, we motivated the need for what we called *Meta-diversity* search, arguing that there is not a unique ground truth *interesting* diversity but rather many types of possible diversity depending on the macrostates than one decides to observe. Meta-diversity search aims to enable continuous seeking of novel source of diversities while being able to quickly adapt the search toward a new unknown type of diversity. In particular, we proposed to integrate a dynamic and modular model architecture, HOLMES, for the unsupervised learning of *diverse representation spaces* within the IMGEP exploration process, an approach referred to as IMGEP-HOLMES. We showed that an IMGEP equipped with a monolithic representation, despite being pre-designed or learned incrementally

during exploration, fails to continuously adapt its representation to the different niches of discovered patterns, limiting their discovery. On the other hand, we showed that the IMGEP-HOLMES is better suited to escape this bias by learning divergent feature representations, leading to the effective exploration of diverse sources of diversities within the Lenia system, among which some led to the particularly interesting discovery of previously unseen glider-emitting lifeforms.

As final ingredient of the "curious discovery assistant", proposed in Chapter 6, we further argued that among the many potential sources of emergent diversities, an "interesting" source of diversity is one that aligns with the preferences and motives of the final end-user. We therefore proposed to transition from an entirely automated approach to an assisted approach integrating *human guidance* into the IMGEP exploration process. In particular, we proposed a preference-guided variant of IMGEP-HOLMES where the goal selection mechanism requested and leveraged human feedback to prioritize the search in the *preferred* niches of discovered patterns. Considering two end-user models respectively interested in two types of diversities (diverse spatially localized and diverse turing-like patterns), we showed that a monolithic IMGEP will make discoveries that are strongly uneven in relation to these user preferences. The proposed approach, on the other hand, was shown to escape this bias and effecitvely adapt the search to align with the end-user preferences, while requiring minimal feedback. However, these experiments were conducted with *simulated* user feedback, lacking the complexity and adaptivity that might arise from interactions with actual human users.

Whereas the first part of this manuscript centered on modular developments of the curiosity-driven exploration algorithms, Part II was the occasion to delve into our second primary research focus: the *practical applications* of these algorithms for assisting scientific discovery, which we investigated around two concrete use cases.

The first use case, presented in Chapter 7, targeted the *automated discovery* and *systematic characterization* of "robust sensorimotor agents" in Lenia. Studying forms of self-organized *agency* and investigating whether these structures can further give rise to forms of *sensorimotricity*, *i.e.* whether they can develop some form of sensorimotor apparatus enabling them to make "decision" and "sense" at the macro scale through local interactions only, is of particular interest to scientists. Yet, their study and characterization has so far relied on manual search and qualitative evaluations, limiting their analysis. We demonstrated how the curiosity-driven exploration algorithms can be particularly useful in that case, successfully finding environmental rules leading to the progressive emergence of individuality, locomotion, and sensorimotor abilities in the Lenia system. Additionally, testing the robustness of the discovered agents via a battery of quantitative and qualitative tests, we further demonstrated the impressive generalization capabilities that these self-organizing agents can achieve, opening various perspectives for future research in AI.

The second use case, presented in Chapter 8, targeted the *automated discovery* and *systematic characterization* of diverse "robust goal states" that gene regulatory networks (GRNs) could exhibit in transcriptional space. Again, the exploration, understanding and characterization of these behaviors is of particular interest for scientists, yet there is a lack of systematic methods to reveal and optimize those behaviors via external

interventions. We demonstrated how the curiosity-driven algorithms, even in their most basic version, can also be very useful in this endeavor, efficiently uncovering a wide spectrum of possible GRN behaviors while using a low budget of experiments. We further characterized the *robustness* of the discovered GRN behaviors via a battery of empirical tests, suggesting that GRNs might possess relatively advanced forms of non-genetic robustness, and that do not require structural changes of network properties or connectivity. We also explored the implications that the discovered competencies could have for biomedicine (control of gene expression via stimuli, not genetic rewiring) and bioengineering (design of synthetic morphology with desired system-level behaviors), opening several perspectives for future research in biology.

Finally, in the third part of this manuscript (Part III), we presented *preliminary* experiments and *software* contributions aimed at expanding the practical applications to a broader range of scientific problems.

In Chapter 9, around preliminary and prospective experiments, we presented several scientific challenges in *biological morphogenetic systems* for which AI-driven curious discovery assistants could be very useful to assist the exploration and shaping of cellular self-organization toward desired morphological or functional outcomes at the tissue-level.

In Chapter 10, we presented the different software contributions of this research, centered around three objectives. The first one, via the development of the ADOOL software, is to encourage and facilitate the reuse of the developed algorithms to a broader audience of scientists. The second one, via the development of SBMLtoODEjax, is to facilitate research in biological networks leveraging recent advances in high-performance computing and differentiable programming. The final one, via the release of several materials accompanying the algorithmic and scientific contributions presented in Part I and Part II of this manuscript, is to disseminate the present research to the general public.

12. Perspectives

12.1. Algorithmic Perspectives Towards Open-Ended Discovery Assistants

IMGEP-HOLMES was a first step in implementing the concepts of *meta-diversity search* (Chapter 5) and *human guidance* (Chapter 6), both of which are tied to what we consider as two pivotal dimensions in defining, simulating, and evaluating open-endedness. The first dimension, as discussed in Banzhaf et al. [238], is the one of *meta*-level changes where there is a notion of abstraction of different "types" of novelty: an open-ended system is one that continually generates novel types of novelties. The second dimension, as outlined in Stanley and Soros [466], is the one of *subjectivity*: open-endedness is, at least to some extent, a subjective concept in the eyes of the evaluator. As the goals of the curious AI discovery assistant become more and more diverse, abstract and creative (*i.e.* novel and interesting), its *autotelic learning process* - the pursuit of self-generated goals - will progressively adopt a more open-ended nature. These dimensions are particularly beneficial for autotelic algorithms in general, extending beyond the role of a scientific discovery assistant.

In this section we discuss how future discovery assistants can make progress on those two dimension, *i.e.* how they could learn to represent and target a richer "meta" diversity of goals (Subsection 12.1.1), and how they could leverage richer interactions with end-users to better align their exploratory strategy with human interests (Subsection 12.1.2).

12.1.1. Towards a Richer "Meta" Diversity of Goals

In the first part of this manuscript (Part I), we proposed various goal-directed search variants that led to increasingly "meta" levels of novelties: novelty in fixed representation space (type-0 novelty), novelty in online-learned representation space (type-1 novelty), and novelty in diverse online-learned representation spaces (type-2 novelty). However, even though the last-variant was operating in diverse representation spaces, *i.e.* the learned features encoded diverse aspects of the input patterns, the representation spaces where all of the same "type": the latent space of a VAE that is trained on (static) input images. They are several ways in which we could aim for a richer diversity of goal space representations, which could in turn lead to higher-level types of novelty.

First, all the goal spaces used in this manuscript were focusing the search on visual aspects of experimental outcome final state (Lenia pattern or GRN node activations at $t = T$). Although this led, as a side effect, to the discovery of interesting dynamics in some cases, targeting the learning of features encoding *dynamical* aspects of the full video sequences should lead to a richer diversity of achievable goals (*e.g.* diverse glider trajectories in Lenia). Notably, the use of neural network architectures tailored for handling long-range interactions, such as Long Short-Term Memory (LSTM)

recurrent models [467] or Transformer attention-based models [468], could be very useful for learning spatiotemporal representations [469, 470].

Secondly, it would be interesting to target *multimodal* goal space representations, *e.g.* using multimodal Transformer [471] or contrastive [472] architectures. For instance, it might be useful to combine image-based representations with ones learned from the sparse data structures (point clouds) of agent-based systems like the ones presented in Chapter 9, or of particle-based and graph-based systems in general.

Notably, one particularly interesting modality to investigate in future work is the one of *language*. Here, language could be useful as a *communicative* tool, notably to facilitate interactions with human end-users (see next section), as well as a *cognitive* tool to imagine new goals[1] as proposed by Colas et al. [173]. In a recent publication, Colas et al. [59] introduced the use of pretrained language model (LM) to support three key components of the autotelic architecture: a captioner for the outcome characterization (R), a goal-generator the the generation of new goals ($\mathcal{G}$) and a reward function for measuring progress toward each of these goals ($\mathcal{L}_g$). They suggest that, as language models have internalized aspects of humans' common-sense, intuitive physics and overall interests, they are particularly useful to support the representation, generation and learning of diverse, abstract, human-relevant goals [59]. It would be interesting to study whether the use of such language-based models could also shape the exploration of the targeted self-organizing systems in interesting ways. However, whereas language has emerged has a particular useful tool to explore the 3D environments that humans are familiar with, its use would also strongly bias the learned representations. Whether this bias would be beneficial for exploring the targeted self-organizing systems which are full of unfamiliar and yet poorly-described behaviors, remains an open question.

1: Autotelic agents could, as humans, leverage language to form categories, make abstract analogies, and combine old ideas to imagine novel goals [173]

Thirdly, in addition to using spatiotemporal and multimodal data structures to learn diverse representation spaces of the experimental outcomes, one could *explicitly* train those representation spaces to be *diverse*. In Chapter 5 we used Representation Similarity Analysis (RSA) metric, a technique coming from systems neuroscience [253], to measure the dissimilarity of HOLMES modular learned representations by comparing their patterns of response to various stimuli (various input Lenia patterns). One could imagine integrating such metric as training loss or regularizer for the learned representations. One relevant architecture in that direction is the *divergent discriminative feature accumulation* (DDFA) proposed by Szerlip et al. [246]. DDFA combines a mechanism to generate neural networks (Hyper-NEAT [419]) with some novelty search algorithm where novelty measures, akin to RSA, the dissimilarity of the newly generated features with respect to an archive of already-acquired representations. The intuitive idea of DDFA is to accumulate an archive of representations that discriminate the training dataset in maximally novel ways. Combining DDFA with some human guidance, *e.g.* by filtering non-interesting instances from the training set, and some hierarchical clustering, *e.g.* to reorganize the incoming dataset in various niches of patterns, could be a promising way to perform meta-diversity search in the targeted self-organizing systems.

Finally, with "meta" comes the essential notion of *abstraction*. But what's an appropriate level of abstraction? Self-organizing systems, and biological ones in particular, are notorious for their ability to coarse-grain information which is believed to lead to top-down causation mechanisms, where individual components adapt their actions based on estimates of coarse-grained, aggregated characteristics [473]. This coarse-grained estimates are also called *goal states* of the system by some biologists, in the sense that low-level elements spend energy toward achieving these specific, higher-level states, and this despite perturbations or changes in the local conditions [45][2]. From the perspective of the AI discovery assistant, knowing how to detect and characterize these specific goal states requires learning the appropriate scales of observation that are the most useful for predicting and exploring the system dynamics. Aligning the AI's target goal states with the system's endogenous goal states might be crucial for effectively understanding and exploring these systems [145].

2: This is for instance the position of Dr. Michael Levin, which we investigate more in depth in Chapter 8

12.1.2. Towards Richer Interactions with Humans

In Chapter 6, we have argued in favor of a shift from fully automated discovery to a concept of "assisted discovery" where human end-users, *e.g.* scientists, provide interactive guidance and instructions to the AI agent throughout exploration. Recently, Sigaud et al. [474] also advocated for what they called *teachable autotelic agents*: agents that could pursue their own goals but that would also be endowed with an additional capability to be taught so that they choose their goals in accordance with the expectation of their users. In Chapter 6, we proposed IMGEP-HOLMES with preference-based guidance as a potential implementation of this concept. However, there are several avenues to expand upon this work to foster a richer diversity of interactions, which may also be central in developing agents with more open-ended learning capabilities.

A first limitation of this work is that experiments were made with simulated end-users. A natural next step for future research would involve conducting experiments with actual human end-users, possibly utilizing the ADTOOL software discussed in Chapter 10.

A second limitation of the HOLMES architecture is that it remains quite rigid in the way it is isolating the different niches of patterns, as it employs a binary tree structure with fixed boundaries. This rigidity hinders flexible interactions, preventing humans from influencing the generation of these niches. Future work could incorporate human feedback to influence the organization of system discoveries, thereby enhancing the ability to guide exploration along desired trajectories.

Finally, offering humans greater control over when and how they provide feedback to the AI could facilitate more diverse and meaningful forms of communication. As discussed in the previous section, as the AI's goals become increasingly varied, multimodal, and abstract, the interactions with human users should evolve accordingly. Modular autotelic architectures, like HOLMES, may be used to that end by handling multiple goal spaces and channels of interactions in parallel.

A potential avenue for future investigation involves the utilization of *language* as a means of communication. As argued by Sigaud et al. [474], for autotelic agents to be considered *teachable*, it may be essential to

integrate enhanced natural language learning capabilities. As an initial step, humans could provide labels or textual descriptions to characterize the properties of AI-generated discoveries and to communicate the properties they want the AI agent to search for within the system. However, as we discussed earlier, it remains again uncertain whether natural human concepts would effectively characterize properties of the targeted self-organizing systems or help conceive new goals, particularly when dealing with systems that challenge even the understanding of scientists regarding the potential range of outcomes and their characterization.

Another promising avenue for future research, which we initiated in Chapter 7 and Chapter 8, involves environment-based guidance. In this approach, human input takes a different form: instead of communicating with the AI directly, humans introduce environmental cues or constraints within the explored environment. For example, in the sensorimotor Lenia web demo (available at this link), users can interactively influence the behaviors of discovered creatures by adding obstacles, manipulating matter to affect creature birth or death, or introducing attractive elements to control the creature trajectories. Similarly, in the work with gene regulatory networks (GRNs), we applied specific tests conditioned on the GRN's behavior that align with how humans typically study the behavior of more conventional animals. While these interactions occurred during evaluation, leveraging controllable human cues or constraints within the AI exploration process could offer a practical means of conveying *functional* goals to the AI system. Indeed, describing verbally desired morphological or dynamical attributes of a cellular collective can be quite challenging. A more practical approach for the human end-user to communicate with the AI agent could be to place specific elements in the environment and define target goals as desired interactions with those elements, *e.g.* aiming for the cellular collective consume certain chemicals and avoid others. Importantly, this could in turn help the human to find appropriate ways to communicate with the cellular collective, *e.g.* placing elements that can attract or repel the collective to steer its trajectory.

12.2. Applicative Perspectives in Sciences

In the second part of this manuscript (Part II), we have shown how even simple variants of goal-directed diversity search can be very helpful to address several challenging questions in Science, such as the search for robust forms of sensorimotor agency in cellular automata (Chapter 7) and the search for diverse "native" behaviors in gene regulatory networks (Chapter 8). They are many other applicative perspectives in which such digital discovery assistants may prove to be very useful. In this section we identify few specific directions that we believe are particularly promising or interesting, which we organize by applicative domain.

12.2.1. For Biology

One challenging but particularly interesting domain of application that we would like to target in future work, and that we have started to explore during the collaboration with the Poietis biotechnology company, is the one of tissue engineering. We refer the interested reader to Section 9.3

in Chapter 9 of this manuscript, where we detail three possible experimental campaigns that could be envisaged with actual morphogenetic systems (in particular fibroblast-collagen models) utilizing Poietis bioprinting device. We believe the proposed experimental directions could be useful for biologists both for fundamental research purposes but also for clinical applications, as discoveries might help shaping tissue maturation toward desired morphological and functional properties.

Following our work on gene regulatory networks (GRNs) discussed in Chapter 8, another potential direction is to apply the same framework to investigate the dynamics of gene expression in single cells *in vitro*. Additionally, within the scope of bio-engineered systems, the Levin lab has been working on *xenobots*, also called "living robots" (see Figure 2.10). The team has already employed evolutionary algorithms to optimize parameters (printing patterns) for creating xenobots with specific functionalities [61–63]. Utilizing the proposed diversity search algorithm could facilitate the discovery of a broader range of potentially interesting behaviors and functionalities within these systems.

12.2.2. For Enactive Artificial Intelligence

Researchers in AI increasingly recognize the potential of drawing inspiration from biology and self-organization to enhance the development of artificial agents and artificial neural networks [475, 476]. Unlike classical AI, which often separates control from the agent's physical characteristics, biological systems integrate functionality within their morphology, which has been coined "morphological computation" [269]. Although numerous challenges lie ahead, we are convinced that insights from biology and morphological computation will enable progress toward *enactivist AI* [270] where artificial agents could exhibit qualities found in biological organisms, such as adaptability, resilience, and versatility.

Discoveries in Chapter 7 were a first promising step in that direction: the discovered self-organized entities, *a.k.a.* sensorimotor agents, showcased strong robustness and generalization with respect to their mechanistic counterpart in classical AI. However, the discovered sensorimotor behaviors remained quite limited as we searched for sensorimotor creatures maximizing a specific reward (distance traveled in the grid) under a specific type of constraint (obstacles). We believe that searching for more advanced behavioral and environmental complexity in enactivist models like the Lenia environment has several promising avenues for future work. In particular, the following scientific questions could be investigated:

- Can forms of *goal-directedness* emerge?
- Can forms of *learning* and *memory* mechanisms emerge?
- Can forms of *cooperation* emerge between several individuals?
- Can forms of *intrinsic evolution* emerge within the environment?

To study the conditions that can lead to the emergence of such complex behaviors from low-level elements and local rules, more advanced search methods like the ones presented in the first part of this manuscript, notably *meta-diversity search* and *human guidance*, might be needed.

In addition to studying forms of agency, life and cognition in cellular automata models, transposing those insights for the growth and learning

of differentiable neural networks and graph-based systems is another promising avenue of research. We refer to Ha and Tang [476] for a survey of recent developments in collective intelligence for deep learning.

12.2.3. For Physics and Chemistry

Grizou et al. [64] served as the first proof of concept that curiosity-driven exploration algorithms, even in their simplest form, could be very useful to explore the dynamics of physico-chemical systems, in their case to explore the diverse behaviors of an oil-droplet system (see Figure 2.12).

Recently, Falk et al. [139] shown how the more advanced variants of these algorithms proposed in the first part of this manuscript, notably ones combining autotelic exploration with unsupervised learning of representations (Chapter 4) and human guidance (Chapter 6), could also be very useful to explore the space of behaviors of models used in physics and chemistry to describe synchronization of a set of coupled oscillators[3]. There are ongoing discussions with their team to apply similar algorithms to explore the phases of chiral matter in a real physics experimental setup, akin to the ones explored in Bililign et al. [477].

3: we discuss the results of Falk et al. [139] in Section 4.6 and Section 6.5

12.2.4. For Arts and Creativity

Finally, although our primary emphasis has been on the scientific utility of curiosity-driven algorithms, it's worth noting that many of the concepts and techniques developed in this research could be harnessed for purely artistic endeavors. Notably it would be very interesting to see whether algorithms combining meta-diversity search and human guidance could help foster human creativity and exploratory design in "artistic" self-organizing systems. Potentially interesting systems to explore span digital art systems such as cellular forms by Artist Andy Lomas [478, 479], mechanical drawing machines like the one explored by Roussel et al. [480], as well as "musical encounters" between human and artificial agents in self-organizing systems as explored by Armitage and Magnusson [481].

Appendices

Figure credits A.

B. Appendix of IMGEP-VAE

(a) Paper Website

(b) Interactive Visualizations

(c) Codebase

Figure B.1.: Scan (or click on) the above QR codes for accessing the paper's (a) companion website, (b) interactive visualizations, and (c) github repository

B.1. Additional Results and Figures

B.1.1. Discovered Patterns

Figure B.2 to Figure B.6 illustrate examples of discovered patterns per class (SLP, TLP, dead) for each algorithm. The shown examples patterns have been randomly sampled from the results of a single exploration repetition, a link with all patterns can be found on the paper website.

B.1.2. Visualization of Goal Spaces

The goal spaces of all IMGEP algorithms are visualized (Figure B.7) via two-dimensional reductions: PCA [482] and t-Distributed Stochastic Neighbor Embedding (t-SNE) [225]. The visualizations were constructed by using for each algorithm its goal space representations of all its discovered patterns from a single repetition experiment. All goal representations were normalized so that the overall minimum value became 0 and the maximum value 1 for each dimension. PCA was performed to detect the two principle components. T-SNE was executed using the default standard Euclidean distance metric and perplexity set to 50.

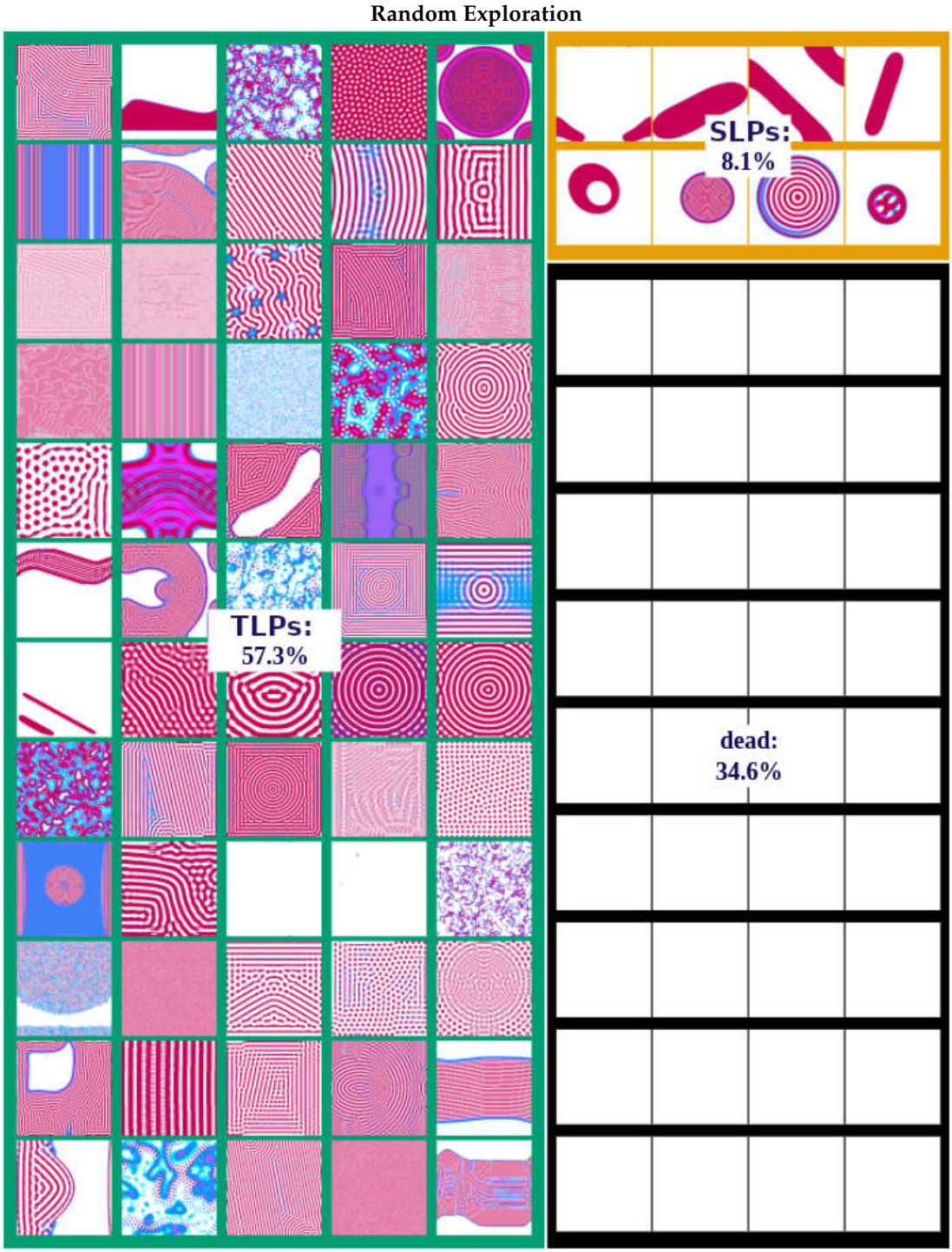


Figure B.2.: Randomly selected examples of patterns discovered by the random exploration algorithm during a single exploration with 5000 iterations.

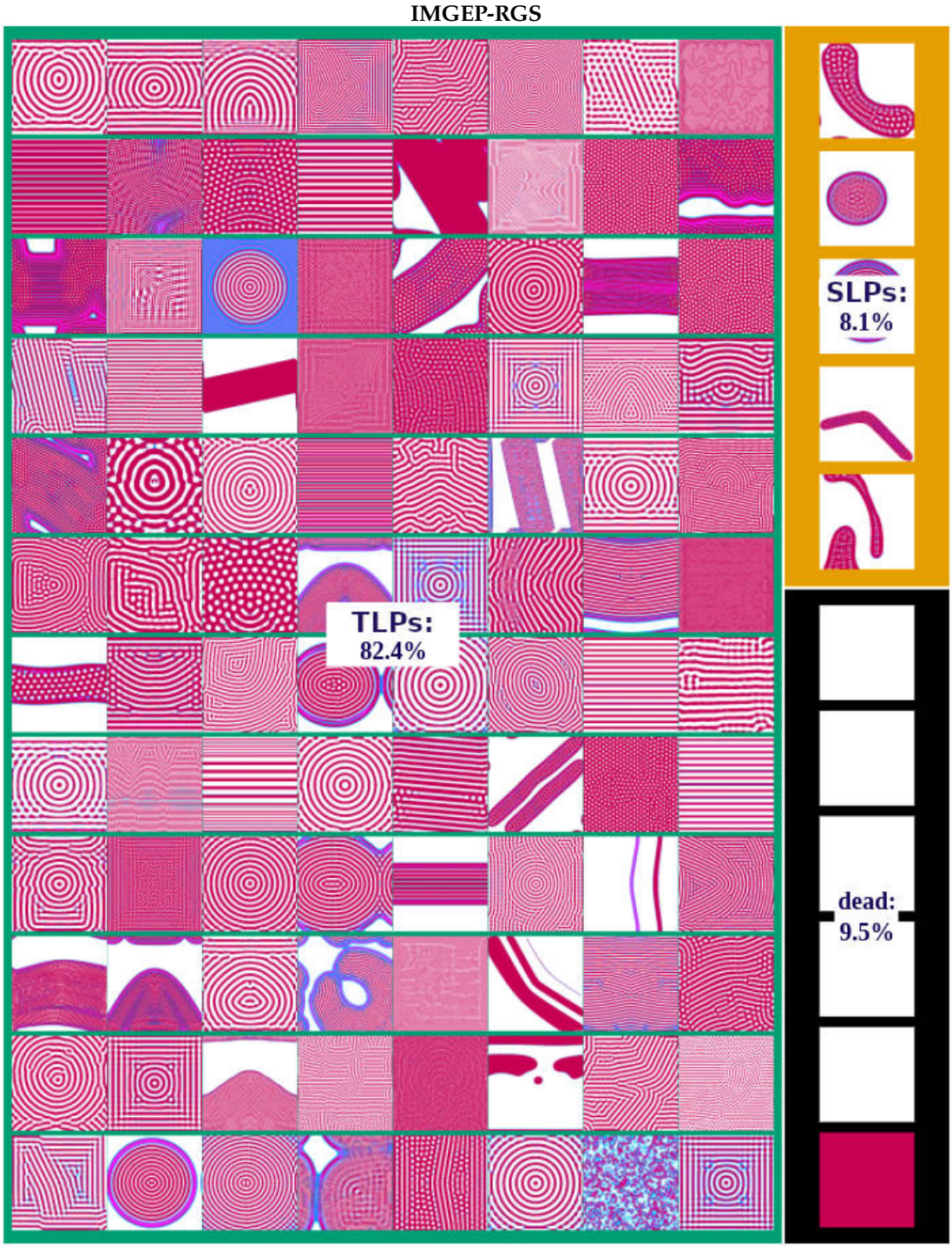


Figure B.3.: Randomly selected examples of patterns discovered by the IMGEP-RGS algorithm during a single exploration with 5000 iterations. The full database and an interactive visualization of the IMGEP-RGS goal space can be found at this link:

IMGEP-RGS discoveries

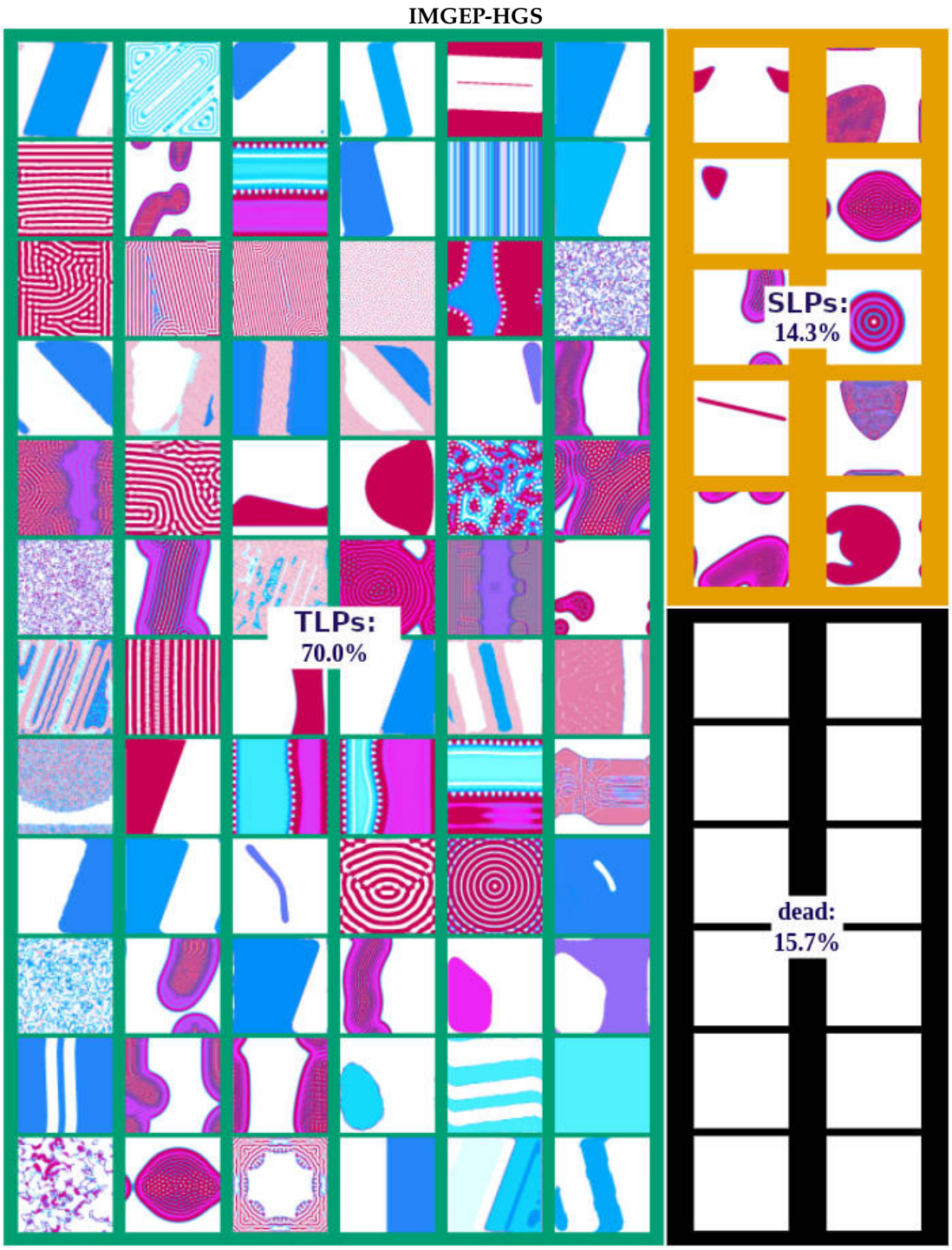


Figure B.4.: Randomly selected examples of patterns discovered by the IMGEP-HGS algorithm during a single exploration with 5000 iterations. The full database and an interactive visualization of the IMGEP-RGS goal space can be found at this link:

IMGEP-HGS discoveries

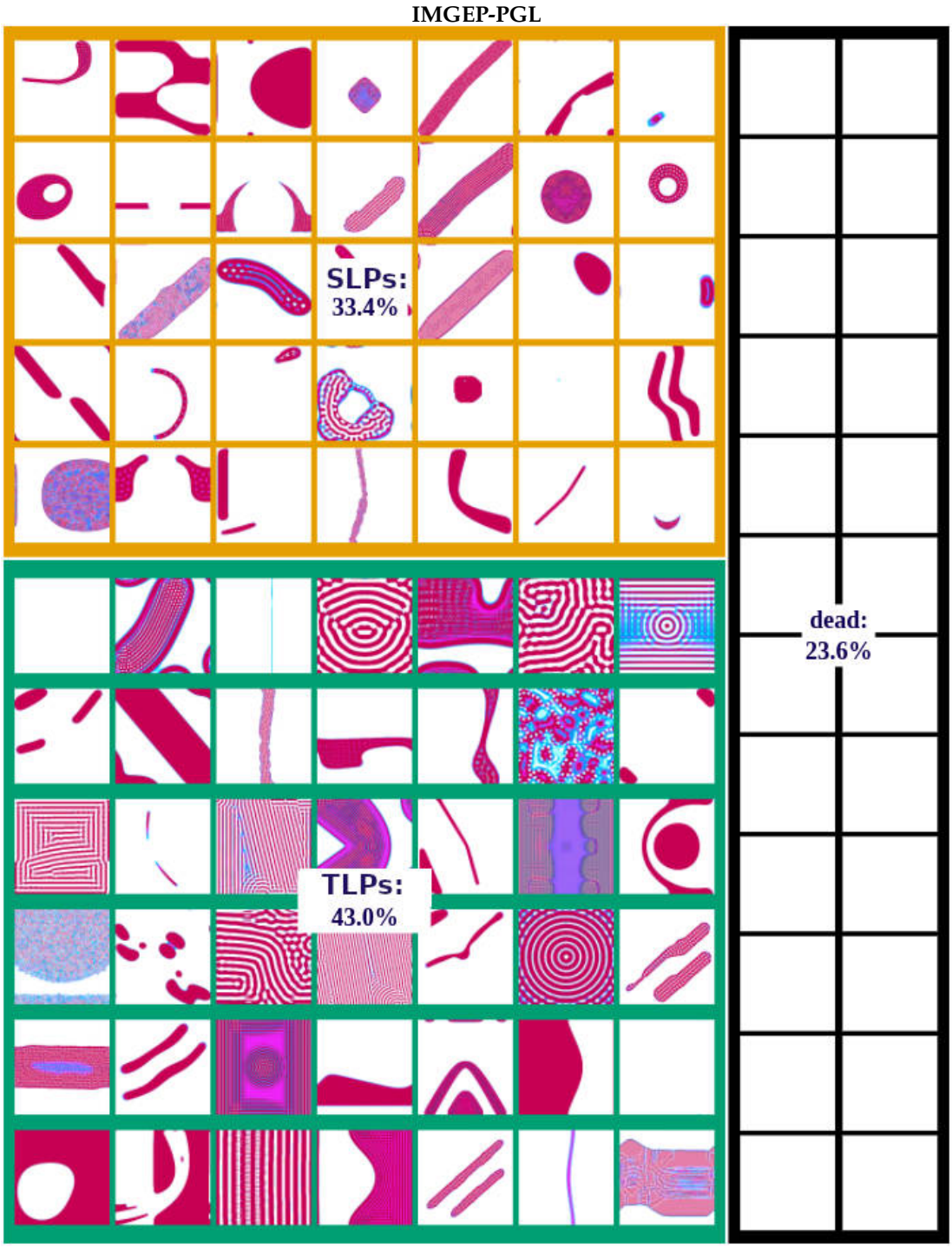


Figure B.5.: Randomly selected examples of patterns discovered by the IMGEP-PGL algorithm during a single exploration with 5000 iterations.

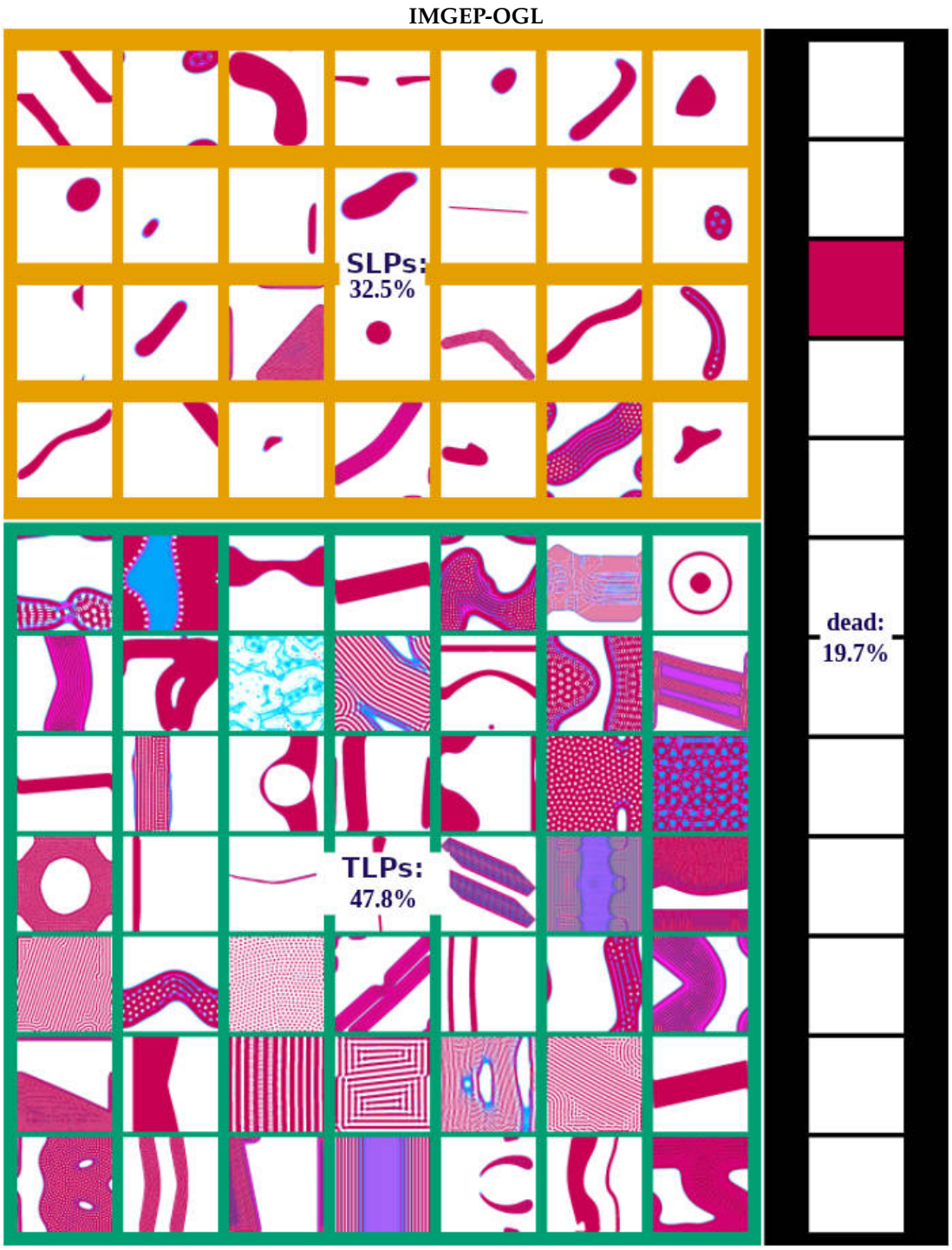


Figure B.6.: Randomly selected examples of patterns discovered by the IMGEP-OGL algorithm during a single exploration with 5000 iterations. The full database and an interactive visualization of the IMGEP-RGS goal space can be found at this link:

IMGEP-OGL discoveries

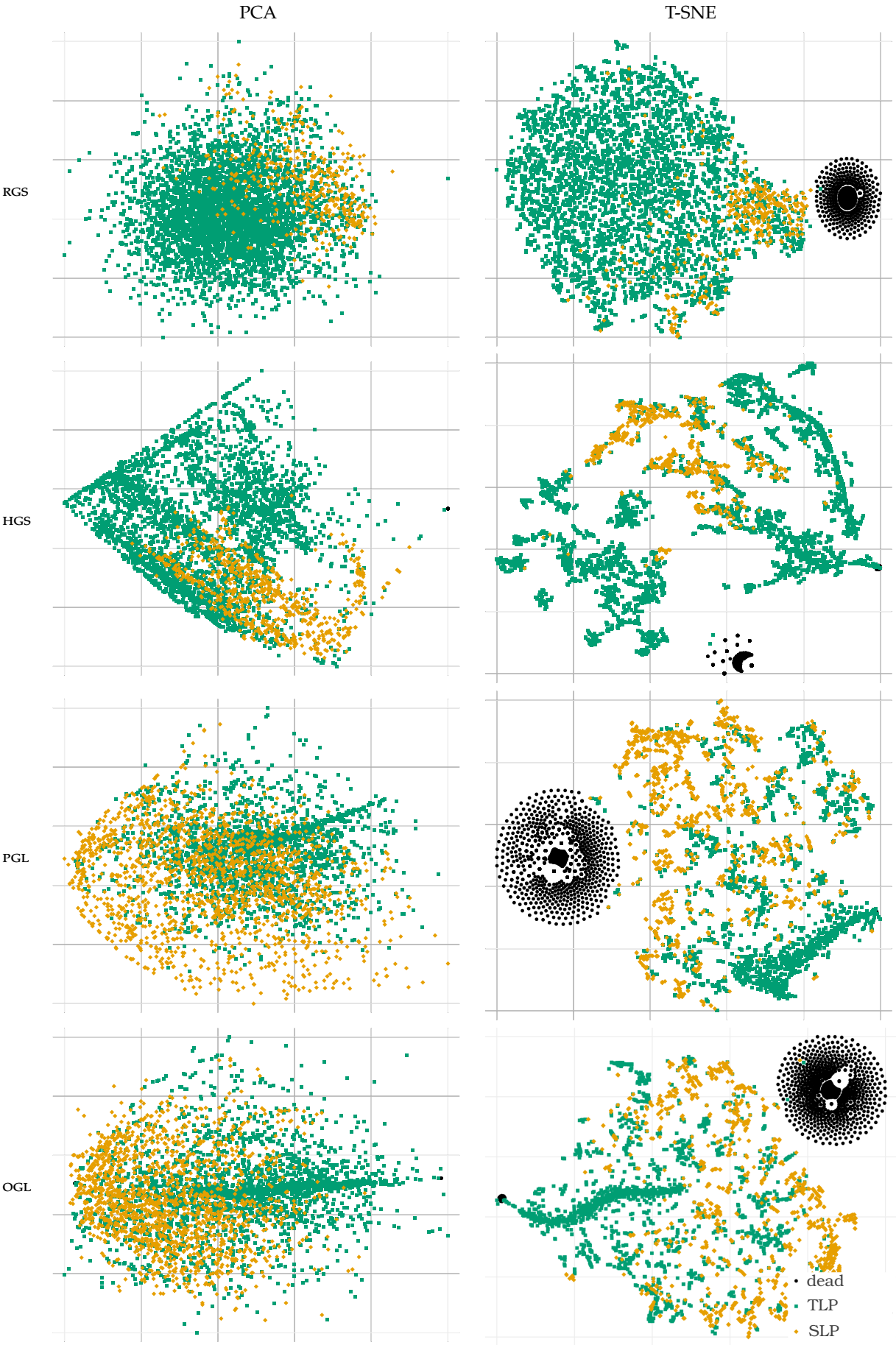


Figure B.7.: PCA and t-SNE visualization of the goal spaces for the IMGEP variants show that HGS has more area (PCA) and clusters (t-SNE) for TLPs compared to learned goal spaces (PGL/OGL) and vice versa for SLPs. t-SNE shows that the hand-defined goal space (HGS) and learned goal spaces (PGL/OGL) structure and cluster more the discovered patterns compared to random goal space (RGS).

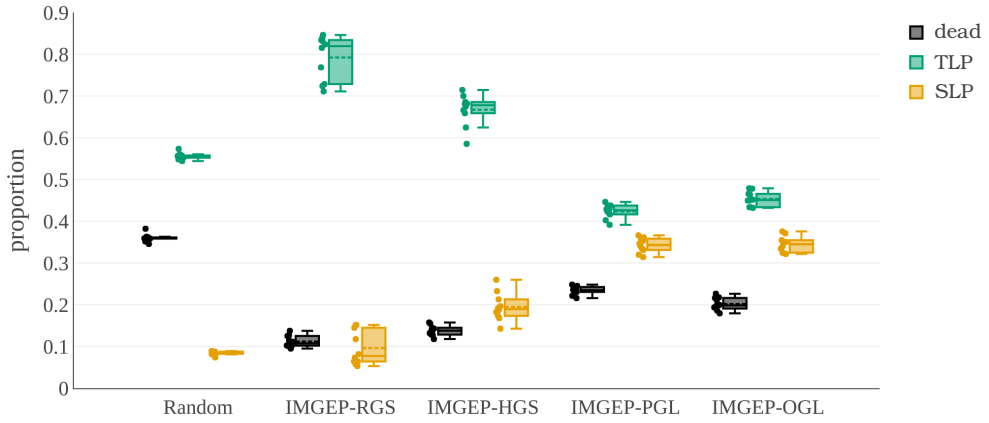


Figure B.8.: Proportion of patterns for each class and algorithm. Each dot besides the boxplot shows the proportion of found patterns for each repetition ($n = 10$). The box ranges from the upper to the lower quartile. The whiskers represent the upper and lower fence. The mean is indicated by the dashed line and the median by the solid line.

B.1.3. Proportion of Discovered Patterns of Different Classes

We used the measure of diversity of the found patterns to compare the performance of algorithms (Figure 4.7). Another measure to compare the algorithms is the average proportion of dead, SLPs, TLPs patterns discovered by each algorithm (Figure B.8). For SLP patterns the results follow the diversity results from Figure 4.7, *i.e.* OGL and PGL find the highest proportion of SLP patterns, followed by HGS, then RGS and Random. A corollary is that RGS, random and HGS find in proportion a higher percentage of TLP patterns than OGL and PGL. As the number of *diverse* TLP patterns is as high for OGL and PGL (Figure B.12d), this shows the higher sample efficiency of OGL and PGL to find diverse patterns. In contrast, Random, RGS and HGS approaches tend to find TLP patterns that are more similar to each other on average.

B.2. Implementation Details and Hyperparameter Settings

B.2.1. Lenia Settings

A full description of Lenia can be found in Subsection 3.2.2. For all experiments the following configurations of Lenia were used:

- Grid size: 256×256 ($A \in [0,1]^{256\times256}$)
- Number of steps: $T = 200$
- Exponential growth mapping: $G(u;\mu,\sigma) = 2\exp\left(-\frac{(u-\mu)^2}{2\sigma^2}\right) - 1$
- Exponential kernel function: $K_C(r) = \exp\left(\alpha - \frac{\alpha}{4r(1-r)}\right)$, with $\alpha = 4$
- Kernel shell: $K_S(r;\beta) = \beta_{\lfloor Br \rfloor} K_C(Br \bmod 1)$, with $\beta = (\beta_1, \beta_2, \beta_3)$

The controllable parameters of Lenia are the kernel size $R \in \mathbb{N}$, time step $T \in \mathbb{N}$, $\mu \in \mathbb{R}$ and $\sigma \in \mathbb{R}$ that control the growth mapping, and $\beta_1, \beta_2, \beta_3 \in \mathbb{R}$ that control the kernel shell. Additionally the initial state $A^{t=1} \in [0,1]^{256\times256}$ controls the system dynamics.

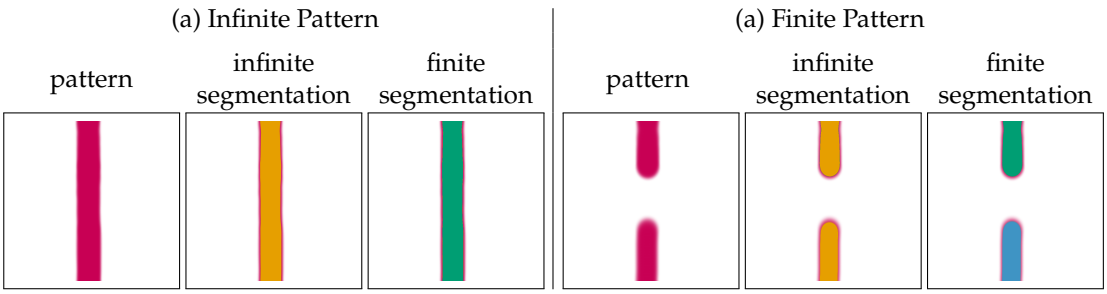


Figure B.9.: Classification of finite and infinite patterns. Infinite patterns form loops between image borders which are identified if a segment is connected between two borders in the infinite and finite segmentation. Finite patterns form no loops. They have connected segments between borders in the infinite but not finite segmentation. Segments are colorized in yellow, green and blue.

B.2.2. Classification of Lenia Patterns

We classified 3 types of Lenia patterns: dead, SLPs and TLPs. The classifiers only categorize the final pattern into which the Lenia system morphs after $T = 200$ time steps.

Dead Classifier: A pattern is classified as dead if the activity of all cells is either 0 or 1.

SLP Classifier: A pattern is classified as an SLP if it is a *finite* and *connected* pattern of activity. Cells x, y are connected as a pattern if both are active ($A(x) \geq 0.1$ and $A(y) \geq 0.1$) and if they influence each other. Cells influence each other when they are within their radius of the kernel K as defined by the parameter R.

Furthermore, the connected pattern must be finite. In Lenia finite and infinite patterns can be differentiated because the opposite borders of Lenia's cell grid are connected, so that the space is similar to a ball surface. Thus, a pattern can loop around this surface making it infinite. We identify infinite patterns as follows. First, all connected patterns are identified for the case of assuming an infinite grid cell, *i.e.* opposite grid cell borders are connected. Second, all connected patterns for the case of a finite grid cell, *i.e.* opposite grid cell borders are not connected, are identified. Third, for each border pair (north-south and east-west) it is tested if cells within a distance of R from both borders exists, that are part of a connected pattern for the infinite and finite grid cell case. If such a pattern exists than it is assumed to be infinite, because it loops around the grid cell surface of Lenia (Figure B.9, a). All other patterns are considered to be finite (Figure B.9, b). Please note that this method can not identify certain infinite patterns that loop over several borders, for example, if a pattern connects the north to east and then the west to south border (see the third SLP in Figure B.2 for an example).

Moreover, there are two additional constraints that an SLP pattern must fulfill. First, the cells of the connected pattern $P = \{x_1, \ldots, x_n\}$ must have at least 80% of all activation, *i.e.* $\sum_{x \in P} A(x) \geq 0.8 \sum_{\forall y} A(y)$. Second, a pattern must exists for the last two time steps ($t = T$ and $t = T - 1$). Both constraint are used to avoid that too small patterns or chaotic entities which change drastically between time steps are classified as SLPs.

TLP Classifier: We also classified TLP patterns which are all entities that were not dead and not an SLP. These patterns spread usually over the whole state space and are connected via borders.

B.2.3. Statistical Measures for Lenia Patterns

We defined five statistical measurements for the final patterns $A^{t=T}$ that emerge in Lenia. The measures were used as features for hand-defined goal spaces of IMGEPs and to define partly the analytic behavior space in which the results of the exploration experiments were compared.

Activation mass M_A: Measures the sum over the total activation of the final pattern and normalizes it according to the size of the Lenia grid:

$$M_A = \frac{1}{L^2} \sum_x A^{t=T}(x) \,,$$

where $L^2 = 256 \cdot 256$ is the number of cells of the Lenia system.

Activation volume V_A: Measures the number of active cells and normalizes it according to the size of the Lenia grid:

$$V_A = \frac{1}{L^2} \left| \{ \forall x : A^{t=T}(x) > \epsilon \} \right| \text{ with } \epsilon = 10^{-4}.$$

Activation density D_A: Measures how dense activation is distributed on average over active cells:

$$D_A = \frac{M_A}{V_A}.$$

Activation asymmetry A_A: Measures how symmetrical the activation is distributed according to an axis that starts in the center of the patterns activation mass and goes along the last movement direction of this center. This measure was introduced to especially characterize SLP patterns. The center of the activity mass is usually also the center of the SLPs and analyzing the activity along their movement axis measures how symmetrical they are.

As a first step, the center of the activation mass is computed for every time step of the Lenia simulation and the Lenia pattern recentered to this location. This ensures that the center is all the time correctly computed in the case the SLP moves and reaches one border to appear on the opposite border in the uncentered pattern. The center $(\bar{x}, \bar{y})_t$ for time step t is calculated by:

$$(\bar{x}, \bar{y})_t = \left(\frac{M_{10}}{M_{00}}, \frac{M_{01}}{M_{00}} \right) \text{ with } M_{pq} = \sum_x \sum_y x^p y^q A^t(x, y) \,,$$

where M_{pq} measures the image moment (or raw moment) of order $(p+q)$ for $p, q \in \mathbb{N}$.

Based on the center $(\bar{x}, \bar{y})_t$ the pattern A^t is recentered to A^t_C by shifting the x and y indexes according to the center:

$$A^t_C(x, y) = A^t((x - \bar{x}) \bmod L, (y - \bar{y}) \bmod L) \,, \tag{B.1}$$

where L is width and length of the Lenia grid and the indexing is $x, y = 0, \ldots, L-1$. After each time step the center is recomputed and the pattern recentered:

$$A^{t=1} \underset{\text{recenter}}{\longrightarrow} A^{t=1}_C \underset{\text{Lenia step}}{\longrightarrow} A^{t=2} \underset{\text{recenter}}{\longrightarrow} A^{t=2}_C \underset{\text{Lenia step}}{\longrightarrow} \ldots \,.$$

Please note, the simulations and all figures of patterns in the paper are done with the uncentered pattern. The centered version is only computed for the purpose of statistical measurements.

The recenter step by $(\bar{x}, \bar{y})_t$ defines also the movement direction of the activity center:

$$(m_x, m_y)_t = (\bar{x}, \bar{y})_t - \left(x^{\text{mid}}, y^{\text{mid}}\right) = \left(\bar{x} - x^{\text{mid}}, \bar{y} - y^{\text{mid}}\right) ,$$

where $x^{\text{mid}}, y^{\text{mid}} = \frac{L-1}{2}$ are the coordinates for the grid's middle point. A line can be defined that starts in the midpoint $\left(x^{\text{mid}}, y^{\text{mid}}\right)$ of the final centered pattern $A_C^{t=T}$ and goes in and opposite to the movement direction of the activity mass center $(m_x, m_y)_{t=T}$. This line separates the grid in two equal areas. The asymmetry is computed by comparing the activity amount in the grid right M_A^{right} and left M_A^{left} of the line. The normalized difference between both sides is the final measure:

$$A_A = \frac{1}{M_A}(M_A^{right} - M_A^{left}).$$

Activation centeredness C_A: Measures how strong the activation is distributed around the activity mass center:

$$C_A = \frac{1}{M_A} \sum_x \sum_y w_{xy} \cdot A_C^{t=T}(x, y) \quad \text{with} \quad w_{xy} = \left(1 - \frac{d(x, y)}{\max_{y,x} d(x, y)}\right)^2 ,$$

where $d(x, y) = \sqrt{(x - x^{\text{mid}})^2 + (y - y^{\text{mid}})^2}$ is the distance from the point (x, y) to the center point $\left(x^{\text{mid}}, y^{\text{mid}}\right)$. $A_C^{t=T}(x, y)$ is the centered activation that is updated every time step as for the asymmetry measure (Eq. B.1). The weights w_{xy} decrease the farer a point is from the center. Thus, patterns that are concentrated around the center have a high value for C_A close to 1. Whereas, patterns whos activity is distributed throughout the whole grid have a smaller value. For patterns that are equally distributed $(\forall_{x,x'} : A(x) = A(x'))$ is $C_A = 0$ defined as centeredness measure.

B.2.4. Sampling of Parameters for Lenia

There are two operations to sample parameters: 1) random initialization and 2) mutating an existing parameter θ. CPPNs are used for the random initialization and mutation of the initial pattern $A^{t=1}$. The details of this process are described in the next section. Afterwards, the initialization and mutation of Lenia's parameter that control its dynamics are described.

B.2.4.1. Sampling of Start Patterns for Lenia via CPPNs

The CPPNs generate Lenia activity patterns cell by cell by taking as input a bias value, the x and y coordinate of the cell (mapped to $x = [-2, 2]$ and $y = [-2, 2]$) and its distance d to the grid center (Figure 4.3a). Their output p defines the activity of the cell ($A(x, y) = 1 - |p|$) between 0 and 1 for the given (x, y) coordinate.

Table B.1.: Settings for the sampling of CPPNs to generate Lenia's initial states.

Parameter	Value
Initial number of hidden neurons	4
Initial activation functions	gauss, sigm
Initial connections	random connections with probability 0.6
Initial synapse weight	Gaussian distribution with $\mu = 0$, $\sigma = 0.4$
Synapse weight range	$[-3, 3]$
Mutation neuron add probability	0.02
Mutation neuron delete probability	0.02
Mutation connection add probability	0.05
Mutation connection delete probability	0.01
Mutation rate of activation functions	0.1
Mutation rate of synapse weights	0.05
Mutation replace rate of synapse weights	0.06
Mutation power of synapse weights σ_M	1
Mutation enable/disable rate of synapse weights	0.02

CPPNs consist of several hidden neurons (typically 4 to 6 in our experiments) that can have recurrent and self connections. Each CPPN has one output neuron. Two activation functions, gaussian and sigmoidal, were used for the hidden neurons and the output neuron:

$$\text{gauss}(x) = 2\exp\left(-(2.5x)^2\right) - 1\,,\ \text{sigm}(x) = 2\left(\frac{1}{1+\exp(-5x)}\right) - 1\,. \quad \text{(B.2)}$$

To randomly initialize a Lenia initial pattern $A^{t=1}$ a CPPN is randomly sampled by sampling the number of hidden neurons, the connections between inputs and neurons and neurons to neurons, their connection weights and the activation functions for neurons. Afterwards the initial pattern is generated by it. In the history $\mathcal{H}$ of the IMGEPs is then the CPPN as part of the parameter θ added. If the parameter is mutated, then the weights, connections and activation functions of the CPPN are mutated and the new initial pattern $A^{t=1}$ generated by it. A CPPN is defined over its network structure (number of nodes, connections of nodes) and its connection weights.

We used the NEAT-PYTHON[1] package for the random generation and mutation of CPPNs. It is based on the NeuroEvolution of Augmenting Topologies (NEAT) algorithm for the evolution of neural networks [483]. The meta-parameters for the initialization and mutation of CPPNs are listed in Table B.1. The random sampling and mutation of CPPNs allows to generate complex patterns as illustrated in Figure 4.3a and Figure 4.3b. Please note, the NEAT-PYTHON package allows also the setting and mutation of response and bias weights for each neuron. Those settings were not used for the experiments. Moreover, we adjusted the sigmoid and Gaussian function in the neat-python package to the ones defined in Eq. B.2 to be able to replicate similar images as in [194].

1: https://github.com/CodeReclaimers/neat-python

B.2.4.2. Sampling of Lenia's Dynamic Parameters

The parameters that control the dynamics of Lenia ($R, T, \mu, \sigma, \beta_1, \beta_2, \beta_3$) are initialized and mutated via uniform and Gaussian distributions. Table

B.2 lists for each parameter the meta-parameters for their initialization and mutation. Each parameter is initialized by an uniform sampling $\theta_i \sim \mathcal{U}(a, b)$ with a and b as upper and lower border. An existing parameter θ_i is mutated by the following equation:

$$\theta_i \leftarrow [\theta_i + \mathcal{N}(0, \sigma_M)]_b^a \ ,$$

where σ_M is the mutation power and $[n]_b^a = \min(\max(n, a), b)$ is the clip function. For natural numbers $\theta_i \in \mathbb{N}$ the resulting value is rounded.

Parameter	Type	Value Range	Mutation σ_M
R	$\mathbb{N}$	$[2, 20]$	0.5
T	$\mathbb{N}$	$[1, 20]$	0.5
μ	$\mathbb{R}$	$[0, 1]$	0.05
σ	$\mathbb{R}$	$[0.001, 0.3]$	0.01
$\beta_1, \beta_2, \beta_3$	$\mathbb{R}$	$[0, 1]$	0.05

Table B.2.: Settings for the initialization and mutation of Lenia system parameters θ.

B.2.5. IMGEP Details

B.2.5.1. VAE Framework and Implementation Details

We applied deep variational autoencoders (VAEs) to learn latent representations of Lenia patterns. VAEs have two components: a neural *encoder* and *decoder*. The encoder $q(\mathbf{z}|\mathbf{x}, \chi)$ represents a given data point x in a latent representation $\mathbf{z}$. In variational approaches the encoder describes a data point by a representative distribution in the latent space of reduced dimension d. A standard Gaussian prior $p(\mathbf{z}) = \mathcal{N}(0, I)$ and a diagonal Gaussian posterior $q(\mathbf{z}|\mathbf{x}, \chi) = \mathcal{N}(\mu, \sigma)$ are used. Given a data point x, the encoder outputs the mean μ and variance σ of the representative distribution in the latent space. The decoder $p(\mathbf{x}|\mathbf{z}, \psi)$ tries to reconstruct the original data x from a sampled latent representation $\mathbf{z}$ for the distribution given by the encoder. Under these assumptions, training is done by maximizing the computationally tractable evidence lower bound (with $\beta = 1$):

$$\mathcal{L}(\chi, \psi) = \underbrace{\mathbb{E}_{\mathbf{z}\sim q_\chi(\mathbf{z}|\mathbf{x})}[\log p_\psi(\mathbf{x}|\mathbf{z})]}_{a} - \beta \times \underbrace{\mathbb{D}_{KL}[q_\chi(\mathbf{z}|\mathbf{x})\|p(\mathbf{z})]}_{b} \ . \tag{B.3}$$

The first term represents the expected reconstruction accuracy (a), the second the KL divergence of the approximate posterior to the prior (b).

$$b = \mathbb{D}_{KL}[\mathcal{N}(\mu(\mathbf{x}), \Sigma(\mathbf{x}))\|\mathcal{N}(0, I)] = \sum_{i=1}^{d} \underbrace{\mathbb{D}_{KL}[\mathcal{N}(\mu(\mathbf{x})_i, \sigma(\mathbf{x})_i)\|\mathcal{N}(0, 1)]}_{b_i} \ . \tag{B.4}$$

In particular, we used the β-VAE framework [209] which re-weights the b term by a factor $\beta > 1$, aiming to enhance the disentangling properties of the learned latent factors. The weight factor was set to $\beta = 5$ for all experiments. All β-VAEs used for this paper use the same architecture (Table B.3). The encoder network has as input the Lenia pattern and as outputs for each latent variable $\mathbf{z}_i$ the mean μ_i and log-variance $\log(\sigma_i^2)$.

Encoder	Decoder
Input pattern A: $256 \times 256 \times 1$	Input latent vector z: 8×1
Conv layer: 32 kernels 4×4, stride 2, 1-padding + ReLU	FC layers : 256 + ReLU, $16 \times 16 \times 32$ + ReLU
Conv layer: 32 kernels 4×4, stride 2, 1-padding + ReLU	TransposeConv layer: 32 kernels 4×4, stride 2, 1-padding + ReLU
Conv layer: 32 kernels 4×4, stride 2, 1-padding + ReLU	TransposeConv layer: 32 kernels 4×4, stride 2, 1-padding + ReLU
Conv layer: 32 kernels 4×4, stride 2, 1-padding + ReLU	TransposeConv layer: 32 kernels 4×4, stride 2, 1-padding + ReLU
FC layers : 256 + ReLU, 256 + ReLU, FC: 2×8	TransposeConv layer: 32 kernels 4×4, stride 2, 1-padding

Table B.3.: β-VAE architecture for the pretrained and online experiments.

The decoder takes as input during the training for each latent variable a sampled value $z_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$. For validation runs and the generation of all reconstructed patterns shown in figures the decoder takes the mean $z_i = \mu_i$ as input. Its output is the reconstructed pattern.

The training objective (Eq. B.3) results in the following loss function:

$$\text{Loss}(x, \hat{x}, \mu, \sigma) = -a + \beta \sum_{i=1}^{d} b_i \ , \tag{B.5}$$

where x are the input patterns, $\hat{x}$ are the reconstructed patterns, μ, σ are the outputs of the decoder network and d is the number of latent dimensions. The reconstruction accurray part a of the loss is given by a binary cross entropy with logits:

$$a = \frac{1}{N} \sum_{n=1}^{N} \sum_{j=1}^{L^2} \left(x_{j,n} \cdot \log \sigma(\hat{x}_{j,n}) + (1 - x_{j,n}) \cdot \log(1 - \sigma(\hat{x}_{j,n})) \right) \ ,$$

where the index j is for the single cells (pixel) of the pattern and n for the datapoint in the current batch, N is the batch size and $\sigma(x) = \frac{1}{1+e^{-x}}$. The KL divergence terms b_i are given by:

$$b_i = \frac{1}{2 \cdot N} \sum_{n=1}^{N} \left(\sigma_{i,n}^2 + \mu_{i,n}^2 - \log(\sigma_{i,n}^2) - 1 \right) \ .$$

All β-VAEs were trained for 2000 epochs and initialized with pytorch default initialization. We used the Adam optimizer [484] (lr = 1e−3, $\beta_1 = 0.9$, $\beta_2 = 0.999$, ϵ = 1e−8, weight decay=1e−5) and batch size=64.

The patterns from the datasets were augmented by random x and y translations (up to half the pattern size and with probability 0.3), rotation (up to 40 degrees and with probability 0.3), horizontal and vertical flipping (with probability 0.2). The translations and rotations were preceded by spherical padding to preserve Lenia spherical continuity.

B.2.5.2. IMGEP Variants

IMGEP-HGS (hand-defined goal space): The 5 statistical features that are given in Section B.2.3 were used to define the goal space of the IMGEP-HGS approach (Algorithm. 4). Goals in this space were sampled from a uniform distribution within the ranges defined in Table B.4.

Feature	min	max	Feature	min	max
mass M_A	0	1	asymmetry A_A	−1	1
volume V_A	0	1	centeredness C_A	0	1
density D_A	0	1			

Table B.4.: HGS Goal Space Ranges

IMGEP-RGS (random goal space): IMGEP with a goal space defined by an encoder network with random weights (Algorithm. 4). The encoder has the same architecture than the VAE encoder used for IMGEP with learned goal spaces, and weights are initialized with the Xavier method [485]. In the other IMGEP algorithms, goals are sampled uniformly within fixed-range boundaries. In the case of random goal spaces, we do not know in advance in which region of the space goals will be encoded. We therefore set the sampling range to the minimum and maximum value of the latent representations of all so far explored patterns.

IMGEP-PGL (prelearned goal space): IMGEP with a goal space defined by a β-VAE that was trained before the exploration starts (Algorithm. 4). The β-VAE was trained on a dataset of 558 pre-collected Lenia patterns (train: 75%, validation: 10%, test: 15%), half being manually identified SLP patterns by [40], the other half random CPPN patterns. The best β-VAE model from training, achieving the highest validation accuracy, was used as goal space for the exploration. Exploration involved uniform sampling within a $[-3, 3]^8$ hyperrectangle, chosen to align with the β-VAE's prior standard normal distribution.

IMGEP-OGL (online learned goal space): The β-VAE model is trained every $K = 100$ explorations for 40 epochs, totaling 2000 epochs (less if insufficient data is available after the initial T runs to start training). The dataset comprises non-dead patterns collected during exploration, with every tenth pattern designated for validation (10%). At the initial period of training, the training dataset has approximately 50 patterns and at the last period approximately 3425 patterns. The validation dataset only serves for checking purposes and has no influence on the learned goal space. Importance sampling is employed, favoring newly discovered patterns with higher weights in training. A weighted random sampler is utilized, drawing half of the patterns from recent discoveries and the rest from the dataset. Newer patterns have a probability of $\frac{0.5}{N}$, while older ones have $\frac{0.5}{|D_{\mathcal{I}}|-N}$, enhancing the influence of newer patterns on training. Goal sampling follows a uniform distribution within the $[-3, 3]^8$ hyperrectangle, similar to PGL.

Algorithm 4: IMGEP-HGS, IMGEP-RGS and IMGEP-PGL

Initialize goal space encoder R with handdefined features (HGS), random weights (RGS), or pretrained weights (PGL)
for i=1 *to* N **do**
  if $i < N_{init}$ **then** `// Initial random iterations`
    Sample random parameter $\theta \sim \mathcal{U}(\Theta)$
  else `// Goal-directed IM iterations`
    Sample target goal $g \sim \mathcal{G}(\mathcal{H})$
    Infer experiment parameters to achieve goal $\theta \sim \Pi(g, \mathcal{H})$
  Execute experiment with θ and observe system outcome o
  Characterize behavior $z = R(o)$
  Write (θ, o, z) to history $\mathcal{H}$

B.2.5.3. VAE Training Results

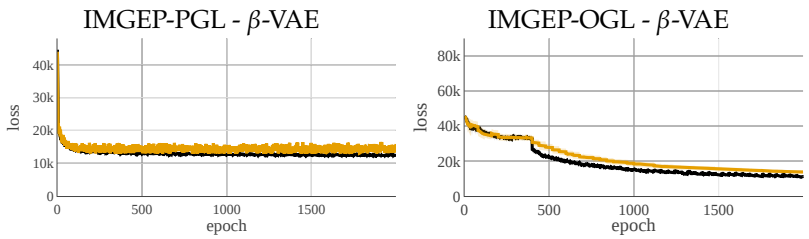


Figure B.10.: Averaged learning curves ($n = 10$) of the β-VAEs for the IMGEP-PGL and OGL experiments.

The β-VAE learning saturates for both the precollected dataset and the online collected dataset (Figure B.10). Their ability to reconstruct patterns based on the encoded latent representation is also qualitatively similar. For both datasets the β-VAEs are able to learn the general form of the activity pattern. However, the compression of the images to a 8-dimensional vector results in a general blurriness in the reconstructed patterns. β-VAEs are not able to encode finer details and textures of patterns.

B.2.6. Measurement of Diversity in the Analytic Parameter and Behavior Space

Analytic Parameter and Behavior Space The analytic parameter space was constructed by the features defined in Table B.5. The β-VAE was trained on initial patterns $A^{t=1}$ used during the experiments. The analytic behavior space was constructed by the features defined in Table B.5. This β-VAE was trained on final patterns $A^{t=200}$ observed during the experiments. The datasets for both β-VAEs were constructed by randomly selecting 42500 patterns (37500 as training set, 5000 as validation set) from the experiments of all algorithms and each of their 10 repetitions. The β-VAEs used the same structure, hyper-parameters, loss function and learning algorithm as described in Section D.3. They were trained for more than 1400 epochs. The encoder which resulted in the minimal validation set error during the training was used. According to their reconstructed patterns they can represent the general form of patterns but often not individual details such as their texture (Figure B.11).

Table B.5.: Features of the analytic parameter and behavior space with their min and max values.

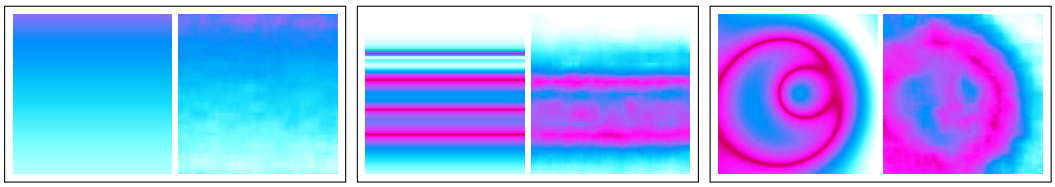
Analytic Parameter Space Definition

Parameter	min	max
R	1	20
T	2	10
μ	0	1
σ	0	0.3
$\beta_1, \beta_2, \beta_3$	0	1
β-VAE latent 1 to 8 (trained on intial states $A^{t=1}$)	-5	5

Analytic Behavior Space Definition

Parameter	min	max
mass M_A	0	1
volume V_A	0	1
density D_A	0	1
asymmetry A_A	-1	1
centeredness C_A	0	1
β-VAE latent 1 to 8 (trained on final patterns $A^{t=T}$)	-5	5

a) Reconstruction Examples of the Analytic Parameter Space β-VAE



b) Reconstruction Examples of the Analytic Behavior Space β-VAE

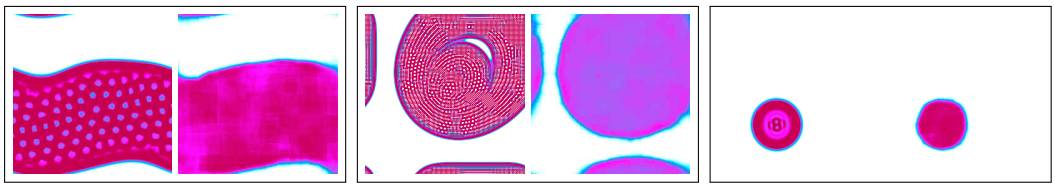
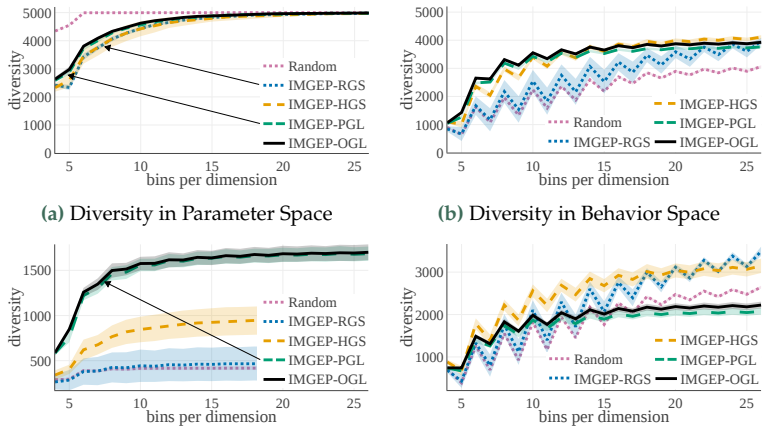


Figure B.11.: Examples of patterns (left) and their reconstruction (right) by the β-VAE used for the analytic parameter (a) and behavior space (b). The patterns are sampled from the validation dataset.

Diversity Metric We use a binning-based diversity metric which measures how much area the algorithms explored in the analytic behavior and parameter spaces (see Subsection 2.1.2). Here we use a regular binning where the spaces are discretized with a spatial grid between a minimum and maximum border defined for each space dimension (Table B.5). The areas with values falling below the minimum or above the maximum border are counted as two additional bins. In the main paper, we used 7 bins per dimensions to measure diversity (Figure 4.7). We analyzed the impact of the number of bins per dimension on the final diversity measure in Figure B.12. Although the diversity difference between algorithms depends on the number of bins per dimension for each space, the order of the algorithms is generally invariant to it. Only if the number of bins per dimension grows large (>10) the order of the algorithms changes in some cases. The order starts to follow the same order as seen for the proportion of identified patterns (Figure B.8). In this case the discretization of the space becomes too fine. Each pattern falls into its own discretized area. We chose therefore a smaller number of bins per dimension of 7 to compare the algorithms in a meaningful way.



(a) Diversity in Parameter Space

(b) Diversity in Behavior Space

(c) Behavior Space Diversity for SLPs

(d) Behavior Space Diversity for TLPs

Figure B.12.: Influence of the number of bins per dimensions on the diversity measure. Depicted is the average diversity ($n = 10$) with the standard deviation as shaded area.

C. Appendix of IMGEP-HOLMES

(a) Paper Website

(b) Codebase

Figure C.1.: Scan (or click on) the above QR codes for accessing the paper's (a) companion website with interactive visualizations and (b) github repository

C.1. Evaluation of the diversity for the monolithic BC spaces variants

C.1.1. Construction of 5 analytic BC spaces

Each set of BC features relies either on *engineered* representation based on existing image descriptors from the literature or on *pretrained* representations unsupervisedly learned on Lenia patterns. Those BCs were constructed to characterize different *types* of diversities in the scope of evaluating *meta*-diversity, but obviously many others could be envisaged. The 5 BC models are provided with the source code of this paper.

Each set of BC features is defined by a mapping function $BC_X : o \in [0,1]^{256\times256} \mapsto \hat{z} \in [0,1]^8$ where X is the corresponding BC space, o is a Lenia pattern and $\hat{z}$ represents its 8-dimensional behavioral descriptor in the corresponding BC space.

We denote $\mathcal{D}_{ref}$ an external dataset of 15000 Lenia patterns. The patterns in $\mathcal{D}_{ref}$ were randomly collected from prior exploration experiments in Lenia, experiments that include different random seeds and different exploration variants and comport 50% SLPs and 50% TLPs. $\mathcal{D}_{ref}$ is a large database that is intended to cover a diversity of patterns orders of magnitude larger than what could be found in any single algorithm experiment, and that we use as reference dataset to construct and normalize the different evaluation BC spaces.

Spectrum-Fourier The 2-dimensional discrete Fourier transform is a mathematical method that projects an image (2D spatial signal) into the frequency domain, from which frequency characteristics can be extracted and used as texture descriptors [486]. Applications range from material description [487], leaf texture description in biology [488] and rule classification in cellular automata [230].

The construction of $\text{BC}_{\text{Spectrum-Fourier}}$ is summarized in Figure C.2 and follows the below procedure:

1. The 2D Fast Fourier Transform transforms the image $o = f(x, y)$ into the u, v frequency domain function F, the zero-frequency component is shifted to the center of the array and the power specrum PS (or power spectral density) is computed:

$$F(u,v) = \frac{1}{256 \times 256} \sum_{x=0}^{255} \sum_{y=0}^{255} f(x,y) \exp^{-j2\pi \frac{ux}{256} \frac{vx}{256}}$$
$$F(u,v) \leftarrow Roll(F(u,v), (\frac{256}{2}, \frac{256}{2}))$$
$$PS(u,v) = Real(F(u,v))^2 + Imaginary(F(u,v))^2$$

2. The power spectrum is filtered to keep only the lower half (symmetry property of the FFT) and the significant values:

$$PS(u,v) = \{PS(u,v), 0 \leq u \leq \frac{256}{2}, -\frac{256}{2} \leq v \leq \frac{256}{2} - 1\}$$
$$PS(u,v) = 0 \; if \; PS(u,v) < mean(PS(u,v))$$

3. The power spectrum is partitioned into 20 ring-shaped sectors:

$$\left[R_i = \{PS(u,v) | r_1^2 \leq u^2 + v^2 \leq r_2^2\} \; with \; (r_1, r_2) = (\frac{i}{20} \times \frac{256}{2}, \frac{i+1}{20} \times \frac{256}{2}); \; for \; i \in [0..19] \right]$$

4. A 40-dimensional feature vector (FV) representing radially-aggregating measures (mean μ_i and standard deviation σ_i of each sector) is extracted:

$$FV(o) = [\mu_1, \sigma_1, \ldots, \mu_{20}, \sigma_{20}],$$
$$\text{where } \mu_i = mean(PS[R_i]), \sigma_i = std(PS[R_i])$$

5. The 40-dimensional feature vector FV is projected into a normalized 8-dimensional behavioral descriptor $\hat{z}$ using a transformation $\hat{T}$: $FV \mapsto \hat{z}$. $\hat{T}$ is constructed with Principal Component Analysis (PCA) [489] dimensionality reduction on $\mathcal{D}_{ref}$:

$X_{ref} = \{FV(o), o \in \mathcal{D}_{ref}\}$

Fit a PCA with 8 components on X_{ref}, $PCA : FV \in \mathbb{R}^{40} \mapsto z \in \mathbb{R}^8$

$z_{ref} = PCA(X_{ref}), z_{min} = percentile(z_{ref}, 0.01), z_{max} = percentile(z_{ref}, 99.9)$

$$\hat{T} : FV \mapsto \hat{z} = \frac{PCA(FV) - z_{min}}{z_{max} - z_{min}}$$

6. $\text{BC}_{\text{Spectrum-Fourier}}(o) = \hat{T} \circ FV(o)$

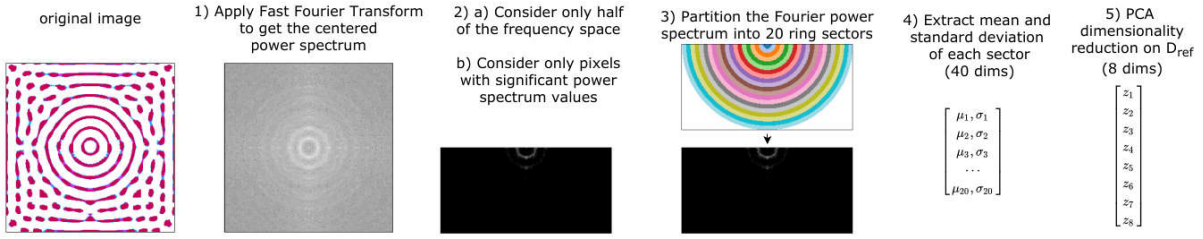


Figure C.2.: Construction of Spectrum-Fourier analytic space. See text for details. Please note that for visualization purposes: (left) the original image is colorized but is originally a 256 × 256 grayscale image; (step 1-2-3) the power spectrum is depicted in logarithmic scale.

Elliptical-Fourier Elliptical Fourier analysis (EFA) [229] is a mathematical method for contour description which has been widely-used for shape description in image processing [490]. These descriptors have been applied to morphometrical analysis in biology [491], for instance to characterize the phenotype of plants leaf and petal contours [492] or anatomical shape changes [493, 494].

A closed contour $\{x_p, y_p\}_{p=1}^{K}$ (K points polygon) can be seen as a continuous periodic function of the *length* parameter $T = \sum_{p=1}^{K} \Delta t_p$ where t_p is the distance from the $p-1^{th}$ to the p^{th} point. Therefore it can be represented as a sum of cosine and sine functions of growing frequencies (harmonics) under Fourier approximation. Each harmonic is an ellipse which is defined by 4 coefficients a, b, c, d.

The construction of $BC_{\text{Elliptical-Fourier}}$ is summarized in Figure C.3 and follows the below procedure:

1. Binarize the image $o_{\texttt{binary}}$ `= o > 0.2` and extract the external contour as the a list of the (x,y) positions of the pixels that make up the boundary using OpenCV function `contour = cv2.findContours(`$o_{\texttt{binary}}$`, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)`
2. Extract the set of $\{a_n, b_n, c_n, d_n\}_{n=1}^{N}$ coefficients for a series of N ellipses (N=25) from the x- and y-deltas (Δx_p and Δy_p) between each consecutive point p in the K points polygon:

$$a_n = \frac{T}{2n^2\pi^2} \sum_{p=1}^{K} \frac{\Delta x_p}{\Delta t_p} \left[\cos\frac{2n\pi t_p}{T} - \cos\frac{2n\pi t_{p-1}}{T} \right]$$

$$b_n = \frac{T}{2n^2\pi^2} \sum_{p=1}^{K} \frac{\Delta x_p}{\Delta t_p} \left[\sin\frac{2n\pi t_p}{T} - \sin\frac{2n\pi t_{p-1}}{T} \right]$$

$$c_n = \frac{T}{2n^2\pi^2} \sum_{p=1}^{K} \frac{\Delta y_p}{\Delta t_p} \left[\cos\frac{2n\pi t_p}{T} - \cos\frac{2n\pi t_{p-1}}{T} \right]$$

$$d_n = \frac{T}{2n^2\pi^2} \sum_{p=1}^{K} \frac{\Delta y_p}{\Delta t_p} \left[\sin\frac{2n\pi t_p}{T} - \sin\frac{2n\pi t_{p-1}}{T} \right]$$

3. The coefficients are standardized (i.e. made invariant to size, rotation and shift):
$\begin{bmatrix} a_n^* & b_n^* \\ c_n^* & d_n^* \end{bmatrix} = \frac{1}{L} \begin{bmatrix} \cos\phi & \sin\phi \\ -\sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} a_n & b_n \\ c_n & d_n \end{bmatrix} \begin{bmatrix} \cos N\theta & \sin N\theta \\ -\sin N\theta & \cos N\theta \end{bmatrix}$, where
$L = \sqrt{[(A_0 - x_m)^2 + (C_0 - x_m)^2]}$, (A_0, C_0) is the center of the 1^{st} harmonic ellipse, (x_m, y_m) is the location of the modified starting point

(on the major axis of the ellipse), $\theta = \frac{2\pi t_m}{T}$ and $\phi = \tan^{-1}\frac{y_m - C_0}{x_m - A_0}$ (angle between the major axis of the ellipse and xaxis).

4. The 100-dimensional feature vector $FV = \{a_n^*, b_n^*, c_n^*, d_n^*\}_{n=1}^{25}$ is projected into a normalized 8-dimensional behavioral descriptor using a transformation $\hat{T} : FV \mapsto \hat{z}$. $\hat{T}$ is constructed with Principal Component Analysis (PCA) dimensionality reduction on $\mathcal{D}_{ref}$ (similar procedure as in point 5 of $\text{BC}_{\text{Spectrum-Fourier}}$).
5. $\text{BC}_{\text{Elliptical-Fourier}}(o) = \hat{T} \circ FV(o)$

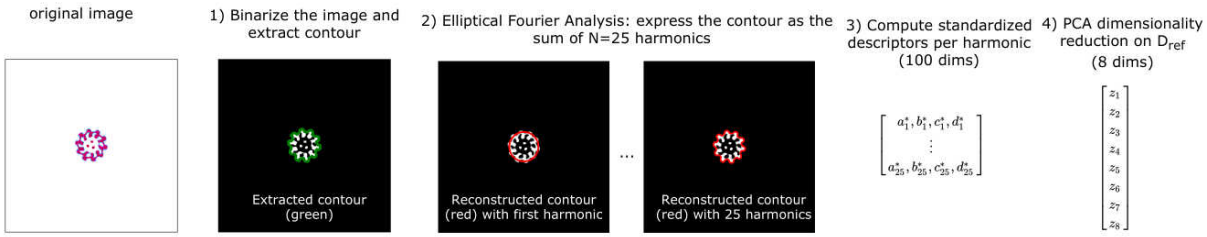


Figure C.3.: Construction of Elliptical-Fourier analytic space. See text for details. (step 1) The contour depicted in green is extracted with OpenCV `findContours()` function (step 2) The contours depicted in red are reconstructed from the EFA coefficients (like in other Fourier series transforms the shape signal can be approximated by summing the harmonics [229]).

Lenia-Statistics The original Lenia paper proposes several measures for statistical analysis of the Lenia patterns (section 2.4.2 in [39]).

$\text{BC}_{\text{Lenia-Statistics}}$ is constructed on top of these measures as follows:

1. Among all the statistical measures proposed in [39] we selected the 17 measures that are time-independent, i.e. that can be computed from the final Lenia pattern $o = I(x, y)$, namely:
   - the activation mass $m = \frac{1}{256\times256}\sum_{(x,y)\in I} I(x,y)$
   - the activation volume $V_m = \frac{1}{256\times256}\sum_{(x,y)\in I} \delta_{I(x,y)>\epsilon}$ $(\epsilon = 10^{-4})$
   - the activation density $\rho_m = \frac{m}{V_m}$
   - the centeredness of the activation mass distribution $C_m = \frac{1}{m}\sum_{(x,y)\in I} w_{xy} \cdot I(x - \bar{x}_m, y - \bar{y}_m)$ where $(\bar{x}_m, \bar{y}_m)$ is the activation centroid and $w_{x,y} = \left(1 - \frac{d(x,y)}{\max_{x,y} d(x,y)}\right)^2$ with $d(x,y) = \sqrt{(x-\bar{x}_m)^2 + (y-\bar{y}_m)^2}$
   - the 8 invariant image moments by Hu [495]
   - the 5 extra invariant image moments by Flusser [496]
2. The 17-dimensional feature vector $FV = \left[m, V_m, \rho_m, C_m, hu_1, \ldots, hu_7, flusser_8, \ldots, flusser_{13}\right]$ is projected into a normalized 8-dimensional behavioral descriptor using a transformation $\hat{T} : FV \mapsto \hat{z}$. $\hat{T}$ is constructed with Principal Component Analysis (PCA) dimensionality reduction on $\mathcal{D}_{ref}$ (similarly than for $\text{BC}_{\text{Spectrum-Fourier}}$).
3. $\text{BC}_{\text{Lenia-Statistics}}(o) = \hat{T} \circ FV(o)$

BetaVAE Reinke et al. [•1] propose to train a β-VAE [210] on a large database of Lenia pattterns and to reuse the learned features as behavioral descriptors for the analytic BC space.

$\text{BC}_{\text{BetaVAE}}$ is constructed according to the below procedure :

1. A β-VAE with 8-dimensional latent space is instantiated with the architecture detailed in table Table C.1.
2. The construction of the training dataset, training procedure and hyperparameters follow [•1]:
    - The β-VAE is trained on an external database $\mathcal{D}_{ref}^{(big)}$ of 42500 Lenia patterns (with 50% SLP and 50% TLP, 37500 as training set, 5000 as validation set) which were randomly collected from independent previous experiments (with the same procedure than $\mathcal{D}_{ref}$).
    - The β-VAE is trained for more than 1250 epochs with hyperparameters $\beta = 5$, Adam optimizer ($lr = 1\mathrm{e}{-3}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 1\mathrm{e}{-8}$, weight decay=1e−5) and a batch size of 64.
    - The network weights which resulted in the minimal validation set error during the training are kept.
3. The resulting pretrained encoder serves as mapping function from a Lenia pattern o to a 8-dimensional feature vector $FV(o) = \left[z_1, z_2, z_3, z_4, z_5, z_6, z_7, z_8\right]$
4. Similarly to the other analytic BC spaces in this paper, we use the reference dataset $\mathcal{D}_{ref}$ to normalize the 8-dimensional behavioral descriptors between $[0, 1]$:

$$z_{ref} = \{FV(o), o \in \mathcal{D}_{ref}\}, z_{min} = percentile(z_{ref}, 0.01), z_{max} = percentile(z_{ref}, 99.9)$$

$$\hat{T} : FV \mapsto \hat{z} = \frac{FV - z_{min}}{z_{max} - z_{min}}$$

5. $\mathrm{BC}_{\text{BetaVAE}}(o) = \hat{T} \circ FV(o)$

Table C.1.: β-VAE architecture used for $\mathrm{BC}_{\text{BetaVAE}}$.

Encoder	**Decoder**
Input pattern A: 256 × 256 × 1	Input latent vector z: 8 × 1
Conv layer: 32 kernels 4 × 4, stride 2, 1-padding + ReLU	FC layers : 256 + ReLU, 256 + ReLU, 4 × 4 × 32 + ReLU
Conv layer: 32 kernels 4 × 4, stride 2, 1-padding + ReLU	TransposeConv layer: 32 kernels 4 × 4, stride 2, 1-padding + ReLU
Conv layer: 32 kernels 4 × 4, stride 2, 1-padding + ReLU	TransposeConv layer: 32 kernels 4 × 4, stride 2, 1-padding + ReLU
Conv layer: 32 kernels 4 × 4, stride 2, 1-padding + ReLU	TransposeConv layer: 32 kernels 4 × 4, stride 2, 1-padding + ReLU
Conv layer: 32 kernels 4 × 4, stride 2, 1-padding + ReLU	TransposeConv layer: 32 kernels 4 × 4, stride 2, 1-padding + ReLU
Conv layer: 32 kernels 4 × 4, stride 2, 1-padding + ReLU	TransposeConv layer: 32 kernels 4 × 4, stride 2, 1-padding + ReLU
FC layers : 256 + ReLU, 256 + ReLU, FC: 2 × 8	TransposeConv layer: 32 kernels 4 × 4, stride 2, 1-padding

Patch-BetaVAE Rhe β-VAE is not able to encode finer details and texture of patterns as the compression of the images to a 8-dimensional vector results in a general blurriness in the reconstructed patterns (see Figure 4.9). Therefore, we also implemented an additional variant denoted as Patch-BetaVAE where the β-VAE is trained on "zoomed" 32 × 32 patches. A preprocessing step extracts the cropped patch around the image activation centroid $P : o \mapsto o[\bar{x}_m - 16 : \bar{x}_m + 16, \bar{y}_m - 16 : \bar{y}_m + 16]$. Then, the construction of $\mathrm{BC}_{\text{Patch-BetaVAE}}$ follows exactly the construction of $\mathrm{BC}_{\text{BetaVAE}}$, except that the network architecture has only 3 convolutional layers instead of 6. Following the notations of the previous paragraph, $\mathrm{BC}_{\text{Patch-BetaVAE}}(o) = \hat{T} \circ FV \circ P(o)$ with FV the pretrained model on image patches and $\hat{T}$ a normalizing function computed on $\mathcal{D}_{ref}$.

C.1.2. Diversity Metric

In Figure 5.2 we (a) measure diversity of the IMGEP variants within each of the 5 BC spaces using a binning-based metric and (b) select a representative sets of discoveries within each BC space using a distance-based diversity metric.

For the binning-based metric (Figure 5.2a), each BC space is discretized into a collection of t bins $N_1, \ldots, N_t$ and the diversity is quantified as the number of bins filled over the course of exploration: $D_{|BC} = \sum_{i=1}^{t} \delta_i$ where $\delta_i = 1$ if the N_i^{th} bin is filled, $\delta_i = 0$ otherwise. We opt for a regular binning where each dimension of the BC space is discretized into equally sized bins. Here, 20 bins per dimension are used for the discretization of the BC spaces. For recall, all the analytic BC spaces used in this paper are 8-dimensional and bounded in $[0, 1]^8$ (see previous section). Note however that for a given BC space, the maximum number of bins that can be filled by all possible Lenia patterns is unknown.

For selecting the representative sets in each BC space (Figure 5.2b), we use the following procedure:

- Randomly draw 750 candidate sets of 5 images among the 5000 discoveries of the IMGEP that was operating that BC space (*i.e.* using it as a goal space)
- Select the most *dissimilar* (i.e. diverse) set among those 750 sets. The dissimilarity of a set of 5 images is measured as a function of all the distances between each pair of images in the set, with distances being computed in the current BC space. This distance-based measure of diversity D, proposed in [497], measures the magnitude M (dispersion) and variability E (equability) of the set of S=6 points in the BC:

  $M = \frac{S}{S-1} \sum_{i=1}^{S} \sum_{j=1}^{S} \frac{d_{ij}}{S^2}$, where d_{ij} is the pairwise euclidean distance

  $E = \frac{1+\sqrt{1+4H}}{2S}$, where $H = \left[\sum_{i=1}^{S} \sum_{j=1}^{S} \left(\frac{d_{ij}}{\sum_{i=1}^{S} \sum_{j=1}^{S} d_{ij}} \right)^2 \right]^{\frac{1}{1-2}}$

  $D = 1 + (S-1) \times E \times M, M \in [0,1] and E \in [0,1]$

  This measure replaces the binning-based measure which can hardly be used here (as they are only 5 images most candidate sets are likely to fall in the same number of bins and be equally diverse).
- display the corresponding set of images

C.2. Focus on HOLMES design choices

Algorithm 5: HOLMES continual learning of modular BC spaces

Require: saturation signal SATURATE, node redirection criteria REDIRECT, connection scheme CONNECTION

Initialize root representation $R = \{\mathcal{R}_0\}$
for incoming data $o \in \{o^{(1)}, \dots, o^{(n)}, \dots\}$ **do**
  Observe incoming data o
  `// Encode observation in HOLMES hierarchy`
  Start with root node $i \leftarrow 0$, $parent(i) \leftarrow \emptyset$
  while i exists in the hierarchy (until leaf) **do**
    $z_i = \mathcal{R}_i(o, \mathcal{R}_{parent(i)}(o))$
    Append (θ, o, z_i) to the history $\mathcal{H}$
    $i \leftarrow i.c, c = \text{REDIRECT}_i(z_i)$ `// go to left or right child`
  `// Augment HOLMES representational capacity if needed`
  for i in HOLMES leaf nodes **do**
    if $\textit{SATURATE}(Z_i)$ **then**
      Freeze $\mathcal{R}_i$ weights
      Fit node-specific boundary
      $\text{REDIRECT}_i : Z_i \rightarrow \{left, right\}$
      Instantiate child modules $\mathcal{R}_{i0}, \mathcal{R}_{i1} = \text{CONNECTION}(\mathcal{R}_i)$
      `// Project past observations to children BCs`
      for $(\theta, o, z) \in \mathcal{H}[Z_i]$ **do**
        $\mathcal{R}_j \leftarrow \mathcal{R}_{i.c}$, $c = \text{REDIRECT}_i(z)$
        Append $(\theta, o, \mathcal{R}_j(o))$ to $\mathcal{H}[j]$
  `// Train HOLMES modules`
  Train the hierarchy R on observations in $\mathcal{H}$
  `// Relabel the previous observations`
  for i in HOLMES nodes **do**
    for $(\theta, o, z) \in \mathcal{H}[i]$ **do**
      $\mathcal{H}[i][z] \leftarrow \mathcal{R}_i(o, \mathcal{R}_{parent(i)}(o))$

The HOLMES architecture has 4 main components:

1. a base *module* embedding neural network ($\mathcal{R}$)
2. a *saturation* signal that triggers the instantiating of new nodes in the hierarchy (SATURATE)
3. a *boundary* redirection criteria that unsupervisedly clusters the incoming patterns into the different modules (REDIRECT)
4. a *connection* scheme that allows to instantiate new children modules in the hierarchy with feature-wise transfer from their parent module (CONNECTION)

Given those base components, Algorithm. 5 details how, assuming a continual source of incoming data $\{o^{(1)}, \dots, o^{(n)}, \dots\}$, HOLMES progressively learns and expands its hierarchy of BC spaces.

While several design choices could be made for the *modules*, *connection scheme*, and *splitting criteria*, we summarize the choices made in [•3] below.

Choice for the base module Each module has an embedding network R_i that maps an observation o to a low-dimensional vector $r = R(o)$. To learn such embedding, we rely on a variational autoencoder network [137] for the base module. The encoder network $R_i : q_\phi(r|x)$ is coupled with a decoder network $D_i : p_\theta(x|r)$ that enables a generative process from the latent space, and the networks are jointly trained to maximize the marginal log-likelihood of the training data with a regularizer on the latent structure. The training loss is

$$\mathcal{L}_{\text{VAE}}(\theta, \phi; \mathbf{x}, \mathbf{r}) = \underbrace{\mathbb{E}_{\hat{p}(\mathbf{x})}\left(\mathbb{E}_{q_\phi(\mathbf{r}|\mathbf{x})}\left(-\log p_\theta(\mathbf{x}|\mathbf{r})\right)\right)}_{a} + \underbrace{\mathbb{E}_{\hat{p}(\mathbf{x})}\left(D_{\text{KL}}\left(q_\phi(\mathbf{r}|\mathbf{x})||p(\mathbf{r})\right)\right)}_{b}$$

where (a) represents the expected reconstruction error (computed with binary cross entropy) and (b) is the regularizer KL divergence loss of the approximate diagonal Gaussian posterior $q_\phi(r|x)$ from the standard Gaussian prior $p(z) = \mathcal{N}(0, I)$. Please note that input observations are partitioned between the different nodes in HOLMES, therefore each module VAE is trained only on its niche of patterns. Only the encoder network R_i is kept in IMGEP-HOLMES, therefore other choices for the base module and training strategy could be envisaged in future work, for instance with contrastive approaches instead of generative approaches.

Choice for the connection scheme The connection scheme takes inspiration from *Progressive Neural Networks* (PNN) [242]), where transfer is enabled by connecting the different modules via learned *lateral connections*. To mitigate the growing number of parameters, we opted for a sparser connection scheme that in [242]. The connection scheme is summarized in Figure C.4. Connections are only instantiated between a child and its parent (hierarchical passing of information). Connections are only instantiated between a reduced number of layers (denoted as l_f, gfi_c, lfi_c, recon_c in the figure). We hypothesize that transfer is beneficial in the decoder network so a child module can reconstruct "as well as" its parent, however connections are removed between encoders as new complementary type of features should be learned. We preserve the connections only at the local feature level, as the CNN first layers tend to learn similar features [498]. Connections between linear layers are defined as linear layers and connections between convolutional layers are defined as convolutions with 1×1 kernel. At each connection level, the output of the connection is summed to the current feature map in the VAE. Other connection schemes could be envisaged in future work, for instance with FiLM layers [499] (feature-wise affine transformation instead of sum) which have recently been proposed for vision models.

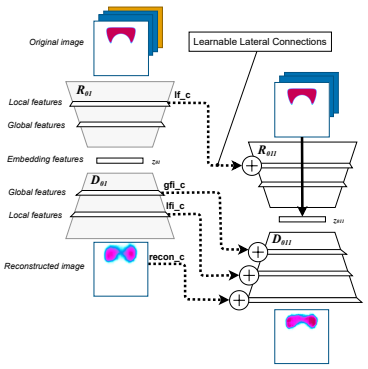


Figure C.4.: Lateral connections in HOLMES

Choice for the splitting criteria There are two main choices: *when* to split a node and *how* to redirect the patterns toward either the left or right children. For both, we opted for simple design choices that allow the split to be unsupervisedly and autonomously handled during the exploration loop. We trigger a split in a node when the reconstruction loss of its VAE reaches a plateau, with additional conditions to prevent premature splitting (minimal node population and minimal number of training steps) or to limit the total number of splits. When splitting a node, we use K-means algorithm in the embedding space to fit 2 clusters

on the points that are currently in the node. This generates a boundary in the latent space of the node, that we keep fixed for the rest of the exploration loop. Again, many other choices could be envisaged in future work, for instance by including human feedback to fit the boundary or with more advanced clustering algorithms.

C.3. Representational Similarity Analysis

We denote $\mathcal{D}_{ref}^{(small)}$ an external dataset of 3000 Lenia patterns (50% SLPs, 50% TLPs) which were collected with the same procedure than $\mathcal{D}_{ref}$.

Given two representations embedding networks R_i and R_j with 8-dimensional latent space, the RSA similarity index RSA_{ij} is computed with the linear Centered Kernel Alignment index (CKA) proposed in [254]:

1. Compute the matrix of behavioral descriptors responses from each representation
   $Z_i = [R_i(o), o \in \mathcal{D}_{ref}^{(small)}] \in [0,1]^{3000\times 8}$ and $Z_j = [R_j(o), o \in \mathcal{D}_{ref}^{(small)}] \in [0,1]^{3000\times 8}$
2. Center the matrices responses:
   $Z_i \leftarrow Z_i - mean(Z_i, axis = 0)$ and $Z_j \leftarrow Z_j - mean(Z_j, axis = 0)$
3. $RSA_{ij} = CKA(Z_i Z_i^T, Z_j Z_j^T) = \frac{||Z_i \cdot Z_j^T||_F^2}{||Z_i \cdot Z_i^T||_F ||Z_j \cdot Z_j^T||_F}$
   where $||\cdot||_F$ represents the Frobenius norm

Representation Similarity Analysis (RSA) is used in Figure 5.5 of the main paper in two ways:

- To compare representations in *time*, i.e. where the the embedding networks R_i and R_j come from the same network but from different training stages
- To compare representations from different *modules* in HOLMES where the the embedding networks R_i and R_j are taken from the same time step (end of exploration) but from different networks.

C.4. Experimental Settings

C.4.1. Sampling of parameters θ

All experiments are done in the Lenia environment, and we use the exact same Lenia settings as in Reinke et al. [•1], described in Section B.2.1. The set of *controllable* parameters θ of the artificial agent include:

- The update rules parameters $[R, \mathcal{T}, \mu, \sigma, \beta_1, \beta_2, \beta_3]$
- CPPN-parameters that control the generation of the initial state $A^{t=1}$

For each exploration run, the IMGEP agent samples a set of parameters $\theta \in \Theta$ that generates a rollout $A^{t=1} \rightarrow \cdots \rightarrow A^{t=200}$.

There are two ways parameters $\theta \in \Theta$ are sampled:

1. During the N_{init} initial runs, parameters are randomly sampled $\theta \sim \mathcal{U}(\Theta)$
2. During the goal-directed exploration runs, parameters are sampled from a policy $\theta \sim \Pi(B\hat{C}_i, \hat{g}, \mathcal{H})$. The Π policy operates in two steps:
   a) given a goal $g \in B\hat{C}_i$, select parameters $\hat{\theta} \in \mathcal{H}$ whose corresponding outcome is closest to g in $B\hat{C}_i$
   b) mutate the parameters by a random process $\theta = \text{MUTATION}(\hat{\theta})$

We therefore need to define 1) the random process $\mathcal{U}$ used to randomly initialize the parameters θ and 2) the random MUTATION process used to mutate an existing set of parameters $\hat{\theta}$.

For both we follow exactly the implementation proposed in the IMGEP-VAE approach. We refer to Section B.2.4 for a complete description of the implementation of the random initialization process and random mutation process. This includes the procedure used for parameters that control the generation of the initial pattern $A^{t=1}$ and for parameters that control Lenia's update rule. In this paper, we use the exact same hyperparameters as in Reinke et al. [•1] for initialization $\mathcal{U}$ and MUTATION of the CPPN-parameters that control the generation of the initial state $A^{t=1}$. We use slightly different hyper-parameters for the MUTATION of the parameters that control the generation of the update rule $[R, \mathcal{T}, \mu, \sigma, \beta_1, \beta_2, \beta_3]$, as detailed in table Table C.2.

	R	$\mathcal{T}$	μ	σ	$(\beta_1, \beta_2, \beta_3)$
$[a, b]$	$[2, 20]$	$[1, 20]$	$[0, 1]$	$[0.001, 0.3]$	$[0, 1]$
σ_M	0.5	0.5	0.1	0.05	0.1

Table C.2.: Sampling of parameters for the update rule. The random initialization process $\mathcal{U}$ uses uniform sampling in an interval $[a, b]$. The random MUTATION is a Gaussian process $\theta = [\hat{\theta} + \mathcal{N}(\sigma_M)]_a^b$.

C.4.2. Incremental Training of the BC Spaces

Training Procedure The networks are trained 100 epochs every 100 runs of exploration (resulting in 50 training stages and 5000 training epochs in total). The networks are initialized with *kaiming* uniform initialization. We used the Adam optimizer (lr = 1e−3, $\beta_1 = 0.9$, $\beta_2 = 0.999$, ϵ = 1e−8, weight decay=1e−5) with a batch size of 128.

Training Dataset The datasets are incrementally constructed during exploration by gathering the discovered patterns. One pattern every ten is added to the validation set (10%) and the rest is used in the training set (the validation dataset only serves for checking purposes and has no influence on the learned BC spaces). Importance sampling is used to give the newly-discovered patterns more weights. A weighted random sampler is used as follow: at each training stage t, there are X patterns discovered so far among which X_{new} have been discovered during the last 100 steps, we create a dataset D_t of X images that we construct by sampling 30% among the X_{new} lastly discovered images and 70% among the $X - X_{new}$ old patterns. We also use data-augmentation, i.e at each training stage t, the images in D_t are augmented online by random x and y translations (up to half the pattern size and with probability 0.6), rotation (up to 20 degrees and with probability 0.6), horizontal and

vertical flipping (with probability 0.2), zooming (up to factor 3 with probability 0.6). The augmentations are preceded by spherical padding to preserve Lenia spherical continuity.

IMGEP-VAE The monolithic VAE architecture used in the IMGEP-VAE baseline is detailed in table Table C.3. It has a total neural capacity of 2258657 parameters.

Table C.3.: VAE architecture used for IMGEP-VAE.

Encoder	**Decoder**
Input pattern A: $256 \times 256 \times 1$	Input latent vector z: 16×1
Conv layer: 64 kernels 4×4, stride 2, 1-padding + ReLU	FC layers : 512+ ReLU, 512+ ReLU, $4 \times 4 \times 64$ + ReLU
Conv layer: 64 kernels 4×4, stride 2, 1-padding + ReLU	TransposeConv layer: 64 kernels 4×4, stride 2, 1-padding + ReLU
Conv layer: 64 kernels 4×4, stride 2, 1-padding + ReLU	TransposeConv layer: 64 kernels 4×4, stride 2, 1-padding + ReLU
Conv layer: 64 kernels 4×4, stride 2, 1-padding + ReLU	TransposeConv layer: 64 kernels 4×4, stride 2, 1-padding + ReLU
Conv layer: 64 kernels 4×4, stride 2, 1-padding + ReLU	TransposeConv layer: 64 kernels 4×4, stride 2, 1-padding + ReLU
Conv layer: 64 kernels 4×4, stride 2, 1-padding + ReLU	TransposeConv layer: 64 kernels 4×4, stride 2, 1-padding + ReLU
FC layers : 512+ ReLU, 512+ ReLU, FC: 2×16	TransposeConv layer: 1 kernels 4×4, stride 2, 1-padding

IMGEP-HOLMES For the IMGEP-HOLMES variant, the hierarchical representation starts with a single root module R_0 at the beginning of exploration. During each training stage, one node is *split* if it meets the following conditions:

- the reconstruction loss for that node reaches a plateau (running average over the last 50 training epochs is below $\epsilon = 20$)
- at least 500 patterns populate the node
- the node has not just been created (must has been trained for at least 200 epochs)
- it is not too early in the exploration loop (there must be at least 2000 patterns are explored)
- the total number of nodes in the hierarchy is below the maximum number allowed (here expansion is stopped at 11 splits i.e. 23 modules)

Each time a split is triggered in a BC space node of the hierarchy BC_i, the boundary $\mathcal{B}_i$ is fitted in the latent space as follows: K-Means algorithm with 2 clusters is ran on the patterns that currently populate the node. The resulting clusters are kept fixed for the rest of the exploration, therefore when a pattern is projected in the split node, it is sent to the left children if it belongs to the first cluster on the latent space and to the right children otherwise. Here, the final hierarchy has a total of 23 VAE modules. The architecture is identical for each module and is detailed in table Table C.4. At the end of exploration, HOLMES has a total neural capacity of 2085981 parameters. Each base module VAE has a capacity of 86225 parameters and connections of 4673 parameters ($2085981 = 23 \times 86225 + 22 \times 4673$).

Table C.4.: Module architecture used for IMGEP-HOLMES. All the modules R_i have this architecture for the base VAE network as well as the connections (except R_0 which does not have the connections).

Encoder	
Input pattern A: $256 \times 256 \times 1$	
Conv layer: 16 kernels 4×4, stride 2, 1-padding + ReLU	
Conv layer: 16 kernels 4×4, stride 2, 1-padding + ReLU	
Conv layer: 16 kernels 4×4, stride 2, 1-padding + ReLU	**lf_c:** 16 kernels 1×1, stride 1, 1-padding
Conv layer: 16 kernels 4×4, stride 2, 1-padding + ReLU	
Conv layer: 16 kernels 4×4, stride 2, 1-padding + ReLU	
Conv layer: 16 kernels 4×4, stride 2, 1-padding + ReLU	
FC layers : 64+ ReLU, 64+ ReLU, FC: 2×16	
Decoder	
Input latent vector z: 16×1	
FC layers : 64+ ReLU,	**gfi_c:** 64+ReLU
FC layers: 64+ ReLU, $4 \times 4 \times 16$ + ReLU	
TransposeConv layer: 16 kernels 4×4, stride 2, 1-padding + ReLU	
TransposeConv layer: 16 kernels 4×4, stride 2, 1-padding + ReLU	
TransposeConv layer: 16 kernels 4×4, stride 2, 1-padding + ReLU	**lfi_c:** 16 kernels 1×1, stride 1, 1-padding
TransposeConv layer: 16 kernels 4×4, stride 2, 1-padding + ReLU	
TransposeConv layer: 16 kernels 4×4, stride 2, 1-padding + ReLU	
TransposeConv layer: 1 kernel 4×4, stride 2, 1-padding	**recon_c:** 1 kernel 1×1, stride 1, 1-padding

C.5. Ablation Study: Impact of the Lateral Connections

We conducted 5 ablation experiments of the IMGEP-HOLMES variant presented in the main paper, each has 3 repetitions with different seeds. Each ablation experiment considered a different connection scheme with either zero or only one connection among **lf_c**, **gfi_c**, **lfi_c** and **recon_c** (proposed connections in HOLMES, see Figure C.4 and Table C.4). As shown in Figure C.5, the lateral connections are *essential* to learn *diverse* behavioral characterizations among the different modules of the hierarchy. Indeed we can see that IMGEP-HOLMES without any connection (first row in the figure) learns BCs that are highly similar from one module to another (histogram concentrated around [0.8, 1] RSA indexes, i.e. very similar). We can also see that connections toward the last layers of the decoder are seemingly the more important (lfi_c and recon_c) as IMGEP-HOLMES with only one of such connection succeeds to learn dissimilar BCs per module (the histogram is shifted toward lower RSA indexes). However, the connection at the encoder level (lf_c) and close to the embedding level (gfi_c) seem less necessary, or at least alone are not sufficient to allow HOLMES modules escaping the bias inherent to the VAE learning (show a similar histogram of RSA indexes than the no-connection variant). The connection scheme used in the main paper (last row in the figure) seems to be the best suited to learn diverse BCs (histogram concentrated around [0.8, 1] RSA indexes, i.e. very dissimilar).

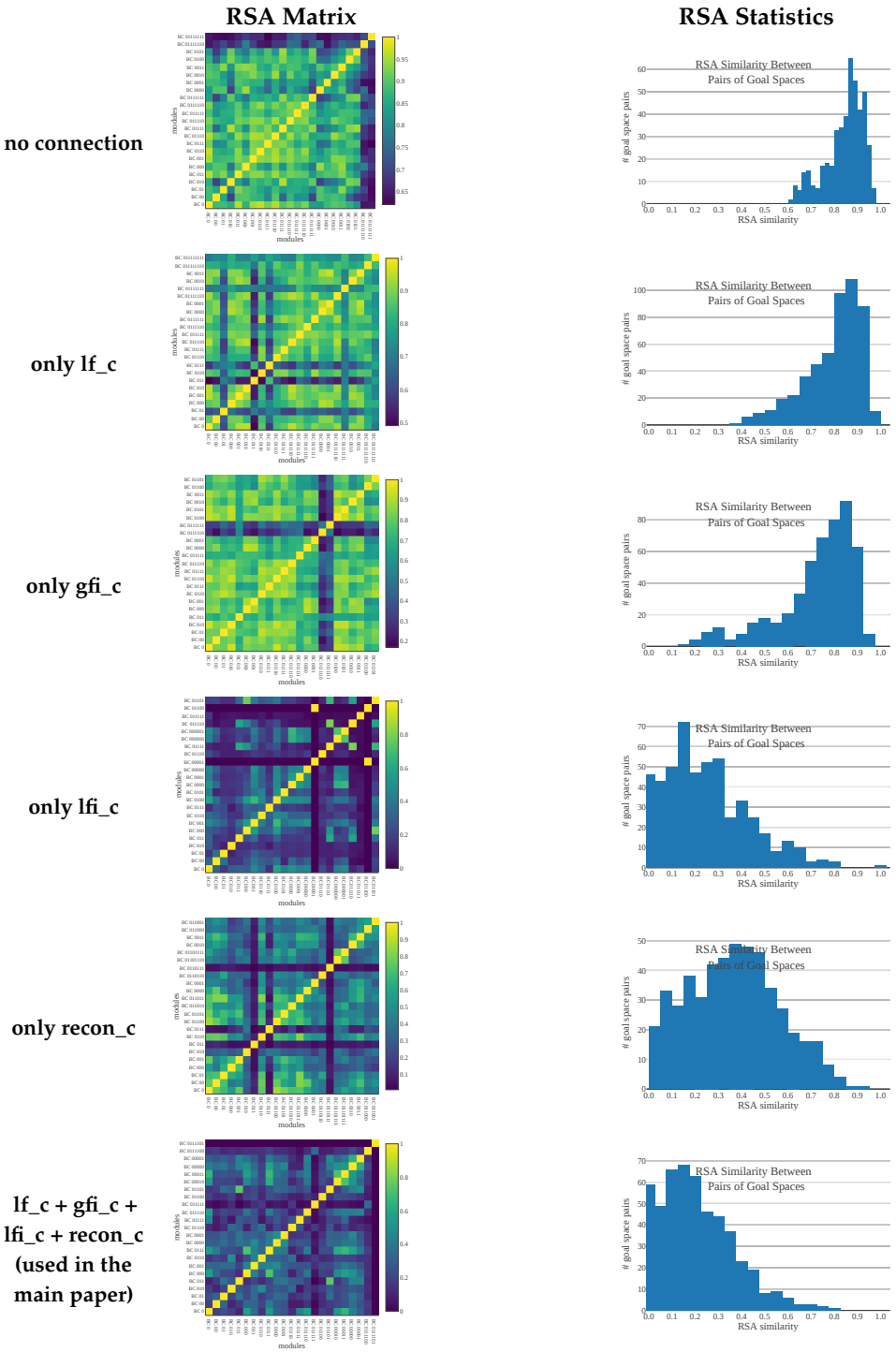


Figure C.5.: RSA Analysis of the effect of the *lateral connections* on the ability for HOLMES to learn *diverse* module BCs. Each row is an ablation experiment with the corresponding connection scheme. (Left) RSA matrix for one experiment repetition (seed 0), with similarity index between 0 (dark blue, not similar at all) and 1 (yellow, identical). Representations are compared at the end of exploration between the different modules (ordered by their creation time on the left-to-right x-axis). (Right) Histogram of RSA index similarity between all pair of modules (aggregated over the 10 repetitions).

C.6. Comparison of HOLMES with Related Methods

In this section we provide a comparison of HOLMES with recent work in the literature: CURL [243], CN-FPM [244] and pro-VLAE [245]. Those

approaches also propose to dynamically expand the network capacity of a VAE in the context of *continual* representation learning, and therefore share similarities with HOLMES. In Table C.5, we provide a high-level comparison of the proposed approaches which compare the different architectures according to their *structural bias*, handling of *catastrophic forgetting*, architecture for *dynamic expansion*, handling of *transfer* between the different group of features, criteria for the *expansion trigger*, and if they are performing *data partitioning* (i.e. learn different set of features for different niches of observations).

	CURL	CN-DPM	pro-VLAE	HOLMES
Structural Bias	Mixture of Gaussians (in a single VAE latent space)	Dirichlet Process Mixture (flat set of VAE modules)	Hierarchical Levels (in a single VAE)	Hierarchical Mixture (binary tree of VAE modules)
Catastrophic Forgetting	Generative Replay	Freeze	"fade-in" coefficient (same network)	Freeze
Dynamic Expansion	New component in the MoG	New module VAE	New feature layer in the VAE	New module VAE
Transfer	Single Shared Network with several "heads"	Lateral connections (exhaustive as in PNN [242])	Single Shared Network with several "levels"	Lateral connections (parent-to-children only)
Expansion Trigger	Short-Term Memory Size	Short-Term Memory Size	Predetermined	Node Saturation
Data Partitioning	Soft partitioning	Soft partitioning (coupling of each VAE with discriminator)	None (same network)	Hard Partitionning (boundary in BC_i)

Table C.5.: High-level comparison of the general choices of HOLMES with those of previous methods: CURL [243], CN-FPM [244] and pro-VLAE [245]. Please refer to the original papers for more details.

As we can see, while HOLMES shares *conceptual* ideas with those approaches, our approach has key differences:

1. It uses a hierarchy of different latent spaces whereas CURL uses a single latent space, CN-DPM uses a flat set of different latent spaces and pro-VLAE uses a fixed-set of latent spaces (different levels in one network)
2. CURL and CN-DPM show results in the context of continual multi-task classification and demonstrate that their modular architecture can separate well the latents allowing to unsupervisedly discriminate between the different input observations / tasks (eg: discriminate digits in MNIST at test time when they have been sequentially observed at train time). However, CURL does not use different features for the different niches of observations and it is not clear if the flat approach of CN-DPM does learn different features between the different modules. However HOLMES targets to learn dissimilar set of features per BC in order to achieve *meta-diversity*.
3. Pro-VLAE is not applied in the context of continual learning but rather proposes to progressively learn features at different levels in the VAE layers, showing that it can successfully disentangle the features. Even though disentanglement is a key property to avoid redundant features, we believe that it is also key to have diverse set of features for the different niches of observed instances.

C.7. Additional IMGEP baselines with a monolithic BC space

In this section, we consider different baselines for the training strategy of an online-learned monolithic architecture used by the IMGEP as goal space representation: **BetaVAE** [210], **BetaTCVAE** [500], **TripletCLR** [501, 502], **SimCLR** [503] and **BigVAE**. All the baselines have the same encoder architecture and training procedure than the main baseline IMGEP-VAE (as detailed in Subsection C.4.2). The baselines differ in their approach

to train the encoder network, including several variants of variational-autoencoders and contrastive approaches.

The first two variants BetaVAE [210] and BetaTCVAE [500] build on the VAE framework and augment the VAE objective with the aim to enhance interpretability and disentanglement of the latent variables. Therefore only the training loss of the VAE (see Section C.2) differs:

- The BetaVAE objective re-weights the b term by a factor $\beta > 1$:
  $\mathcal{L}_{\text{BetaVAE}}(\theta, \phi; \mathbf{x}, \mathbf{r}) = \underbrace{\mathbb{E}_{\hat{p}(\mathbf{x})}\left(\mathbb{E}_{q_\phi(\mathbf{r}|\mathbf{x})}\left(-\log p_\theta(\mathbf{x}|\mathbf{r})\right)\right)}_{a} + \beta \times \underbrace{\mathbb{E}_{\hat{p}(\mathbf{x})}\left(D_{\text{KL}}\left(q_\phi(\mathbf{r}|\mathbf{x})||p(\mathbf{r})\right)\right)}_{b}$
  Our baseline uses $\beta = 10$.
- The BetaTCVAE objective augments the VAE objective with an additional regularizer that penalizes the *total correlation* (dependencies between the dimensions of the representation):

  $$\mathcal{L}_{\text{BetaTCVAE}}(\theta, \phi; \mathbf{x}, \mathbf{r}) = \mathbb{E}_{\hat{p}(\mathbf{x})}\left(\mathbb{E}_{q_\phi(\mathbf{r}|\mathbf{x})}\left(-\log p_\theta(\mathbf{x}|\mathbf{r})\right)\right) +$$
  $$\alpha \times \underbrace{I_{q_\phi}(\mathbf{x}|\mathbf{r})}_{mutual\ information} + \beta \times \underbrace{TC(q_\phi(\mathbf{r}))}_{total\ correlation} + \gamma \times \underbrace{\sum_j D_{KL}(q_\phi(z_j)||p(z_j))}_{elementwise\ KL}$$

  Because TC is not tractable, they [500] propose two methods based on importance sampling to estimate it: *minibatch weighted sampling* (mws) and *minibatch stratified sampling (mss)*. Our baselines uses $\alpha = 1, \beta = 10, \gamma = 1$ and mss importance sampling.

The second two variants TripletCLR [501, 502] and SimCLR [503] use contrastive approaches as training strategy for the encoder. Contrary to the VAE variants, these approaches drop the decoder networks and pixel-wise reconstruction as their training objective operates directly in the latent space. The encoders are trained to maximize agreement between differently augmented versions of the same observation o. We used to 2 variants for the contrastive loss:

- Triplet Loss: $\mathcal{L}_{\text{TripletCLR}}(A, P, N) = \max\left(d(R(A), R(P)) - d(R(A), R(N)) + \alpha, 0\right)$ where R is the embedding network, A is an anchor input (pattern o in the training dataset), P is the positive input (augmented version of o), N is the negative input (other pattern o' randomly sampled in the training dataset), $d(\cdot, \cdot)$ is the distance in the latent space (we use cosine similarity) and α is a margin between positive and negative pairs (we use $\alpha = 1$)
- SimCLR Loss: $\mathcal{L}_{\text{SimCLR}}(A, P) = -\log \frac{\exp sim(z_A, z_P)/\tau}{\sum_N \mathbb{1}_{[N \neq A]} \exp sim(z_A, z_N)/\tau}$ where τ denotes the temperature parameter (we use $\tau = 0.1$); sim the similarity distance (we use cosine similarity); and z represent the latent features onto which operates the contrastive loss. Please note that for this variant the encoder is coupled to a *projection head* network $g(\cdot)$ such that $o \xrightarrow{R} r \xrightarrow{g} z$ (We use g: FC 16→16 + RelU, FC 16→16). Here the positive pair (A, P) is contrasted with all negative pairs (A, N) in the current batch. The final loss is computed across all positive pairs in a mini-batch.

Finally the BigVAE baseline uses the same architecture and training strategy than the main VAE baseline, but with a much larger embedding

capacity (368 dims instead of 16) corresponding to the total embedding capacity of HOLMES if we would concatenate its 23 BC latent spaces.

The results, illustrated in Figure C.6, show that monolithic architecture lack either of stability, enabling efficient diversity search, or plasticity, enabling adaption to novel discoveries throughout exploration:

- Lack of *plasticity* for the all VAE variants, *i.e.* inability to adapt the learned features to novel niches of patterns. Even when changing the training objective or the encoding capacity, the VAE tendency to saturate early during exploration observed in Figure 5.5 seems to hold for all variants, although to a lesser extent for BigVAE. Interestingly, IMGEP-BetaVAE and IMGEP-BetaTCVAE show the same profile of discovered diversities than VAE (good at finding a diversity of SLPs but bad for TLPs) whereas the IMGEP-BigVAE seems to have a reversed bias (good at finding a diversity of TLPs but bad for SLPs). We attribute this effect to the difficulty of VAEs with low embedding capacity to capture textures with fine-grained structures (*i.e.* TLPs) whereas when given a higher encoding-capacity they can more accurately represent TLPs. Therefore the variants with small capacity representations seem better suited for exploring diverse SLPs (to the detriment of TLPs) whereas BigVAE seem better suited for exploring diverse TLPs (to the detriment of SLPs).
- Lack of *stability* for all the contrastive variants, where features are drastically different from one training stage to the other. Contrary to the VAE variants, those approaches do not exhibit a strong *bias* in their BC and therefore do not seem to differ much from the *default* diversity found in Lenia (represented with the black curve), at least for the two *types* of diversity measured in Figure C.6.

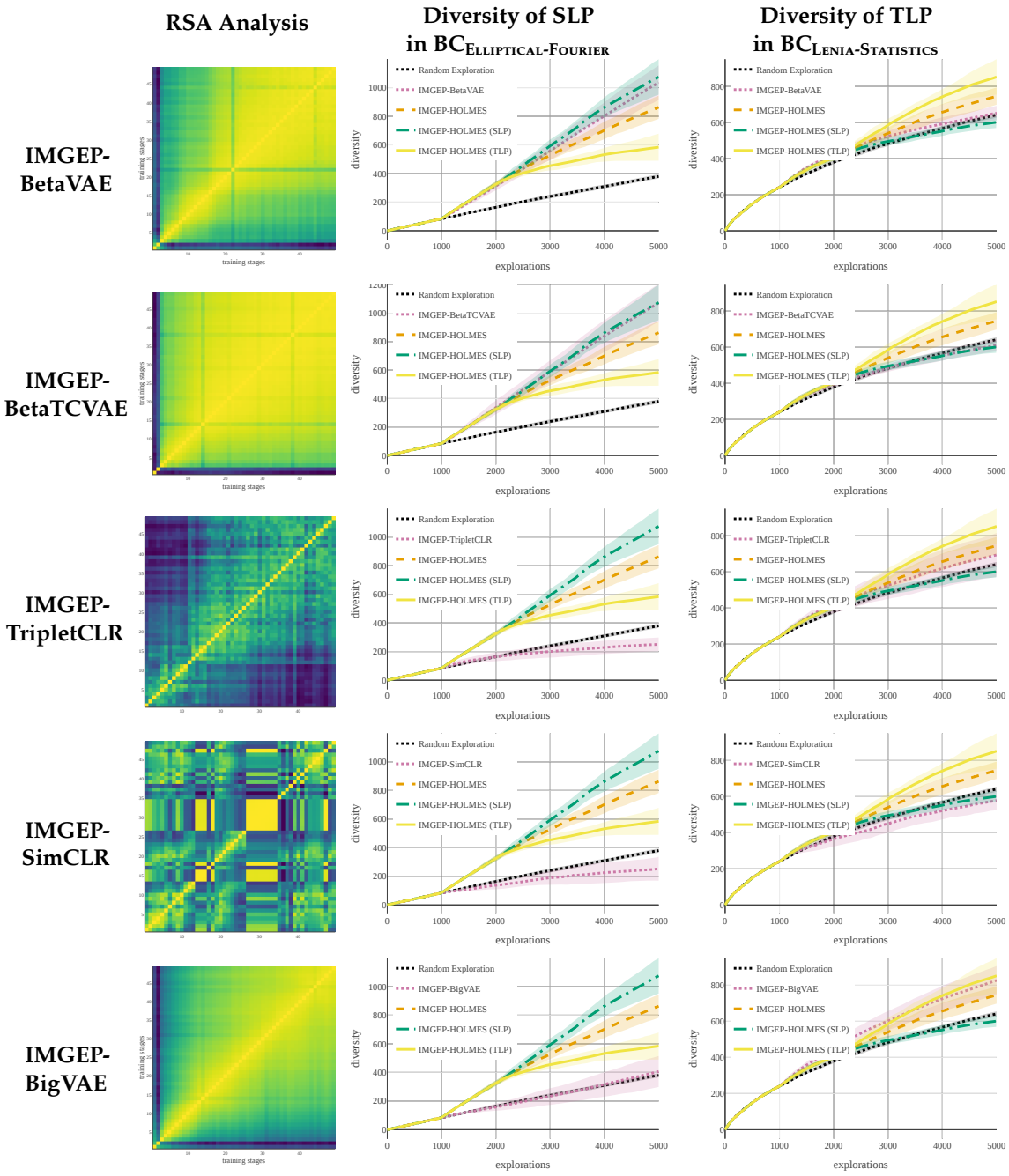


Figure C.6.: This figure complements the results presented in Subsection 6.4.2, where we replace the IMGEP-VAE baseline with different architectures and training strategies for the monolithic architecture. Each row is a baseline denoted as IMGEP-X (where X represents the training strategy used for training the monolithic representation). For each row, we display: (left) RSA matrix where representations are compared in time between the different training stages, as shown in Figure 5.5; (middle-right) exact same plots than Figure 6.6 where we replace the monolithic IMGEP-VAE baseline (pink curve) by the other baseline (of the current row). Therefore only the pink curve vary between the different graphs of one column.

D.

Appendix of Sensorimotor Lenia

(a) Demo and Videos

(b) Notebook

(c) Codebase

Figure D.1.: Scan (or click on) the above QR codes for accessing the paper's (a) companion website with web-demo and supplementary videos, (b) jupyter notebook, and (c) github repository

D.1. Additional Results

D.1.1. Curriculum phylogeny

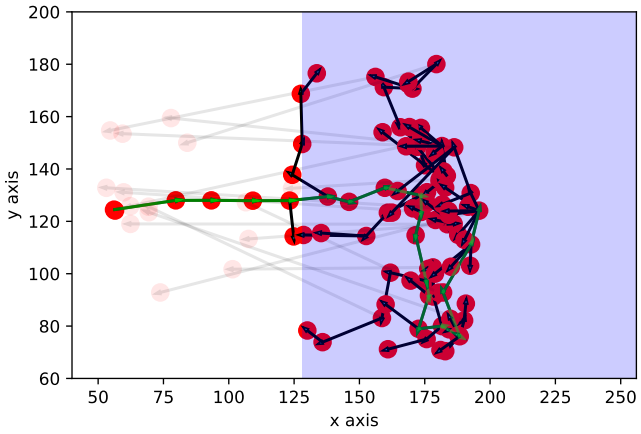


Figure D.2.: "phylogeny tree" of one run of IMGEP. The red dot are reached positions (by a step of IMGEP). The blue zone correspond to the zone where obstacles can be placed. Black arrows indicate optimization progress (*i.e.* the point at the end of the arrow was obtained after optimizing the one at the start of the arrow). The path leading to the best agent (point reaching the furthest position on the x axis) is highlighted in green. Interestingly we can see that the best path is not necessarily a straight path. For visibility reasons, we put transparency on the optimization steps that led to reached goals far from the reached goal of the parameters that was used to initialize the optimization (often due to failing).

In Figure D.2, we visualize the curriculum path that emerged from the IMGEP by plotting the achieved position (reached goal) a each step of the IMGEP. Arrows show, for each step, what was the previous step result used as initialization. In addition, we highlight in green the sequence of reached position leading to the furthest position attained in the grid. We observe that the path to this furthest position is not necessarily a straight path with might indicate the usefulness of trying goals that are not necessarily totally in the direction you want to improve. This figure also shows the diversity of position reached by the IMGEP with some steps leading to parameters reaching different part of the grid.

D.1.2. Variability per IMGEP seed

We report in Table D.1 the variability of the results of the method across the 10 seeds. The variability in result might indicate that some parameter area are easier to navigate or more prone to certain behavior. Overall we still observe that every seed finds a good amount of moving agents and most of them find at least 1 robust agent(ie an agent with a score >0.95 to the "basic obstacle test").

Table D.1.: Seed variability

Seed Number	Number of agents	Number of moving (agents)	Number of robust (agents)	max speed (agents)	max speed obs (agents)
Seed 0	107	93	91	2.8	1.4
Seed 1	64	54	26	2.7	1.5
Seed 2	33	32	1	2.0	1.1
Seed 3	18	7	0	0.5	0.3
Seed 4	35	26	6	1.9	0.4
Seed 5	66	52	38	2.9	1.8
Seed 6	54	54	2	2.5	0.3
Seed 7	30	30	1	3.0	0.9
Seed 8	44	44	4	2.3	0.3
Seed 9	104	94	92	3.2	2.3

D.1.3. Generalization results

We refer to Table D.2 for the full results of the generalization tests (detailed in Subsection D.4.5) among all three algorithm variants: IMGEP search, Random Seach and Handmade Search (detailed in Section D.5).

Table D.2.: Generalization results

Tests	IMGEP		Random	Handmade
	speed>1	10 best	10 best	10 best
speed	1.33 ± 0.28	1.94 ± 0.15	0.53 ± 0.25	0.34 ± 0.10
obstacle number				
24	0.98 ± 0.07	0.99 ± 0.03	0.99 ± 0.03	0.99 ± 0.03
30	0.98 ± 0.07	1.00 ± 0.00	0.99 ± 0.03	0.99 ± 0.03
36	0.99 ± 0.06	1.00 ± 0.00	0.99 ± 0.03	0.97 ± 0.09
42	0.99 ± 0.03	1.00 ± 0.00	0.99 ± 0.03	0.97 ± 0.09
48	0.99 ± 0.04	1.00 ± 0.00	1.00 ± 0.00	0.98 ± 0.06
radius				
4	0.92 ± 0.18	0.90 ± 0.13	0.92 ± 0.12	0.95 ± 0.09
7	0.98 ± 0.08	1.00 ± 0.00	1.00 ± 0.00	0.97 ± 0.09
10	0.98 ± 0.07	0.99 ± 0.03	0.99 ± 0.03	0.99 ± 0.03
13	0.98 ± 0.08	0.99 ± 0.03	1.00 ± 0.00	0.99 ± 0.03
16	0.98 ± 0.08	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
speed				
1/3	0.99 ± 0.04	1.00 ± 0.00	0.77 ± 0.27	0.74 ± 0.28
1/2	0.97 ± 0.07	1.00 ± 0.00	0.61 ± 0.38	0.51 ± 0.38
1	0.81 ± 0.23	0.97 ± 0.05	0.42 ± 0.41	0.02 ± 0.04
2	0.34 ± 0.32	0.71 ± 0.25	0.13 ± 0.29	0.00 ± 0.00
3	0.12 ± 0.15	0.32 ± 0.17	0.07 ± 0.12	0.00 ± 0.00
update mask rate				
0.2	0.99 ± 0.08	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
0.6	0.99 ± 0.08	1.00 ± 0.00	0.89 ± 0.30	1.00 ± 0.00
1.0	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
1.4	0.99 ± 0.09	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
1.8	0.99 ± 0.10	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
noise rate				
0.2	0.91 ± 0.28	0.90 ± 0.30	0.77 ± 0.37	0.99 ± 0.03
0.4	0.75 ± 0.42	0.91 ± 0.27	0.74 ± 0.38	0.92 ± 0.18
0.6	0.67 ± 0.45	0.90 ± 0.27	0.58 ± 0.46	0.77 ± 0.38
0.8	0.60 ± 0.47	0.63 ± 0.44	0.50 ± 0.44	0.71 ± 0.44
1.0	0.51 ± 0.47	0.32 ± 0.41	0.44 ± 0.45	0.70 ± 0.46
noise std				
0.2	0.99 ± 0.11	1.00 ± 0.00	0.96 ± 0.12	1.00 ± 0.00
0.6	0.79 ± 0.39	0.90 ± 0.30	0.76 ± 0.39	0.98 ± 0.06
1.0	0.51 ± 0.47	0.32 ± 0.41	0.44 ± 0.45	0.70 ± 0.46
1.4	0.08 ± 0.21	0.03 ± 0.09	0.18 ± 0.32	0.56 ± 0.45
1.8	0.06 ± 0.14	0.06 ± 0.10	0.17 ± 0.30	0.45 ± 0.47
init noise rate				
0.2	1.00 ± 0.01	1.00 ± 0.00	0.89 ± 0.16	1.00 ± 0.00
0.4	0.99 ± 0.09	1.00 ± 0.00	0.91 ± 0.24	0.99 ± 0.03
0.6	0.98 ± 0.13	1.00 ± 0.00	0.88 ± 0.30	0.95 ± 0.15
0.8	0.97 ± 0.14	1.00 ± 0.00	0.88 ± 0.30	0.89 ± 0.24
1.0	0.95 ± 0.21	1.00 ± 0.00	0.88 ± 0.30	0.76 ± 0.29
noise std				
0.5	0.97 ± 0.16	1.00 ± 0.00	0.87 ± 0.30	0.97 ± 0.09
1.5	0.94 ± 0.20	0.98 ± 0.06	0.85 ± 0.30	0.52 ± 0.42
2.5	0.89 ± 0.27	0.92 ± 0.17	0.80 ± 0.36	0.37 ± 0.44
3.5	0.86 ± 0.32	0.91 ± 0.27	0.81 ± 0.34	0.35 ± 0.45
4.5	0.85 ± 0.32	0.94 ± 0.18	0.79 ± 0.38	0.32 ± 0.43
scaling				
0.15	0.91 ± 0.28	0.90 ± 0.30	0.30 ± 0.46	0.00 ± 0.00
0.65	0.99 ± 0.10	1.00 ± 0.00	0.50 ± 0.50	1.00 ± 0.00
1.15	1.00 ± 0.00	1.00 ± 0.00	0.70 ± 0.46	1.00 ± 0.00
1.65	1.00 ± 0.00	1.00 ± 0.00	0.70 ± 0.46	1.00 ± 0.00
2.15	1.00 ± 0.00	1.00 ± 0.00	0.60 ± 0.49	1.00 ± 0.00

D.2. Lenia system

In Lenia , the system is composed of several communicating grids $A = \{A_c\}$ which we call channels. In each of these grids, every cell/pixel can take any value between 0 and 1. Cells at 0 are considered dead while others are alive. The channels are updated in parallel according to their own physics rule.

The update of a cell $a_{x,c}$ at position x in channel c can be decomposed in three steps. First the cell senses its neighborhood in some other channels (its neighborhood in its channel, with cells of the same type but also in other channels with other types of cells) through convolution kernels which are filters K_k of different shapes and sizes. Second, the cell converts this sensing into an update (whether positive or negative growth or neutral) through growth functions G_k associated with the kernels. Finally, the cell modifies its state by summing the scalars obtained after the growth functions and adding it to its current state. After the update of every rule has been applied, the state is clipped between 0 and 1. Each (kernel,growth function) couple is associated to the source channel c_s it senses, and to the target channel c_t it updates. A couple (kernel,growth function) characterizes a rule on how a type of cell c_t reacts to its neighborhood of cells of type c_s. Note that c_s and c_t could be the same, which correspond to interaction of cells of the same type (intra-channel). Note also that several rules, *i.e.* several (kernel,growth function) couples, can govern the interaction between c_s and c_t.

A local update in the grid is summarized with the following formula (where G^k, K^k, c_s^k, c_t^k are respectively the growth function,convolution filter, source channel,target channel associated with the k'th rule):

$$a_x^{t+1} = f(a_x^t, \mathcal{N}(a_x^t)) =$$

$$\begin{bmatrix} a^t_{x,c_0} + \sum_{k \text{ st } c_t^k=0} G^k(K^k(a^t_{x,c_s^k}, \mathcal{N}_{c_s^k}(a_x^t))) \\ \vdots \\ a^t_{x,c_C} + \sum_{k \text{ st } c_t^k=C} G^k(K^k(a^t_{x,c_s^k}, \mathcal{N}_{c_s^k}(a_x^t))) \end{bmatrix}$$

For each rule, the shape of the (kernel, growth function) is parametrized. We are thus able to "tune" the physics of the cells and of their interactions by changing the kernels shape (how the cells perceive their neighborhood) as well as the growth function shape (how the cells react to this perception). For example, $R \in \mathbb{N}$ controls the maximum size of a kernel for all rules (which is the maximum sensing distance) and each rules has a parameter $r \in [0, 1]$ giving the relative size of the kernel compared to R.

D.2.1. Differentiating through Lenia steps

Due to the locality and recurrence of the update rule, there is a close relationship between cellular automata and recurrent convolutional networks [35]. In fact, we can see a rollout in Lenia as applying a recurrent neural network to an initial state. If (some of) the network parameters are

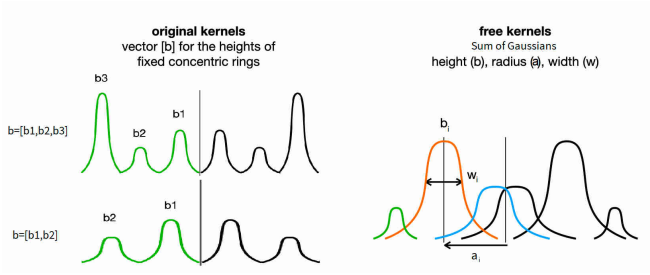


Figure D.3.: Visualization of (left) the convolution kernels used in the original lenia papers [39, 40], (right) the kernel we propose in this paper for more differentiation capabilities. The kernel we propose consists of a sum of free shifted gaussian bumps, which differ from the original fixed concentrated rings.

differentiable, backpropagation can be done by "unfolding" the Lenia rollout and applying a loss at certain time step(s) like in [104].

However, in the classic version of Lenia, the shape of the kernels are not totally differentiable and not very flexible. To allow easier optimization of the Lenia system, we introduce some changes to the kernel parametrization. In fact in [39], the number of bumps in the kernel (see Figure D.3) is given by the length of a list which can not be differentiated. One option is to let the number of bumps fixed to an arbitrary value (3 for example). However, with a 3 bumps kernels parametrization as in the original paper, you can't recover 2 bumps kernels by putting the height of a bump to 0 (unless you allow to change the maximum sensing radius R which seems unnatural). And thus fixing the number of bumps in the base version restricts a lot the shape of the kernels compared to the original paper where various number of bumps are used.

We therefore introduced a new class of CA with differentiable parameters. To do so, the main shift is to use kernels in the form of a sum of k overlapping gaussian bumps:

$$x \rightarrow \sum_{i}^{k} b_i exp(-\frac{(\frac{x}{rR} - a_i)^2}{2w_i^2})$$

The parameters controlling the shape are $3k$-dimensional vectors: k bumps of height b, size w and center a.

These symmetric "free kernels", while very inspired from Lenia's original "vanilla bumps", allow differentiation and more flexibility and expressivity but at the cost of more parameters. In fact, for example an interesting feature of those "free kernels" is that for a fixed k you can easily have the free kernels with $\leq k$ bumps by putting some heights of bumps close to 0 (And you can even have this happening through gradient descent).

In Lenia, a growth function $G : [0,1] \rightarrow [-1,1]$ is any unimodal non-monotonic function that satisfies $G(\mu) = 1$. In this work, we use the continuous exponential growth function $G(x) = 2\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) - 1$ which is differentiable with respect to μ and σ.

To summarize, the parameters of the update rule are thus those controlling the kernel shape (R, r, a, w, b), those controlling the growth function (μ, σ, h) and a time controlling parameter (T). For a total of n rules (all channels included) with k bumps kernels, the number of parameters is $(3k+4)n+2$. In our experiments, R and T are chosen randomly and fixed while all the other parameters are optimized, and we use a total of $n = 10$ rules with $k = 3$ bumps kernels . So in total we have 132 parameters for the

rules from which 130 are optimized. In addition to the rule, parameters we also optimize the initialization square $I_{square} \in [0,1]^{(40,40)}$.

D.2.2. Obstacles

The multi-channel aspect of Lenia allows the implementation of different types of cells/particles. To implement obstacles in Lenia we added a separate "obstacle" channel with a kernel going from this channel to the learnable "creature" channel. This kernel triggers a severe negative growth in the pixels of the learnable channel where there are obstacles but has no impact on other pixels where there are no obstacles (very localized kernel). This way we prevent any growth in the pixels of the learnable channel where there are obstacles. The formula of the growth function is : $G(x) = -clip((x - 1e - 8), 0, 1) * 10$. Hyperparameters of this handmade rule can be found in Subsection D.2.3.

The learnable channel cells can only sense the obstacles through the changes/deformations it implies on it or its neighbors. In fact, as the only kernel that goes from the obstacle channel to the learnable channel is the one we hand-designed, if a macro agent emerges it has to "touch" the obstacle to sense it. More precisely the agent can only sense an obstacle because its interaction with the obstacle will perturb its own configuration and dynamics (i.e. its shape and the interaction between the cells constituting it). This is similar to experiments with swarming bacteria [504], where the swarm must learn to collectively avoid antibiotic zones (externally-added obstacles) where the bacteria can't live.

In our implementation, obstacles stay still meaning that there is no update of the obstacle channel. To test the agents under moving obstacles, we simply shift the channel of obstacles of a certain amount of pixel at every timestep. This shift of the grid, for an integer value of speed, can be written as a rule of the system from the obstacle channel to the obstacle channel. The rule would be the same on all the grid and is localized as it's a function of the fixed neighborhood. Moving obstacles with a speed with a rational value (for example 0.5 pixels/timesteps) is done in our case by doing the shift every few timesteps.

D.2.3. Lenia rules parameters

Here is the list of the parameters associated to the rules of a Lenia system with C channels, nb_k rules wiht kernels with k bumps. We also provide the range used in this work for the learnable channel. In this work we used C=2 channels (one learnable channel and the fixed channel), $nb_k = 10$ learnable rules and 1 fixed rule (for the obstacles).

- Common to all rules
  - T $\in [1, 10]$
- Learnable rules
  - Kernel (convolution filter) parameters:
    - R $\in [15, 40]$ Radius of the kernels (common to all kernels)
    - r $\in [0,1]^{nb_k}$ relative radius of each kernel.
    - b $\in [0,1]^{nb_k, k}$ height of the k bumps.

    - w $\in [0.01, 0.5]^{nb_k,k}$ width of the k bumps.
    - a $\in [0, 1]^{nb_k,k}$ position of the bumps on the radius.
  - Growth function $G(x) = 2\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) - 1$ parameters
    - $\mu \in [0.05, 0.5]^{nb_k}$ mean of the gaussian growth function.
    - $\sigma \in [0.001, 0.18]^{nb_k}$ variance of the gaussian growth function.
    - h $\in [0, 1]^{nb_k}$
  - $c_0 = [0] \times nb_k$ source channel (0 is learnable channel)
  - $c_1 = [0] \times nb_k$ destination channel
- Fixed rule
  - Kernel parameters:
    - R = 4 small radius for very localized action
    - r = [1,1,1]
    - b = [1,0,0]
    - w = [0.5,1,1]
    - a = [0,0,0]
  - Growth function $G(x) = -clip((x - 1e - 8), 0, 1) * 10$
  - $c_0 = 1$ source channel (1 is fixed channel)
  - $c_1 = 0$ destination channel

D.2.4. Lenia rule parameters mutations

- Common to all rules
  - T : $\mathcal{N}(0, 0.1) \times \mathcal{B}(0.01)$ (mutation then integer)
- Learnable rules
  - Kernel (convolution filter) parameters:
    - R $\mathcal{N}(0, 0.1) \times \mathcal{B}(0.01)$ (mutation then integer)
    - r :$\mathcal{N}(0_{nb_k}, 0.2 \times \mathcal{I}_{nb_k})$
    - b : $\mathcal{N}(0_{3nb_k}, 0.2 \times \mathcal{I}_{3nb_k})$
    - w :$\mathcal{N}(0_{3nb_k}, 0.2 \times \mathcal{I}_{3nb_k})$
    - a : $\mathcal{N}(0_{3nb_k}, 0.2 \times \mathcal{I}_{3nb_k})$
  - Growth function $G(x) = 2\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) - 1$ parameters
    - $\mu : \mathcal{N}(0_{nb_k}, 0.2 \times \mathcal{I}_{nb_k}) \times \mathcal{B}(0.1)$
    - σ: $\mathcal{N}(0_{nb_k}, 0.01 \times \mathcal{I}_{nb_k}) \times \mathcal{B}(0.1)$
    - h $\mathcal{N}(0_{nb_k}, 0.2 \times \mathcal{I}_{nb_k}) \times \mathcal{B}(0.1)$

D.3. IMGEP details

In this section, we first recall the basics of the IMGEP procedure and then go into the details of each element of the method.

Our method described in the Algorithm. 6 starts by initializing a pool of (parameters, reached position) couples by random search, this constitutes the initial state of the history $\mathcal{H}$ (details in Subsection D.3.1). Then, at each iteration, the method iterates through the following steps (illustrated in Figure D.4). **1) Sample a new goal** using a goal sampling strategy which takes into account the previously reached positions (details in

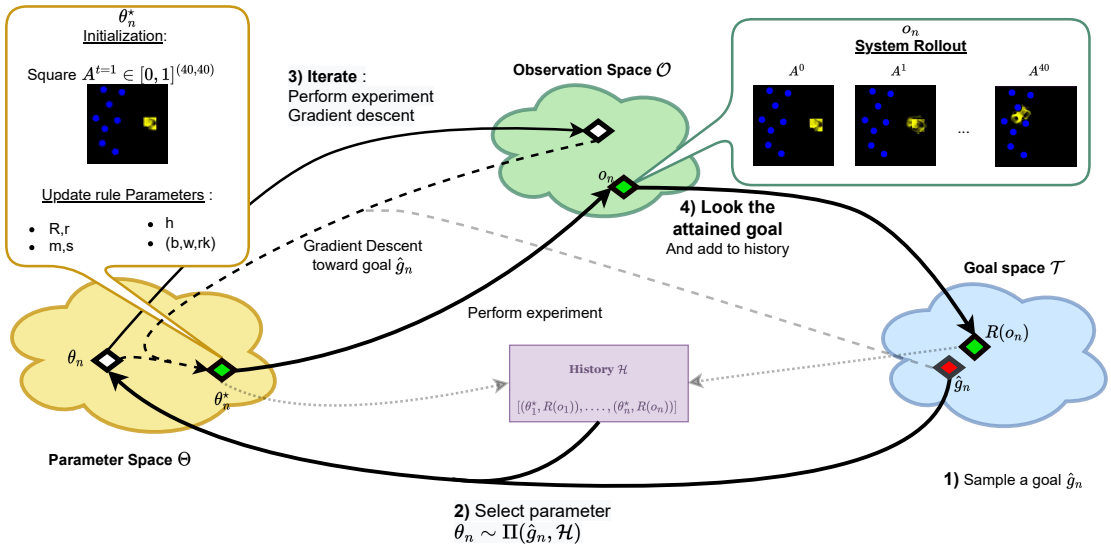


Figure D.4.: IMGEP loop

Algorithm 6: IMGEP pseudo code

Initialization: history $\mathcal{H}$ and models $\mathcal{T}, \Pi, \text{Optim}, R$
for *i=1 to N* **do**
  Generate a target goal $\tau_i \sim \mathcal{T}(\mathcal{H})$; `// use of curriculum learning and diversity search`
  Train parameters on target goal $\theta^*_i = \text{Optim}(\theta_i|\tau_i)$, where $\theta_i \sim \Pi(\mathcal{H}|\tau_i)$; `// use of gradient descent and stochasticity handling`
  Evaluate parameters $x_i \sim R(\theta^*_i)$; `// behavioral characterization`
  Store in history $\mathcal{H} \leftarrow \mathcal{H} \cup (\theta^*_i, x_i)$; `// reuse knowledge for task sampling and training`
return $\mathcal{H}$

Subsection D.3.4). An example of the sampling distribution can be found in green in Figure 7.5.a. **2) Infer starting parameters for that goal** by selecting parameters $\{(\theta_l, A^{t=1}_l)_i\}_{i=1\ldots t-1}$ associated to a previously reached position in history $\mathcal{H}$ that is close to the sampled goal (details in Subsection D.3.8). **3) Optimize parameters toward the sampled goal** by iteratively performing rollouts of the Lenia system under different environmental conditions A_f and applying stochastic gradient descent on the MSE loss between the disk at goal position and the mass of the learnable channel at the last timestep (details in Subsection D.3.6). **4) Update history** $\mathcal{H}$ with the newly obtained parameter point and test it in various environmental conditions A_f to estimate its reached position (details in Subsection D.3.7), such that it can be later reused as a starting point for achieving other sampled goals.

As described in the main text, the behavioral space is the position (x,y) of the center of mass at the last timestep of the rollout. The loss we use is the Mean square error loss between the learnable channel at the last timesteps of the rollout and the same grid with a superposition of 2 disk centered at the goal position in the first channel. The target disk has this formula: $0.9x(0.15x(R_g < 10) + 0.85x(R_g < 5))$ where R_g is the euclidian

distance to the goal position.

To introduce more diversity in the search (and potentially getting out of difficult optimization landscape), some steps of IMGEP add mutation to the promising parameters before applying the optimization through gradient descent. More details can be found in Subsection D.3.5.

Note that we also introduce an automatic way for the method to restart again from scratch in case of not good enough first steps (not present in Algorithm. 6). We refer to Subsection D.3.3 for a detailed description of this restarting mechanism.

The following sections provide additional details about different parts of the method.

D.3.1. Intialization of history

The IMGEP method first applies an initialization of history $\mathscr{H}$ through random search to bootstrap the whole IMGEP procedure.

In this work, the initialization of history consist of 40 trials of random parameters. The range used for this random search are the one presented in Subsection D.2.3 except that we divide the strength of the kernels parameters h by 3. This change is done in order to have weaker/slower updates increasing the chance to have a pattern not exploding or vanishing in 50 timesteps, in order to facilitate further optimization.

This dividing of h by 3 is only to make things go faster (requiring less trials for the initialization of history) but should not be mandatory as random search without this should also get interesting parameters for initialization (although might take more trials).

D.3.2. Warming up goal sampling

To accelerate the curriculum, we start the first 8 steps of the IMGEP with a deterministic goal sampling which tries to go as far as possible on the x axis. The goal position starts at position (-0.19,0) and is shifted of +0.06 along the x axis for every of those deterministic steps. The rest of the goal sampling is stochastic as described in Subsection D.3.4.

D.3.3. Initialization selection

History initialization and the first IMGEP steps have a huge impact on the performance of the method, as it will provide the basis for all subsequent optimization. History initialization and the warm up of goal sampling have a huge impact on the performance of the method, as it will provide the basis for all subsequent optimization.

To mitigate this problem, we also apply initialization selection with the objective of facilitating further optimization. We run the first steps of the method (random initialization and few steps of optimization), and observe the loss for the 3 first deterministic targets (described in Subsection D.3.2). If this loss is above a certain threshold for one of the 3 step, we start over again getting rid of the initialization history and

initializing it again with random search. We perform this until we find a "good" initialization that is below the threshold for the 3 steps.

D.3.4. Goal sampling

The goal sampling we chose in this work intends to sample goals ((x,y) positions) that should be most of the time further in the grid (for harder goals), not too far from previously reached positions (for feasibility of the goal) and also not too close from previously achieved goals (to make progress) . From those heuristic we introduce our engineered goal sampling strategy in pseudo code Algorithm. 7. The objective of this engineered sampling is to accelerate the search but much simpler ones could work if given enough computational budget.

Algorithm 7: Goal Sampling Strategy

Input : history $\mathcal{H}$
while *nb_close* < 1 *or nb_veryclose* > 2 **do**
  if $rand \sim \mathcal{U}(0,1) < 0.2$ **then**
    goal = bestgoal($\mathcal{H}$) +
     $(\mathcal{U}(0,1) \times 0.04 + 0.02, (\mathcal{U}(0,1) \times 0.45 - 0.22)/4)$; `// Try a location slightly further than the previous best`
  else
    if *rand* $\sim \mathcal{U}(0,1) < 0.7$ **then**
      goal = $(-\mathcal{U}(0,1) \times 0.2 + 0.35, -\mathcal{U}(0,1) \times 0.45 - 0.22)$
    else
      goal = $(-\mathcal{U}(0,1) \times 0.35 + 0.35, -\mathcal{U}(0,1) \times 0.45 - 0.22)$
  nb_close, nb_veryclose = calc_distances(goal, $\mathcal{H}$)
return goal

D.3.5. Mutation

We apply mutations on candidates parameters in order to increase diversity. Some mutations can facilitate optimization while others can lead to undesirable configurations impairing it. For this reason, we apply less gradient steps on those mutated parameters. See Subsection D.3.9 for the hyperparameters in this work.

In addition, we generate mutations of a parameter configuration until it results in a pattern not collapsing after 50 timesteps. For this (approximate) collapsing measure, we use a simple soft filter checking if the total mass in the learnable channel at the last timestep is >10 (to test for death of matter) and if the mean square error between the learnable channel at the last timestep and the disk defined in Section D.3 centered on the center of mass of the learnable channel is < 25 (as a proxy for explosion of the mass, more details in Subsection D.3.7). This loop of mutations is counted in the total number of rollout performed by the IMGEP.

We refer to Subsection D.2.4 for the mutation (distribution, mean, variance) applied to each parameters in the method.

D.3.6. Gradient descent

Differentiating through Lenia can be difficult because the gradient must backpropagate through several steps (which moreover have their result clipped between 0 and 1) without vanishing. We should thus limit ourselves to a few iterations when training: in our experiments the loss is applied after 50 steps in Lenia.

Obtaining gradients that are informative for optimization requires an overlapping between the mass in the learnable channel and the disk centered at the goal position. The curriculum we introduce in the goal sampling procedure facilitates this overlap by generating goals that neither too far nor too close from the initial pattern at t=0 and from previously reached goal.

D.3.7. Parameter evaluation

We perform an evaluation of the parameters after each IMGEP step (sampling of goal and optimization of parameters). This evaluation consists of running 20 rollouts of 50 timesteps (the same rollout length as in the optimization rollout) with different random obstacle configurations and measures the average reached position over those rollouts.

For each rollout, we also compute the mean square error between the learnable channel at the last timestep and the disk shape centered on the center of mass of the learnable channel at last timestep. We then take the average value over the rollouts. This is used as a proxy "collapsing measure" (explosion or death of the pattern) to apply a soft filter when selecting promising initialization parameter for a new goal as explained in Subsection D.3.8.

The parameters (A_l, θ_l), the measured reached position (r_x, r_y) and collapsing proxy measure c are then stored in the history $\mathcal{H}$.

D.3.8. Reusing history $\mathcal{H}$ for a new goal

Once a goal is selected, we compute the L2 distance between all vectors (c, r_x, r_y) of the history and (c_{goal}, g_x, g_y), where g_x and g_y are the (x,y) coordinate of the goal and c_{goal} is a constant equal to 0.065 in this work. These L2 distances are used to select a point in the history reaching a position close to the goal while mitigating the risk of collapsing.

In addition to these L2 distances for the selection of potential candidates for a new goal, we also filter out the points in the history having $c > 0.11$ allowing to remove the potential collapsing ones even though they might be close to the goal, as collapsing parameters are hard to recover from through gradient descent.

The candidate parameter for a goal is therefore the point in the history which has $c <= 0.11$ and which minimize the L2 distance to the goal.

D.3.9. IMGEP search hyperparameters

- Number of IMGEP steps : 120
- History initialization : 40 trials of random parameters.
- In 4 out of 5 IMGEP step, we mutate the candidate parameter before gradient descent.
- Number of gradient steps : 125 when no mutation beforehand (1 out of 5 IMGEP steps) , 15 when mutation beforehand.
- Rollout length : 50 timesteps
- Grid size : 256x256
- Number of obstacle during the search: 8
- Initialization position on the 256x256 grid: [36:76,105:145]

D.4. Battery of Empirical Tests

Note that the tests we provide are proxy measure of agency/stability, and so what we present here are what we consider in this paper as agency. It is for example impossible to test for infinite time stability in finite time budget. Our stability tests are based on previous work on Lenia [•1].

D.4.1. Empirical agency test

For the agency test, we first apply a prefilter to the obtained parameters by running a rollout of 500 steps with the obtained parameters. From this rollout, we measure if the mass at the last timestep was strictly above 0 (not dead) and below 6400 (explosion). The number are arbitrary and relatively "loose" so that we reject nearly no "false positive". This prefilter allows to throw out obvious non interesting parameters to reduce the computational cost of testing all obtained parameters – especially for the random search method where many of them are not interesting.

We then do rollout of 2000 timesteps for the empirical agency and moving test. The rollout is long (especially relative to the 50 timesteps of the search) in order to probe for long term stability. We compute some stats, from the rollout observations, which are used for the empirical agency test (and moving test) of the parameters inspired by [•1].

The empirical agency test consist of :

- Measuring if the mass of the learnable channel is > 0 and <6400 (~10% of the map) at the last timestep of the rollout as those correspond to collapse and explosion.
- Measuring if the average mass is augmenting or decreasing too much between 2 windows of the rollout. This is a proxy measure for long term instability meaning that a big loss or increase of mass between the 2 windows is most of the time an indicator for long term instability. In this work, we measure the ratio between the average mass during the 0 to 500 window and 1500 to 2000 window. If this ratio is greater than 2, the parameters do not pass the test. The windows are relatively large to still allow for variation of mass during a rollout and the formation of a pattern in the first window.

- We also want the emerging pattern to be a spatially localized **Soliton** (ie pattern forming a single entity not expanding indefinitely, with a bounded radius). To measure this, we perform a connectivity analysis of the pattern depending on the kernel radius, rejecting patterns where two distinct blobs of mass cannot influence each other (distance between blobs $\geq R * max(r)$).

D.4.2. Moving test

To test if a pattern passing the empirical agency test is moving, we measure if the center of masse of the learnable channel moved further than 100 pixels from the initialization position at any point during the 1000 first steps of the rollout.

D.4.3. Speed measure

To measure speed of agents, we use the 2000 timesteps rollout computed in the filter phase and track the average distance travelled by the center of mass of the agent on sliding overlapping windows of size 25 starting from timestep 150 to timestep 2000. The result is divided by 25 (the size of the sliding window) in order to have a per timestep average distance travelled. We use a sliding window to filter slight back and forth movement of the center of mass (which can even be due to self organization without clear "movement" of the whole). Note that we compute the speed only for agents passing the filters above.

The same is done to measure speed with obstacles but we average on the 50 rollouts with random obstacles computed in the robustness test. The only small modification is that if an agent does not pass the survival tests above on the rollout (for example its mass reaches 0), we set the speed for this rollout to 0.

D.4.4. Basic obstacles tests

We then test the parameters leading to moving agents by performing 50 rollouts of 2000 timesteps where obstacles are the same as in training i.e. obstacles of radius 10. We place 24 obstacles in the whole grid (compared to only the right part of the grid in training), from which 23 are randomly placed and one being in the trajectory of the moving agent to be sure that it will encounter at least one obstacle in the rollout. To do this we look at the achieved position of the moving agent without obstacle at timestep 1000 and put an obstacle here in the test for every rollout. We also remove any obstacle pixel in the initialization area (pixel of the learnable channel >0 at the initialization) as well as in a radius of 10 pixels (euclidian distance) of the initialization (to let some space for the initialization to develop).

To get the robustness measure we then measure the fraction of rollout where the pattern pass the empirical agency test.

D.4.5. Generalization tests

Here is a full description of each of the generalization test conducted in the *Generalization* section in the main text. For all the quantitative generalization tests, we used the same robustness test as above except that we do it on 10 random trials instead of 50: we run rollout of 2000 timesteps, then measure if it fulfills the empirical agency test. The measure of robustness is again measured by the proportion of trials where the agent pass the empirical agency test (between 0 and 1).

The resulting robustness obtained for the agents discovered by the IMGEP, random and handmade search variants is given in Table D.2.

- **Initialization noise**. In this experiment, we add a centered gaussian noise to the pixel of the initialization square A^1. In the first test "init noise rate" we vary the proportion of pixels affected by this gaussian noise, testing proportions [0.2,0.4,0.6,0.8,1.], and keep the variance fixed to 1. In the "init noise std" test, we apply the noise to all pixels of the initialization but vary the variance of the gaussian in [0.5,1.5,2.5,3.5,4.5].
- **Obstacles** In all of these test we also remove obstacles pixel from the initialization square and in a radius of 10 pixels (euclidian distance) around it.
  - **Obstacle radius** In this test, we vary the radius of the obstacles in [4,7,10,13,16]. The number of obstacles varies according to the radius of obstacles to keep the same ratio of obstacle pixels with the default one which is 24 obstacles of radius 10. The formula is Number obstacles = $24 \times (10/var)^2$.
  - **Obstacles number** In this test, we vary the obstacle number keeping the radius fixed to the default one (radius=10). We try obstacle number= [24, 30,36,42 ,48] .
  - **Obstacle speed**. In this test, we change the dynamic of the obstacle channel so that obstacle move at a certain speed as detailed in Subsection D.2.2. For a speed of 1, the obstacle channel is shifted of 1 on the left at every timestep, for a speed of 0.5, the obstacle channel is shifted of 1 every 2 timesteps. We tested obstacle speed of [1/3,1/2,1,2,3]. In this test we put 24 obstacles of radius 10.
- **Scale** In this test, we vary the scale of agents by changing their kernel size multiplying the parameter R of the simulation by the factor. A smaller (resp bigger) size of kernel means that the convolution will cover a smaller (resp bigger) neighborhood. We also change the initialization size by a factor α to match the scale. To do this, we use a downscaling (or upscaling) of the initialization 40×40 square with bilinear interpolation. We test both smaller sizes : 0.15,0.65 , as well as bigger sizes: 1.15,1.65,2.15.
- **Update**. In this tests, we perturb the update (what is added to the current state) from step 0 until step 1900. We let the step from 1900 to 2000 free of update perturbation to allow the rule to recover until step 2000 for the statistics computation.
  - **Update mask** In this test, for a value of update mask p<1, every pixel has a probability p of being updated while the rest of the pixels will keep the same value. This does not apply to the update applied by the obstacles. For a value 1<p<2, each pixel

is updated one time using the update rule normally (sensing and add of growth) giving a new state and then each pixel is updated again from this new state with a $p-1$ probability (the sensing on the potential second random update is done by sensing the new state). We test the update mask rate in [0.2,0.6,1.,1.4,1.8].

- **update noise std** In this test, we add noise to the update of the learnable channel before the clipping as such :

$$A_l^{t+1} = A_l^t + \frac{1}{T}\left(G(K * A^t) + \mathcal{N}(0_{256\times256}, \sigma \mathbb{I}_{256\times256})\right)$$

where $\mathcal{N}(\mu, \Sigma)$ is a gaussian vector of mean μ and variance Σ. We vary σ in $[0.5, 1.5, 2.5, 3.5, 4.5]$

- **Update noise rate**. We add noise to the update of the learnable channel before clipping. Every pixel has a probability $p \in [0.2, 0.4, 0.6, 0.8, 1.]$ to have a gaussian noise of mean 0 and variance 1.

- **Morphological computation (Hand damage)**. In this test, we allow an exterior experimenter to pause the simulation and put pixels of the learnable channel to 0. After the damage, we then let the simulation unroll as usual starting from the damaged state.
- **Interactions (Multi agents setting)**. We allow to put several initialization square in the learnable channel. As the update rule apply to all the grid the same way, if a couple (initialization square, update rule) already led to a an agent in the case of a single initialization square then several of them that are not interfering (further enough so that the convolution of a pixel of one does not contains pixels of the other) will lead to several agents.
- **Custom obstacles**. We allow an experimenter to freely draw obstacle in the grid. This allows to have obstacles with shapes not seen during training.
- **Custom init states** In this test, we replace the initialization of the pattern (that was optimized) by simple arbitrary shape such as disk with a gradient (the gradient being to have an asymmetry for movement), disk of large size etc. The web demo at http://developmentalsystems.org/sensorimotor-lenia-companion also allows to load any image as initialization of the system.
- **External control** This experiment consists in adding a new channel (a new type of cell) to the system which we want to act as an attractive element. We conducted a semi handmade search in order to search for a rule, sensing in the attractive channel and updating the learnable channel, leading to this attractive behavior.
  Note that this attractive element should attract but not disturb too much the matter as we don't want the attractive matter to be able to destroy the agent dynamics.
  In fact, we first searched for a rule tuned for one agent found with the IMGEP search (ie one parameter point (A_l, θ_l)). By doing so, the rule is adapted to the dynamic of this specific agent (for example different agents might have different range for pixel value or growth etc).
  The search for a rule (tuned for a specific agent) is semi handmade. We first preselect some rule parameters from a set of random rules. The preselection is done by moving a circle of attractive mass along

a predefined straight trajectory in an environment with a moving agent. We then look if the attractive mass and the agent overlaps at the last timestep which should mean that the agent followed this attractive mass. An experimenter then select by hand the rules that lead to attraction of mass without too much perturbation by controlling the mass of attractive matter in a real time simulation with the moving agent.
After searching for a rule for a specific agent, we then tested it on some other moving agents obtained with IMGEP. Some agents (some parameters (A_f, θ_f)) are more prone to work with it (meaning attraction while not affecting the stability too much) while it destroy the stability of others. The reported qualitative results on this test are performed on agents where the rule leads to stable attraction.

D.5. Comparison baselines

D.5.1. Random search details

We use uniform sampling of parameters with the ranges given in Subsection D.2.3. The initialization 40x40 square is randomly sampled with each of the pixel constituting it being independently sampled following a uniform distribution between 0 and 1.

D.5.2. "Handmade" agents (from original Lenia paper)

The parameters from this dataset are the one from the original Lenia paper [39, 40], which are accessible on this and this links. Contrary to the rest of the paper we use the classic parametrization of Lenia for the agent channel. We filter out the discoveries that have more than one channel or an initialization that has a side bigger than 256. We then apply the pre-filter and filter as explained in Subsection D.4.1.

E. Appendix of Flow Lenia

(a) Website and Videos

(b) Notebook

Figure E.1.: Scan (or click on) the above QR codes for accessing the paper's (a) companion website with supplementary videos, and (b) jupyter notebook

E.1. Model

Let $\mathscr{L}$ be the support of a CA, which is the two-dimensional grid $\mathbb{Z}^2$ in the rest of this work. Let $A^t : \mathscr{L} \to S^C$ be CA's activations at time t, with $A_i^t(x)$ the activation in location $x \in \mathscr{L}$, channel i and time t. C is the number of channels of the system ans S is the state space (the set of states a cell can take in each channel).

E.1.1. Lenia

The state space S in Lenia is the unit range $[0, 1]$. An instance of Lenia is defined by a tuple $< K, G, A^0 >$ where K is a set of convolution kernels where $K_i : \mathscr{L} \to [0, 1]$ satisfies $\int_{\mathscr{L}} K_i = 1$ and G is a set of growth functions with $G_i : [0, 1] \to [-1, 1]$. Each pair (K_i, G_i) is associated to a source channel c_0^i it senses and a target channel c_1^i it updates. Connectivity can be represented through a square adjacency matrix $M_{C,C} = \begin{bmatrix} m_{11} & \cdots & m_{1C} \\ \vdots & \ddots & \vdots \\ m_{C1} & \cdots & m_{CC} \end{bmatrix}$ where $m_{ij} \in \mathbb{N}$ is the number of kernels sensing channel i and updating channel j. A^0 is the initial state of the system. As in Hamon et al. [•5], kernels are radially symmetrical and defined as a sum of concentric Gaussian bumps :

$$K_i(x) = \sum_{j=1}^{k} b_{i,j} \, exp\left(-\frac{(\frac{x}{r_i R} - a_{i,j})^2}{2w_{i,j}^2}\right) \tag{E.1}$$

Where a_i, b_i, w_i and r_i are parameters defining kernel i. k is a parameter defining the number of rings per kernel (set to 3 here) and R is a parameter common to all kernels defining the maximum neighborhood radius. Each

kernel is then defined by $3 \times k + 1$ parameters. Growth functions are defined as Gaussian function scaled in the range $[-1, 1]$:

$$G_i(x) = 2\ exp\left(-\frac{(\mu_i - x)^2}{2\sigma_i^2}\right) - 1 \tag{E.2}$$

Where μ_i and σ_i are parameters of growth function i so each growth function is defined by 2 parameters. A step in Lenia is defined by the following steps:

1. Compute the growth at time t given the actual state A^t:

$$U_j^t = \sum_{i=1}^{|K|} h_i \cdot G_i(K_i * A^t_{c_0^i}) \cdot [c_1^i = j] \tag{E.3}$$

   Where $h \in \mathbb{R}^{|K|}$ is a vector weighting the importance of each pair (K_i, G_i) and $[c_1^i = j]$ is the Iverson bracket which equals 1 if $c_1^i = j$ and 0 otherwise (*i.e.* equals 1 if the ith pair updates channel j).
2. Add a small portion of the growth U^t to the actual state A^t to get the state at the next time step and clip results back to the unit range:

$$A_i^{t+dt} = [A_i^t + dt\ U_i^t]_0^1 \tag{E.4}$$

E.1.2. Flow Lenia

Flow Lenia is a mass-conservative extension to Lenia. By mass-conservative, we mean that the sum of activations across all cells and for each channel is constant over time :

$$\sum_{x \in \mathcal{L}} A_c^t = \sum_{x \in \mathcal{L}} A_c^{t+dt}\ , \forall t, \forall c \in \{1, ..., C\}$$

We propose for this system to interpret activations as concentrations of "matter" in all cells and to refer to the term U^t, previously called the growth in Lenia, as an affinity map. The idea is that the matter will move towards higher affinity regions by following the local gradient of the affinity map U, $\nabla U : \mathcal{L} \to \mathbb{R}^2$. To do so, we define a flow $F : \mathcal{L} \to (\mathbb{R}^2)^C$, which can be interpreted as the instantaneous speed of matter, as:

$$\begin{cases} F_i^t = (1 - \alpha^t)\nabla U_i^t - \alpha^t \nabla A_\Sigma^t \\ \alpha^t(p) = [(A_\Sigma^t(p)/\theta_A)^n]_0^1 \end{cases} \tag{E.5}$$

With $A_\Sigma^t(p) = \sum_{i=1}^C A_i^t(p)$ the total mass in each location p. Here ∇U_i^t is the affinity gradient for channel i. The negative concentration gradient $-\nabla A_\Sigma^t$ is a diffusion term to avoid concentrating all the matter in very small regions akin to the clipping in Lenia which upper bounds concentrations. In practice, gradients are estimated through Sobel filtering. Map $\alpha : \mathcal{L} \to [0, 1]$ is used to weight the importance of each term such that $-\nabla A_\Sigma^t$ dominates when the total mass at a given location is close to a critical mass $\theta_A \in \mathbb{R}_{>0}$. Intuitively, the result is that matter is mainly driven by concentration gradients in high concentrations regions and is more free to move along the affinity gradient in less concentrated areas. We

typically use $n > 1$ such that the affinity gradient dominates on a larger range of masses.

Then, we can move matter in space according to flow F giving us the state at the next time step. To do so we use the reintegration tracking method proposed in Moroz [195]. Reintegration tracking is a semi-Lagrangian grid based algorithm thought as a reformulation of particle tracking in screen space (*i.e.* grid space) aimed at not losing information (*i.e.* particles) which happens when two particles end up in the same cell. Figure E.2 illustrates how reintegration tracking is used in our case. The resulting update rule is the following :

$$A_i^{t+dt}(p) = \sum_{p' \in \mathcal{L}} A_i^t(p') I_i(p', p) \tag{E.6}$$

Where $I_i(p', p)$ is the proportion of incoming matter in channel i going from cell $p' \in \mathcal{L}$ to cell $p \in \mathcal{L}$:

$$I_i(p', p) = \int_{\Omega(p)} \mathcal{D}(p_i'', s) \tag{E.7}$$

With $p_i'' = p' + dt \cdot F_i^t(p')$ the target location of the flow from p'. $\Omega(p)$ is the domain of cell at location p, which is a square of side 1. $\mathcal{D}(m, s)$ is a distribution defined on $\mathcal{L}$ with mean m and variance s satisfying $\int_{\mathcal{L}} \mathcal{D}(m, s) = 1$, which is in practice a uniform square distribution with side length $2s$ centered at m. This distribution emulates a flow of particles from source area $\Omega(p')$ to target area $\mathcal{D}(p'', s)$, where the distribution $\mathcal{D}$ emulates Brownian motion at the low level. s is an hyperparameter of the system which can be seen as form of temperature. The reintegration tracking method is depicted in Fig. E.2. Since the distribution $\mathcal{D}$ integrates to 1, a cell cannot send out more mass than it contains nor less and so the system conserves its total mass. Mass conservation also implies that cells' states are no longer bound to the unit range but can be any positive real valued number ($S \equiv \mathbb{R}_{\geq 0}^C$). This model has been implemented in JAX [198] allowing fast simulation on GPU ($255\mu s \pm 3.11\mu s$ per step on Tesla T4 GPU with 1 channel, 10 kernels and 128×128 world size).

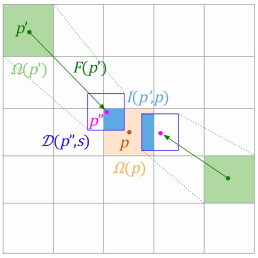


Figure E.2.: Calculation of incoming matter to cell $p \in \mathcal{L}$ through reintegration tracking [195]. Mass contained in cell at location $p' \in \mathcal{L}$ is moved to a square distribution $\mathcal{D}$ centered on $p'' = p' + dt \cdot F^t(p')$. The proportion of mass from p' arriving in p is given by the integral of $\mathcal{D}$ on the cell domain of p, $\Omega(p)$, denoted as $I(p', p)$.

E.1.3. Parameters Localization

Flow Lenia allows to embed the update rule parameters inside the CA. Intuitively, we can "attach" a vector of parameters to the matter locally modifying how it behaves (*i.e.* locally modifying how the affinity map is computed), and let it flow with it. Formally, this comes to defining a parameter map $P : \mathcal{L} \to \Theta$ where Θ is a given parameter set. This map can be used to locally modify the update rule. For instance, we can embed the $h \in \mathbb{R}^{|K| \cdot |K|}$ matrix weighting the importance of each kernel in the affinity map computation (see equation E.3), giving :

$$U_j^t(p) = \sum_{i,k} P_k^t(p) \cdot G_k(K_k * A_i^t)(p) \tag{E.8}$$

Then parameters can be moved along with matter. A question is how to mix parameters arriving in the same cell. Here we propose two different methods which are respectively *average* and *softmax sampling*. The former

makes a weighted average of incoming parameters with respect to the quantities of incoming matter and is formally defined as :

$$P^{t+dt}(p) = \frac{\sum_{p' \in \mathcal{L}} A^t(p') I(p', p) P^t(p')}{\sum_{p' \in \mathcal{L}} A^t(p') I(p', p)} \tag{E.9}$$

Softmax sampling on the other hand samples a parameter in the set of incoming ones following the softmax distribution given by incoming quantities of matter :

$$\mathbb{P}[P^{t+dt}(p) = P^t(x)] = \frac{\exp(A^t(x) I(x, p))}{\sum_{p' \in \mathcal{L}} \exp(A^t(p') I(p', p))} \tag{E.10}$$

Intuitively, the more represented set of parameters has a greater probability of being selected in the cell, like simulating in one step a competition between different parameters in the cell. Note that we could also sample each element of the vector of parameters independently giving some crossover mechanism. We can also add mutations in the simulation. For instance, we can, at a given rate, modify parameters in a randomly sampled zone by adding Gaussian noise to the parameter map.

E.1.4. Physical resources in the Environment

While parameter embedding could lead to the emergence of intrinsic evolutionary processes where parameters would compete for available matter, we can also add intrinsic selective pressures in the form of environmental constraints in the environment.

temperature What we call "temperature" in Figure 7.12a is the size of the reintegration tracking distribution s (see equation E.7). We can vary it locally to obtain varying effects of temperature in the same grid.

Food The idea is to add food resources that creatures would need to collect in order to replenish their own constantly decaying pool of resources. To do so, we let matter decay at a fixed rate ρ_{decay}, and create a food map $\Psi : \mathcal{L} \to \mathbb{R}_{\geq 0}$. When matter is in a cell where there is also food, then food is transformed into matter at a given rate ρ_{digest} giving the following update.

$$\begin{cases} A^{t+dt}(x) = \cdots + [A^t(x) \rho_{digest}]_0^{\Psi^t(x)} - A^t(x) \rho_{decay} \\ \Psi^{t+dt}(x) = \Psi^t(x) - [A^t(x) \rho_{digest}]_0^{\Psi^t(x)} \end{cases} \tag{E.11}$$

Where $\cdots$ refers to the update equation E.6 and $[\cdot]_a^b$ is the clip function between a and b. We enable creatures to sense food by adding kernels and growth function from the food map Ψ to creatures' channels A.

E.2. Results

E.2.1. Random search

Random search is performed in the Flow Lenia parameter and hyperparameter space described in Table E.1. Initial patterns A^0 are set with a 40×40 patch with matter drawn from uniform distribution in the center of the grid and no matter everywhere else.

Neighborhood			Growth functions		
R	$\in [2, 25]$		μ	$\in [0.05, 0.5]$	*
r	$\in [0.2, 1]$	*	σ	$\in [0.001, 0.2]$	*
Kernels			**Flow**		
h	$\in [0, 1]$	*	s	0.65	
a	$\in [0, 1]^3$	*	n	2	
b	$\in [0, 1]^3$	*	dt	0.2	
w	$\in [0.01, 0.5]^3$	*			

Table E.1.: Flow Lenia explored parameter space. Parameters marked with a * must be sampled for each kernel-growth function pair.

E.2.2. Optimizing Flow Lenia creatures

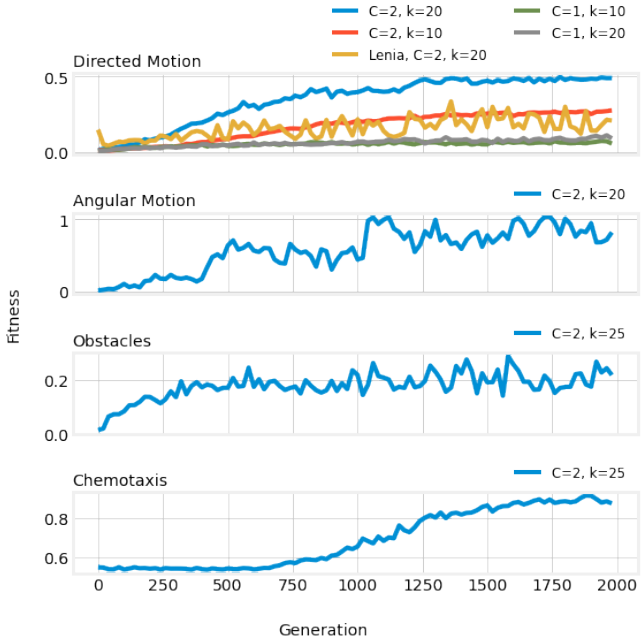


Figure E.3.: Results of evolutionary optimization. C is the number of channels of the system and k is the number of kernels and growth functions. When performing the exact same optimization for directed motion in the original Lenia system (yellow curve), not only optimization is unstable but it only discovers exploding patterns.

Using evolutionary strategies [283] to optimize the update rule parameters and the initial configuration (A^0) with respect to user-defined fitness functions, we have been able to successfully find good solutions for 4 different tasks : directed motion, angular motion, navigation through obstacles and chemotaxis.

We used evosax [505] implementation of the OpenES strategy with population size of 16 and adam optimizer [484] with 0.01 as learning rate. We optimized the Flow Lenia update rule with different number of kernels and either 1 or 2 channels. For comparison, we also trained original Lenia on the directed motion task following the same optimization procedure. The initial pattern is composed, as in random search, of a square patch

with non-zero activations placed at the center of the world and zeros everywhere else.

E.2.2.1. Directed motion

In order to train creatures displaying directed motion, *i.e.* straight line motion, we used the distance traveled by the creature as the fitness function. The distance is calculated by computing the center of mass of the pattern at step 0 and final step 400. Formally, the fitness function is defined as :

$$f(\theta) = dist(\phi(A^0), \phi(A^{400}))$$

Where $A \equiv \{A^0, ..., A^T\}$ is the pattern obtained by making a rollout with parameters θ for T timesteps (here 500). $\phi(A^t) \in [-0.5, 0.5]^2$ is the center of mass of state A^t and $dist$ is the euclidean distance function. We optimized the system with either 1 or 2 channels and 10 or 20 kernels. We used $M = \begin{bmatrix} 5 & 5 \\ 5 & 5 \end{bmatrix}$ as the adjacency matrix with 2 channels and 20 kernels and $M = \begin{bmatrix} 3 & 2 \\ 2 & 3 \end{bmatrix}$ with 10 kernels.

Results (see Figure E.3) show that good solutions can be found in the 2 channels condition but not in the single channel case. However, when running the algorithm for longer (e.g 5000 generations), we have been able to found single channel creatures with similar fitness than their 2 channels counterpart. Increasing the number of kernels led to faster discovery of good solutions. The best performing creature is shown in Figure 7.10. On the other hand, the optimization of the original Lenia model is much less stable and discovered patterns are less successful than their mass-conservative counterparts. Moreover, every Lenia optimized patterns are exploding ones.

E.2.2.2. Angular motion

In this task, we want emerging creatures to display more complex forms of motion. More precisely, we want creatures to be able to move and make turns. As with directed motion, we use the center of mass of the creature through time to compute its trajectory. The fitness function is the following :

$$\begin{aligned} f(\theta) = & \, dist(\phi(A^0), \phi(A^{200})) \\ & + dist(\phi(A^{200}), \phi(A^{400})) \\ & + \angle[\phi(A^{200}) - \phi(A^0)], [\phi(A^{400}) - \phi(A^{200})] \end{aligned}$$

Where $\angle ab$ is the angle between vectors a and b. The first two terms are the distance traveled from step 0 to step 200 and from step 200 to step 400. The last term is the angle between these two trajectories which is maximal when they are opposite. In order to avoid large angles to come from very small movements, the angle is set to 0 when distance traveled either before or after step 200 is below a given threshold. The optimal behavior for this fitness function is then to move fast in one direction, make a 180° turn, and then move fast in the opposite direction. We used

2 channels, 20 kernels and the same connectivity matrix as for directed motion. Result are shown in Figure E.3. The best performing creature is shown in Figure 7.10.

E.2.2.3. Navigation through obstacles

In this task, we want to see if creatures can navigate through obstacles as done in Hamon et al. [•5]. To do so, we added walls which are implemented by adding a strong flow going from the center of walls outwards, thus strongly repelling the creature and acting as a solid obstacle. At each evaluation of the optimization process, we randomly sample points on a circle surrounding the creatures' initial positions to be walls positions thus making a "forest" of walls around the creature. We then optimize the creature with the same fitness function as in the *directed motion* task so creatures have to go as far as possible and so through the forest. We made the experiment with 2 channels creatures, walls are defined in a separate third channel. We used 25 kernels and $M = \begin{bmatrix} 5 & 5 & 0 \\ 5 & 5 & 0 \\ 5 & 0 & 0 \end{bmatrix}$ as the connectivity matrix so creatures are able to sense the walls channel (3rd channel). We have been able to successfully train creatures able to move and stay robust when making contact with walls such as the one shown in Figure 7.10.

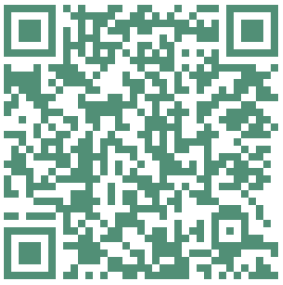
E.2.2.4. Chemotaxis

Another important feature of natural life-forms is the ability to sense their environment in order to find food or avoid dangers through chemotaxis. In this task, we want creatures to be able to sense a "chemical" gradient and climb it towards its maximum. To do so, we added a separate channel $\Gamma : \mathcal{L} \to \mathbb{R}_{\geq 0}$ whose activations are defined following a Gaussian function around a point randomly sampled on a circle surrounding the center of the CA for each evaluation of the optimization process ensuring creatures learn to follow gradient and not a fixed direction while keeping the distance to cover constant. We also added 5 kernels and growth functions from Γ to A, which are also optimized, so the creature is able to sense the chemical. The fitness of an individual is then computed with the following function :

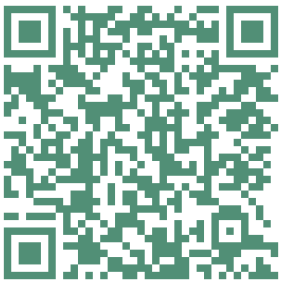
$$f(\theta) = \frac{\sum_{x \in \mathcal{L}} A_{\Sigma}^{500}(x) \times \Gamma(x)}{\sum_{x \in \mathcal{L}} A_{\Sigma}^{500}(x)}$$

Since mass is conserved, the optimal behavior for a creature is to concentrate as much of its mass in the cells where Γ is maximal. We have been able to find good solutions to this task as shown in Figure E.3. Best solutions such as the one shown in Figure 7.10 are perfectly able to climb the gradient towards its maximum.

F. Appendix of GRN



(a) Interactive Paper and Tutorials



(b) Codebase

Figure F.1.: Scan (or click on) the above QR codes for accessing the paper's (a) interactive version and accompanying tutorials, and (b) github repository

F.1. Data Availability

Source code is available on GitHub (click on QR codes). It contains experimental data and an executable notebook version of the paper to reproduce all paper figures, as well as additional step-by-step tutorials to reproduce results from scratch for Figure 8.5, Figure 8.7 and Figure 8.9 (tutorial 1) and Figure 8.10 (tutorial 2), as well as the codebase to reproduce the whole experimental campaign. All our codebase is open-source under MIT License.

F.2. Materials and Methods

F.2.1. GRN models and numerical simulation

This study employs ordinary differential equation (ODE) models to represent molecular pathways, with nodes representing pathway components and edges capturing their interactions. The continuous node states, encompassing variables like gene expression levels and protein concentrations, are interconnected through a system of ODEs, enabling the modeling of complex regulatory dynamics. ODE models are often available in the Systems Biology Markup Language (SBML), a standardized format that contains essential information about variables, parameters, equations, and model metadata in XML files.

To perform numerical simulations of ODE SBML models, we rely on the SBMLtoODEjax python library, a recent development that automates the parsing and conversion of SBML models into python models written entirely in JAX [•7]. Taking advantage of JAX computing capabilities, SBMLtoODEjax enables efficient and parallel numerical solutions for gene expression levels and other node states by recursively invoking the generated python models to integrate the ODE equations with current gene expression levels. Additionally, we have developed a python library (https://github.com/flowersteam/autodiscjax) comprising additional modules and pipelines that facilitate interventions on the GRN models such as genome or drug interventions, as well as other perturbations such as noise, pushes, and walls that can be applied to the states and kinematic parameters of gene regulatory networks.

Given the model species initial state $y(t = 0)$, the desired rollout length $T(secs)$ and step size ΔT, as well as the chosen intervention i and/or perturbation u, the model rollout iteratively 1) integrates the system of ODE-governed equations that specifies the rate of species changes $\frac{dy}{dt}$ using JAX odeint solver to update model species $y(t) \rightarrow y(t+\Delta T)$, 2) calls the model assignment rules to update kinematic parameters if needed, and 3) apply the intervention and/or perturbation function to update $(y(t + \Delta T), w(t + \Delta T), c)$ accordingly. In this paper we use $T = 2500secs$ and $\Delta T = 0.1$ (25 001 time points per rollout including t_0). The ODE solver uses an absolute tolerance of $1e^{-6}$ and relative tolerance of $1e^{-12}$, with maximum number of solver steps of 1000. For a step-by-step guide on utilizing these libraries within the proposed framework, we refer interested readers to our tutorial (https://developmentalsystems.org/curious-exploration-of-grn-competencies/tuto1.html), which offers practical examples and detailed instructions.

F.2.2. Experimental setup

In our computational models, we are able to record the activities of all nodes during a model rollout. The observation space $O \subset \mathbb{R}_+^{n \times \frac{T}{\Delta T}}$ is such that $o = (y(0), \ldots, y(T))$ where $y(t)$ represents the n-dimensional vector of node states at each time step, with T being the total reaction time. The boundaries of the observation space are not known.

Regarding the exploration of problem spaces, namely the intervention space I and behavior space Z, we specify them as follows.

For the main experiments on biological networks, the intervention space $I \subset \mathbb{R}_+^n$consists of initial node states sampled from the hyper-rectangle $[y_{0,\min}, y_{0,\max}]$ where $y_{0,\min} = \frac{1}{r} \times y_{d,\min}$ and $y_{0,\max} = r \times y_{d,\max}$ with $r = 20$ and $(y_{d,\min}, y_{d,\max})$ the minimum and maximum of each node of the model over the default time course simulation (with initial conditions provided in the SBML file and T=25000). On the other hand, the behavior space $Z \subset \mathbb{R}_+^2$ endpoint states $z = \big(y_i(T), y_j(T)\big)$ where (i, j) corresponds to the target phenotype nodes. We ensure that most trajectories have reached stable states at T = 2500 (as elaborated in the next section) such that Z can be viewed as the space of reachable endpoints, whose boundaries are not known.

F.2.3. Database creation

F.2.3.1. Biological networks database

All the ODE models we use in this work are downloaded from the BioModels database [314, 315] in SBML format. From all models referenced on the website, we only consider the ones that are curated, that have at least 3 nodes, and that are handled by the SBMLtoODEjax simulator (as SBMLtoODEjax does not handle models with discrete events, custom functions or other specific cases as detailed in [•7]. To ensure the inclusion of models suitable for our analyses, we applied specific filters to the collected models.

First, we simulated the default model rollout for each model to obtain the concentration profiles of the pathway components over a short time span (T=10 secs and $\Delta T = 0.1$). We discarded simulation results containing invalid values (NaN or negative concentrations) or those that took an excessive amount of time (>1sec). While it is acceptable that a rollout sometimes returns NaN values (when there are no solutions given ODE tolerance options for specific initial conditions), we consider the model invalid if this occurs for the default initial conditions provided in the SBML file.

For the remaining models, we conducted further simulations with an extended time span (T=2500) and 50 random initial conditions uniformly sampled within the model's intervention space I (as defined before). Once again we discarded models whose batch simulations took an excessive amount of time (> 15 secs). From the remaining models, we derived the resulting 50 trajectories for each node pair (i, j) and subjected them to additional filters to refine the database. We removed node pairs where either 1) [filter F1] a substantial proportion of trajectories ($\geq$ 20%) exhibited invalid concentrations (NaN or negative) or unsettled behaviors ($\exists t \geq 2400$ such that $|y(t) - y(T)| \geq 0.02 \times |y(T) - y(0)|$) or periodic patterns ($\exists f > 0$ such that $|S(f)| \geq 40$ where $S = DFT([y(\frac{T}{2}), \ldots, y(T)])$; or [filter F2] the reached space in Z was too small $\left((\max_{k=1\cdots50} y^k(T) - \min_{k=1\cdots5} y^k(T)) < 0.1\right)$ to discard cases where "diversity" could result from floating point rounding errors; or [filter F3] the number of attractors was less than four $\left(\{ y^k(T)\}_{k=1\ldots50} \text{ cover } \leq 4 \text{ bins over a } 20 \times 20 \text{ binning of } Z\right)$. Upon completion of the filtering process, our final database comprised 30 models, consisting of a total of 432 systems, as detailed in Supplementary Table F.1. These curated models and systems served as the foundation for our subsequent analyses and investigations into the navigation competencies of the molecular pathways.

F.2.3.2. Random networks database

Following the methodology proposed in [367], we aimed to create a database of synthetic networks with topologies similar to those of the biological networks, but with random regulatory rules instead of evolved ones. The objective was to compare the versatility and robustness competencies between biological and random networks, akin to the approach used for memory competencies in [367]. To achieve this, we initially generated 30 networks based on the transcriptional gene

circuit model [387], ensuring that they had the same distribution of network size (number of nodes) and connectivity (nodes in-degree) as the biological network database (using fitted gaussian distributions). The kinematic parameters W, b, τ of these networks were randomized ($W \sim [-30, 30]^{n \times n}$, $B \sim [-10, 10]^n$, $\tau \sim [1, 15]$) where model step is defined as $y(t+1) = \frac{\Delta T}{\tau} \times sigmoid(Wy + B) + \left(1 - \frac{\Delta T}{\tau}\right) \times y$ and in-degree connectivity is enforced by setting some weights of W to zero. However, during the creation process, we observed that none of the generated networks met the criterion for exhibiting a sufficient number of steady states (criterion F3). This limitation arose from the inherent constraints imposed by the gene circuit model's shape of ODE equations, limiting the diversity of possible dynamical behaviors. As our focus was on networks with a possible spectrum of steady states, akin to the biological network database, we decided not to pursue further analyses on these networks.

Instead, we selected the systems (models and pairs of nodes) that demonstrated the highest versatility (metric detailed below) from among all the generated systems that passed the filters F1 and F2. The selected networks' versatility is presented in Figure 8.8, but for future research, it would be interesting to explore broader and more complex classes of equations to assess their potential for achieving higher behavioral diversity.

F.2.4. Curiosity-driven exploration

This section provides additional information about the internal models and hyperparameters of the intrinsically-motivated goal exploration process. The overall IMGEP pipeline is illustrated in Figure 8.1-c. To sample a goal, the IMGEP uses a uniform sampling strategy within the bounding hyper-rectangle of currently reached goals (scaled by a factor 1.3). Hence sampling bounds adapt to the discoveries and do not need to be predefined via expert knowledge. The volume of the hyper-rectangle is larger compared to the cloud of currently-reached goals, which incentivizes targeting unexplored areas outside of the cloud and promotes diversity in the exploration process. Then, to generate an intervention for achieving the sampled goal, the IMGEP selects the nearest previously reached goal in Z, identifies its associated intervention, and performs a local random step from that point ($stepsize \sim \mathcal{N}(0, 0.1 \cdot [y_{0,\max} - y_{0,\min}])$ in the intervention space.

While our implementation choices for the IMGEP goal representation, goal generation, and goal-conditioned optimization are relatively straightforward, it is worth noting that alternative strategies could be considered for each of these components for more complex problems. The python library AutoDiscJax (`https://github.com/flowersteam/autodiscjax`) that accompanies this paper can be used to implement this and other IMGEP variants in JAX.

F.2.5. Robustness tests

We define 3 family of perturbations: 1) the noise perturbation $U_n(\sigma_n, p_n|y)$ which is parametrized by its standard-deviation (scaled proportionally

to the extent of the observed trajectory y prior perturbation) and period (secs); 2) the push perturbation $U_p\ (m_p, n_p|y)$ parametrized by its magnitude (proportional to the extent of y) and number of occurrences; 3) the wall perturbation $U_w\ (l_w, n_w|y)$ parametrized by its length (proportional to the extent of y) and number, and where walls are generated in locations of the space that the GRN would "naturally" visit without the induced perturbation. Details about the implementation of walls are provided in Supplementary Figure F.3.

To assess the robustness of the GRN systems in our database, we employ an evaluation procedure, as depicted in Figure 8.1-d. For each system (I, Z) in the database with its corresponding behavioral catalog H discovered using the curiosity-search algorithm, we perform the following steps. We first retrieve K representative trajectories out of the N discoveries, i.e. ones that cover well the reachable space. To do so, we randomly sample tuples of K discoveries (among N) 500 times, and select the one with the maximum diversity. One could test all trajectories with K=N but here we use $K = N/10$ mainly for compute reasons, as we run the experimental campaign on all 432 systems. Next, we subject each of these K trajectories $\{y_k, k = 1 \dots K\}$ to s=18 different perturbation distributions, each representing various levels of difficulty:

$$(\sigma_n, p_n) \in \{(0.001, 5), (0.005, 5), (0.1, 5), (0.005, 10), (0.005, 5), (0.005, 1)\},$$

$$(m_p, n_p) \in \{(0.05, 1), (0.1, 1), (0.15, 1), (1, 0.1), (2, 0.1), (3, 0.1)\},$$

$$(l_w, n_w) \in \{(0.05, 1), (0.1, 1), (0.15, 1), (1, 0.1), (2, 0.1), (3, 0.1)\}.$$

In each perturbation distribution, we sample $r = 3$ random perturbations, resulting in $P = s \times r$ perturbations. For each perturbation in the set $\{u_p, p = 1 \dots P\}$, we re-run the trajectory starting from the same initial state i but with the sampled perturbation applied (i, u_p), and observe the resulting outcome (o_p) and reached endpoint (z_p).

At the end of this process, the behavioral catalog is augmented with the perturbed trajectories $H = \{(i_k, o_k, z_k,\ \{(u_p, o_p, z_p),\ \ p = 1 \dots P\})\ ,\ \ k = 1 \dots K\}$.

F.2.6. Evaluation Metrics

F.2.6.1. Diversity measure

Diversity is measured by the area that explored observations cover in behavior space Z. Each single exploration results in a new point in this space, such that diversity measures how much area the algorithms explored in those spaces.

In general, existing approaches in the NS, QD and IMGEP literature use binning-based metrics [•1, •3, 72] or distance-based metric from ecology [73] to quantify the diversity of a set of explored instances. However, those metrics are sensitive to the binning strategy, or fail to discriminate between qualitatively significantly different explorations [71]. Another approach, called the threshold coverage, measures diversity as the volume of the union of the set of hyperballs of radius ϵ that have for centers the observed effects $\{z \in Z\}$. This diversity measure, while difficult to compute in high-dimensional spaces, avoids the pitfalls

of bin-based and distance-based metrics and is easily computable in 2-dimensional spaces [71].

Threshold coverage quantifies the area of the space that has been reached at a given precision ϵ (the threshold), and is what we used in Figure 8.4 to compare random search and curiosity-driven exploration strategies.

F.2.6.2. Sensitivity measure

In general, existing approaches in systems biology and evolutionary genetics measure sensitivity (opposite of robustness) in a relative manner with respect to 1) a functionality [369] or phenotypic trait [370] of interest, 2) specific perturbations (environmental or genetic changes), and 3) a measure of the degree of variation. Here, we adopt a similar metric where 1) the phenotypic trait of interest is defined as a goal state $z \in Z$ discovered by curiosity search, 2) the set of perturbation $\{u\_\{p\}\}$ is defined in previous section and conditioned on the GRN goal-reaching trajectory $i \rightarrow z$, and 3) variation is measured as the Euclidean distance in behavior space, normalized by the extent of the trajectory prior perturbation in Z.

This distance-based sensitivity measure proves straightforward as we explicitly use "spaces" to observe and analyze behaviors. The results of this sensitivity analysis are presented in Figure 8.5.

F.2.6.3. Versatility-Robustness measure

In this study, we introduce the terms "diversity" and "versatility" to characterize the competencies of the exploration agent (IMGEP) and the gene regulatory network agent (GRN), respectively. Diversity refers to the ability of the IMGEP agent to reveal a wide range of behaviors in the GRN, while versatility refers to the capability of the GRN agent to reach diverse goal states. The GRN versatility is unknown, and can only be approximated via proxy metric. Here, we consider that the diversity of the IMGEP (measured with the threshold coverage metric) is a good approximation of the versatility of a given GRN, as the IMGEP was shown to efficiently drive the GRN into diverse possible goal states. In Figure 8.8a, we employ this diversity metric to categorize the versatility of surveyed networks based on the class of organism they belong to. For the random networks, as they all have less or equal than 4 attractors, the versatility remains below $0.026 = 4 \times \frac{\pi\epsilon^2}{(1+2\epsilon)^2}$.

Figure 8.8b, we introduce the versatility-robustness metric, which conditions the diversity metric on a sensitivity threshold. Only goal states with sensitivity to perturbations below this threshold are considered when computing the reached area of the space. A high versatility-robustness score indicates that diverse goal states are achieved with a high level of precision.

F.2.7. Experiments on the RKIP-ERK signaling pathway

This section details the additional experiments conducted on the RKIP-ERK signaling pathway [375]. We refer to the accompanying notebook tutorial for reproducing these experiments: https://developmentalsystems.org/curious-exploration-of-grn-competencies/tuto1.html.

For Figure 8.5, clustering in behavior space was performed using the HDBSCAN algorithm [376] with hyperparameters set as min_cluster_size=10 and cluster_selection_epsilon=0.1. Points in the 10-dimensional intervention space are visualized by applying a TSNE 2-dimensional reduction. To visualize the clusters in behavior space (and corresponding clusters in intervention space), we fitted polygons on the cluster points using shapely library unary_union, dilatation, and erosion operations [506].

In Figure 8.7, we generated trajectory-based energy landscapes following the method proposed in [383]. Energy landscapes provide an intuitive way to understand how a system with multiple steady states behave, by picturing it as a ball rolling downhill towards low-energy valleys (steady states). Given a set of trajectories in behavior space Z, we constructed a probability distribution (P) of system states and converted it into a pseudopotential energy surface ($U = -\ln(P)$). This energy surface was smoothed using cubic spline interpolation and visualized using Plotly 3D surface plots. Figure 8.7a, 6b, and 6c differed by the input set of trajectories used for generating the landscape: a) employed the set of trajectories discovered by random search, b) used the set of trajectories discovered by curiosity search, and c) utilized the set of trajectories generated by robustness tests.

In Figure 8.9, the "healthy" and "disease" clusters were the same as in Figure 8.5 and visualized similarly. We displayed trajectories with the lowest sensitivity (averaged over all $P = 3 \times 18$ perturbations). The stimuli-based intervention shown in Figure 8.9b was found using a simple random search procedure. First, we defined an arbitrary target node and a stepwise node-activation function, clamping MKEPP values to desired values $x = \left[y^{(1)}_{MEKPP}, \cdots, y^{(10)}_{MEKPP}\right]$ every 10 seconds for 100 seconds. Then, we randomly sampled x within a range of values near the MKEPP current steady states (endpoints from the 6 "disease" trajectories, assuming that the drug intervention cannot drastically remodel those values). For each candidate x, we ran new trajectories starting from the disease states and applying the intervention x under a distribution of noise, push, and wall perturbations. Finally, we selected the intervention x that most successfully brought ERK-RKIP levels back to the target setpoint (centroid of the healthy region). The resulting intervention (shown in Figure 8.9b) succeeds to robustly reset all 6 disease state points despite perturbations, as shown in Figure 8.9c. We refer to the notebook for reproducing the experiments.

F.2.8. Experiments on synthetic gene networks

This section details the additional experiments conducted on the synthetic gene networks (Figure 8.10). We refer to the second accompanying

tutorial for the full codebase: https://developmentalsystems.org/curious-exploration-of-grn-competencies/tuto1.html.

In these experiments, we consider the target application of gene circuit engineering followed in [361], where parameters of a gene circuit model are optimized to produce target oscillator patterns. The gene circuit model employed in [361] is the same than the one used for the random networks database (Eq 1), with $\tau = 1$. Hence the *intervention space* is a $n^2 + 2n$ dimensional space defined as $I = [y_{t=0,\min}, y_{t=0,\max}] \oplus [W_{\min}, W_{\max}] \oplus [B_{\min}, B_{\max}]$, with $y_{0,\min} = 0$, $y_{0,\max} = 1$, $W_{\min} = -30$, $W_{\max} = 30$, $B_{\min} = -10$, $B_{\max} = 10$. Here we consider networks of n=3 nodes, with the first node being the target phenotype node. Thus, what we seek here is kinematic parameters (W, B) and initial concentrations y_0 that would produce a periodic pattern $y = [y_{n=0}(0), \cdots y_{n=0}(T)]$ with target amplitude A, frequency w and offset b. Here, the target (A, ω, b) are sample randomly with $A \sim U([0.1, 0.5])$, $b \sim U([A, 1 - A])$, $\omega \sim Beta(\alpha = 2, \beta = 8)$.

We then compare three alternative exploration strategies: 1) curiosity search, 2) random search and 3) gradient descent, i.e. pure optimization-driven search as proposed in [361], all given the same experimental budget $N = 5000$.

For curiosity search, the behavior space Z is defined as the image space of the discrete Fourier transform of the 1d-signal y, where distance in the space measures average difference in spectral amplitude. The IMGEP algorithm is then the same that the one previously used, as detailed in Figure 8.1-c, but operating within the novel problem spaces (I, Z).

For random search, interventions are sample uniformly $(i_1, \cdots, i_N) \sim U(I)$.

For gradient descent, we follow the procedure proposed in [361]. We define a loss function which, for a set of parameters $i = (W, B, y_0)$, measures the mean square error between the phenotype node activation levels y and the target oscillation represented as a cosine wave with the desired $(A, \omega, b) : L = \sum_t (y(t) - (A\cos(2\pi\omega \times t) + b))^2$. We then sample a random parameter $i \sim U(I)$ and use Adam optimizer with $l_r = 10^{-3}$, $b1 = 0.02$, $b3 = 0.001$, $\epsilon = 10^{-8}$ for $N = 5000$ optimization steps (same number of model rollouts allowed than for curiosity search and random search).

In addition, we use gradient descent for *local* refinement of the best discoveries made by the other exploration strategies (curiosity search and random search), this time with a limited budget of $N = 100$ optimization steps.

Visualizations in Figure 8.10 show: (a-b) the oscillators discovered by random search and curiosity search (gradient descent did not find any oscillator in this example) in the (A, ω, b) space, (c) the corresponding diversity (using this time a binning-based space coverage measure with 20^3bins as the space is 3-dimensional), (d) the evolution of the training loss L throughout the N=5000 trials for the three exploration strategies, (e-f-g) the corresponding best discoveries (for which L is minimal) for the three exploration strategies, and (h-i) the local training loss and resulting finetuning of the best discoveries with gradient descent.

F.3. Supplementary Material

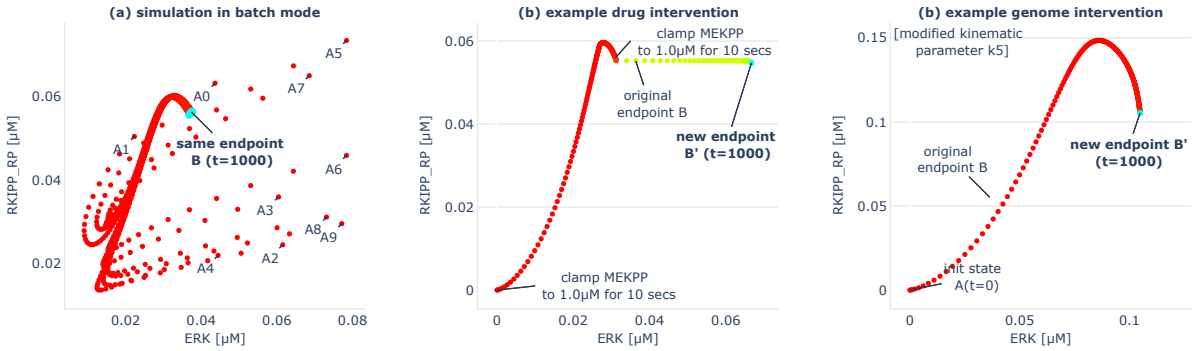


Figure F.2.: Examples of interventions that can be implemented within the accompanying AutodiscJax software. All those examples can be reproduced in the accompanying tutorial 1. (a) Numerical simulations with interventions can be performed in parallel by vectorizing simulations over different intervention parameters, simply using the jax vmap operator. This offers a convenient (and fast) way to test several interventions in the biological network, as shown here for testing the network under various initial conditions in batch mode. Examples of other possible "drug" or "genome" interventions that can be implemented in the accompanying software, as well as the possibility to perform interventions (or perturbations) in parallel using batched computations. In this example, despite the numerous interventions, the GRN trajectories still converge to the same endpoint B. (b) Example intervention where species amounts are clamped to specific values. In this example the node MEKPP is clamped to $2.5\mu M$ for 10 seconds at t=0 and then to $1\mu M$ for 10 additional seconds at t=400. We observe that, after the first clamping the GRN trajectory still follows a similar S-shape curve and arrives close to the original endpoint B but after the second clamping, ERK expression levels are shifted to a higher steady state B'. (c) Example intervention where the numerical value of one kinematic parameter of the model (k5) is changed from 0.0315 to 0.1. In this example we can see that changing the parameter k5 shifts the trajectory end point quite significantly, but qualitatively the trajectory seems to preserve a similar S-shape.

BioModel ID reference	number of nodes	organism class	systems (observed nodes pairs)
BIOMD0000000524	16	Homo sapiens	(PrER, mGFP), (DISC, tBid), (p18inactive, PrER), (p18inactive, tBid), (p18inactive, mGFP), (p18inactive, PrNES), (DISC, mCherry), (PrNES, PrER), (Bid, mCherry), (tBid, PrNES), (p18inactive, PrER_mGFP), (PrNES, mGFP), (mCherry, PrER_mGFP), (PrER_mGFP, PrER), (FADD, PrNES), (tBid, mGFP), (DISC, PrER), (FADD, mGFP), (mCherry, mGFP), (FADD, mCherry), (Bid, PrER), (FADD, tBid), (DISC, p18inactive), (Bid, PrNES), (p18inactive, Bid), (tBid, PrER_mGFP), (FADD, PrER), (tBid, mCherry), (PrNES, PrER_mGFP), (FADD, p18inactive), (FADD, Bid), (Bid, mGFP), (mCherry, PrER), (PrNES, mCherry), (DISC, Bid), (DISC, PrNES), (Bid, tBid), (DISC, PrER_-mGFP), (FADD, DISC), (DISC, mGFP), (p18inactive, mCherry), (PrER_mGFP, mGFP), (FADD, PrER_mGFP), (tBid, PrER)
BIOMD0000000050	14	n/a	(Gly, Fru), (FA, Mel), (FA, MG), (FA, Fru), (Mel, Fru), (Gly, Mel), (FA, Glu), (Gly, AA), (AA, Fru), (Glu, Mel), (MG, Fru), (AA, Glu), (FA, AA), (Glu, Fru), (Gly, MG), (Glu, MG), (AA, MG), (AA, Mel), (Mel, MG), (Gly, FA)
BIOMD0000000647	11	n/a	(RKIPP, MEKPP_ERK), (RKIPP, RKIPP_RP), (Raf1_RKIP, ERK), (ERK, RP), (ERK, MEKPP), (Raf1, MEKPP), (MEKPP, RKIPP_RP), (MEKPP, RP), (RKIP, RP), (ERKPP, RKIPP), (Raf1, RKIPP), (ERKPP, RP), (Raf1_RKIP_ERKPP, ERK), (Raf1_RKIP, MEKPP), (Raf1, ERKPP), (ERKPP, MEKPP_ERK), (ERK, MEKPP_ERK), (RKIP, RKIPP_RP), (Raf1, ERK), (ERKPP, Raf1_RKIP_ERKPP), (RKIP, ERKPP), (Raf1_RKIP_ERKPP, RKIPP_RP), (Raf1_RKIP_ERKPP, MEKPP), (Raf1_RKIP, ERKPP), (RP, RKIPP_RP), (MEKPP_ERK, RKIPP_RP), (ERK, RKIPP_RP), (ERKPP, ERK), (Raf1_RKIP_ERKPP, RP), (Raf1, Raf1_-RKIP_ERKPP), (Raf1, Raf1_RKIP), (MEKPP, MEKPP_ERK), (Raf1_RKIP, RKIPP), (Raf1_RKIP_ERKPP, RKIPP), (Raf1_RKIP_ERKPP, MEKPP_ERK), (MEKPP_ERK, RP), (RKIP, MEKPP), (Raf1, RKIP), (RKIPP, RP), (RKIP, MEKPP_ERK), (Raf1_RKIP, Raf1_-RKIP_ERKPP), (Raf1, RP), (Raf1_RKIP, RKIPP_RP), (Raf1, RKIPP_RP), (Raf1_RKIP, MEKPP_ERK), (ERKPP, MEKPP), (RKIP, RKIPP), (Raf1, MEKPP_ERK), (RKIP, ERK), (RKIPP, MEKPP), (ERKPP, RKIPP_RP), (RKIP, Raf1_RKIP), (ERK, RKIPP), (Raf1_RKIP, RP), (RKIP, Raf1_RKIP_ERKPP)
BIOMD0000000520	3	Rodents	(N0, N1), (N1, N2), (N0, N2)

BioModel ID reference	number of nodes	organism class	systems (observed nodes pairs)
BIOMD0000000523	16	Homo sapiens	(tBid, mCherry), (PrER, mGFP), (PrNES, mGFP), (DISC, PrNES), (DISC, tBid), (FADD, tBid), (p18inactive, tBid), (FADD, mCherry), (mCherry, mGFP), (p18inactive, mCherry), (tBid, PrNES), (tBid, mGFP), (FADD, DISC), (p18inactive, mGFP), (mCherry, PrER), (p18inactive, PrER), (FADD, p18inactive), (DISC, p18inactive), (DISC, PrER), (FADD, PrER), (DISC, mGFP), (PrNES, mCherry), (p18inactive, PrNES), (FADD, PrNES), (tBid, PrER), (FADD, mGFP), (DISC, mCherry), (PrNES, PrER)
BIOMD0000000454	8	Generic	(x3, y3), (y4, y2), (y1, y5), (x3, y2), (y4, y5), (x2, y4), (x1, y3), (x1, y5), (y2, y3), (y1, y3), (x1, y4), (x1, x3), (y4, y3), (y1, y2), (y1, y4), (y1, x3), (y5, y2), (y1, x1), (x2, x1), (x1, y2), (x2, x3), (x3, y4), (x3, y5), (y1, x2), (x2, y2), (x2, y5), (x2, y3), (y5, y3)
BIOMD0000000069	10	Homo sapiens	(srco, srca), (srci, srca), (srco, srcc), (srca, Cbp_P), (srcc, Cbp_P_CSK), (srci, Cbp_P), (srci, PTP), (Cbp_P, PTP_pY789), (srcc, PTP_pY789), (CSK_cytoplasm, PTP), (srci, PTP_pY789), (srca, PTP_pY789), (srcc, CSK_cytoplasm), (CSK_cytoplasm, Cbp_-P_CSK), (srca, CSK_cytoplasm), (srca, PTP), (srcc, Cbp_P), (srco, Cbp_P_CSK), (CSK_cytoplasm, PTP_pY789), (srcc, PTP), (srca, Cbp_P_CSK), (srci, CSK_cytoplasm), (Cbp_P, Cbp_P_CSK), (Cbp_P, PTP), (srci, srco), (srco, PTP), (CSK_cytoplasm, Cbp_P), (srca, srcc), (srci, Cbp_P_CSK), (Cbp_P_CSK, PTP_pY789), (PTP, PTP_pY789), (Cbp_-P_CSK, PTP), (srci, srcc), (srco, Cbp_P), (srco, PTP_pY789), (srco, CSK_cytoplasm)
BIOMD0000000455	9	Generic	(x2, y5), (y5, y2), (x2, y2), (y1, y3), (x3, y3), (y1, x2), (x2, x1), (x2, y4), (x3, y2), (x2, y3), (y1, y4), (y5, y3), (y1, y5), (x1, x3), (x1, y4), (y4, y5), (x1, y3), (y1, x3), (y4, y3), (x3, y4), (x3, y5), (x1, y5), (y1, x1), (y1, y2), (y2, y3), (y4, y2), (x2, x3), (x1, y2)
BIOMD0000000526	16	Homo sapiens	(FADD, PrER_mGFP), (PrNES, mCherry), (p18inactive, PrER_mGFP), (DISC, p18inactive), (FADD, p18inactive), (p18inactive, mCherry), (DISC, mGFP), (FADD, mGFP), (p18inactive, tBid), (p18inactive, mGFP), (FADD, DISC), (tBid, PrER), (DISC, tBid), (p18inactive, PrER), (p18inactive, PrNES), (PrNES, PrER_mGFP), (PrNES, PrER), (FADD, PrNES), (DISC, PrER), (DISC, mCherry), (FADD, tBid), (PrER_mGFP, PrER), (PrNES, mGFP), (FADD, PrER), (mCherry, mGFP), (DISC, PrNES), (tBid, PrNES), (mCherry, PrER), (PrER_mGFP, mGFP), (tBid, mCherry), (FADD, mCherry), (tBid, mGFP), (mCherry, PrER_mGFP), (DISC, PrER_mGFP), (PrER, mGFP), (tBid, PrER_mGFP)
BIOMD0000000284	9	Bacteria	(D, E), (A, D), (X, C), (B, E), (C, Z), (X, E), (A, Z), (X, Y), (C, D), (Y, Z), (E, F), (F, Z), (D, F), (B, F), (C, F), (X, Z), (A, Y), (X, D), (A, F), (B, Z), (X, F), (A, B), (A, C), (B, D), (Y, E), (B, C), (B, Y), (C, Y), (Y, F), (E, Z), (X, B), (D, Z), (D, Y), (A, E), (C, E), (X, A)
BIOMD0000000084	8	n/a	(Rin, x2), (x1, x3), (x2, x3), (x1, x2), (Rin, x3), (Rin, x1)
BIOMD0000000052	11	n/a	(Triose, lys_R), (Cn, lys_R), (Acetic_acid, lys_R), (lys_R, Melanoidin), (Triose, Melanoidin), (Cn, Melanoidin), (Formic_acid, lys_R), (C5, lys_R), (C5, Formic_-acid), (Triose, Acetic_acid), (C5, Acetic_acid), (Acetic_acid, Melanoidin), (C5, Cn), (C5, Melanoidin), (Cn, Acetic_acid), (C5, Triose), (Formic_acid, Triose), (Formic_acid, Cn), (Triose, Cn), (Formic_acid, Melanoidin), (Formic_acid, Acetic_acid)
BIOMD0000000271	6	Rodents	(EpoR, dEpoe), (Epo_EpoRi, dEpoe), (Epo_EpoR, dEpoe), (Epo_EpoRi, dEpoi), (EpoR, dEpoi), (Epo, dEpoe), (Epo, dEpoi), (Epo_EpoR, dEpoi), (dEpoi, dEpoe)
BIOMD0000000461	4	Bacteria	(lacz, x), (IPTG, sigb), (sigb, x), (sigb, lacz), (IPTG, x), (IPTG, lacz)
BIOMD0000000525	16	Homo sapiens	(p18inactive, PrNES), (p18inactive, PrER), (tBid, PrNES), (DISC, PrER), (mCherry, PrER), (PrNES, mCherry), (DISC, tBid), (FADD, mCherry), (p18inactive, tBid), (FADD, PrER), (FADD, p18inactive), (tBid, mGFP), (DISC, p18inactive), (mCherry, mGFP), (DISC, PrNES), (PrER, mGFP), (DISC, mGFP), (FADD, DISC), (tBid, PrER), (p18inactive, mCherry), (tBid, mCherry), (PrNES, mGFP), (PrNES, PrER), (FADD, PrNES), (DISC, mCherry), (p18inactive, mGFP), (FADD, tBid), (FADD, mGFP)
BIOMD0000000521	4	Homo sapiens	(P, Q)
BIOMD0000000010	8	Amphibians	(MKK_P, MAPK), (MKK, MAPK_PP), (MKK, MKK_PP), (MKK_P, MKK_PP), (MKK, MAPK), (MKK_PP, MAPK_P), (MAPK, MAPK_PP), (MKK_PP, MAPK), (MKK, MAPK_P), (MKK_PP, MAPK_PP), (MAPK_P, MAPK_PP), (MKK_P, MAPK_PP), (MKK, MKK_P), (MKK_P, MAPK_P)
BIOMD0000000029	4	Amphibians	(M, MpT), (M, MpY), (MpY, MpT)
BIOMD0000000197	5	n/a	(x1, x2), (x1, x4), (x1, x3), (x1, x5), (x5, x4), (x3, x5), (x5, x2)
BIOMD0000000272	6	Rodents	(SAv_EpoR, SAv_EpoRi), (EpoR, SAv_EpoR), (EpoR, SAv_EpoRi)
BIOMD0000000167	7	Generic	(Pstat_nuc, stat_nuc), (stat_nuc, species_test), (Pstat_nuc, species_test)
BIOMD0000000262	11	Rodents	(Akt, S6)
BIOMD0000000240	6	Bacteria	(DegU, mDegU), (Dim, mDegU), (DegUP, mDegU)
BIOMD0000000037	12	Slime Mold	(Xi, Ya), (Xi, Pi), (Ya, Pi)
BIOMD0000000263	11	Rodents	(Akt, pro_TrkA), (S6, pro_TrkA), (Akt, S6)
BIOMD0000000641	5	Homo sapiens	(CellCact, Effectoract)

BioModel ID reference	number of nodes	organism class	systems (observed nodes pairs)
BIOMD0000000413	5	Plants	(TIR1, auxinTIR1), (auxinTIR1, VENUS), (TIR1, VENUS)
BIOMD0000000624	7	Homo sapiens	(APAP, APAPconj_Glu)
BIOMD0000000945	5	Homo sapiens	(L_m, H_m)
BIOMD0000000459	4	Bacteria	(IPTG, sigb)

Table F.1.: List of biological networks from Biomodels used in this study. The resulting database includes 30 biological networks (one row per network) and a total of 432 systems, which is defined as a (GRN model, behavior space (Z)) tuple and where the pairs of observed nodes (used as behavior spaces) per network are given in the last column.

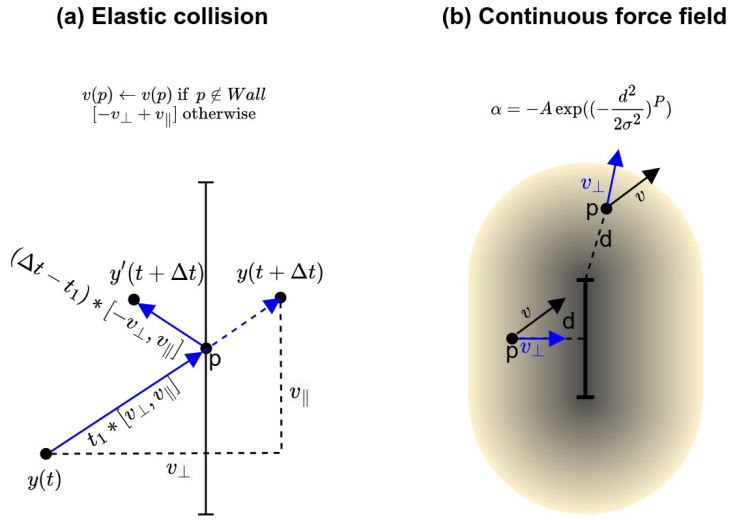


Figure F.3.: Wall implementation. Walls are implemented within the 2D space spanned by the 2 observed nodes. Within that space, we can interpret the node activation levels $y(0), \cdots, y(t)$ as the trajectory of a particle moving. In order to simulate the interaction with "walls" in that space, several implementations could be envisaged. Within the accompanying software AutoDiscJax we propose two possible variants: perfectly elastic collision (equivalent to a discontinuous force field) and some continuous force field variant. The second variant (continuous force field) is employed for the main results of this paper. (a) For the first variant, we consider a perfectly elastic collision without loss of kinetic energy. In that case, when the trajectory is touching the wall at position p with speed $v = v_\perp + v_\parallel$ we deviate the trajectory in such a way that is "bouncing" against the wall such that $v_\parallel$ is unchanged and $v_\perp \leftarrow -v_\perp$. To implement it, we simply check whether the segment $[y(t), y(t+\Delta t)]$ intersect the wall at each time step. It it does, we compute the intersection point p and time t_1, and set the activation level $y(t+\Delta t)$ to $p + (\Delta t - t1) \cdot \left[-v_\perp + v_\parallel\right]$. (b) For the second variant, we implement walls as energy barriers acting as a new force field in the environment, constraining the GRN traversal of the space. This time, instead of having a discontinuous effect on the perpendicular speed $v_\perp$ we define a wall force $f_\perp = \pm\alpha v_\perp$ (+ if $v_\perp$ is going toward wall, - otherwise) and use it to update the perpendicular component of the trajectory speed as $v_\perp \leftarrow v_\perp + f_\perp \cdot \Delta T$. Here $\alpha \in [0, -2]$ is calculated as a function of the distance between the point and the wall. As illustrated in the figure, this basically results in a stadium-shaped force field around the wall.

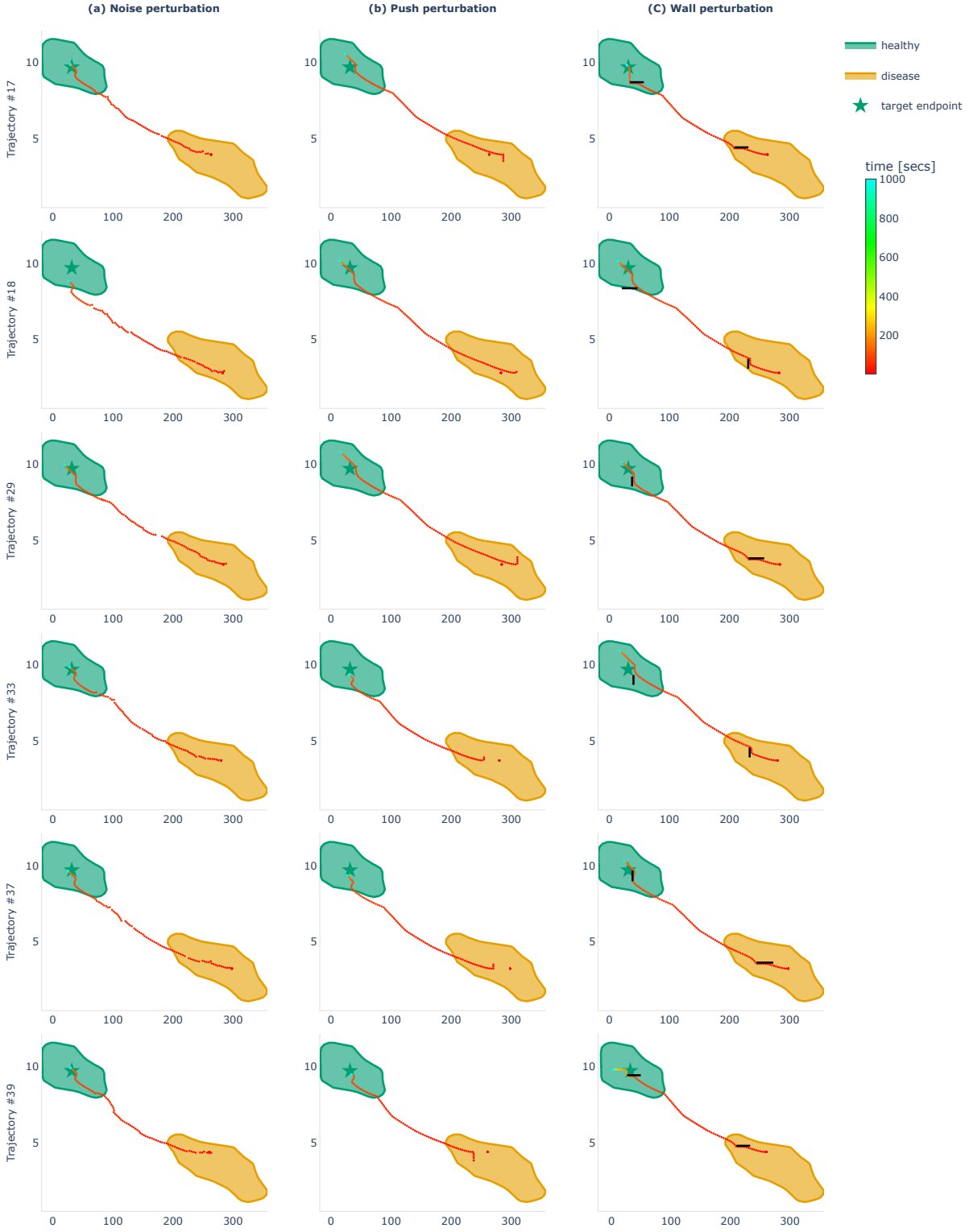


Figure F.4.: Additional results complementing Figure 8.9 of the main paper. This figure shows the resulting trajectories after applying the discovered stimuli-based intervention (shown in Figure 8.9-b) to the example RKIP-ERK signaling pathway [375] for the 6 "disease" trajectories originally discovered in the behavioral catalog (shown in Figure 8.9-a). (a) For each trajectory (one per row), we see that the intervention successfully re-sets the disease setpoint (startpoint of the trajectory shown in red in the orange region) to a healthy set-point (endpoint of the trajectory shown in cyan in the green region). (b-c) Similar results are achieved despite adding push perturbations (b) or wall perturbations (c) in addition to the stimuli-based intervention.

Bibliography

Here are the references in citation order.

[9] Sara Imari Walker and Paul C. W. Davies. "The Algorithmic Origins of Life". In: *Journal of The Royal Society Interface* 10.79 (2013), p. 20120869. DOI: 10.1098/rsif.2012.0869
Cited on page 2

[10] Silvana S. S. Cardoso et al. "Chemobrionics: From Self-Assembled Material Architectures to the Origin of Life". In: *Artificial Life* 26.3 (2020), pp. 315–326. DOI: 10.1162/artl_a_00323
Cited on page 2

[11] Jeremy P. Brockes and Anoop Kumar. "Comparative Aspects of Animal Regeneration". In: *Annual Review of Cell and Developmental Biology* 24.1 (2008), pp. 525–549. DOI: 10.1146/annurev.cellbio.24.110707.175336
Cited on page 2

[12] Richard C. Mohs and Nigel H. Greig. "Drug Discovery and Development: Role of Basic Biological Research". In: *Alzheimer's & Dementia: Translational Research & Clinical Interventions* 3.4 (2017), pp. 651–657. DOI: 10.1016/j.trci.2017.10.005
Cited on page 2

[13] G. Pezzulo and M. Levin. "Re-Membering the Body: Applications of Computational Neuroscience to the Top-down Control of Regeneration of Limbs and Other Complex Organs". In: *Integrative Biology: Quantitative Biosciences from Nano to Macro* 7.12 (2015), pp. 1487–1517. DOI: 10.1039/c5ib00221d
Cited on pages 2, 122, 126

[14] Giovanni Pezzulo and Michael Levin. "Top-down Models in Biology: Explanation and Control of Complex Living Systems above the Molecular Level". In: *Journal of The Royal Society Interface* 13.124 (2016), p. 20160555. DOI: 10.1098/rsif.2016.0555
Cited on pages 2, 122, 126

[15] Daniel P. Tabor et al. "Accelerating the Discovery of Materials for Clean Energy in the Era of Smart Automation". In: *Nature Reviews Materials* 3.5 (2018), pp. 5–20. DOI: 10.1038/s41578-018-0005-z
Cited on page 2

[16] Mo R. Ebrahimkhani and Michael Levin. "Synthetic Living Machines: A New Window on Life". In: *iScience* 24.5 (2021), p. 102505. DOI: 10.1016/j.isci.2021.102505
Cited on pages 2, 123

[17] Florian Häse, Loïc Roch, and Alán Aspuru-Guzik. "Next-Generation Experimentation with Self-Driving Laboratories". In: *Trends in Chemistry* 1 (2019). DOI: 10.1016/j.trechm.2019.02.007
Cited on page 2

[18] Milad Abolhasani and Eugenia Kumacheva. "The Rise of Self-Driving Labs in Chemical and Materials Sciences". In: *Nature Synthesis* 2.6 (2023), pp. 483–492. DOI: 10.1038/s44160-022-00231-0
Cited on page 2

[19] Hector G Martin et al. "Perspectives for Self-Driving Labs in Synthetic Biology". In: *Current Opinion in Biotechnology* 79 (2023), p. 102881. DOI: 10.1016/j.copbio.2022.102881
Cited on page 2

[20] Bertrand Guillotin and Fabien Guillemot. "Cell Patterning Technologies for Organotypic Tissue Fabrication". In: *Trends in Biotechnology* 29.4 (2011), pp. 183–190. DOI: 10.1016/j.tibtech.2010.12.008
Cited on pages 2, 149

[21] J von Neumann. “Theory of Self-Reproducing Automata”. In: *Mathematics of Computation* 21 (1966), p. 745
Cited on pages 3, 43

[22] Christopher G Langton. “Self-Reproduction in Cellular Automata”. In: *Physica D: Nonlinear Phenomena* 10.1-2 (1984), pp. 135–144
Cited on page 3

[23] Alan Mathison Turing. “The Chemical Basis of Morphogenesis”. In: *Bulletin of mathematical biology* 52.1-2 (1990), pp. 153–197
Cited on page 3

[24] G. Bard Ermentrout and Leah Edelstein-Keshet. “Cellular Automata Approaches to Biological Modeling”. In: *Journal of Theoretical Biology* 160.1 (1993), pp. 97–133. DOI: 10.1006/jtbi.1993.1007
Cited on page 3

[25] Philip Ball. *The Self-Made Tapestry: Pattern Formation in Nature*. Oxford University Press Oxford, 1999
Cited on page 3

[26] Shigeru Kondo and Takashi Miura. “Reaction-Diffusion Model as a Framework for Understanding Biological Pattern Formation”. In: *Science* 329.5999 (2010), pp. 1616–1620. DOI: 10.1126/science.1179047
Cited on page 3

[27] Chad M Glen, Melissa L Kemp, and Eberhard O Voit. “Agent-Based Modeling of Morphogenetic Systems: Advantages and Challenges”. In: *PLoS computational biology* 15.3 (2019)
Cited on page 3

[28] Stephen Wolfram. “Statistical Mechanics of Cellular Automata”. In: *Reviews of Modern Physics* 55.3 (1983), pp. 601–644. DOI: 10.1103/RevModPhys.55.601
Cited on page 3

[29] Gérard Y. Vichniac. “Simulating Physics with Cellular Automata”. In: *Physica D: Nonlinear Phenomena* 10.1 (1984), pp. 96–116. DOI: 10.1016/0167-2789(84)90253-7
Cited on page 3

[30] Lemont B. Kier, Paul G. Seybold, and Chao-Kun Cheng. *Modeling Chemical Systems Using Cellular Automata*. Springer Science & Business Media, 2005
Cited on page 3

[31] Christopher G Langton. “Studying Artificial Life with Cellular Automata”. In: *Physica D: Nonlinear Phenomena*. Proceedings of the Fifth Annual International Conference 22.1 (1986), pp. 120–149. DOI: 10.1016/0167-2789(86)90237-X
Cited on page 3

[32] Craig W. Reynolds. “Flocks, Herds and Schools: A Distributed Behavioral Model”. In: *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH ’87. New York, NY, USA: Association for Computing Machinery, 1987, pp. 25–34. DOI: 10.1145/37401.37406
Cited on page 3

[33] Mark A. Bedau. “Artificial Life: Organization, Adaptation and Complexity from the Bottom Up”. In: *Trends in Cognitive Sciences* 7.11 (2003), pp. 505–512. DOI: 10.1016/j.tics.2003.09.012
Cited on page 3

[34] Kenneth O. Stanley and Risto Miikkulainen. “A Taxonomy for Artificial Embryogeny”. In: *Artificial Life* 9.2 (2003), pp. 93–130. DOI: 10.1162/106454603322221487
Cited on page 3

[35] William Gilpin. “Cellular Automata as Convolutional Neural Networks”. In: *Physical Review E* 100.3 (2019), p. 032402
Cited on pages 3, 47, 225

[36] Deepak Pathak, Christopher Lu, Trevor Darrell, Phillip Isola, and Alexei A Efros. "Learning to Control Self-Assembling Morphologies: A Study of Generalization via Modularity". In: *Advances in Neural Information Processing Systems*. 2019, pp. 2292–2302
Cited on pages 3, 24, 30

[37] Alexander Mordvintsev, Ettore Randazzo, Eyvind Niklasson, Michael Levin, and Sam Greydanus. "Thread: Differentiable Self-Organizing Systems". In: *Distill* (2020). DOI: 10.23915/distill.00027. https://distill.pub/2020/selforg
Cited on pages 3, 23, 53, 106

[38] Bert Wang-Chak Chan. "Towards Large-Scale Simulations of Open-Ended Evolution in Continuous Cellular Automata". In: *arXiv preprint arXiv:2304.05639* (2023)
Cited on page 3

[39] Bert Wang-Chak Chan. "Lenia-Biology of Artificial Life". In: *Complex Systems* 28.3 (2019), pp. 251–286
Cited on pages 3, 11, 43–45, 48, 72, 83, 103, 104, 106, 109, 111, 208, 226, 237

[40] Bert Wang-Chak Chan. "Lenia and Expanded Universe". In: *The 2020 Conference on Artificial Life*. Online: MIT Press, 2020, pp. 221–229. DOI: 10.1162/isal_a_00297
Cited on pages 3, 11, 43–46, 48, 63, 65, 87, 103–106, 109, 111, 202, 226, 237

[41] Christopher Lipinski and Andrew Hopkins. "Navigating Chemical Space for Biology and Medicine". In: *Nature* 432.7019 (2004), pp. 855–861
Cited on page 3

[42] Jack W Scannell, Alex Blanckley, Helen Boldon, and Brian Warrington. "Diagnosing the Decline in Pharmaceutical R&D Efficiency". In: *Nature reviews Drug discovery* 11.3 (2012), pp. 191–200
Cited on page 3

[43] Timothy S. Gardner. "Synthetic Biology: From Hype to Impact". In: *Trends in Biotechnology* 31.3 (2013), pp. 123–125. DOI: 10.1016/j.tibtech.2013.01.018
Cited on page 3

[44] Pablo Carbonell, Tijana Radivojevic, and Héctor García Martín. "Opportunities at the Intersection of Synthetic Biology, Machine Learning, and Automation". In: *ACS Synthetic Biology* 8.7 (2019), pp. 1474–1477. DOI: 10.1021/acssynbio.8b00540
Cited on page 3

[45] Michael Levin. "Technological Approach to Mind Everywhere: An Experimentally-Grounded Framework for Understanding Diverse Bodies and Minds". In: *Frontiers in Systems Neuroscience* 16 (2022)
Cited on pages 3, 28, 49, 115, 126, 129, 140, 151, 181

[46] Chris G. Langton. "Computation at the Edge of Chaos: Phase Transitions and Emergent Computation". In: *Physica D: Nonlinear Phenomena* 42.1 (1990), pp. 12–37. DOI: 10.1016/0167-2789(90)90064-V
Cited on pages 3, 20

[47] Petra Schneider and Gisbert Schneider. "De Novo Design at the Edge of Chaos". In: *Journal of Medicinal Chemistry* 59.9 (2016), pp. 4077–4086. DOI: 10.1021/acs.jmedchem.5b01849
Cited on pages 3, 20

[48] A Corma, JM Serra, P Serna, S Valero, E Argente, and V Botti. "Optimisation of Olefin Epoxidation Catalysts with the Application of High-Throughput and Genetic Algorithms Assisted by Artificial Neural Networks (Softcomputing Techniques)". In: *Journal of Catalysis* 229.2 (2005), pp. 513–524
Cited on pages 4, 24, 30

[49] Pavel Nikolaev, Daylond Hooper, Frederick Webber, Rahul Rao, Kevin Decker, Michael Krein, Jason Poleski, Rick Barto, and Benji Maruyama. "Autonomy in Materials Research: A Case Study in Carbon Nanotube Growth". In: *npj Computational Materials* 2.1 (2016), pp. 1–6
Cited on pages 4, 24, 25, 30

[50] Dezhen Xue, Prasanna V Balachandran, John Hogden, James Theiler, Deqing Xue, and Turab Lookman. "Accelerated Search for Materials with Targeted Properties by Adaptive Design". In: *Nature communications* 7.1 (2016), pp. 1–9
Cited on pages 4, 24, 30

[51] Vasilios Duros, Jonathan Grizou, Weimin Xuan, Zied Hosni, De-Liang Long, Haralampos N Miras, and Leroy Cronin. "Human versus Robots in the Discovery and Crystallization of Gigantic Polyoxometalates". In: *Angewandte Chemie* 129.36 (2017), pp. 10955–10960
Cited on pages 4, 22, 30

[52] Ross D King, Kenneth E Whelan, Ffion M Jones, Philip GK Reiser, Christopher H Bryant, Stephen H Muggleton, Douglas B Kell, and Stephen G Oliver. "Functional Genomic Hypothesis Generation and Experimentation by a Robot Scientist". In: *Nature* 427.6971 (2004), pp. 247–252
Cited on pages 4, 11, 13, 21, 30

[53] Tijana Radivojević, Zak Costello, Kenneth Workman, and Hector Garcia Martin. "A Machine Learning Automated Recommendation Tool for Synthetic Biology". In: *Nature communications* 11.1 (2020), pp. 1–14
Cited on pages 4, 30

[54] Hiroaki Kitano. "Nobel Turing Challenge: Creating the Engine for Scientific Discovery". In: *npj Systems Biology and Applications* 7.1 (2021), p. 29. DOI: 10.1038/s41540-021-00189-3
Cited on pages 5, 16, 18, 34, 35

[55] Cédric Colas, Tristan Karch, Olivier Sigaud, and Pierre-Yves Oudeyer. "Autotelic Agents with Intrinsically Motivated Goal-Conditioned Reinforcement Learning: A Short Survey". In: *Journal of Artificial Intelligence Research* 74 (2022), pp. 1159–1199. DOI: 10.1613/jair.1.13554
Cited on pages 5, 28, 31, 33, 40, 41

[56] Vieri Giuliano Santucci, Pierre-Yves Oudeyer, Andrew Barto, and Gianluca Baldassarre. "Editorial: Intrinsically Motivated Open-Ended Learning in Autonomous Robots". In: *Frontiers in Neurorobotics* 13 (2020)
Cited on pages 5, 32

[57] Adrien Baranes and Pierre-Yves Oudeyer. "Active Learning of Inverse Models with Intrinsically Motivated Goal Exploration in Robots". In: *Robotics and Autonomous Systems* 61.1 (2013), pp. 49–73. DOI: 10.1016/j.robot.2012.05.008
Cited on pages 5, 25–27, 32, 33, 40, 41, 58, 71, 103, 128, 132, 145

[58] Sébastien Forestier, Rémy Portelas, Yoan Mollard, and Pierre-Yves Oudeyer. "Intrinsically Motivated Goal Exploration Processes with Automatic Curriculum Learning". In: *arXiv preprint arXiv:1708.02190* (2017)
Cited on pages 5, 26, 27, 32, 40, 41, 71, 103

[59] Cédric Colas, Laetitia Teodorescu, Pierre-Yves Oudeyer, Xingdi Yuan, and Marc-Alexandre Côté. *Augmenting Autotelic Agents with Large Language Models*. 2023. DOI: 10.48550/arXiv.2305.12487
Cited on pages 5, 33, 180

[60] Ross D King, Jem Rowland, Stephen G Oliver, Michael Young, Wayne Aubrey, Emma Byrne, Maria Liakata, Magdalena Markham, Pinar Pir, Larisa N Soldatova, et al. "The Automation of Science". In: *Science (New York, N.Y.)* 324.5923 (2009), pp. 85–89
Cited on pages 11, 13, 21

[61] Sam Kriegman, Douglas Blackiston, Michael Levin, and Josh Bongard. "A Scalable Pipeline for Designing Reconfigurable Organisms". In: *Proceedings of the National Academy of Sciences* 117.4 (2020), pp. 1853–1859. DOI: 10.1073/pnas.1910837117
Cited on pages 11, 13, 22, 30, 123, 183

[62] Douglas Blackiston, Emma Lederer, Sam Kriegman, Simon Garnier, Joshua Bongard, and Michael Levin. "A Cellular Platform for the Development of Synthetic Living Machines". In: *Science Robotics* 6.52 (2021), eabf1571. DOI: 10.1126/scirobotics.abf1571
Cited on pages 11, 13, 23, 183

[63] Sam Kriegman, Douglas Blackiston, Michael Levin, and Josh Bongard. "Kinematic Self-Replication in Reconfigurable Organisms". In: *Proceedings of the National Academy of Sciences* 118.49 (2021), e2112672118. DOI: 10.1073/pnas.2112672118
Cited on pages 11, 13, 23, 183

[64] Jonathan Grizou, Laurie J. Points, Abhishek Sharma, and Leroy Cronin. "A Curious Formulation Robot Enables the Discovery of a Novel Protocell Behavior". In: *Science Advances* 6.5 (2020), eaay4237. DOI: 10.1126/sciadv.aay4237
Cited on pages 11, 13, 25, 27, 28, 30, 50, 52, 53, 63, 71, 103, 128, 132, 184

[65] Francesco Ruscelli, Arturo Laurenzi, Nikos G. Tsagarakis, and Enrico Mingo Hoffman. "Horizon: A Trajectory Optimization Framework for Robotic Systems". In: *Frontiers in Robotics and AI* 9 (2022)
Cited on page 15

[66] Nirosha J. Murugan, Daniel H. Kaltman, Paul H. Jin, Melanie Chien, Ramses Martinez, Cuong Q. Nguyen, Anna Kane, Richard Novak, Donald E. Ingber, and Michael Levin. "Mechanosensation Mediates Long-Range Spatial Decision-Making in an Aneural Organism". In: *Advanced Materials (Deerfield Beach, Fla.)* 33.34 (2021), e2008161. DOI: 10.1002/adma.202008161
Cited on pages 15, 124

[67] Tom Leinster. *Entropy and Diversity: The Axiomatic Approach*. 2022. DOI: 10.48550/arXiv.2012.02113. Comment: Book, viii + 442 pages. Version 3: small number of minor corrections
Cited on page 16

[68] Pierre Delarboulas, Marc Schoenauer, and Michèle Sebag. *Open-Ended Evolutionary Robotics: An Information Theoretic Approach*. 2010
Cited on page 16

[69] Elad Hazan, Sham Kakade, Karan Singh, and Abby Van Soest. "Provably Efficient Maximum Entropy Exploration". In: *Proceedings of the 36th International Conference on Machine Learning*. PMLR, 2019, pp. 2681–2691
Cited on page 16

[70] Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. "Unifying Count-Based Exploration and Intrinsic Motivation". In: *Advances in Neural Information Processing Systems*. 2016, pp. 1471–1479
Cited on pages 16, 27

[71] Fabien Benureau. *Self Exploration of Sensorimotor Spaces in Robots*. Doctoral Dissertation, Université de Bordeaux
Cited on pages 16, 17, 20, 134, 249, 250

[72] Justin K Pugh, Lisa B Soros, Paul A Szerlip, and Kenneth O Stanley. "Confronting the Challenge of Quality Diversity". In: *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation*. 2015, pp. 967–974
Cited on pages 16, 17, 71, 249

[73] Samuel M Scheiner. "A Compilation of and Typology for Abundance-, Phylogenetic-and Functional-Based Diversity Metrics". In: *bioRxiv : the preprint server for biology* (2019), p. 530782
Cited on pages 17, 249

[74] Alexandre Péré, Sébastien Forestier, Olivier Sigaud, and Pierre-Yves Oudeyer. "Unsupervised Learning of Goal Spaces for Intrinsically Motivated Goal Exploration". In: *ICLR2018 - 6th International Conference on Learning Representations*. Vancouver, Canada, 2018
Cited on pages 17, 33, 40, 58, 62

[75] Fabien C. Y. Benureau and Pierre-Yves Oudeyer. "Behavioral Diversity Generation in Autonomous Exploration through Reuse of Past Experience". In: *Frontiers in Robotics and AI* 3 (2016)
Cited on page 17

[76] Lisa B Soros, Joel Lehman, and Kenneth O. Stanley. *Open-Endedness: The Last Grand Challenge You've Never Heard Of*. https://www.oreilly.com/radar/open-endedness-the-last-grand-challenge-youve-never-heard-of/. 2017
Cited on pages 18, 48, 75, 122

[77] Kenneth O. Stanley. "Why Open-Endedness Matters". In: *Artificial Life* 25.3 (2019), pp. 232–235. DOI: 10.1162/artl_a_00294
Cited on pages 18, 116

[78] John Jumper et al. "Highly Accurate Protein Structure Prediction with AlphaFold". In: *Nature* 596.7873 (2021), pp. 583–589. DOI: 10.1038/s41586-021-03819-2
Cited on page 18

[79] Jenny Zhang, Joel Lehman, Kenneth Stanley, and Jeff Clune. *OMNI: Open-endedness via Models of Human Notions of Interestingness*. 2023. DOI: 10.48550/arXiv.2306.01711. Comment: 33 pages, 22 figures
Cited on pages 19, 33

[80] Stephen Wolfram. "Universality and Complexity in Cellular Automata". In: *Physica D: Nonlinear Phenomena* (1984), p. 37
Cited on page 20

[81] Charles M Macal and Michael J North. "Agent-Based Modeling and Simulation". In: *Proceedings of the 2009 Winter Simulation Conference (WSC)*. IEEE, 2009, pp. 86–98
Cited on page 20

[82] Ettore Randazzo, Eyvind Niklasson, and Alexander Mordvintsev. "MPLP: Learning a Message Passing Learning Protocol". In: *arXiv:2007.00970 [cs, stat]* (2020). Comment: Code at https://github.com/google-research/self-organising-systems/tree/master/mplp; code base link fixed
Cited on page 20

[83] James Bergstra and Yoshua Bengio. "Random Search for Hyper-Parameter Optimization". In: *Journal of Machine Learning Research* 13.10 (2012), pp. 281–305
Cited on page 20

[84] Selim M Senkan. "High-Throughput Screening of Solid-State Catalyst Libraries". In: *Nature* 394.6691 (1998), pp. 350–353
Cited on page 20

[85] Stefano Curtarolo, Gus LW Hart, Marco Buongiorno Nardelli, Natalio Mingo, Stefano Sanvito, and Ohad Levy. "The High-Throughput Highway to Computational Materials Design". In: *Nature materials* 12.3 (2013), pp. 191–201
Cited on page 20

[86] Jürgen Bajorath. "Integration of Virtual and High-Throughput Screening". In: *Nature Reviews Drug Discovery* 1.11 (2002), pp. 882–894
Cited on page 20

[87] Ricardo Macarron, Martyn N Banks, Dejan Bojanic, David J Burns, Dragan A Cirovic, Tina Garyantes, Darren VS Green, Robert P Hertzberg, William P Janzen, Jeff W Paslay, et al. "Impact of High-Throughput Screening in Biomedical Research". In: *Nature reviews Drug discovery* 10.3 (2011), pp. 188–195
Cited on page 20

[88] Allen P. Liu, Ovijit Chaudhuri, and Sapun H. Parekh. "New Advances in Probing Cell–Extracellular Matrix Interactions". In: *Integrative Biology* 9.5 (2017), pp. 383–405
Cited on page 20

[89] Ovijit Chaudhuri, Justin Cooper-White, Paul A. Janmey, David J. Mooney, and Vivek B. Shenoy. "Effects of Extracellular Matrix Viscoelasticity on Cellular Behaviour". In: *Nature* 584.7822 (2020), pp. 535–546. DOI: 10.1038/s41586-020-2612-2
Cited on page 20

[90] Lilian Weng. "Learning with Not Enough Data Part 2: Active Learning". In: *lilianweng.github.io* (2022)
Cited on page 22

[91] Burr Settles. "Active Learning Literature Survey". In: *Computer Sciences Technical Report 1648, University of Wisconsin-Madison* ()
Cited on page 22

[92] Yuriy Sverchkov and Mark Craven. "A Review of Active Learning Approaches to Experimental Design for Uncovering Biological Networks". In: *PLOS Computational Biology* 13.6 (2017). Ed. by Haiyan Huang, e1005466. DOI: 10.1371/journal.pcbi.1005466
Cited on page 22

[93] Turab Lookman, Prasanna V. Balachandran, Dezhen Xue, and Ruihao Yuan. "Active Learning in Materials Science with Emphasis on Adaptive Sampling Using Uncertainties for Targeted Design". In: *npj Computational Materials* 5.1 (2019), pp. 1–17. DOI: 10.1038/s41524-019-0153-8
Cited on page 22

[94] Jie Yu, Xutong Li, and Mingyue Zheng. "Current Status of Active Learning for Drug Discovery". In: *Artificial Intelligence in the Life Sciences* 1 (2021), p. 100023. DOI: 10.1016/j.ailsci.2021.100023
Cited on page 22

[95] David Lowell, Zachary C. Lipton, and Byron C. Wallace. *Practical Obstacles to Deploying Active Learning*. 2019. DOI: 10.48550/arXiv.1807.04801
Cited on page 22

[96] Josh Attenberg and Foster Provost. "Inactive Learning? Difficulties Employing Active Learning in Practice". In: *ACM SIGKDD Explorations Newsletter* 12.2 (2011), pp. 36–41. DOI: 10.1145/1964897.1964906
Cited on page 22

[97] Melanie Mitchell, James P Crutchfield, Rajarshi Das, et al. "Evolving Cellular Automata with Genetic Algorithms: A Review of Recent Work". In: *Proceedings of the First International Conference on Evolutionary Computation and Its Applications (EvCA'96)*. Vol. 8. Moscow, 1996
Cited on pages 23, 30

[98] Gianluca Baldassarre, Vito Trianni, Michael Bonani, Francesco Mondada, Marco Dorigo, and Stefano Nolfi. "Self-Organized Coordinated Motion in Groups of Physically Connected Robots". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 37.1 (2007), pp. 224–239
Cited on pages 23, 30

[99] Vito Trianni and Stefano Nolfi. "Self-Organizing Sync in a Robotic Swarm: A Dynamical System View". In: *IEEE Transactions on Evolutionary Computation* 13.4 (2009), pp. 722–741
Cited on pages 23, 30

[100] Miguel Duarte, Vasco Costa, Jorge Gomes, Tiago Rodrigues, Fernando Silva, Sancho Moura Oliveira, and Anders Lyhne Christensen. "Evolution of Collective Behaviors for a Real Swarm of Aquatic Surface Robots". In: *PloS one* 11.3 (2016), e0151834
Cited on pages 23, 30

[101] Nathanael Aubert-Kato, Charles Fosseprez, Guillaume Gines, Ibuki Kawamata, Huy Dinh, Leo Cazenille, Andre Estevez-Tores, Masami Hagiya, Yannick Rondelez, and Nicolas Bredeche. "Evolutionary Optimization of Self-Assembly in a Swarm of Bio-Micro-Robots". In: *Proceedings of the Genetic and Evolutionary Computation Conference*. 2017, pp. 59–66
Cited on pages 23, 26, 30

[102] Jason E Kreutz, Anton Shukhaev, Wenbin Du, Sasha Druskin, Olafs Daugulis, and Rustem F Ismagilov. "Evolution of Catalysts Directed by Genetic Algorithms in a Plug-Based Microfluidic Device Tested with Oxidation of Methane by Oxygen". In: *Journal of the American Chemical Society* 132.9 (2010), pp. 3128–3132
Cited on pages 23, 30

[103] Daniel Salley, Graham Keenan, Jonathan Grizou, Abhishek Sharma, Sergio Martín, and Leroy Cronin. "A Nanomaterials Discovery Robot for the Darwinian Evolution of Shape Programmable Gold Nanoparticles". In: *Nature Communications* 11.1 (2020), p. 2771. DOI: 10.1038/s41467-020-16501-4
Cited on pages 23, 30

[104] Alexander Mordvintsev, Ettore Randazzo, Eyvind Niklasson, and Michael Levin. "Growing Neural Cellular Automata". In: *Distill* (2020). DOI: 10.23915/distill.00023. https://distill.pub/2020/growing-ca
https://distill.pub/2020/growing-ca
Cited on pages 23, 30, 46, 47, 106, 113, 226

[105] Shyam Sudhakaran, Djordje Grbic, Siyan Li, Adam Katona, Elias Najarro, Claire Glanois, and Sebastian Risi. "Growing 3D Artefacts and Functional Machines with Neural Cellular Automata". In: *ALIFE 2021: The 2021 Conference on Artificial Life*. MIT Press, 2021. DOI: 10.1162/isal_a_00451
Cited on page 23

[106] Eyvind Niklasson, Alexander Mordvintsev, Ettore Randazzo, and Michael Levin. "Self-Organising Textures". In: *Distill* 6.2 (2021), e00027.003. DOI: 10.23915/distill.00027.003
Cited on pages 23, 106

[107] Ettore Randazzo, Alexander Mordvintsev, Eyvind Niklasson, Michael Levin, and Sam Greydanus. "Self-Classifying MNIST Digits". In: *Distill* 5.8 (2020), e00027.002. DOI: 10.23915/distill.00027.002
Cited on pages 23, 106

[108] Mark Sandler, Andrey Zhmoginov, Liangcheng Luo, Alexander Mordvintsev, Ettore Randazzo, and Blaise Agúera y Arcas. *Image Segmentation via Cellular Automata*. 2020. DOI: 10.48550/arXiv.2008.04965
Cited on page 23

[109] Samuel Schoenholz and Ekin Dogus Cubuk. "JAX MD: A Framework for Differentiable Physics". In: *Advances in Neural Information Processing Systems*. Vol. 33. Curran Associates, Inc., 2020, pp. 11428–11441
Cited on pages 23, 24, 172

[110] Dmitrii Kochkov, Jamie A. Smith, Ayya Alieva, Qing Wang, Michael P. Brenner, and Stephan Hoyer. "Machine Learning–Accelerated Computational Fluid Dynamics". In: *Proceedings of the National Academy of Sciences* 118.21 (2021), e2101784118. DOI: 10.1073/pnas.2101784118
Cited on page 23

[111] Neythen J. Treloar, Alex J. H. Fedorec, Brian Ingalls, and Chris P. Barnes. "Deep Reinforcement Learning for the Control of Microbial Co-Cultures in Bioreactors". In: *PLOS Computational Biology* 16.4 (2020), e1007783. DOI: 10.1371/journal.pcbi.1007783
Cited on pages 23, 30

[112] Jonas Degrave et al. "Magnetic Control of Tokamak Plasmas through Deep Reinforcement Learning". In: *Nature* 602.7897 (2022), pp. 414–419. DOI: 10.1038/s41586-021-04301-9
Cited on pages 23, 30

[113] Zhenpeng Zhou, Xiaocheng Li, and Richard N. Zare. "Optimizing Chemical Reactions with Deep Reinforcement Learning". In: *ACS Central Science* 3.12 (2017), pp. 1337–1344. DOI: 10.1021/acscentsci.7b00492
Cited on pages 23, 30

[114] Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P. Adams, and Nando de Freitas. "Taking the Human Out of the Loop: A Review of Bayesian Optimization". In: *Proceedings of the IEEE* 104.1 (2016), pp. 148–175. DOI: 10.1109/JPROC.2015.2494218
Cited on page 24

[115] Florian Hase, Loïc M Roch, Christoph Kreisbeck, and Alán Aspuru-Guzik. "Phoenics: A Bayesian Optimizer for Chemistry". In: *ACS central science* 4.9 (2018), pp. 1134–1145
Cited on pages 24, 30

[116] Melodie Christensen et al. “Data-Science Driven Autonomous Process Optimization”. In: *Communications Chemistry* 4.1 (2021), pp. 1–12. DOI: 10.1038/s42004-021-00550-x
Cited on pages 24, 30

[117] Benjamin J Shields, Jason Stevens, Jun Li, Marvin Parasram, Farhan Damani, Jesus I Martinez Alvarado, Jacob M Janey, Ryan P Adams, and Abigail G Doyle. “Bayesian Reaction Optimization as a Tool for Chemical Synthesis”. In: *Nature* 590.7844 (2021), pp. 89–96
Cited on pages 24, 30

[118] Paul B Wigley, Patrick J Everitt, Anton van den Hengel, John W Bastian, Mahasen A Sooriyabandara, Gordon D McDonald, Kyle S Hardman, Ciaron D Quinlivan, P Manju, Carlos CN Kuhn, et al. “Fast Machine-Learning Online Optimization of Ultra-Cold-Atom Experiments”. In: *Scientific reports* 6.1 (2016), pp. 1–6
Cited on pages 24, 30

[119] Christof Angermueller, David Dohan, David Belanger, Ramya Deshpande, Kevin Murphy, and Lucy Colwell. “Model-Based Reinforcement Learning for Biological Sequence Design”. In: *International Conference on Learning Representations*. 2019
Cited on page 24

[120] Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. “Curiosity-Driven Exploration by Self-Supervised Prediction”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 2017, pp. 16–17
Cited on pages 24, 27, 79

[121] Pierre-Yves Oudeyer and Frederic Kaplan. “What Is Intrinsic Motivation? A Typology of Computational Approaches”. In: *Frontiers in Neurorobotics* 1 (2007)
Cited on pages 24, 27

[122] Luca A. Thiede, Mario Krenn, AkshatKumar Nigam, and Alán Aspuru-Guzik. “Curiosity in Exploring Chemical Spaces: Intrinsic Rewards for Molecular Reinforcement Learning”. In: *Machine Learning: Science and Technology* 3.3 (2022), p. 035008. DOI: 10.1088/2632-2153/ac7ddc
Cited on pages 24, 25, 27, 30

[123] Mario Krenn, Florian Häse, AkshatKumar Nigam, Pascal Friederich, and Alan Aspuru-Guzik. “Self-Referencing Embedded Strings (SELFIES): A 100% Robust Molecular String Representation”. In: *Machine Learning: Science and Technology* 1.4 (2020), p. 045024. DOI: 10.1088/2632-2153/aba947
Cited on page 24

[124] Germán Kruszewski and Tomas Mikolov. “Combinatory Chemistry: Towards a Simple Model of Emergent Evolution”. In: *ALIFE 2020: The 2020 Conference on Artificial Life*. MIT Press, 2020, pp. 411–419. DOI: 10.1162/isal_a_00258
Cited on page 24

[125] P. Dittrich, J. Ziegler, and W. Banzhaf. “Artificial Chemistries–a Review”. In: *Artificial Life* 7.3 (2001), pp. 225–275. DOI: 10.1162/106454601753238636
Cited on page 24

[126] J. Lehman and Kenneth O. Stanley. “Exploiting Open-Endedness to Solve Problems Through the Search for Novelty”. In: *IEEE Symposium on Artificial Life*. 2008. [TLDR] Decoupling the idea of open-ended search from only artificial life worlds, the raw search for novelty can be applied to real world problems and significantly outperforms objective-based search in the deceptive maze navigation task.
Cited on pages 26, 71, 132

[127] Joel Lehman and Kenneth O. Stanley. “Abandoning Objectives: Evolution Through the Search for Novelty Alone”. In: *Evolutionary Computation* 19.2 (2011), pp. 189–223. DOI: 10.1162/EVCO_a_00025
Cited on pages 26, 71, 116, 132

[128] Jorge Gomes, Paulo Urbano, and Anders Lyhne Christensen. “Evolution of Swarm Robotics Systems with Novelty Search”. In: *Swarm Intelligence* 7.2 (2013), pp. 115–144
Cited on pages 26, 27, 30

[129] Leo Cazenille, Nicolas Bredeche, and Nathanael Aubert-Kato. "Exploring Self-Assembling Behaviors in a Swarm of Bio-micro-robots Using Surrogate-Assisted MAP-Elites". In: *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*. 2019, pp. 238–246. DOI: 10.1109/SSCI44817.2019.9003047
Cited on pages 26, 27, 30, 71

[130] Jean-Baptiste Mouret and Jeff Clune. *Illuminating Search Spaces by Mapping Elites*. 2015. DOI: 10.48550/arXiv.1504.04909. Comment: Early draft
Cited on page 26

[131] Antoine Cully, Jeff Clune, Danesh Tarapore, and Jean-Baptiste Mouret. "Robots That Can Adapt like Animals". In: *Nature* 521.7553 (2015), pp. 503–507. DOI: 10.1038/nature14422
Cited on pages 26, 71, 132

[132] Justin K. Pugh, Lisa B. Soros, and Kenneth O. Stanley. "Quality Diversity: A New Frontier for Evolutionary Computation". In: *Frontiers in Robotics and AI* 3 (2016)
Cited on pages 26, 71, 132

[133] Alexandre Chenu, Nicolas Perrin-Gilbert, Stéphane Doncieux, and Olivier Sigaud. "Selection-Expansion: A Unifying Framework for Motion-Planning and Diversity Search Algorithms". In: *arXiv:2104.04768 [cs]* (2021)
Cited on pages 26, 52, 53

[134] Stephane Doncieux, Alban Laflaquière, and Alexandre Coninx. "Novelty Search: A Theoretical Perspective". In: *Proceedings of the Genetic and Evolutionary Computation Conference*. Prague Czech Republic: ACM, 2019, pp. 99–106. DOI: 10.1145/3321707.3321752
Cited on pages 26, 134

[135] Cédric Colas, Olivier Sigaud, and Pierre-Yves Oudeyer. "GEP-PG: Decoupling Exploration and Exploitation in Deep Reinforcement Learning Algorithms". In: *Proceedings of the 35th International Conference on Machine Learning*. PMLR, 2018, pp. 1039–1048
Cited on pages 27, 33, 143

[136] Kei Terayama, Masato Sumita, Ryo Tamura, Daniel T. Payne, Mandeep K. Chahal, Shinsuke Ishihara, and Koji Tsuda. "Pushing Property Limits in Materials Discovery via Boundless Objective-Free Exploration". In: *Chemical Science* 11.23 (2020), pp. 5959–5968. DOI: 10.1039/D0SC00982B
Cited on pages 27, 28, 30, 71

[137] Diederik P Kingma and Max Welling. "Auto-Encoding Variational Bayes". In: *arXiv:1312.6114* (2013)
Cited on pages 28, 33, 58, 60, 61, 75, 76, 158, 212

[138] Pamela Lyon, Fred Keijzer, Detlev Arendt, and Michael Levin. *Reframing Cognition: Getting down to Biological Basics*. 2021
Cited on pages 28, 49, 101

[139] Martin J. Falk, Finnegan D. Roach, William Gilpin, and Arvind Murugan. *Curiosity-Driven Search for Novel Non-Equilibrium Behaviors*. 2023
Cited on pages 28, 30, 68, 98, 103, 128, 132, 184

[140] Shyam Sudhakaran, Elias Najarro, and Sebastian Risi. *Goal-Guided Neural Cellular Automata: Learning to Control Self-Organising Systems*. 2022. DOI: 10.48550/arXiv.2205.06806
Cited on pages 28, 30

[141] Mohammad HamediRad, Ran Chao, Scott Weisberg, Jiazhang Lian, Saurabh Sinha, and Huimin Zhao. "Towards a Fully Automated Algorithm Driven Platform for Biosystems Design". In: *Nature communications* 10.1 (2019), pp. 1–10
Cited on page 30

[142] Jie Zhang, Søren D Petersen, Tijana Radivojevic, Andrés Ramirez, Andrés Pérez-Manríquez, Eduardo Abeliuk, Benjamín J Sánchez, Zak Costello, Yu Chen, Michael J Fero, et al. "Combining Mechanistic and Machine Learning Models for Predictive Engineering and Optimization of Tryptophan Metabolism". In: *Nature communications* 11.1 (2020), pp. 1–13
Cited on page 30

[143] Andrew J Elliot and James W Fryer. "The Goal Construct in Psychology". In: *Handbook of motivation science* 18 (2008), pp. 235–250
Cited on page 31

[144] James Y Shah and Wendi L Gardner. *Handbook of Motivation Science*. Guilford Press, 2008
Cited on page 31

[145] Wesley P Clawson and Michael Levin. "Endless Forms Most Beautiful 2.0: Teleonomy and the Bioengineering of Chimaeric and Synthetic Organisms". In: *Biological Journal of the Linnean Society* (2022), blac073. DOI: 10.1093/biolinnean/blac073
Cited on pages 31, 126, 181

[146] Junyi Chu and Laura Schulz. "Exploratory Play, Rational Action, and Efficient Search." In: *CogSci*. 2020
Cited on page 31

[147] D. E. Berlyne. "Curiosity and Exploration". In: *Science (New York, N.Y.)* 153.3731 (1966), pp. 25–33. DOI: 10.1126/science.153.3731.25
Cited on page 31

[148] Mihaly Csikszentmihalhi. *Finding Flow: The Psychology of Engagement with Everyday Life*. Hachette UK, 2020
Cited on page 31

[149] Alison Gopnik, Andrew N Meltzoff, and Patricia K Kuhl. *The Scientist in the Crib: Minds, Brains, and How Children Learn*. William Morrow & Co, 1999
Cited on pages 31, 35

[150] Celeste Kidd and Benjamin Y Hayden. "The Psychology and Neuroscience of Curiosity". In: *Neuron* 88.3 (2015), pp. 449–460
Cited on page 31

[151] Pierre-Yves Oudeyer and Linda B. Smith. "How Evolution May Work Through Curiosity-Driven Developmental Process". In: *Topics in Cognitive Science* 8.2 (2016), pp. 492–502. DOI: 10.1111/tops.12196
Cited on pages 31, 32

[152] Jacqueline Gottlieb and Pierre-Yves Oudeyer. "Towards a Neuroscience of Active Sampling and Curiosity". In: *Nature Reviews Neuroscience* 19.12 (2018), pp. 758–770
Cited on page 31

[153] Michael Tomasello. *The Cultural Origins of Human Cognition*. Harvard university press, 2009
Cited on page 32

[154] Michael Tomasello, Malinda Carpenter, Josep Call, Tanya Behne, and Henrike Moll. "Understanding and Sharing Intentions: The Origins of Cultural Cognition". In: *Behavioral and brain sciences* 28.5 (2005), pp. 675–691
Cited on page 32

[155] Michael Tomasello. *Becoming Human: A Theory of Ontogeny*. Harvard University Press, 2019
Cited on page 32

[156] Kelly Brewer, Nancy Pollock, and F Virginia Wright. "Addressing the Challenges of Collaborative Goal Setting with Children and Their Families". In: *Physical & Occupational Therapy in Pediatrics* 34.2 (2014), pp. 138–152
Cited on page 32

[157] Pierre-Yves Oudeyer and Frédéric Kaplan. "Discovering Communication". In: *Connection Science* 18.2 (2006), pp. 189–206. DOI: 10.1080/09540090600768567
Cited on pages 32, 33

[158] Pierre-Yves Oudeyer, Frederic Kaplan, Verena Hafner, and Andrew Whyte. "The Playground Experiment: Task-independent Development of a Curious Robot". In: *Proceedings of the AAAI Spring Symposium on Developmental Robotics* (2005)
Cited on pages 32, 33

[159] Clément Moulin-Frier, Sao Mai Nguyen, and Pierre-Yves Oudeyer. "Self-Organization of Early Vocal Development in Infants and Machines: The Role of Intrinsic Motivation". In: *Frontiers in Psychology* 4 (2014)
Cited on page 32

[160] Alexandr Ten, Pierre-Yves Oudeyer, and Clément Moulin-Frier. "Curiosity-Driven Exploration". In: *The Drive for Knowledge: The Science of Human Information Seeking* (2022), p. 53
Cited on page 32

[161] Alexandr Ten, Pramod Kaushik, Pierre-Yves Oudeyer, and Jacqueline Gottlieb. "Humans Monitor Learning Progress in Curiosity-Driven Exploration". In: *Nature Communications* 12.1 (2021), p. 5972. DOI: 10.1038/s41467-021-26196-w
Cited on page 32

[162] Georg Martius, Ralf Der, and Nihat Ay. "Information Driven Self-Organization of Complex Robotic Behaviors". In: *PLOS ONE* 8.5 (2013), e63400. DOI: 10.1371/journal.pone.0063400
Cited on page 33

[163] Matthias Rolf and Jochen J. Steil. "Efficient Exploratory Learning of Inverse Kinematics on a Bionic Elephant Trunk". In: *IEEE Transactions on Neural Networks and Learning Systems* 25.6 (2014), pp. 1147–1160. DOI: 10.1109/TNNLS.2013.2287890
Cited on page 33

[164] Sao Mai Nguyen and Pierre-Yves Oudeyer. "Socially Guided Intrinsic Motivation for Robot Learning of Motor Skills". In: *Autonomous Robots* 36.3 (2014), p. 273. DOI: 10.1007/s10514-013-9339-y
Cited on pages 33, 92

[165] Luca Lonini, Sébastien Forestier, Céline Teulière, Yu Zhao, Bertram Shi, and Jochen Triesch. "Robust Active Binocular Vision through Intrinsically Motivated Learning". In: *Frontiers in Neurorobotics* 7 (2013)
Cited on page 33

[166] Sébastien Forestier and Pierre-Yves Oudeyer. "Modular Active Curiosity-Driven Discovery of Tool Use". In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 3965–3972
Cited on pages 33, 39, 41, 58

[167] Open Ended Learning Team et al. *Open-Ended Learning Leads to Generally Capable Agents*. 2021
Cited on pages 33, 102, 108

[168] Adrien Laversanne-Finot, Alexandre Pere, and Pierre-Yves Oudeyer. "Curiosity Driven Exploration of Learned Disentangled Goal Spaces". In: ed. by Aude Billard, Anca Dragan, Jan Peters, and Jun Morimoto. Vol. 87. Proceedings of Machine Learning Research. PMLR, 2018, pp. 487–504
Cited on pages 33, 40, 58, 60, 62, 65, 71, 79

[169] Ashvin V Nair, Vitchyr Pong, Murtaza Dalal, Shikhar Bahl, Steven Lin, and Sergey Levine. "Visual Reinforcement Learning with Imagined Goals". In: *Advances in Neural Information Processing Systems*. 2018, pp. 9191–9200
Cited on pages 33, 40, 61, 79

[170] Vitchyr H Pong, Murtaza Dalal, Steven Lin, Ashvin Nair, Shikhar Bahl, and Sergey Levine. "Skew-Fit: State-covering Self-Supervised Reinforcement Learning". In: *arXiv preprint arXiv:1903.03698* (2019)
Cited on pages 33, 40, 41, 61, 79

[171] Grgur Kovač, Adrien Laversanne-Finot, and Pierre-Yves Oudeyer. "GRIMGEP: Learning Progress for Robust Goal Sampling in Visual Deep Reinforcement Learning". In: *IEEE Transactions on Cognitive and Developmental Systems* (2022), pp. 1–1. DOI: 10.1109/TCDS.2022.3216911
Cited on pages 33, 41

[172] Cédric Colas, Pierre Fournier, Olivier Sigaud, Mohamed Chetouani, and Pierre-Yves Oudeyer. "CURIOUS: Intrinsically Motivated Modular Multi-Goal Reinforcement Learning". In: *arXiv preprint arXiv:1810.06284* (2018)
Cited on pages 33, 103

[173] Cédric Colas, Tristan Karch, Nicolas Lair, Jean-Michel Dussoux, Clément Moulin-Frier, Peter Dominey, and Pierre-Yves Oudeyer. "Language as a Cognitive Tool to Imagine Goals in Curiosity Driven Exploration". In: *Advances in Neural Information Processing Systems*. Vol. 33. Curran Associates, Inc., 2020, pp. 3761–3774
Cited on pages 33, 103, 180

[174] Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. *Voyager: An Open-Ended Embodied Agent with Large Language Models*. 2023. Comment: Project website and open-source codebase: https://voyager.minedojo.org/
Cited on page 33

[175] P. -Y. Oudeyer, J. Gottlieb, and M. Lopes. "Chapter 11 - Intrinsic Motivation, Curiosity, and Learning: Theory and Applications in Educational Technologies". In: *Progress in Brain Research*. Ed. by Bettina Studer and Stefan Knecht. Vol. 229. Motivation. Elsevier, 2016, pp. 257–284. DOI: 10.1016/bs.pbr.2016.05.005
Cited on page 34

[176] Benjamin Clement, Didier Roy, Pierre-Yves Oudeyer, Manuel Lopes, et al. "Multi-Armed Bandits for Intelligent Tutoring Systems". In: *Journal of Educational Data Mining* 7.2 (2015), pp. 20–48
Cited on page 34

[177] Rania Abdelghani, Edith Law, Chloé Desvaux, Pierre-Yves Oudeyer, and Hélène Sauzéon. "Interactive Environments for Training Children's Curiosity through the Practice of Metacognitive Skills : A Pilot Study". In: *Proceedings of the 22nd Annual ACM Interaction Design and Children Conference*. IDC '23. New York, NY, USA: Association for Computing Machinery, 2023, pp. 495–501. DOI: 10.1145/3585088.3593880
Cited on page 34

[178] Rania Abdelghani, Yen-Hsiang Wang, Xingdi Yuan, Tong Wang, Pauline Lucas, Hélène Sauzéon, and Pierre-Yves Oudeyer. "GPT-3-Driven Pedagogical Agents to Train Children's Curious Question-Asking Skills". In: *International Journal of Artificial Intelligence in Education* (2023). DOI: 10.1007/s40593-023-00340-7
Cited on page 34

[179] Yolanda Gil. "Thoughtful Artificial Intelligence: Forging a New Partnership for Data Science and Scientific Discovery". In: *Data Science* 1.1-2 (2017), pp. 119–129
Cited on page 34

[180] Kenneth O Stanley and Joel Lehman. *Why Greatness Cannot Be Planned: The Myth of the Objective*. Springer, 2015
Cited on page 34

[181] Jonathan M. Spector, Rosemary S. Harrison, and Mark C. Fishman. "Fundamental Science behind Today's Important Medicines". In: *Science Translational Medicine* 10.438 (2018), eaaq1787. DOI: 10.1126/scitranslmed.aaq1787
Cited on page 34

[182] Jean Piaget, Margaret Cook, et al. *The Origins of Intelligence in Children*. Vol. 8. International Universities Press New York, 1952
Cited on pages 35, 37

[183] Pierre-Yves Oudeyer, Frdric Kaplan, and Verena V Hafner. "Intrinsic Motivation Systems for Autonomous Mental Development". In: *IEEE transactions on evolutionary computation* 11.2 (2007), pp. 265–286
Cited on page 35

[184] Alison Gopnik. "The Scientist as Child". In: *Philosophy of Science* 63.4 (1996), pp. 485–514. DOI: 10.1086/289970
Cited on page 35

[185] Simon Kirby, Mike Dowman, and Thomas L. Griffiths. "Innateness and Culture in the Evolution of Language". In: *Proceedings of the National Academy of Sciences* 104.12 (2007), pp. 5241–5245. DOI: 10.1073/pnas.0608222104

[186] Gianluca Baldassarre and Marco Mirolli. *Intrinsically Motivated Learning in Natural and Artificial Systems*. Springer, 2013
Cited on page 40

[187] Rémy Portelas, Cédric Colas, Katja Hofmann, and Pierre-Yves Oudeyer. "Teacher Algorithms for Curriculum Learning of Deep RL in Continuously Parameterized Environments". In: *Proceedings of the Conference on Robot Learning*. PMLR, 2020, pp. 835–853
Cited on page 41

[188] Rémy Portelas, Cédric Colas, Lilian Weng, Katja Hofmann, and Pierre-Yves Oudeyer. "Automatic Curriculum Learning for Deep RL: A Short Survey". In: *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*. IJCAI'20. Yokohama, Yokohama, Japan, 2021, pp. 4819–4825
Cited on page 41

[189] Clément Romac, Rémy Portelas, Katja Hofmann, and Pierre-Yves Oudeyer. "TeachMyAgent: A Benchmark for Automatic Curriculum Learning in Deep RL". In: *Proceedings of the 38th International Conference on Machine Learning*. PMLR, 2021, pp. 9052–9063
Cited on page 41

[190] Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. "Hindsight Experience Replay". In: *Advances in Neural Information Processing Systems*. 2017, pp. 5048–5058
Cited on pages 41, 58, 60

[191] Randall D. Beer. "The Cognitive Domain of a Glider in the Game of Life". In: *Artificial Life* 20.2 (2014), pp. 183–206. DOI: 10.1162/ARTL_a_00125
Cited on pages 44, 103, 115

[192] Randall D Beer. "Bittorio Revisited: Structural Coupling in the Game of Life". In: *Adaptive Behavior* 28.4 (2020), pp. 197–212. DOI: 10.1177/1059712319859907
Cited on pages 44, 103, 108

[193] Djordje Grbic, Rasmus Berg Palm, Elias Najarro, Claire Glanois, and Sebastian Risi. "EvoCraft: A New Challenge for Open-Endedness". In: *Applications of Evolutionary Computation*. Ed. by Pedro A. Castillo and Juan Luis Jiménez Laredo. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2021, pp. 325–340. DOI: 10.1007/978-3-030-72699-7_21
Cited on page 46

[194] Kenneth O Stanley. "Exploiting Regularity without Development." In: *AAAI Fall Symposium: Developmental Systems*. 2006, p. 49
Cited on pages 46, 53, 62, 64, 152, 158, 199

[195] Mykhailo Moroz. *Reintegration Tracking*. https://michaelmoroz.github.io/Reintegration-Tracking/
Cited on pages 47, 117, 240

[196] Tim Hoverd and Susan Stepney. "Energy as a Driver of Diversity in Open-Ended Evolution (Full Article)". In: *ECAL 2011: The 11th European Conference on Artificial Life*. MIT Press, 2011. DOI: 10.7551/978-0-262-29714-1-ch055
Cited on page 47

[197] Simon Hickinbotham and Susan Stepney. "Conservation of Matter Increases Evolutionary Activity". In: *ECAL 2015: The 13th European Conference on Artificial Life*. MIT Press, 2015, pp. 98–105. DOI: 10.1162/978-0-262-33027-5-ch024
Cited on page 47

[198] James Bradbury et al. *JAX: Composable Transformations of Python+NumPy Programs*. 2018
Cited on pages 47, 172, 240

[199] Alexander Mordvintsev, Eyvind Niklasson, and Ettore Randazzo. *Particle Lenia and the Energy-Based Formulation*. 2022
Cited on pages 47, 117

[200] Takako Kawaguchi, Reiji Suzuki, Takaya Arita, and Bert Chan. "Introducing Asymptotics to the State-Updating Rule in Lenia". In: *ALIFE 2022: The 2022 Conference on Artificial Life*. MIT Press, 2021
Cited on page 47

[201] Hiroki Kojima and Takashi Ikegami. "Implementation of Lenia as a Reaction-Diffusion System". In: *ALIFE 2023: Ghost in the Machine: Proceedings of the 2023 Artificial Life Conference*. MIT Press, 2023. DOI: 10.1162/isal_a_00638
Cited on page 47

[202] Q Tyrell Davis and Josh Bongard. "Glaberish: Generalizing the Continuously-Valued Lenia Framework to Arbitrary Life-like Cellular Automata". In: *Artificial Life Conference Proceedings 34*. Vol. 2022. MIT Press One Rogers Street, Cambridge, MA 02142-1209, USA journals-info . . ., 2022, p. 47
Cited on page 47

[203] Santosh Manicka and Michael Levin. "The Cognitive Lens: A Primer on Conceptual Tools for Analysing Information Processing in Developmental and Regenerative Morphogenesis". In: *Philosophical Transactions of the Royal Society B: Biological Sciences* 374.1774 (2019), p. 20180369. DOI: 10.1098/rstb.2018.0369
Cited on pages 49, 127

[204] Yoshua Bengio, Aaron Courville, and Pascal Vincent. "Representation Learning: A Review and New Perspectives". In: *IEEE transactions on pattern analysis and machine intelligence* 35.8 (2013), pp. 1798–1828. DOI: 10.1109/tpami.2013.50
Cited on page 51

[205] Sebastian Risi and Kenneth O. Stanley. "Deep Neuroevolution of Recurrent and Discrete World Models". In: *Proceedings of the Genetic and Evolutionary Computation Conference*. GECCO '19. New York, NY, USA: Association for Computing Machinery, 2019, pp. 456–462. DOI: 10.1145/3321707.3321817
Cited on page 53

[206] Yujin Tang, Yingtao Tian, and David Ha. "EvoJAX: Hardware-Accelerated Neuroevolution". In: *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. 2022, pp. 308–311. DOI: 10.1145/3520304.3528770. Comment: GECCO 2022. Project website at https://github.com/google/evojax
Cited on page 53

[207] Dominik Dellermann, Adrian Calma, Nikolaus Lipusch, Thorsten Weber, Sascha Weigel, and Philipp Ebel. "The Future of Human-AI Collaboration: A Taxonomy of Design Knowledge for Hybrid Intelligence Systems". In: *arXiv:2105.03354 [cs]* (2021)
Cited on pages 55, 91

[208] Carlos Florensa, David Held, Xinyang Geng, and Pieter Abbeel. "Automatic Goal Generation for Reinforcement Learning Agents". In: *Proceedings of the 35th International Conference on Machine Learning*. PMLR, 2018, pp. 1515–1528
Cited on page 58

[209] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. "Beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework". In: *International Conference on Learning Representations*. 2016
Cited on pages 58, 61, 200

[210] Christopher P Burgess, Irina Higgins, Arka Pal, Loic Matthey, Nick Watters, Guillaume Desjardins, and Alexander Lerchner. "Understanding Disentangling in β-VAE". In: *arXiv preprint arXiv:1804.03599* (2018)
Cited on pages 58, 71, 208, 218, 219

[211] Yoshua Bengio, Aaron Courville, and Pascal Vincent. "Representation Learning: A Review and New Perspectives". In: *IEEE transactions on pattern analysis and machine intelligence* 35.8 (2013), pp. 1798–1828
Cited on page 60

[212] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative Adversarial Nets". In: *Advances in Neural Information Processing Systems*. Vol. 27. Curran Associates, Inc., 2014
Cited on page 60

[213] L. Theis, A. van den Oord, and M. Bethge. "A Note on the Evaluation of Generative Models". In: *International Conference on Learning Representations*. 2016
Cited on page 60

[214] Irina Higgins, Arka Pal, Andrei Rusu, Loic Matthey, Christopher Burgess, Alexander Pritzel, Matthew Botvinick, Charles Blundell, and Alexander Lerchner. "Darla: Improving Zero-Shot Transfer in Reinforcement Learning". In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 1480–1490
Cited on pages 60, 79

[215] Manuel Watter, Jost Springenberg, Joschka Boedecker, and Martin Riedmiller. "Embed to Control: A Locally Linear Latent Dynamics Model for Control from Raw Images". In: *Advances in Neural Information Processing Systems*. 2015, pp. 2746–2754
Cited on pages 60, 79

[216] Valentin Thomas, Jules Pondard, Emmanuel Bengio, Marc Sarfati, Philippe Beaudoin, Marie-Jean Meurs, Joelle Pineau, Doina Precup, and Yoshua Bengio. "Independently Controllable Features". In: *arXiv preprint arXiv:1708.01289* (2017)
Cited on pages 60, 79

[217] Arash Vahdat and Jan Kautz. "NVAE: A Deep Hierarchical Variational Autoencoder". In: *Advances in Neural Information Processing Systems*. Vol. 33. Curran Associates, Inc., 2020, pp. 19667–19679
Cited on page 60

[218] Antoine Cully. "Autonomous Skill Discovery with Quality-Diversity and Unsupervised Descriptors". In: *Proceedings of the Genetic and Evolutionary Computation Conference*. 2019, pp. 81–89
Cited on pages 61, 71

[219] Luca Grillotti and Antoine Cully. "Unsupervised Behavior Discovery With Quality-Diversity Optimization". In: *IEEE Transactions on Evolutionary Computation* 26.6 (2022), pp. 1539–1552. DOI: 10.1109/TEVC.2022.3159855
Cited on page 61

[220] Peter Pastor, Mrinal Kalakrishnan, Franziska Meier, Freek Stulp, Jonas Buchli, Evangelos Theodorou, and Stefan Schaal. "From Dynamic Movement Primitives to Associative Skill Memories". In: *Robotics and Autonomous Systems*. Models and Technologies for Multi-modal Skill Training 61.4 (2013), pp. 351–361. DOI: 10.1016/j.robot.2012.09.017
Cited on page 63

[221] Martin Gardener. "MATHEMATICAL GAMES: The Fantastic Combinations of John Conway's New Solitaire Game" Life,"" in: *Scientific American* 223 (1970), pp. 120–123
Cited on page 64

[222] Randall D Beer. "Autopoiesis and Cognition in the Game of Life". In: *Artificial Life* 10.3 (2004), pp. 309–326
Cited on pages 64, 102

[223] Yuri Burda, Harri Edwards, Deepak Pathak, Amos Storkey, Trevor Darrell, and Alexei A. Efros. "Large-Scale Study of Curiosity-Driven Learning". In: *ICLR* (2019)
Cited on page 66

[224] Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. "Exploration by Random Network Distillation". In: *International Conference on Learning Representations*. 2019
Cited on page 66

[225] Laurens van der Maaten and Geoffrey Hinton. "Visualizing Data Using T-SNE". In: *Journal of machine learning research* 9.Nov (2008), pp. 2579–2605
Cited on pages 67, 188

[226] Shengjia Zhao, Jiaming Song, and Stefano Ermon. "Towards Deeper Understanding of Variational Autoencoding Models". In: *preprint arXiv:1702.08658* (2017)
Cited on pages 67, 81

[227] Yoshiki Kuramoto. "Self-Entrainment of a Population of Coupled Non-Linear Oscillators". In: *International Symposium on Mathematical Problems in Theoretical Physics*. Ed. by Huzihiro Araki. Lecture Notes in Physics. Berlin, Heidelberg: Springer, 1975, pp. 420–422. DOI: 10.1007/BFb0013365
Cited on pages 68, 98

[228] Juan A. Acebrón, L. L. Bonilla, Conrad J. Pérez Vicente, Félix Ritort, and Renato Spigler. "The Kuramoto Model: A Simple Paradigm for Synchronization Phenomena". In: *Reviews of Modern Physics* 77.1 (2005), pp. 137–185. DOI: 10.1103/RevModPhys.77.137
Cited on pages 68, 98

[229] Frank P Kuhl and Charles R Giardina. "Elliptic Fourier Features of a Closed Contour". In: *Computer graphics and image processing* 18.3 (1982), pp. 236–258
Cited on pages 72, 207, 208

[230] Jeaneth Machicao, Lucas C Ribas, Leonardo FS Scabini, and Odermir M Bruno. "Cellular Automata Rule Characterization and Classification Using Texture Descriptors". In: *Physica A: Statistical Mechanics and its Applications* 497 (2018), pp. 109–117
Cited on pages 72, 206

[231] James S Cope, David Corney, Jonathan Y Clark, Paolo Remagnino, and Paul Wilkin. "Plant Species Identification Using Digital Morphometrics: A Review". In: *Expert Systems with Applications* 39.8 (2012), pp. 7562–7573
Cited on page 72

[232] Ying Zhang, Chunjiang Zhao, Jianjun Du, Xinyu Guo, Wenliang Wen, Shenghao Gu, Jinglu Wang, and Jiangchuan Fan. "Crop Phenomics: Current Status and Perspectives". In: *Frontiers in Plant Science* 10 (2019), p. 714
Cited on page 72

[233] Margaret A. Boden. "Creativity and Artificial Intelligence". In: *Artificial Intelligence*. Artificial Intelligence 40 Years Later 103.1 (1998), pp. 347–356. DOI: 10.1016/S0004-3702(98)00055-1
Cited on pages 73, 74, 90

[234] John Maynard Smith and Eors Szathmary. *The Major Transitions in Evolution*. OUP Oxford, 1997
Cited on page 74

[235] Mark A. Bedau. "Weak Emergence". In: *Philosophical Perspectives* 11 (1997), pp. 375–399
Cited on page 74

[236] David J Chalmers. "Strong and Weak Emergence". In: *The re-emergence of emergence* 675 (2006), pp. 244–256
Cited on page 74

[237] Alyssa M. Adams. "A Graph-Theoretic Approach to Understanding Emergent Behavior in Physical Systems". In: *ALIFE 2021: The 2021 Conference on Artificial Life*. MIT Press, 2021. DOI: 10.1162/isal_a_00382
Cited on pages 74, 75

[238] Wolfgang Banzhaf, Bert Baumgaertner, Guillaume Beslon, René Doursat, James A Foster, Barry McMullin, Vinicius Veloso De Melo, Thomas Miconi, Lee Spector, Susan Stepney, et al. "Defining and Simulating Open-Ended Novelty: Requirements, Guidelines, and Challenges". In: *Theory in Biosciences* 135 (2016), pp. 131–161
Cited on pages 74, 179

[239] Tim Taylor. "Evolutionary Innovations and Where to Find Them: Routes to Open-Ended Evolution in Natural and Artificial Systems". In: *Artificial life* 25.2 (2019), pp. 207–224
Cited on page 74

[240] Susan Stepney. "Modelling and Measuring Open-Endedness". In: *Artificial Life* 25.1 (2021), p. 9
Cited on page 74

[241] Jeffrey L Elman. "Learning and Development in Neural Networks: The Importance of Starting Small". In: *Cognition* 48.1 (1993), pp. 71–99
Cited on page 75

[242] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. "Progressive Neural Networks". In: *arXiv preprint arXiv:1606.04671* (2016)
Cited on pages 76, 212, 218

[243] Dushyant Rao, Francesco Visin, Andrei Rusu, Razvan Pascanu, Yee Whye Teh, and Raia Hadsell. "Continual Unsupervised Representation Learning". In: *Advances in Neural Information Processing Systems*. 2019, pp. 7645–7655
Cited on pages 77, 217, 218

[244] Soochan Lee, Junsoo Ha, Dongsu Zhang, and Gunhee Kim. "A Neural Dirichlet Process Mixture Model for Task-Free Continual Learning". In: *arXiv preprint arXiv:2001.00689* (2020)
Cited on pages 77, 217, 218

[245] Zhiyuan Li, Jaideep Vitthal Murkute, Prashnna Kumar Gyawali, and Linwei Wang. "Progressive Learning and Disentanglement of Hierarchical Representations". In: *arXiv preprint arXiv:2002.10549* (2020)
Cited on pages 77, 217, 218

[246] Paul A Szerlip, Gregory Morse, Justin K Pugh, and Kenneth O Stanley. "Unsupervised Feature Learning through Divergent Discriminative Feature Accumulation". In: *Twenty-Ninth AAAI Conference on Artificial Intelligence*. 2015
Cited on pages 77, 180

[247] Justin K Pugh, Lisa B Soros, and Kenneth O Stanley. "Searching for Quality Diversity When Diversity Is Unaligned with Quality". In: *International Conference on Parallel Problem Solving from Nature*. Springer, 2016, pp. 880–889
Cited on page 78

[248] Timothée Lesort, Natalia Díaz-Rodríguez, Jean-François Goudou, and David Filliat. "State Representation Learning for Control: An Overview". In: *Neural Networks* 108 (2018), pp. 379–392. DOI: 10.1016/j.neunet.2018.07.006
Cited on page 79

[249] Herke Van Hoof, Nutan Chen, Maximilian Karl, Patrick van der Smagt, and Jan Peters. "Stable Reinforcement Learning with Autoencoders for Tactile and Visual Data". In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 3928–3934
Cited on page 79

[250] Maximilian Karl, Maximilian Soelch, Justin Bayer, and Patrick Van der Smagt. "Deep Variational Bayes Filters: Unsupervised Learning of State Space Models from Raw Data". In: *arXiv preprint arXiv:1605.06432* (2016)
Cited on page 79

[251] Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. "Learning Latent Dynamics for Planning from Pixels". In: *arXiv preprint arXiv:1811.04551* (2018)
Cited on page 79

[252] Amy Zhang, Harsh Satija, and Joelle Pineau. "Decoupling Dynamics and Reward for Transfer Learning". In: *arXiv preprint arXiv:1804.10689* (2018)
Cited on page 79

[253] Nikolaus Kriegeskorte, Marieke Mur, and Peter A Bandettini. "Representational Similarity Analysis-Connecting the Branches of Systems Neuroscience". In: *Frontiers in systems neuroscience* 2 (2008), p. 4
Cited on pages 79, 180

[254] Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. "Similarity of Neural Network Representations Revisited". In: *arXiv preprint arXiv:1905.00414* (2019)
Cited on pages 80, 213

[255] Leo Cazenille. "Ensemble Feature Extraction for Multi-Container Quality-Diversity Algorithms". In: *Proceedings of the Genetic and Evolutionary Computation Conference* (2021), pp. 75–83. DOI: 10.1145/3449639.3459392. Comment: Draft version. 10 pages, 4 figures, 4 tables, Accepted at the GECCO2021 Conference
Cited on page 87

[256] H. Takagi. "Interactive Evolutionary Computation: Fusion of the Capabilities of EC Optimization and Human Evaluation". In: *Proceedings of the IEEE* 89.9 (2001), pp. 1275–1296. DOI: 10.1109/5.949485
Cited on page 94

[257] Karl Sims. "Interactive Evolution of Dynamical Systems". In: *Toward a Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life*. 1992, pp. 171–178
Cited on page 94

[258] WB Langdon. "Pfeiffer–A Distributed Open-Ended Evolutionary System". In: Citeseer, 2005
Cited on page 94

[259] Jimmy Secretan, Nicholas Beato, David B. D Ambrosio, Adelein Rodriguez, Adam Campbell, and Kenneth O. Stanley. "Picbreeder: Evolving Pictures Collaboratively Online". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '08. New York, NY, USA: Association for Computing Machinery, 2008, pp. 1759–1768. DOI: 10.1145/1357054.1357328
Cited on page 94

[260] Jimmy Secretan, Nicholas Beato, David B D'Ambrosio, Adelein Rodriguez, Adam Campbell, Jeremiah T Folsom-Kovarik, and Kenneth O Stanley. "Picbreeder: A Case Study in Collaborative Evolutionary Exploration of Design Space". In: *Evolutionary computation* 19.3 (2011), pp. 373–403
Cited on page 94

[261] Joost Huizinga, Kenneth O. Stanley, and Jeff Clune. "The Emergence of Canalization and Evolvability in an Open-Ended, Interactive Evolutionary System". In: *Artificial Life* 24.3 (2018), pp. 157–181. DOI: 10.1162/artl_a_00263
Cited on page 94

[262] Martial Mermillod, Aurélia Bugaiska, and Patrick BONIN. "The Stability-Plasticity Dilemma: Investigating the Continuum from Catastrophic Forgetting to Age-Limited Learning Effects". In: *Frontiers in Psychology* 4 (2013)
Cited on page 97

[263] David Krakauer, Nils Bertschinger, Eckehard Olbrich, Jessica C. Flack, and Nihat Ay. "The Information Theory of Individuality". In: *Theory in Biosciences* 139.2 (2020), pp. 209–223. DOI: 10.1007/s12064-020-00313-7
Cited on pages 101, 103, 122

[264] Humberto R Maturana and Francisco J Varela. *Autopoiesis and Cognition: The Realization of the Living*. 1980
Cited on pages 101, 108

[265] Ezequiel A Di Paolo. "Process and Individuation: The Development of Sensorimotor Agency". In: *Human Development* 63.3-4 (2019), pp. 202–226
Cited on pages 101, 102, 111

[266] Rui Wang, Joel Lehman, Jeff Clune, and Kenneth O Stanley. "Paired Open-Ended Trailblazer (Poet): Endlessly Generating Increasingly Complex and Diverse Learning Environments and Their Solutions". In: *arXiv preprint arXiv:1901.01753* (2019)
Cited on page 102

[267] Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, et al. "Solving Rubik's Cube with a Robot Hand". In: *arXiv preprint arXiv:1910.07113* (2019)
Cited on page 102

[268] Bowen Baker, Ingmar Kanitscheider, Todor Markov, Yi Wu, Glenn Powell, Bob McGrew, and Igor Mordatch. "Emergent Tool Use from Multi-Agent Autocurricula". In: *arXiv preprint arXiv:1909.07528* (2019)
Cited on pages 102, 122

[269] Rolf Pfeifer and Josh Bongard. *How the Body Shapes the Way We Think: A New View of Intelligence*. MIT press, 2006
Cited on pages 102, 116, 183

[270] Tom Froese and Tom Ziemke. "Enactive Artificial Intelligence: Investigating the Systemic Organization of Life and Mind". In: *Artificial intelligence* 173.3-4 (2009), pp. 466–500
Cited on pages 102, 116, 183

[271] F.G. Varela, H.R. Maturana, and R. Uribe. "Autopoiesis: The Organization of Living Systems, Its Characterization and a Model". In: *Bio Systems* 5.4 (1974), pp. 187–196. DOI: 10.1016/0303-2647(74)90031-8
Cited on page 102

[272] Barry McMullin. "Thirty Years of Computational Autopoiesis: A Review". In: *Artificial life* 10.3 (2004), pp. 277–295
Cited on page 102

[273] Eran Agmon, Alexander J Gates, and Randall D Beer. "Ontogeny and Adaptivity in a Model Protocell". In: *Artificial Life Conference Proceedings 13*. MIT Press, 2015, pp. 216–223
Cited on pages 102, 103

[274] Martin Biehl, Takashi Ikegami, and Daniel Polani. "Towards Information Based Spatiotemporal Patterns as a Foundation for Agent Representation in Dynamical Systems". In: *Proceedings of the Artificial Life Conference 2016*. Cancun, Mexico: MIT Press, 2016, pp. 722–729. DOI: 10.7551/978-0-262-33936-0-ch115
Cited on pages 103, 122

[275] Arta Cika, Elissa Cohen, Germán Kruszewski, Luther Seet, Patrick Steinmann, and Wenqian Yin. *Resilient Life: An Exploration of Perturbed Autopoietic Patterns in Conway's Game of Life*. Vol. ALIFE 2020: The 2020 Conference on Artificial Life. ALIFE 2021: The 2021 Conference on Artificial Life. 2020, pp. 656–664
Cited on page 103

[276] Hiroaki Kitano. "Biological Robustness". In: *Nature Reviews Genetics* 5.11 (2004), pp. 826–837
Cited on page 103

[277] Alexandre Variengien, Sidney Pontes-Filho, Tom Eivind Glover, and Stefano Nichele. "Towards Self-organized Control: Using Neural Cellular Automata to Robustly Control a Cart-pole Agent". In: *Innovations in Machine Intelligence* (2021). DOI: 10.54854/imi2021.01
Cited on page 107

[278] Laura N Vandenberg, Dany S Adams, and Michael Levin. "Normalized Shape and Location of Perturbed Craniofacial Structures in the Xenopus Tadpole Reveal an Innate Ability to Achieve Correct Morphology". In: *Developmental Dynamics* 241.5 (2012), pp. 863–878
Cited on pages 111, 113

[279] Gerhard Fankhauser. "Maintenance of Normal Structure in Heteroploid Salamander Larvae, through Compensation of Changes in Cell Size by Adjustment of Cell Number and Cell Shape". In: *Journal of Experimental Zoology* 100.3 (1945), pp. 445–455
Cited on pages 111, 114

[280] Jörg Stelling, Uwe Sauer, Zoltan Szallasi, Francis J Doyle III, and John Doyle. "Robustness of Cellular Functions". In: *Cell* 118.6 (2004), pp. 675–685
Cited on page 111

[281] Michael Levin. "The Computational Boundary of a "Self": Developmental Bioelectricity Drives Multicellularity and Scale-Free Cognition". In: *Frontiers in Psychology* 10 (2019), p. 2688
Cited on page 114

[282] Wei Li, Xiaoran Wu, Hong Qin, Zhongqiang Zhao, and Hewen Liu. "Light-Driven and Light-Guided Microswimmers". In: *Advanced Functional Materials* 26.18 (2016), pp. 3164–3171
Cited on page 115

[283] Tim Salimans, Jonathan Ho, Xi Chen, Szymon Sidor, and Ilya Sutskever. "Evolution Strategies as a Scalable Alternative to Reinforcement Learning". In: *arXiv preprint arXiv:1703.03864* (2017)
Cited on pages 118, 242

[284] Peter Godfrey-Smith. "Environmental Complexity and the Evolution of Cognition". In: *The Evolution of Intelligence*. Mahwah, NJ, US: Lawrence Erlbaum Associates Publishers, 2002, pp. 223–249
Cited on pages 119, 122

[285] Nils Bertschinger, Eckehard Olbrich, Nihat Ay, and Jürgen Jost. "Autonomy: An Information Theoretic Perspective". In: *Bio Systems* 91.2 (2008), pp. 331–345. DOI: 10.1016/j.biosystems.2007.05.018
Cited on page 122

[286] Samuel Arbesman. *Emergent Microcosms*. Substack Newsletter. 2022
Cited on page 122

[287] Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. "Webgpt: Browser-assisted Question-Answering with Human Feedback". In: *arXiv preprint arXiv:2112.09332* (2021)
Cited on page 122

[288] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. "Toolformer: Language Models Can Teach Themselves to Use Tools". In: *arXiv preprint arXiv:2302.04761* (2023)
Cited on page 122

[289] Victor G. Laties. "Society for the Experimental Analysis of Behavior: The First Thirty Years (1957–1987)". In: *Journal of the Experimental Analysis of Behavior* 48.3 (1987), pp. 495–512. DOI: 10.1901/jeab.1987.48-495
Cited on page 124

[290] Gro Amdam and Anne Hovland. "Measuring Animal Preferences and Choice Behavior". In: *Nature Education Knowledge* 3 (2012), p. 74
Cited on page 124

[291] Saul McLeold. *Behavioral Perspective in Psychology [Behaviorism Theory]*. 2022
Cited on page 124

[292] Anatoly Mikhaltsov. *Paramecium Bursaria*. 16 September 2013, 19:32:09
Cited on page 124

[293] Joshua Bongard and Michael Levin. "There's Plenty of Room Right Here: Biological Systems as Evolved, Overloaded, Multi-Scale Machines". In: *Biomimetics* 8.1 (2023), p. 110. DOI: 10.3390/biomimetics8010110
Cited on page 124

[294] Juan J. Sanz-Ezquerro, Andrea E. Münsterberg, and Sigmar Stricker. "Editorial: Signaling Pathways in Embryonic Development". In: *Frontiers in Cell and Developmental Biology* 5 (2017)
Cited on page 125

[295] Elena R. Alvarez-Buylla, Enrique Balleza, Mariana Benítez, Carlos Espinosa-Soto, and Pablo Padilla-Longoria. "Gene Regulatory Network Models: A Dynamic and Integrative Approach to Development". In: *SEB experimental biology series* 61 (2008), pp. 113–139
Cited on page 125

[296] Sui Huang, Gabriel Eichler, Yaneer Bar-Yam, and Donald E. Ingber. "Cell Fates as High-Dimensional Attractor States of a Complex Gene Regulatory Network". In: *Physical Review Letters* 94.12 (2005), p. 128701. DOI: 10.1103/PhysRevLett.94.128701
Cited on page 125

[297] Eric H. Davidson. "Emerging Properties of Animal Gene Regulatory Networks". In: *Nature* 468.7326 (2010), pp. 911–920. DOI: 10.1038/nature09645
Cited on page 125

[298] Isabelle S. Peter and Eric H. Davidson. "Evolution of Gene Regulatory Networks Controlling Body Plan Development". In: *Cell* 144.6 (2011), pp. 970–985. DOI: 10.1016/j.cell.2011.02.017
Cited on page 125

[299] Kirsten H. ten Tusscher and Paulien Hogeweg. "Evolution of Networks for Body Plan Patterning; Interplay of Modularity, Robustness and Evolvability". In: *PLOS Computational Biology* 7.10 (2011), e1002208. DOI: 10.1371/journal.pcbi.1002208
Cited on page 125

[300] Hyobin Kim and Hiroki Sayama. "How Criticality of Gene Regulatory Networks Affects the Resulting Morphogenesis under Genetic Perturbations". In: *Artificial Life* 24.02 (2018), pp. 85–105. DOI: 10.1162/ARTL_a_00262
Cited on page 125

[301] Mansi Srivastava. "Beyond Casual Resemblance: Rigorous Frameworks for Comparing Regeneration Across Species". In: *Annual Review of Cell and Developmental Biology* 37.1 (2021), pp. 415–440. DOI: 10.1146/annurev-cellbio-120319-114716
Cited on page 125

[302] Arun J. Singh, Stephen A. Ramsey, Theresa M. Filtz, and Chrissa Kioussi. "Differential Gene Regulatory Networks in Development and Disease". In: *Cellular and Molecular Life Sciences* 75.6 (2018), pp. 1013–1025. DOI: 10.1007/s00018-017-2679-6
Cited on page 125

[303] Guimin Qin, Luqiong Yang, Yuying Ma, Jiayan Liu, and Qiuyan Huo. "The Exploration of Disease-Specific Gene Regulatory Networks in Esophageal Carcinoma and Stomach Adenocarcinoma". In: *BMC Bioinformatics* 20.22 (2019), p. 717. DOI: 10.1186/s12859-019-3230-6
Cited on page 125

[304] Hassan Fazilaty, Luciano Rago, Khalil Kass Youssef, Oscar H. Ocaña, Francisco Garcia-Asencio, Aida Arcas, Juan Galceran, and M. Angela Nieto. "A Gene Regulatory Network to Control EMT Programs in Development and Disease". In: *Nature Communications* 10.1 (2019), p. 5115. DOI: 10.1038/s41467-019-13091-8
Cited on page 125

[305] Jamie Davies and Michael Levin. *Synthetic Morphology via Active and Agential Matter*. 2022. DOI: 10.31219/osf.io/xrv8h
Cited on pages 125, 126

[306] Satoshi Toda, Lucas R. Blauch, Sindy K. Y. Tang, Leonardo Morsut, and Wendell A. Lim. "Programming Self-Organizing Multicellular Structures with Synthetic Cell-Cell Signaling". In: *Science* 361.6398 (2018), pp. 156–162. DOI: 10.1126/science.aat0271
Cited on page 125

[307] Satoshi Toda, Wesley L. McKeithan, Teemu J. Hakkinen, Pilar Lopez, Ophir D. Klein, and Wendell A. Lim. "Engineering Synthetic Morphogen Systems That Can Program Multicellular Patterning". In: *Science* 370.6514 (2020), pp. 327–331. DOI: 10.1126/science.abc0033
Cited on page 125

[308] Christine Ho and Leonardo Morsut. "Novel Synthetic Biology Approaches for Developmental Systems". In: *Stem Cell Reports* 16.5 (2021), pp. 1051–1064. DOI: 10.1016/j.stemcr.2021.04.007
Cited on page 125

[309] Marco Santorelli, Calvin Lam, and Leonardo Morsut. "Synthetic Development: Building Mammalian Multicellular Structures with Artificial Genetic Programs". In: *Current Opinion in Biotechnology*. Tissue, Cell and Pathway Engineering 59 (2019), pp. 130–140. DOI: 10.1016/j.copbio.2019.03.016
Cited on page 125

[310] Hidde de Jong. "Modeling and Simulation of Genetic Regulatory Systems: A Literature Review". In: *Journal of Computational Biology* 9.1 (2002), pp. 67–103. DOI: 10.1089/10665270252833208
Cited on page 125

[311] Thomas Schlitt and Alvis Brazma. "Current Approaches to Gene Regulatory Network Modelling". In: *BMC Bioinformatics* 8.6 (2007), S9. DOI: 10.1186/1471-2105-8-S6-S9
Cited on page 125

[312] Jacquelyn S. Fetrow and Patricia C. Babbitt. "New Computational Approaches to Understanding Molecular Protein Function". In: *PLOS Computational Biology* 14.4 (2018), e1005756. DOI: 10.1371/journal.pcbi.1005756
Cited on page 125

[313] Fernando M. Delgado and Francisco Gómez-Vela. "Computational Methods for Gene Regulatory Networks Reconstruction and Analysis: A Review". In: *Artificial Intelligence in Medicine* 95 (2019), pp. 133–145. DOI: 10.1016/j.artmed.2018.10.006
Cited on page 125

[314] Mihai Glont et al. "BioModels: Expanding Horizons to Include More Modelling Approaches and Formats". In: *Nucleic Acids Research* 46.D1 (2018), pp. D1248–D1253. DOI: 10.1093/nar/gkx1023
Cited on pages 125, 129, 171, 247

[315] Rahuman S Malik-Sheriff et al. "BioModels—15 Years of Sharing Computational Models in Life Science". In: *Nucleic Acids Research* 48.D1 (2020), pp. D407–D415. DOI: 10.1093/nar/gkz1055
Cited on pages 125, 129, 171, 247

[316] Stuart A Kauffman. *The Origins of Order: Self-organization and Selection in Evolution*. Oxford University Press, USA, 1993
Cited on pages 125, 145

[317] Stuart A Kauffman. *At Home in the Universe: The Search for Laws of Self-Organization and Complexity*. Oxford University Press, USA, 1995
Cited on pages 125, 145

[318] Charles I. Abramson and Michael Levin. "Behaviorist Approaches to Investigating Memory and Learning: A Primer for Synthetic Biology and Bioengineering". In: *Communicative & Integrative Biology* 14.1 (2021), pp. 230–247. DOI: 10.1080/19420889.2021.2005863
Cited on pages 125, 127

[319] František Baluška and Michael Levin. "On Having No Head: Cognition throughout Biological Systems". In: *Frontiers in Psychology* 7 (2016)
Cited on pages 125, 126

[320] Gordana Dodig-Crnkovic. "Cognition as Morphological/Morphogenetic Embodied Computation In Vivo". In: *Entropy (Basel, Switzerland)* 24.11 (2022), p. 1576. DOI: 10.3390/e24111576
Cited on page 125

[321] Youri Timsit and Sergeant-Perthuis Grégoire. "Towards the Idea of Molecular Brains". In: *International Journal of Molecular Sciences* 22.21 (2021), p. 11868. DOI: 10.3390/ijms222111868
Cited on pages 125, 126

[322] Yarden Katz, Michael Springer, and Walter Fontana. *Embodying Probabilistic Inference in Biochemical Circuits*. 2018. DOI: 10.48550/arXiv.1806.10161. Comment: 11 figures
Cited on page 125

[323] Péter Csermely, Nina Kunsic, Péter Mendik, Márk Kerestély, Teodóra Faragó, Dániel V. Veres, and Péter Tompa. "Learning of Signaling Networks: Molecular Mechanisms". In: *Trends in Biochemical Sciences* 45.4 (2020), pp. 284–294. DOI: 10.1016/j.tibs.2019.12.005
Cited on pages 125, 127

[324] Dávid M. Gyurkó, Dániel V. Veres, Dezső Módos, Katalin Lenti, Tamás Korcsmáros, and Peter Csermely. "Adaptation and Learning of Molecular Networks as a Description of Cancer Development at the Systems-Level: Potential Use in Anti-Cancer Therapies". In: *Seminars in Cancer Biology*. Cancer-Related Networks: A Help to Understand, Predict and Change Malignant Transformation 23.4 (2013), pp. 262–269. DOI: 10.1016/j.semcancer.2013.06.005
Cited on page 125

[325] Chris Fields and Michael Levin. "Competency in Navigating Arbitrary Spaces as an Invariant for Analyzing Cognition in Diverse Embodiments". In: *Entropy* 24.6 (2022), p. 819. DOI: 10.3390/e24060819
Cited on pages 125, 127

[326] Richard Watson, C. L. Buckley, Rob Mills, and Adam Davies. "Associative Memory in Gene Regulation Networks". In: ed. by Harold Fellerman, Mark Dörr, Martin M. Hanczyc, Lone Ladegaard Laursen, Sarah Maurer, Daniel Merkle, Pierre-Alain Monnard, Kasper Stoy, and Steen Rasmussen. MIT Press, 2010, pp. 659–666
Cited on page 125

[327] Juanita Mathews, Alan (Jaelyn) Chang, Liam Devlin, and Michael Levin. "Cellular Signaling Pathways as Plastic, Proto-Cognitive Systems: Implications for Biomedicine". In: *Patterns* 4.5 (2023), p. 100737. DOI: 10.1016/j.patter.2023.100737
Cited on page 125

[328] Eric Lagasse and Michael Levin. "Future Medicine: From Molecular Pathways to the Collective Intelligence of the Body". In: *Trends in Molecular Medicine* 29.9 (2023), pp. 687–710. DOI: 10.1016/j.molmed.2023.06.007
Cited on page 125

[329] Kathleen T. Krist, Ayusman Sen, and W. G. Noid. "A Simple Theory for Molecular Chemotaxis Driven by Specific Binding Interactions". In: *The Journal of Chemical Physics* 155.16 (2021), p. 164902. DOI: 10.1063/5.0061376
Cited on page 126

[330] Jitka Čejková, Taisuke Banno, Martin M. Hanczyc, and František Štěpánek. "Droplets As Liquid Robots". In: *Artificial Life* 23.4 (2017), pp. 528–549. DOI: 10.1162/ARTL_a_00243
Cited on page 126

[331] Martin M. Hanczyc, Filippo Caschera, and Steen Rasmussen. "Models of Minimal Physical Intelligence". In: *Procedia Computer Science*. Proceedings of the 2nd European Future Technologies Conference and Exhibition 2011 (FET 11) 7 (2011), pp. 275–277. DOI: 10.1016/j.procs.2011.09.058
Cited on page 126

[332] Arturo Rosenblueth, Norbert Wiener, and Julian Bigelow. "Behavior, Purpose and Teleology". In: *Philosophy of Science* 10.1 (1943), pp. 18–24. DOI: 10.1086/286788
Cited on page 126

[333] Joshua Bongard and Michael Levin. "Living Things Are Not (20th Century) Machines: Updating Mechanism Metaphors in Light of the Modern Science of Machine Behavior". In: *Frontiers in Ecology and Evolution* 9 (2021)
Cited on page 126

[334] Pamela Lyon. "The Biogenic Approach to Cognition". In: *Cognitive Processing* 7.1 (2006), pp. 11–29. DOI: 10.1007/s10339-005-0016-8
Cited on page 126

[335] Xabier Barandiaran and Alvaro Moreno. "On What Makes Certain Dynamical Systems Cognitive: A Minimally Cognitive Organization Program". In: *Adaptive Behavior* 14.2 (2006), pp. 171–185. DOI: 10.1177/105971230601400208
Cited on page 126

[336] Franco di Primio, Bernd S Müller, and Joseph W Lengeler. "Minimal Cognition in Unicellular Organisms". In: *From Animals to Animats* (2000), pp. 3–12
Cited on page 126

[337] Patrick McGivern. "Active Materials: Minimal Models of Cognition?" In: *Adaptive Behavior* 28.6 (2020), pp. 441–451. DOI: 10.1177/1059712319891742
Cited on page 126

[338] Michael Levin. "Darwin's Agential Materials: Evolutionary Implications of Multiscale Competency in Developmental Biology". In: *Cellular and Molecular Life Sciences* 80.6 (2023), p. 142. DOI: 10.1007/s00018-023-04790-z
Cited on pages 126, 146

[339] David J. Wong, Dimitry S.A. Nuyten, Aviv Regev, Meihong Lin, Adam S. Adler, Eran Segal, Marc J. van de Vijver, and Howard Y. Chang. "Revealing Targeted Therapy for Human Cancer by Gene Module Maps". In: *Cancer Research* 68.2 (2008), pp. 369–378. DOI: 10.1158/0008-5472.CAN-07-0382
Cited on page 126

[340] T. Jake Samuel, Ryan P. Rosenberry, Seungyong Lee, and Zui Pan. "Correcting Calcium Dysregulation in Chronic Heart Failure Using SERCA2a Gene Therapy". In: *International Journal of Molecular Sciences* 19.4 (2018), p. 1086. DOI: 10.3390/ijms19041086
Cited on page 126

[341] Rafał Krzysztoń, Yiming Wan, Julia Petreczky, and Gábor Balázsi. "Gene-Circuit Therapy on the Horizon: Synthetic Biology Tools for Engineered Therapeutics". In: *Acta biochimica Polonica* 68.3 (2021), pp. 377–383. DOI: 10.18388/abp.2020_5744
Cited on page 126

[342] Christopher Baum. "Insertional Mutagenesis in Gene Therapy and Stem Cell Biology". In: *Current Opinion in Hematology* 14.4 (2007), p. 337. DOI: 10.1097/MOH.0b013e3281900f01
Cited on page 126

[343] Daniel Lobo, Mauricio Solano, George A. Bubenik, and Michael Levin. "A Linear-Encoding Model Explains the Variability of the Target Morphology in Regeneration". In: *Journal of The Royal Society Interface* 11.92 (2014), p. 20130918. DOI: 10.1098/rsif.2013.0918
Cited on page 126

[344] Philipp Städter, Yannik Schälte, Leonard Schmiester, Jan Hasenauer, and Paul L. Stapor. "Benchmarking of Numerical Integration Methods for ODE Models of Biological Systems". In: *Scientific Reports* 11.1 (2021), p. 2696. DOI: 10.1038/s41598-021-82196-2
Cited on pages 127, 174

[345] Brian P. Ingalls. "A Frequency Domain Approach to Sensitivity Analysis of Biochemical Networks". In: *The Journal of Physical Chemistry B* 108.3 (2004), pp. 1143–1152. DOI: 10.1021/jp036567u
Cited on pages 127, 145

[346] Brian Ingalls. "Sensitivity Analysis: From Model Parameters to System Behaviour". In: *Essays in Biochemistry* 45 (2008). Ed. by Olaf Wolkenhauer, Peter Wellstead, and Kwang-Hyun Cho, pp. 177–194. DOI: 10.1042/bse0450177
Cited on pages 127, 145

[347] Alexandre Donzé, Gilles Clermont, and Christopher J. Langmead. "Parameter Synthesis in Nonlinear Dynamical Systems: Application to Systems Biology". In: *Journal of Computational Biology: A Journal of Computational Molecular Cell Biology* 17.3 (2010), pp. 325–336. DOI: 10.1089/cmb.2009.0172
Cited on page 127

[348] Thao Dang, Colas Le Guernic, and Oded Maler. "Computing Reachable States for Nonlinear Biological Models". In: *Theoretical Computer Science* 412.21 (2011), pp. 2095–2107. DOI: 10.1016/j.tcs.2011.01.014
Cited on pages 127, 145

[349] Alexandre Donzé, Eric Fanchon, Lucie Martine Gattepaille, Oded Maler, and Philippe Tracqui. "Robustness Analysis and Behavior Discrimination in Enzymatic Reaction Networks". In: *PLOS ONE* 6.9 (2011), e24246. DOI: 10.1371/journal.pone.0024246
Cited on pages 127, 128, 136

[350] Jordan Rozum and Réka Albert. "Leveraging Network Structure in Nonlinear Control". In: *npj Systems Biology and Applications* 8.1 (2022), pp. 1–8. DOI: 10.1038/s41540-022-00249-2
Cited on page 127

[351] Steven Nathaniel Steinway, Jorge Gomez Tejeda Zañudo, Paul J Michel, David J Feith, Thomas P Loughran, and Reka Albert. "Combinatorial Interventions Inhibit TGFβ-driven epithelial-to-mesenchymal transition and support hybrid cellular phenotypes". In: *NPJ systems biology and applications* 1.1 (2015), pp. 1–12
Cited on pages 127, 129

[352] Jorge G. T. Zañudo and Réka Albert. "Cell Fate Reprogramming by Control of Intracellular Network Dynamics". In: *PLOS Computational Biology* 11.4 (2015), e1004193. DOI: 10.1371/journal.pcbi.1004193
Cited on page 127

[353] Jorge Gomez Tejeda Zañudo, Gang Yang, and Réka Albert. "Structure-Based Control of Complex Networks with Nonlinear Dynamics". In: *Proceedings of the National Academy of Sciences* 114.28 (2017), pp. 7234–7239. DOI: 10.1073/pnas.1617387114
Cited on page 127

[354] Laura Cifuentes Fontanals, Elisa Tonello, and Heike Siebert. "Control Strategy Identification via Trap Spaces in Boolean Networks". In: *Computational Methods in Systems Biology*. Ed. by Alessandro Abate, Tatjana Petrov, and Verena Wolf. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2020, pp. 159–175. DOI: 10.1007/978-3-030-60327-4_9
Cited on page 127

[355] David Murrugarra, Alan Veliz-Cuba, Boris Aguilar, and Reinhard Laubenbacher. "Identification of Control Targets in Boolean Molecular Network Models via Computational Algebra". In: *BMC Systems Biology* 10.1 (2016), p. 94. DOI: 10.1186/s12918-016-0332-x
Cited on page 127

[356] Sang-Mok Choo, Byunghyun Ban, Jae Il Joo, and Kwang-Hyun Cho. "The Phenotype Control Kernel of a Biomolecular Regulatory Network". In: *BMC Systems Biology* 12.1 (2018), p. 49. DOI: 10.1186/s12918-018-0576-8
Cited on page 127

[357] Sang-Mok Choo, Sang-Min Park, and Kwang-Hyun Cho. "Minimal Intervening Control of Biomolecular Networks Leading to a Desired Cellular State". In: *Scientific Reports* 9.1 (2019), p. 13124. DOI: 10.1038/s41598-019-49571-6
Cited on page 127

[358] S. R. Paladugu, V. Chickarmane, A. Deckard, J. P. Frumkin, M. McCormack, and H. M. Sauro. "In Silico Evolution of Functional Modules in Biochemical Networks". In: *IEE Proceedings - Systems Biology* 153.4 (2006), pp. 223–235. DOI: 10.1049/ip-syb:20050096
Cited on page 127

[359] Paul François. "Evolving Phenotypic Networks in Silico". In: *Seminars in Cell & Developmental Biology*. Regulated Necrosis & Modeling Developmental Signaling Pathways & Development of Connective Maps in the Brain 35 (2014), pp. 90–97. DOI: 10.1016/j.semcdb.2014.06.012
Cited on page 127

[360] Nasimul Noman, Taku Monjo, Pablo Moscato, and Hitoshi Iba. "Evolving Robust Gene Regulatory Networks". In: *PLOS ONE* 10.1 (2015), e0116258. DOI: 10.1371/journal.pone.0116258
Cited on pages 127, 128, 136

[361] Tom W. Hiscock. "Adapting Machine-Learning Algorithms to Design Gene Circuits". In: *BMC bioinformatics* 20.1 (2019), p. 214. DOI: 10.1186/s12859-019-2788-3
Cited on pages 127, 144, 252

[362] Jingxiang Shen, Feng Liu, Yuhai Tu, and Chao Tang. "Finding Gene Network Topologies for given Biological Function with Recurrent Neural Network". In: *Nature Communications* 12.1 (2021), p. 3125. DOI: 10.1038/s41467-021-23420-5
Cited on page 127

[363] Diogo M. Camacho, Katherine M. Collins, Rani K. Powers, James C. Costello, and James J. Collins. "Next-Generation Machine Learning for Biological Networks". In: *Cell* 173.7 (2018), pp. 1581–1592. DOI: 10.1016/j.cell.2018.05.015
Cited on pages 127, 171

[364] Michael Jeffrey Volk, Ismini Lourentzou, Shekhar Mishra, Lam Tung Vo, Chengxiang Zhai, and Huimin Zhao. "Biosystems Design by Machine Learning". In: *ACS Synthetic Biology* 9.7 (2020), pp. 1514–1533. DOI: 10.1021/acssynbio.0c00129
Cited on pages 127, 143

[365] Hiroaki Kitano. "A Robustness-Based Approach to Systems-Oriented Drug Design". In: *Nature Reviews Drug Discovery* 6.3 (2007), pp. 202–210. DOI: 10.1038/nrd2195
Cited on page 127

[366] Surama Biswas, Santosh Manicka, Erik Hoel, and Michael Levin. "Gene Regulatory Networks Exhibit Several Kinds of Memory: Quantification of Memory in Biological and Random Transcriptional Networks". In: *iScience* 24.3 (2021), p. 102131. DOI: 10.1016/j.isci.2021.102131
Cited on page 127

[367] Surama Biswas, Wesley Clawson, and Michael Levin. "Learning in Transcriptional Network Models: Computational Discovery of Pathway-Level Memory and Effective Interventions". In: *International Journal of Molecular Sciences* 24.1 (2023), p. 285. DOI: 10.3390/ijms24010285
Cited on pages 127, 141, 247

[368] Sébastien Forestier, Rémy Portelas, Yoan Mollard, and Pierre-Yves Oudeyer. *Intrinsically Motivated Goal Exploration Processes with Automatic Curriculum Learning*. 2022. Comment: Accepted at JMLR 2022
Cited on pages 128, 132, 145

[369] Hiroaki Kitano. "Towards a Theory of Biological Robustness". In: *Molecular Systems Biology* 3.1 (2007), p. 137. DOI: 10.1038/msb4100179
Cited on pages 128, 136, 250

[370] Marie-Anne Félix and Michalis Barkoulas. "Pervasive Robustness in Biological Systems". In: *Nature Reviews Genetics* 16.8 (2015), pp. 483–496. DOI: 10.1038/nrg3949
Cited on pages 128, 136, 250

[371] Nicholas T. Ingolia. "Topology and Robustness in the Drosophila Segment Polarity Network". In: *PLOS Biology* 2.6 (2004), e123. DOI: 10.1371/journal.pbio.0020123
Cited on pages 128, 136

[372] Wenzhe Ma, Luhua Lai, Qi Ouyang, and Chao Tang. "Robustness and Modular Design of the Drosophila Segment Polarity Network". In: *Molecular Systems Biology* 2.1 (2006), p. 70. DOI: 10.1038/msb4100111
Cited on pages 128, 136

[373] David Deutscher, Isaac Meilijson, Martin Kupiec, and Eytan Ruppin. "Multiple Knockout Analysis of Genetic Robustness in the Yeast Metabolic Network". In: *Nature Genetics* 38.9 (2006), pp. 993–998. DOI: 10.1038/ng1856
Cited on page 128

[374] George von Dassow, Eli Meir, Edwin M. Munro, and Garrett M. Odell. "The Segment Polarity Network Is a Robust Developmental Module". In: *Nature* 406.6792 (2000), pp. 188–192. DOI: 10.1038/35018085
Cited on pages 128, 136

[375] Cho Kwang-Hyun, Shin Sung-Young, Kim Hyun-Woo, Olaf Wolkenhauer, Brian McFerran, and Walter Kolch. "Mathematical Modeling of the Influence of RKIP on the ERK Signaling Pathway". In: *Computational Methods in Systems Biology*. Ed. by Gerhard Goos, Juris Hartmanis, Jan Van Leeuwen, and Corrado Priami. Vol. 2602. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 127–141. DOI: 10.1007/3-540-36481-1_11
Cited on pages 131, 132, 135, 139, 141, 142, 251, 256

[376] Leland McInnes, John Healy, and Steve Astels. "Hdbscan: Hierarchical Density Based Clustering". In: *The Journal of Open Source Software* 2.11 (2017), p. 205
Cited on pages 135, 251

[377] Charles C. Bell and Omer Gilan. "Principles and Mechanisms of Non-Genetic Resistance in Cancer". In: *British Journal of Cancer* 122.4 (2020), pp. 465–472. DOI: 10.1038/s41416-019-0648-6
Cited on pages 136, 138, 141

[378] Aurélien Rizk, Gregory Batt, François Fages, and Sylvain Soliman. "A General Computational Method for Robustness Analysis with Applications to Synthetic Gene Networks". In: *Bioinformatics* 25.12 (2009), pp. i169–i178. DOI: 10.1093/bioinformatics/btp200
Cited on page 136

[379] Charles Walcott. "Pigeon Homing: Observations, Experiments and Confusions". In: *Journal of Experimental Biology* 199.1 (1996), pp. 21–27. DOI: 10.1242/jeb.199.1.21
Cited on page 136

[380] Paolo Luschi, Susanne Åkesson, Annette C. Broderick, Fiona Glen, Brendan J. Godley, Floriano Papi, and Graeme C. Hays. "Testing the Navigational Abilities of Ocean Migrants: Displacement Experiments on Green Sea Turtles (Chelonia Mydas)". In: *Behavioral Ecology and Sociobiology* 50.6 (2001), pp. 528–534
Cited on page 136

[381] Sonja Bisch-Knaden and Rüdiger Wehner. "Egocentric Information Helps Desert Ants to Navigate around Familiar Obstacles". In: *Journal of Experimental Biology* 204.24 (2001), pp. 4177–4184. DOI: 10.1242/jeb.204.24.4177
Cited on page 136

[382] Charles I Abramson et al. *A Primer of Invertebrate Learning: The Behavioral Perspective.* American Psychological Association, 1994
Cited on page 136

[383] Harish Venkatachalapathy, Samira M. Azarin, and Casim A. Sarkar. "Trajectory-Based Energy Landscapes of Gene Regulatory Networks". In: *Biophysical Journal* 120.4 (2021), pp. 687–698. DOI: 10.1016/j.bpj.2020.11.2279
Cited on pages 138, 139, 251

[384] Chunhe Li and Jin Wang. "Landscape and Flux Reveal a New Global View and Physical Quantification of Mammalian Cell Cycle". In: *Proceedings of the National Academy of Sciences* 111.39 (2014), pp. 14130–14135. DOI: 10.1073/pnas.1408628111
Cited on page 138

[385] Chunhe Li and Jin Wang. "Quantifying Cell Fate Decisions for Differentiation and Reprogramming of a Human Stem Cell Network: Landscape and Biological Paths". In: *PLOS Computational Biology* 9.8 (2013), e1003165. DOI: 10.1371/journal.pcbi.1003165
Cited on page 138

[386] Han Chu Lee, Bo Tian, John M. Sedivy, Jack R. Wands, and Miran Kim. "Loss of Raf Kinase Inhibitor Protein Promotes Cell Proliferation and Migration of Human Hepatoma Cells". In: *Gastroenterology* 131.4 (2006), pp. 1208–1217. DOI: 10.1053/j.gastro.2006.07.012
Cited on pages 139, 142

[387] John Reinitz and David H. Sharp. "Mechanism of Eve Stripe Formation". In: *Mechanisms of Development* 49.1 (1995), pp. 133–158. DOI: 10.1016/0925-4773(94)00310-J
Cited on pages 141, 248

[388] Johannes Jaeger et al. "Dynamical Analysis of Regulatory Interactions in the Gap Gene System of Drosophila Melanogaster". In: *Genetics* 167.4 (2004), pp. 1721–1737. DOI: 10.1534/genetics.104.027334
Cited on page 141

[389] James Cotterell and James Sharpe. "An Atlas of Gene Regulatory Networks Reveals Multiple Three-Gene Mechanisms for Interpreting Morphogen Gradients". In: *Molecular Systems Biology* 6.1 (2010), p. 425. DOI: 10.1038/msb.2010.74
Cited on page 141

[390] Evan J. Molinelli et al. "Perturbation Biology: Inferring Signaling Networks in Cellular Systems". In: *PLOS Computational Biology* 9.12 (2013), e1003290. DOI: 10.1371/journal.pcbi.1003290
Cited on page 141

[391] Jordi Vallverdú, Oscar Castro, Richard Mayne, Max Talanov, Michael Levin, Frantisek Baluška, Yukio Gunji, Audrey Dussutour, Hector Zenil, and Andrew Adamatzky. "Slime Mould: The Fundamental Mechanisms of Biological Cognition". In: *Biosystems* 165 (2018), pp. 57–70. DOI: 10.1016/j.biosystems.2017.12.011
Cited on page 141

[392] Madeleine Beekman and Tanya Latty. "Brainless but Multi-Headed: Decision Making by the Acellular Slime Mould Physarum Polycephalum". In: *Journal of Molecular Biology*. Cooperative Behaviour in Microbial Communities 427.23 (2015), pp. 3734–3743. DOI: 10.1016/j.jmb.2015.07.007
Cited on page 141

[393] Tetsu Saigusa, Atsushi Tero, Toshiyuki Nakagaki, and Yoshiki Kuramoto. "Amoebae Anticipate Periodic Events". In: *Physical Review Letters* 100.1 (2008), p. 018101. DOI: 10.1103/PhysRevLett.100.018101
Cited on page 141

[394] Toshiyuki Nakagaki and Robert D. Guy. "Intelligent Behaviors of Amoeboid Movement Based on Complex Dynamics of Soft Matter". In: *Soft Matter* 4.1 (2008), pp. 57–67. DOI: 10.1039/B706317M
Cited on page 141

[395] Amir Pandi et al. "A Versatile Active Learning Workflow for Optimization of Genetic and Metabolic Networks". In: *Nature Communications* 13.1 (2022), p. 3876. DOI: 10.1038/s41467-022-31245-z
Cited on page 145

[396] Ashley R. G. Libby, Demarcus Briers, Iman Haghighi, David A. Joy, Bruce R. Conklin, Calin Belta, and Todd C. McDevitt. "Automated Design of Pluripotent Stem Cell Self-Organization". In: *Cell Systems* 9.5 (2019), 483–495.e10. DOI: 10.1016/j.cels.2019.10.008
Cited on page 146

[397] Alexis Pietak and Michael Levin. "Exploring Instructive Physiological Signaling with the Bioelectric Tissue Simulation Engine". In: *Frontiers in Bioengineering and Biotechnology* 4 (2016)
Cited on pages 146, 150, 151, 153

[398] Aneta Koseska and Philippe IH Bastiaens. "Cell Signaling as a Cognitive Process". In: *The EMBO Journal* 36.5 (2017), pp. 568–582. DOI: 10.15252/embj.201695383
Cited on page 146

[399] František Baluška, Arthur S. Reber, and William B. Miller. "Cellular Sentience as the Primary Source of Biological Order and Evolution". In: *Biosystems* 218 (2022), p. 104694. DOI: 10.1016/j.biosystems.2022.104694
Cited on page 146

[400] František Baluška, William B Miller, and Arthur S Reber. "Cellular and Evolutionary Perspectives on Organismal Cognition: From Unicellular to Multicellular Organisms". In: *Biological Journal of the Linnean Society* 139.4 (2023), pp. 503–513. DOI: 10.1093/biolinnean/blac005
Cited on page 146

[401] Arthur S. Reber and František Baluška. "Cognition in Some Surprising Places". In: *Biochemical and Biophysical Research Communications*. Rethinking Cognition: From Animal to Minimal 564 (2021), pp. 150–157. DOI: 10.1016/j.bbrc.2020.08.115
Cited on page 146

[402] František Baluška and Arthur S. Reber. "Cellular and Organismal Agency - Not Based on Genes: A Comment on Baverstock". In: *Progress in Biophysics and Molecular Biology* 167 (2021), pp. 161–162. DOI: 10.1016/j.pbiomolbio.2021.11.001
Cited on page 146

[403] Anne Bernheim-Groswasser, Nir S. Gov, Samuel A. Safran, and Shelly Tzlil. "Living Matter: Mesoscopic Active Materials". In: *Advanced Materials* 30.41 (2018), p. 1707028. DOI: 10.1002/adma.201707028
Cited on page 146

[404] Bertrand Guillotin et al. "Laser Assisted Bioprinting of Engineered Tissue with High Cell Density and Microscale Organization". In: *Biomaterials* 31.28 (2010), pp. 7250–7256. DOI: 10.1016/j.biomaterials.2010.05.055
Cited on pages 149, 159

[405] Virginie Keriquel et al. "In Situ Printing of Mesenchymal Stromal Cells, by Laser-Assisted Bioprinting, for in Vivo Bone Regeneration Applications". In: *Scientific Reports* 7.1 (2017), p. 1778. DOI: 10.1038/s41598-017-01914-x
Cited on page 149

[406] Maxime Abellan Lopez et al. "In Vivo Efficacy Proof of Concept of a Large-Size Bioprinted Dermo-Epidermal Substitute for Permanent Wound Coverage". In: *Frontiers in Bioengineering and Biotechnology* 11 (2023)
Cited on pages 149, 161

[407] Alexis Pietak and Michael Levin. "Bioelectric Gene and Reaction Networks: Computational Modelling of Genetic, Biochemical and Bioelectrical Dynamics in Pattern Regulation". In: *Journal of The Royal Society Interface* 14.134 (2017), p. 20170425. DOI: 10.1098/rsif.2017.0425
Cited on pages 150, 152

[408] A. Douillet, C. Douillet, M. Garcia, M. Nicodem, F. Guillemot, and P. Ballet. "A Computational Framework to Simulate Bio-Printed Cells and Extracellular Matrix Mechanobiochemical Interactions". In: *Computer Methods in Biomechanics and Biomedical Engineering* 22.sup1 (2019), S338–S339. DOI: 10.1080/10255842.2020.1714935
Cited on pages 150, 154, 155

[409] Pascal Ballet. "SimCells, an Advanced Software for Multicellular Modeling Application to Tumoral and Blood Vessel Co-Development". 2018. working paper or preprint
Cited on pages 150, 154

[410] Dany S. Adams and Michael Levin. "General Principles for Measuring Resting Membrane Potential and Ion Concentration Using Fluorescent Bioelectricity Reporters". In: *Cold Spring Harbor Protocols* 2012.4 (2012), pp. 385–397. DOI: 10.1101/pdb.top067710
Cited on page 151

[411] Michael Levin and Christopher J. Martyniuk. "The Bioelectric Code: An Ancient Computational Medium for Dynamic Control of Growth and Form". In: *Biosystems*. Code Biology 164 (2018), pp. 76–93. DOI: 10.1016/j.biosystems.2017.08.009

[412] Michael Levin. "Bioelectric Signaling: Reprogrammable Circuits Underlying Embryogenesis, Regeneration, and Cancer". In: *Cell* 184.8 (2021), pp. 1971–1989. DOI: 10.1016/j.cell.2021.02.034
Cited on page 151

[413] Javier Cervera, Alexis Pietak, Michael Levin, and Salvador Mafe. "Bioelectrical Coupling in Multicellular Domains Regulated by Gap Junctions: A Conceptual Approach". In: *Bioelectrochemistry (Amsterdam, Netherlands)* 123 (2018), pp. 45–61. DOI: 10.1016/j.bioelechem.2018.04.013
Cited on page 151

[414] Vaibhav P. Pai, Alexis Pietak, Valerie Willocq, Bin Ye, Nian-Qing Shi, and Michael Levin. "HCN2 Rescues Brain Defects by Enforcing Endogenous Voltage Pre-Patterns". In: *Nature Communications* 9.1 (2018), p. 998. DOI: 10.1038/s41467-018-03334-5
Cited on page 151

[415] Alexis Pietak, Johanna Bischof, Joshua LaPalme, Junji Morokuma, and Michael Levin. "Neural Control of Body-Plan Axis in Regenerating Planaria". In: *PLOS Computational Biology* 15.4 (2019), e1006904. DOI: 10.1371/journal.pcbi.1006904
Cited on page 151

[416] Hananel Hazan and Michael Levin. "Exploring The Behavior of Bioelectric Circuits Using Evolution Heuristic Search". In: *Bioelectricity* 4.4 (2022), pp. 207–227
Cited on pages 152, 153

[417] Camille Douillet, Marc Nicodeme, Loïc Hermant, Vanessa Bergeron, Fabien Guillemot, Jean-Christophe Fricain, Hugo Oliveira, and Mikael Garcia. "From Local to Global Matrix Organization by Fibroblasts: A 4D Laser-Assisted Bioprinting Approach". In: *Biofabrication* 14.2 (2022). DOI: 10.1088/1758-5090/ac40ed
Cited on pages 154, 157, 160, 161

[418] Arthur Douillet. "Simulation de La Morphogenèse de Tissus Bio-Imprimés". These de Doctorat. Brest, 2020. Sous la direction de Pascal Ballet et de Fabien Guillemot. Soutenue le 17-09-2020,à Brest , dans le cadre de École doctorale Biologie-Santé (Nantes) , en partenariat avec Laboratoire de traitement de l'information médicale (Brest, Finistère) (laboratoire) .
Cited on page 155

[419] Kenneth O Stanley, David B D'Ambrosio, and Jason Gauci. "A Hypercube-Based Encoding for Evolving Large-Scale Neural Networks". In: *Artificial life* 15.2 (2009), pp. 185–212
Cited on pages 158, 180

[420] Nick Cheney, Robert MacCurdy, Jeff Clune, and Hod Lipson. "Unshackling Evolution: Evolving Soft Robots with Multiple Materials and a Powerful Generative Encoding". In: *ACM SIGEVOlution* 7.1 (2014), pp. 11–23
Cited on page 158

[421] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. "Learning Representations and Generative Models for 3d Point Clouds". In: *International Conference on Machine Learning*. PMLR, 2018, pp. 40–49
Cited on page 158

[422] Camille Douillet. "La Bio-Impression Assistée Par Laser de Fibroblastes : Contrôler l'organisation Du Collagène Pour l'ingénierie Tissulaire de La Peau". These de Doctorat. Bordeaux, 2021. Sous la direction de Jean-Christophe Fricain. Soutenue le 11-06-2021,à Bordeaux , dans le cadre de École doctorale Sciences de la vie et de la santé (Bordeaux) , en partenariat avec Bioingénierie tissulaire (laboratoire) .
Cited on pages 159, 161

[423] Jipeng Li, Mingjiao Chen, Xianqun Fan, and Huifang Zhou. "Recent Advances in Bioprinting Techniques: Approaches, Applications and Future Prospects". In: *Journal of Translational Medicine* 14.1 (2016), p. 271. DOI: 10.1186/s12967-016-1028-0
Cited on page 159

[424] Ulrike Baranyi, Birgitta Winter, Alfred Gugerell, Balazs Hegedus, Christine Brostjan, Günther Laufer, and Barbara Messner. "Primary Human Fibroblasts in Culture Switch to a Myofibroblast-like Phenotype Independently of TGF Beta". In: *Cells* 8.7 (2019), p. 721
Cited on page 160

[425] Allen P Liu, Ovijit Chaudhuri, and Sapun H Parekh. "New Advances in Probing Cell–Extracellular Matrix Interactions". In: *Integrative Biology* 9.5 (2017), pp. 383–405
Cited on page 160

[426] Altug Ozcelikkale, J Craig Dutton, Frederick Grinnell, and Bumsoo Han. "Effects of Dynamic Matrix Remodelling on En Masse Migration of Fibroblasts on Collagen Matrices". In: *Journal of The Royal Society Interface* 14.135 (2017), p. 20170287
Cited on page 160

[427] Frederick Grinnell. "Fibroblast Biology in Three-Dimensional Collagen Matrices". In: *Trends in cell biology* 13.5 (2003), pp. 264–269
Cited on page 160

[428] Allen P Liu. "Biophysical Tools for Cellular and Subcellular Mechanical Actuation of Cell Signaling". In: *Biophysical Journal* 111.6 (2016), pp. 1112–1118
Cited on page 160

[429] Ju An Park, Hwa-Rim Lee, Seung-Yeol Park, and Sungjune Jung. "Self-Organization of Fibroblast-Laden 3D Collagen Microstructures from Inkjet-Printed Cell Patterns". In: *Advanced biosystems* 4.5 (2020), p. 1900280
Cited on pages 160, 162, 163

[430] Yee Han Tee, Tom Shemesh, Visalatchi Thiagarajan, Rizal Fajar Hariadi, Karen L Anderson, Christopher Page, Niels Volkmann, Dorit Hanein, Sivaraj Sivaramakrishnan, Michael M Kozlov, et al. "Cellular Chirality Arising from the Self-Organization of the Actin Cytoskeleton". In: *Nature cell biology* 17.4 (2015), pp. 445–457
Cited on page 160

[431] John L Tan, Joe Tien, Dana M Pirone, Darren S Gray, Kiran Bhadriraju, and Christopher S Chen. "Cells Lying on a Bed of Microneedles: An Approach to Isolate Mechanical Force". In: *Proceedings of the National Academy of Sciences* 100.4 (2003), pp. 1484–1489
Cited on page 160

[432] Carlos Matellan and Armando E del Río Hernández. "Engineering the Cellular Mechanical Microenvironment–from Bulk Mechanics to the Nanoscale". In: *Journal of cell science* 132.9 (2019), jcs229013
Cited on page 160

[433] Huizhi Chen, Yuan Siang Lui, Zhen Wei Tan, Justin Yin Hao Lee, Nguan Soon Tan, and Lay Poh Tan. "Migration and Phenotype Control of Human Dermal Fibroblasts by Electrospun Fibrous Substrates". In: *Advanced healthcare materials* 8.9 (2019), p. 1801378
Cited on page 160

[434] Megan E Smithmyer, Samantha E Cassel, and April M Kloxin. "Bridging 2D and 3D Culture: Probing Impact of Extracellular Environment on Fibroblast Activation in Layered Hydrogels". In: *AIChE Journal* 65.12 (2019), e16837
Cited on page 160

[435] Jia Shi, Jinchun Song, Bin Song, and Wen F Lu. "Multi-Objective Optimization Design through Machine Learning for Drop-on-Demand Bioprinting". In: *Engineering* 5.3 (2019), pp. 586–593
Cited on page 161

[436] Kalani Ruberu, Manisha Senadeera, Santu Rana, Sunil Gupta, Johnson Chung, Zhilian Yue, Svetha Venkatesh, and Gordon Wallace. "Coupling Machine Learning with 3D Bioprinting to Fast Track Optimisation of Extrusion Printing". In: *Applied Materials Today* 22 (2021), p. 100914
Cited on page 161

[437] Chunling Yu and Jingchao Jiang. "A Perspective on Using Machine Learning in 3D Bioprinting". In: *International Journal of Bioprinting* 6.1 (2020)
Cited on page 161

[438] Tânia Baltazar et al. "Three Dimensional Bioprinting of a Vascularized and Perfusable Skin Graft Using Human Keratinocytes, Fibroblasts, Pericytes, and Endothelial Cells". In: *Tissue Engineering. Part A* 26.5-6 (2020), pp. 227–238. DOI: 10.1089/ten.tea.2019.0201
Cited on pages 162, 163

[439] Hironobu Takahashi, Tatsuya Shimizu, Masamichi Nakayama, Masayuki Yamato, and Teruo Okano. "The Use of Anisotropic Cell Sheets to Control Orientation during the Self-Organization of 3D Muscle Tissue". In: *Biomaterials* 34.30 (2013), pp. 7372–7380. DOI: 10.1016/j.biomaterials.2013.06.033
Cited on page 162

[440] M. G. Tonnesen, X. Feng, and R. A. Clark. "Angiogenesis in Wound Healing". In: *The Journal of Investigative Dermatology. Symposium Proceedings* 5.1 (2000), pp. 40–46. DOI: 10.1046/j.1087-0024.2000.00014.x
Cited on page 163

[441] Hiroaki Kitano. "Systems Biology: A Brief Overview". In: *Science* 295.5560 (2002), pp. 1662–1664. DOI: 10.1126/science.1069492
Cited on page 171

[442] M. Hucka et al. "The Systems Biology Markup Language (SBML): A Medium for Representation and Exchange of Biochemical Network Models". In: *Bioinformatics* 19.4 (2003), pp. 524–531. DOI: 10.1093/bioinformatics/btg015
Cited on page 171

[443] Michael Hucka et al. "The Systems Biology Markup Language (SBML): Language Specification for Level 3 Version 2 Core Release 2". In: *Journal of Integrative Bioinformatics* 16.2 (2019), p. 20190021. DOI: 10.1515/jib-2019-0021
Cited on page 171

[444] Giulia Muzio, Leslie O'Bray, and Karsten Borgwardt. "Biological Network Analysis with Deep Learning". In: *Briefings in Bioinformatics* 22.2 (2021), pp. 1515–1530. DOI: 10.1093/bib/bbaa257
Cited on page 171

[445] Mohammed AlQuraishi and Peter K. Sorger. "Differentiable Biology: Using Deep Learning for Biophysics-Based and Data-Driven Modeling of Molecular Mechanisms". In: *Nature Methods* 18.10 (2021), pp. 1169–1180. DOI: 10.1038/s41592-021-01283-4
Cited on page 171

[446] Stefan Hoops, Sven Sahle, Ralph Gauges, Christine Lee, Jürgen Pahle, Natalia Simus, Mudita Singhal, Liang Xu, Pedro Mendes, and Ursula Kummer. "COPASI—a COmplex PAthway SImulator". In: *Bioinformatics* 22.24 (2006), pp. 3067–3074. DOI: 10.1093/bioinformatics/btl485
Cited on page 172

[447] Frank T. Bergmann. *Copasi/Basico: Release 0.48*. 2023
Cited on page 172

[448] Leslie M. Loew and James C. Schaff. "The Virtual Cell: A Software Environment for Computational Cell Biology". In: *Trends in Biotechnology* 19.10 (2001), pp. 401–406. DOI: 10.1016/S0167-7799(01)01740-1
Cited on page 172

[449] Boris M. Slepchenko, James C. Schaff, Ian Macara, and Leslie M. Loew. "Quantitative Cell Biology with the Virtual Cell". In: *Trends in Cell Biology* 13.11 (2003), pp. 570–576. DOI: 10.1016/j.tcb.2003.09.002
Cited on page 172

[450] Akira Funahashi, Mineo Morohashi, Hiroaki Kitano, and Naoki Tanimura. "CellDesigner: A Process Diagram Editor for Gene-Regulatory and Biochemical Networks". In: *BIOSILICO* 1.5 (2003), pp. 159–162. DOI: 10.1016/S1478-5382(03)02370-9
Cited on page 172

[451] Akira Funahashi, Yukiko Matsuoka, Akiya Jouraku, Mineo Morohashi, Norihiro Kikuchi, and Hiroaki Kitano. "CellDesigner 3.5: A Versatile Modeling Tool for Biochemical Networks". In: *Proceedings of the IEEE* 96.8 (2008), pp. 1254–1265. DOI: 10.1109/JPROC.2008.925458
Cited on page 172

[452] Kiri Choi, J. Kyle Medley, Matthias König, Kaylene Stocking, Lucian Smith, Stanley Gu, and Herbert M. Sauro. "Tellurium: An Extensible Python-Based Modeling Environment for Systems and Synthetic Biology". In: *Biosystems* 171 (2018), pp. 74–79. DOI: 10.1016/j.biosystems.2018.07.006
Cited on page 172

[453] J. Kyle Medley, Kiri Choi, Matthias König, Lucian Smith, Stanley Gu, Joseph Hellerstein, Stuart C. Sealfon, and Herbert M. Sauro. "Tellurium Notebooks—An Environment for Reproducible Dynamical Modeling in Systems Biology". In: *PLOS Computational Biology* 14.6 (2018), e1006220. DOI: 10.1371/journal.pcbi.1006220
Cited on page 172

[454] Steve M. Ruggiero and Ashlee N. Ford Versypt. "SBMLtoODEpy: A Software Program for Converting SBML Models into ODE Models in Python". In: *Journal of Open Source Software* 5.53 (2019), p. 1643. DOI: 10.21105/joss.01643
Cited on page 172

[455] Nicolas Rodriguez et al. "The Systems Biology Format Converter". In: *BMC Bioinformatics* 17.1 (2016), p. 154. DOI: 10.1186/s12859-016-1000-2
Cited on page 172

[456] Eric J. Ma and Arkadij Kummer. *Reimplementing Unirep in JAX*. 2020. DOI: 10.1101/2020.05.11.088344
Cited on page 172

[457] Giuseppe Carleo et al. "NetKet: A Machine Learning Toolkit for Many-Body Quantum Systems". In: *SoftwareX* 10 (2019), p. 100311. DOI: 10.1016/j.softx.2019.100311
Cited on page 172

[458] Jean-Eric Campagne, François Lanusse, Joe Zuntz, Alexandre Boucaud, Santiago Casas, Minas Karamanis, David Kirkby, Denise Lanzieri, Yin Li, and Austin Peel. "JAX-COSMO: An End-to-End Differentiable and GPU Accelerated Cosmology Library". In: *The Open Journal of Astrophysics* 6 (2023), 10.21105/astro.2302.05163. DOI: 10.21105/astro.2302.05163
Cited on page 172

[459] Dion Häfner, Roman Nuterman, and Markus Jochum. "Fast, Cheap, and Turbulent—Global Ocean Modeling With GPU Acceleration in Python". In: *Journal of Advances in Modeling Earth Systems* 13.12 (2021), e2021MS002717. DOI: 10.1029/2021MS002717. e2021MS002717 2021MS002717
Cited on page 172

[460] Sean Mann, Eric Fadel, Samuel S. Schoenholz, Ekin D. Cubuk, Steven G. Johnson, and Giuseppe Romano. "∂PV: An End-to-End Differentiable Solar-Cell Simulator". In: *Computer Physics Communications* 272 (2022), p. 108232. DOI: 10.1016/j.cpc.2021.108232
Cited on page 172

[461] Antonio Stanziola, Simon R. Arridge, Ben T. Cox, and Bradley E. Treeby. "J-Wave: An Open-Source Differentiable Wave Simulator". In: *SoftwareX* 22 (2023), p. 101338. DOI: 10.1016/j.softx.2023.101338
Cited on page 172

[462] Deniz A. Bezgin, Aaron B. Buhendwa, and Nikolaus A. Adams. "JAX-Fluids: A Fully-Differentiable High-Order Computational Fluid Dynamics Solver for Compressible Two-Phase Flows". In: *Computer Physics Communications* 282 (2023), p. 108527. DOI: 10.1016/j.cpc.2022.108527
Cited on page 172

[463] B. N. Kholodenko. "Negative Feedback and Ultrasensitivity Can Bring about Oscillations in the Mitogen-Activated Protein Kinase Cascades". In: *European Journal of Biochemistry* 267.6 (2000), pp. 1583–1588. DOI: 10.1046/j.1432-1327.2000.01197.x
Cited on page 173

[464] Patrick Kidger and Cristian Garcia. "Equinox: Neural Networks in JAX via Callable PyTrees and Filtered Transformations". In: *Differentiable Programming workshop at Neural Information Processing Systems 2021* (2021)
Cited on page 173

[465] Igor Babuschkin et al. *The DeepMind JAX Ecosystem*. 2020
Cited on page 173

[466] Kenneth O Stanley and L B Soros. "The Role of Subjectivity in the Evaluation of Open-Endedness". In: *The Second Workshop on Open-Ended Evolution (OEE2), at ALIFE 2016* (2016)
Cited on page 179

[467] Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-Term Memory". In: *Neural computation* 9.8 (1997), pp. 1735–1780
Cited on page 180

[468] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. "Attention Is All You Need". In: *Advances in Neural Information Processing Systems*. Vol. 30. Curran Associates, Inc., 2017
Cited on page 180

[469] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhutdinov. "Unsupervised Learning of Video Representations Using LSTMs". In: *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*. ICML'15. Lille, France: JMLR.org, 2015, pp. 843–852
Cited on page 180

[470] Javier Selva, Anders S. Johansen, Sergio Escalera, Kamal Nasrollahi, Thomas B. Moeslund, and Albert Clapés. "Video Transformers: A Survey". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 01 (2023), pp. 1–20. DOI: 10.1109/TPAMI.2023.3243465
Cited on page 180

[471] Hassan Akbari, Liangzhe Yuan, Rui Qian, Wei-Hong Chuang, Shih-Fu Chang, Yin Cui, and Boqing Gong. "VATT: Transformers for Multimodal Self-Supervised Learning from Raw Video, Audio and Text". In: *Advances in Neural Information Processing Systems*. Vol. 34. Curran Associates, Inc., 2021, pp. 24206–24221
Cited on page 180

[472] Tristan Karch, Yoann Lemesle, Romain Laroche, Clément Moulin-Frier, and Pierre-Yves Oudeyer. *Contrastive Multimodal Learning for Emergence of Graphical Sensory-Motor Communication*. 2023
Cited on page 180

[473] Jessica C. Flack. "Coarse-Graining as a Downward Causation Mechanism". In: *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 375.2109 (2017), p. 20160338. DOI: 10.1098/rsta.2016.0338
Cited on page 181

[474] Olivier Sigaud, Ahmed Akakzia, Hugo Caselles-Dupré, Cédric Colas, Pierre-Yves Oudeyer, and Mohamed Chetouani. "Towards Teachable Autotelic Agents". In: *IEEE Transactions on Cognitive and Developmental Systems* 15.3 (2023), pp. 1070–1084. DOI: 10.1109/TCDS.2022.3231731.
Cited on page 181

[475] Sebastian Risi. "The Future of Artificial Intelligence Is Self-Organizing and Self-Assembling". In: *sebastianrisi.com* (2021)
Cited on page 183

[476] David Ha and Yujin Tang. "Collective Intelligence for Deep Learning: A Survey of Recent Developments". In: *Collective Intelligence* 1.1 (2022), p. 26339137221114874. DOI: 10.1177/26339137221114874
Cited on pages 183, 184

[477] Ephraim S. Bililign, Florencio Balboa Usabiaga, Yehuda A. Ganan, Alexis Poncet, Vishal Soni, Sofia Magkiriadou, Michael J. Shelley, Denis Bartolo, and William T. M. Irvine. "Motile Dislocations Knead Odd Crystals into Whorls". In: *Nature Physics* 18.2 (2022), pp. 212–218. DOI: 10.1038/s41567-021-01429-3
Cited on page 184

[478] Andy Lomas. "Cellular Forms: An Artistic Exploration of Morphogenesis". In: *ACM SIGGRAPH 2014 Studio*. Vancouver Canada: ACM, 2014, pp. 1–1. DOI: 10.1145/2619195.2656282. [TLDR] The aim is to create structures emergently: exploring generic similarities between many different forms in nature rather than recreating any particular organism, revealing universal archetypal forms that can come from growth-like processes rather than top-down externally engineered design.
Cited on page 184

[479] Andy Lomas. "Species Explorer: An Interface for Artistic Exploration of Multi-Dimensional Parameter Spaces". In: *Electronic Visualisation and the Arts*. BCS Learning & Development, 2016. DOI: 10.14236/ewic/EVA2016.23
Cited on page 184

[480] Robin Roussel, Marie-Paule Cani, Jean-Claude Léon, and Niloy J. Mitra. "Exploratory Design of Mechanical Devices with Motion Constraints". In: *Computers & Graphics* 74 (2018), pp. 244–256. DOI: 10.1016/j.cag.2018.05.023
Cited on page 184

[481] Jack Armitage and Thor Magnusson. "Agential Scores: Exploring Emergent, Self-Organising and Entangled Music Notation". In: *Proceedings of the 8th International Conference on Technologies for Music Notation and Representation (Northeastern University, Boston, Massachusetts, USA, 2023)*. 2023
Cited on page 184

[482] I. T. Jolliffe. "Principal Component Analysis and Factor Analysis". In: *Principal Component Analysis*. New York, NY: Springer New York, 1986, pp. 115–128. DOI: 10.1007/978-1-4757-1904-8_7
Cited on page 188

[483] Kenneth O Stanley and Risto Miikkulainen. "Efficient Evolution of Neural Network Topologies". In: *Proceedings of the 2002 Congress on Evolutionary Computation.* Vol. 2. IEEE, 2002
Cited on page 199

[484] Diederik P Kingma and Jimmy Ba. "Adam: A Method for Stochastic Optimization". In: *preprint arXiv:1412.6980* (2014)
Cited on pages 201, 242

[485] Xavier Glorot and Yoshua Bengio. "Understanding the Difficulty of Training Deep Feedforward Neural Networks". In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. 2010, pp. 249–256
Cited on page 202

[486] Fethi Smach, Cedric Lemaître, Jean-Paul Gauthier, Johel Miteran, and Mohamed Atri. "Generalized Fourier Descriptors with Applications to Objects Recognition in SVM Context". In: *Journal of mathematical imaging and Vision* 30.1 (2008), pp. 43–71
Cited on page 206

[487] WH Müller. "Fourier Transforms and Their Application to the Formation of Textures and Changes of Morphology in Solids". In: *IUTAM Symposium on Transformation Problems in Composite and Active Materials*. Springer, 1998, pp. 61–72
Cited on page 206

[488] James S Cope, Paolo Remagnino, Sarah Barman, and Paul Wilkin. "Plant Texture Classification Using Gabor Co-Occurrences". In: *International Symposium on Visual Computing*. Springer, 2010, pp. 669–677
Cited on page 206

[489] Svante Wold, Kim Esbensen, and Paul Geladi. "Principal Component Analysis". In: *Chemometrics and intelligent laboratory systems* 2.1-3 (1987), pp. 37–52
Cited on page 206

[490] John C Russ. "Image Processing". In: *Computer-Assisted Microscopy*. Springer, 1990, pp. 33–69
Cited on page 207

[491] Pete E Lestrel. *Fourier Descriptors and Their Applications in Biology*. Cambridge University Press, 1997
Cited on page 207

[492] João Camargo Neto, George E Meyer, David D Jones, and Ashok K Samal. "Plant Species Identification Using Elliptic Fourier Leaf Shape Analysis". In: *Computers and electronics in agriculture* 50.2 (2006), pp. 121–134
Cited on page 207

[493] Simon YY Chen, Pete E Lestrel, W John S Kerr, and John H McColl. "Describing Shape Changes in the Human Mandible Using Elliptical Fourier Functions". In: *The European Journal of Orthodontics* 22.3 (2000), pp. 205–216
Cited on page 207

[494] Martin Friess and Michel Baylac. "Exploring Artificial Cranial Deformation Using Elliptic Fourier Analysis of Procrustes Aligned Outlines". In: *American Journal of Physical Anthropology: The Official Publication of the American Association of Physical Anthropologists* 122.1 (2003), pp. 11–22
Cited on page 207

[495] Ming-Kuei Hu. "Visual Pattern Recognition by Moment Invariants". In: *IRE transactions on information theory* 8.2 (1962), pp. 179–187
Cited on page 208

[496] Jan Flusser. "Moment Invariants in Image Analysis". In: *Proceedings of World Academy of Science, Engineering and Technology*. Vol. 11. Citeseer, 2006, pp. 196–201
Cited on page 208

[497] Samuel M Scheiner, Evsey Kosman, Steven J Presley, and Michael R Willig. "Decomposing Functional Diversity". In: *Methods in Ecology and Evolution* 8.7 (2017), pp. 809–820
Cited on page 210

[498] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. "How Transferable Are Features in Deep Neural Networks?" In: *Advances in Neural Information Processing Systems*. 2014, pp. 3320–3328
Cited on page 212

[499] Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. "Film: Visual Reasoning with a General Conditioning Layer". In: *Thirty-Second AAAI Conference on Artificial Intelligence*. 2018
Cited on page 212

[500] Ricky T. Q. Chen, Xuechen Li, Roger Grosse, and David Duvenaud. "Isolating Sources of Disentanglement in Variational Autoencoders". In: *Advances in Neural Information Processing Systems*. 2018
Cited on pages 218, 219

[501] Gal Chechik, Varun Sharma, Uri Shalit, and Samy Bengio. "Large Scale Online Learning of Image Similarity through Ranking". In: *Journal of Machine Learning Research* 11.Mar (2010), pp. 1109–1135
Cited on pages 218, 219

[502] Florian Schroff, Dmitry Kalenichenko, and James Philbin. "Facenet: A Unified Embedding for Face Recognition and Clustering". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 815–823
Cited on pages 218, 219

[503] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. "A Simple Framework for Contrastive Learning of Visual Representations". In: *arXiv preprint arXiv:2002.05709* (2020)
Cited on pages 218, 219

[504] Harshitha S. Kotian, Amith Z. Abdulla, K. N. Hithysini, Shalini Harkar, Shubham Joge, Ayushi Mishra, Varsha Singh, and Manoj M. Varma. "Active Modulation of Surfactant-Driven Flow Instabilities by Swarming Bacteria". In: *Physical Review E: Statistical Physics, Plasmas, Fluids, and Related Interdisciplinary Topics* 101.1 (2020), p. 012407. DOI: 10.1103/PhysRevE.101.012407
Cited on page 227

[505] Robert Tjarko Lange. "Evosax: JAX-based Evolution Strategies". In: *arXiv preprint arXiv:2212.04180* (2022)
Cited on page 242

[506] Sean Gillies, Casper van der Wel, Joris Van den Bossche, Mike W. Taves, Joshua Arnott, Brendan C. Ward, et al. *Shapely*. Zenodo. 2022. DOI: 10.5281/zenodo.7428463. Please cite this software using these metadata.
Cited on page 251