Model-based and model-free macroscopic control of the shape of low-dimensional systems with stochastic and deterministic dynamics

Francesco Boccardo

N°d'ordre NNT : 2022LYSE1039

THÈSE de DOCTORAT DE L'UNIVERSITÉ DE LYON

opérée au sein de:

l'Université Claude Bernard Lyon 1

École Doctorale ED 52
Physique et Astrophysique

Spécialité de doctorat: Physique

Soutenue publiquement le 31 mars 2022 par:

Francesco Boccardo

Model-based and model-free macroscopic control of the shape of low-dimensional systems with stochastic and deterministic dynamics

---

Devant le jury composé de:

ARGENTINA Médéric Professeur des Universités, Université Côte d'Azur	**Rapporteur**
COMBE Nicolas Professeur des Universités, Université Toulouse 3	**Rapporteur**
BARENTIN Catherine Professeur des Universités, Université Lyon 1	**Examinatrice**
MATIGNON Laetitia Maître de Conférences, Université Lyon 1	**Examinatrice**
MONTALENTI Francesco Professeur, Université de Milan Bicocca	**Examinateur**
PIERRE-LOUIS Olivier Directeur de Recherche, CNRS Lyon	**Directeur de thèse**
M. BOUCHET Freddy Directeur de Recherche, CNRS Lyon	**Invité**

Acknowledgments

I would first like to express my deepest gratitude to Olivier Pierre-Louis, my thesis advisor, for the inspiration and help he has provided me throughout the process of researching and writing this thesis. Without his precious advice that always guided me in the right direction, this work would not have been possible.

Then, I would like to thank the Institut Lumière Matière and particularly the members of the MMCI (Modélisation de la Matière Condensée et des Interfaces) team, for the pleasant professional and informal interactions. I am also very grateful to the members of the jury for accepting to be part of it, and especially to the referees for taking the time to evaluate this manuscript.

I would also like to thank all the people with whom I have had fruitful discussions and who have provided valuable comments, including Laetitia Matignon, Tung B. T. To, Luca Gagliardi and Riccardo Galafassi. During this project, I also had the great pleasure of working with several talented internship students, whom I thank for their collaboration: Baptiste Filoche, Alexandre Mass, Christopher Greenberg Bonilla, Jingyu Wang, Qianlong He and Jules Vanaret. Their detailed contributions are listed in the concluding chapter of this thesis.

I am also grateful to my colleagues and all the friends I have made over the past four years at ILM, who have made my days at the institute very enjoyable: Carolina, Paolo, Clara, Francesca, Kevin, Matilde, Thomas, Nilankur, Cecilia, Bastien and Yedhir. A very special thanks goes to Luca, Antoine and Riccardo, irreplaceable office and life companions.

I would like to extend my gratitude to all my friends in Lyon outside of academia, especially the fellow mechanics of the atelier "Chat Perché", the musicians of the fanfare "Marcel Frontale" and the amazing jugglers of the "AJIL" association, who have made these French years truly memorable. Another special thanks goes to Carlotta, who supported me emotionally and with love in the last moments of writing, the hardest ones.

Finally, I must express my profound gratitude to my parents Carla and Alberto for always giving me trust and encouragement throughout all my years of study. This achievement would not have been possible without them.

Contents

Abstract

This thesis explores the control of the morphology of extended physical systems involving stochastic or nonlinear dynamics. More precisely, we are interested in the problem of achieving in finite time a target morphology that is as arbitrary as possible through "model-based" or "model-free" control methods. In the model-based approach, complete knowledge of the physical laws governing the system is exploited to compute an optimal control strategy. If this knowledge is not available, a control strategy can be learned by the model-free approach by interacting with the dynamical system itself and "reinforcing" the actions that maximize a certain reward signal.

We have applied model-based control to the case of small two-dimensional islands of a few particles under the effect of an external macroscopic field, such as an electric field or a temperature gradient, which acts as a control parameter to reach a target shape. This model describes single-layer clusters of atoms, nanoparticles, or colloids. We considered the case of a dynamics governed by stochastic diffusion of particles along the periphery of the island, which thus conserves the number of particles during the dynamics. Reaching a target shape for the island can be seen as a first-passage problem in the space of configurations, and the choice of the external field can be studied in the framework of Markov decision processes. The finite number of configurations allowed us to apply tabular algorithms (which list all the states of the system in an array). We have also derived some analytical results using a high temperature expansion.

In the absence of an external field, we showed that sufficiently large compact shapes exhibit an optimal temperature at which they are reached in minimum time. In the presence of field, we used the so-called dynamic programming method to solve the Markov decision process and find an optimal control strategy to reach the target shapes in minimum time. This strategy results in a time gain that increases when increasing the size of the islands or decreasing the temperature. Moreover, the optimal strategy is not unique, and its degeneracy is mainly related to the symmetries of the system. Furthermore, the optimal strategy presents a discrete set of transitions as the temperature varies. As the cluster size increases, a continuous density of transitions emerges.

With model-free control, we modeled a situation that mimics an experimental setup, where an automated controller learns to manipulate a small cluster. We have shown that the reinforcement learning methods of Monte Carlo and Q-learning type allow one to reach control performances close to the optimal performances computed by dynamic programming, except at high temperature where the fluctuations are important and the influence of the external field is weak.

Finally, we have studied the model-free control of the morphology of extended deterministic

continuous systems. The control is obtained by adjusting a global parameter that governs the stability of the system, such as the temperature difference from the critical point in a system undergoing a phase transition. In the framework of a one-dimensional model based on a time-dependent generalization of the Allen-Cahn (or Ginzburg-Landau) equation, we have been able to control the number of phase domains appearing in the system by using neural network-based approximation techniques (deep Q-learning). Preliminary results have also allowed us to classify the shape of domains undergoing a fingering instability obtained by a two-dimensional phase-field model, a result that marks a first step towards the morphological control of these systems.

Résumé

Cette thèse explore le contrôle de la morphologie de systèmes physiques étendus mettant en jeu une dynamique stochastique ou nonlinéaire. Plus précisément, nous nous sommes intéressés au problème d'atteinte en temps fini une morphologie cible aussi arbitraire que possible grâce à des méthodes de contrôle "basées sur un modèle" ou "sans modèle". Dans l'approche basée sur les modèles, la connaissance complète des lois qui régissent le système est exploitée pour calculer une stratégie de contrôle optimale. Si cette connaissance n'est pas disponible, on peut apprendre une stratégie de contrôle par l'approche sans modèle en interagissant avec le système dynamique lui-même et en "renforçant" les actions qui maximisent un certain signal de récompense.

Nous avons appliqué le contrôle basé sur un modèle au cas de petits îlots bidimensionnels de quelques particules sous l'effet d'un champ macroscopique extérieur, tel qu'un champ électrique ou un gradient de température, qui joue le rôle de paramètre de contrôle. Ce modèle décrit des amas monocouche d'atomes, de nanoparticules, ou de colloïdes. Nous avons considéré le cas d'une dynamique régie par la diffusion stochastique des particules le long de la périphérie de l'îlot, qui conserve donc le nombre de particules pendant la dynamique. L'atteinte d'une forme cible pour l'îlot peut être vue comme un problème de premier passage dans l'espace des configurations, et le choix du champ peut être étudié dans le cadre des processus de décision Markoviens. Le nombre fini de configurations nous permet d'appliquer des algorithmes tabulaires (qui listent l'ensemble des états du système dans un tableau). Nous avons également dérivé quelques résultats analytiques à l'aide d'un développement à haute température.

En l'absence de champ externe, nous avons montré que les formes compactes suffisamment grandes présentent une température optimale à laquelle elles sont atteintes en un temps minimum. En présence de champ, nous avons utilisé la méthode dite de programmation dynamique pour résoudre le processus de décision Markovien et trouver une stratégie optimale pour atteindre les formes cibles dans un temps minimal. Cette stratégie entraîne un gain de temps qui croît lorsqu'on augmente la taille des îlots ou que l'on diminue la température. Par ailleurs, la stratégie optimale n'est pas unique, et sa dégénérescence est principalement liée aux symétries du système. De plus, la stratégie optimale présente un ensemble discret de transitions lorsque la température varie. Quand la taille du cluster augmente, une densité continue de transitions émerge.

Avec le contrôle sans modèle, nous avons modélisé une situation qui mime un dispositif expérimental, où un contrôleur automatisé apprend à manipuler un petit îlot. Nous avons montré que les méthodes d'apprentissage par renforcement de type Monte Carlo et Q-learning permettent en général d'atteindre des performances de contrôle proches des performances optimales calculées

par programmation dynamique, sauf à haute température où les fluctuations sont importantes et l'influence du champ extérieur est faible.

Enfin, nous avons étudié le contrôle sans modèle de la morphologie de systèmes déterministes étendus. Le contrôle est obtenu en ajustant un paramètre global qui gouverne la stabilité du système, tel que l'écart de température par rapport au point critique dans un système subissant une transition de phase. Dans le cadre d'un modèle unidimensionnel basé sur une généralisation de l'équation d'Allen-Cahn (ou Ginzburg-Landau dépendante du temps), nous avons pu contrôler le nombre de domaines apparaissant dans le système en utilisant des techniques d'approximation basées sur des réseaux de neurones (deep Q-learning). Des résultats préliminaires ont aussi permis de classifier les formes de domaines subissant une instabilité de digitation obtenues par un modèle de champ de phase bidimensionnel, un résultat qui marque un premier pas vers le contrôle morphologique de ces systèmes.

CHAPTER 1

Introduction

Linking the macroscopic world and the world of atoms, molecules and nanostructures is a long-standing technological challenge. The difficulties we are confronted with were already envisioned by Richard Feynman in 1959, when he described the problems of small-scale manufacturing in his famous talk "There's Plenty of Room at the Bottom" [1]. More than sixty years later, a fundamental question is still driving and inspiring many research investigations: "How do we manipulate and control objects on a small scale?".

Techniques to control and interact with extremely small objects are usually referred to as *micro-* and *nanomanipulation* and are considered to be key technologies for the upcoming decades or even century [2], with applications in various fields, including chemistry, materials science, biology and medicine [3].

In the macroscopic world, matter can be easily manipulated using direct physical methods (e.g. etching and lithography [4]), and an object of a desired shape can be carved out of a larger part. This is a common approach used to make macroscopic components and devices. However, this *top-down* strategy of sculpting matter to form shapes becomes increasingly challenging as dimensions approach the nanoscale [5, 6]. Hence, engineers also tried a radically different approach, mimicking the *bottom-up* strategy that can be observed in nature, where small individual building blocks self-assemble to produce structures with defined geometries and specific functions, such as biomolecules.

In this theoretical work, we investigate a novel paradigm for controlling the morphology of matter at small scales. This approach does not require the manipulation of atoms or particles one by one, but instead relies on the use of a macroscopic control parameter that can impose a global driving force on the system, as illustrated in Fig. 1.1. In this way, it is not necessary to resort to tools that can perform microscopic actions, such as focused electron beams for atoms and molecules or optical tweezers for micro- and nanoparticles, in order to achieve shape control.

The main theme of this thesis is to exploit the dynamics of an out-of-equilibrium physical system, which naturally explores different configurations over time, and to guide this exploration towards a target shape using a macroscopic control parameter. We have considered two sources of shape exploration in physical systems. The first comes from thermal fluctuations. For this case, we have looked at a discrete lattice model, which describes the stochastic dynamics of small clusters of atoms, molecules, or colloidal particles. The second one comes from the presence of

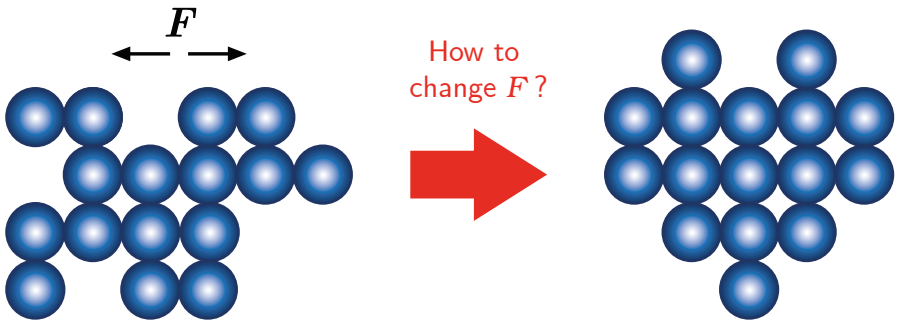


Figure 1.1: Schematic illustration of the main question of this work: how can we tune a macroscopic control parameter acting globally on a microscopic dynamical system (e.g. a cluster of nanoparticles undergoing thermal fluctuations) in order to achieve a desired morphology?

morphological instabilities, such as the Mullins-Sekerka instability in crystal growth or spinodal decomposition in phase transitions, and for this we considered a continuous deterministic model. To these discrete and continuous models we applied several techniques from control theory to determine how to tune the control parameter in order to achieve the desired reshaping of the system.

This introductory chapter is divided into three main parts. In the first, we give a brief historical perspective to the problem of shape manipulation in small-scale physical systems, with an overview of relevant research that has been done on the topic, focusing on the specificity of our approach with respect to what already exists in the literature. In the second part, we give a concise presentation of the original contributions contained in this thesis. Finally, the third part aims at introducing the key concepts of control theory.

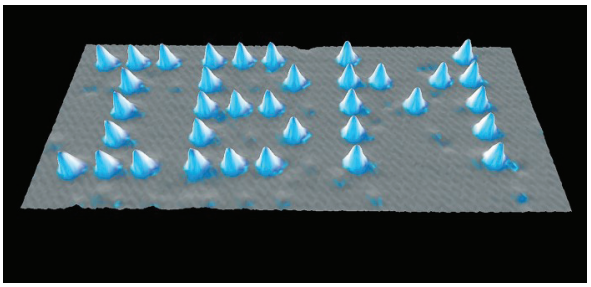
1.1 Shape control at the micro- and nanoscale

A milestone in the history of nanomanipulation were the experiments performed in 1990 at IBM, where 35 individual xenon atoms were positioned on a nickel substrate at low temperature to form the IBM logo [7], reaching atomic-scale control on the organization of matter (see Fig. 1.2a). These experiments were based on the development of the scanning tunnelling microscope (STM) by Binnig and Rohrer [8] for which they were awarded the Nobel Prize in Physics 1986.

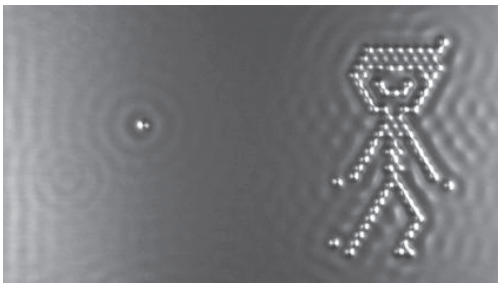
During the 1990s manipulation experiments with instruments based on the STM or the atomic force microscope (AFM) [9, 2] gained momentum and also dedicated equipment for micro- and nanomanipulation were developed. In the following decades, many examples of organization of atoms, molecules, or nanoparticles with STM or AFM, but also colloids with tools like optical tweezers [10, 11, 12] or magnetic tweezers [13, 14] have been obtained (see Figs. 1.2c and 1.2d).

In 2013, scientists at IBM research where able to create a stop-motion animated short film entitled “A Boy And His Atom”, where a cartoon character plays with a ball. Each frame has been created individually by precisely positioning, using STM, carbon monoxide molecules on a copper surface and imaging each configuration (see Fig. 1.2b). This has been recognized as the world’s smallest movie [15].

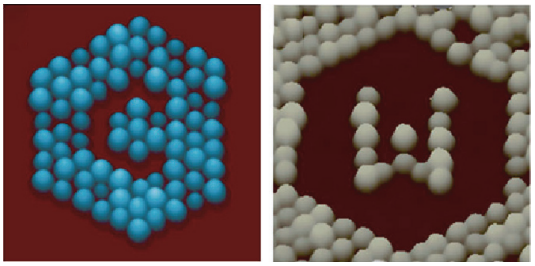
Typically, manipulating single atoms or nanoparticles is an interactive procedure that requires human intervention. This makes it a generally slow, non-repeatable and inaccurate process. For this reason, automated nanomanipulation methods have been developed for AFM [19, 20, 17], where an algorithm detects the particle positions and plans the pushing paths to reach a desired



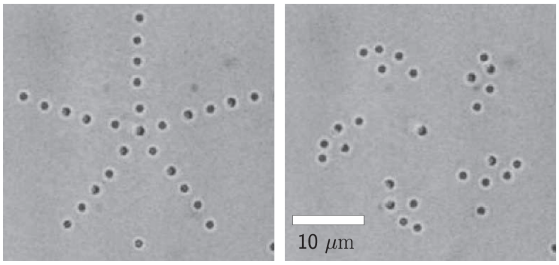
(a) Xe atoms arranged on a Ni substrate at 4 K to form the letters IBM. Each letter is 40 Å from top to bottom. Image from Ref. [7].



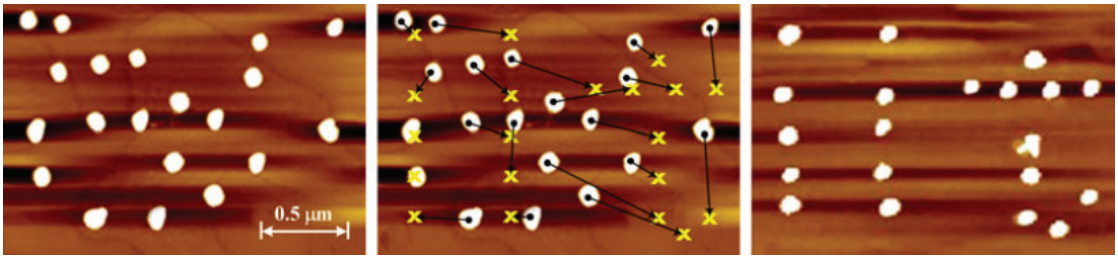
(b) (45×25 nm^2) CO molecules arranged on a Cu(111) substrate at 5 K to form a frame of the world's smallest movie. Image from Ref. [15].



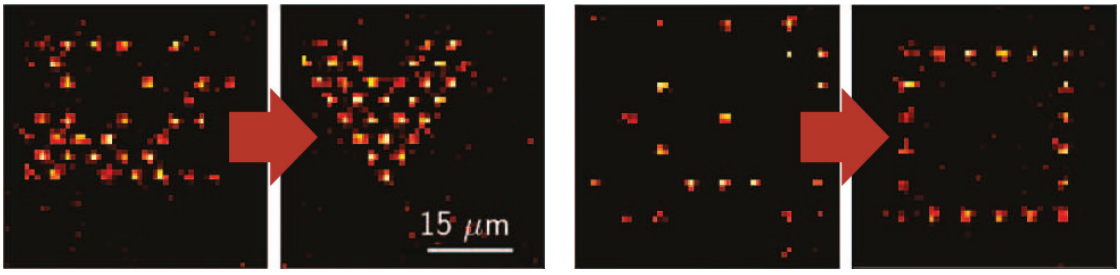
(c) (30×30 nm^2) Cluster of Ir (left) and W (right) nanoparticles on a G/Ir(111) substrate at ∼300 K, each particle contains ∼50 atoms. Image from Ref. [16].



(d) Two different patterns obtained arranging 26 colloidal silica spheres of around 1 µm in diameter using optical tweezers. Image from Ref. [10]



(e) Pattern composed of 20 latex nanoparticles (diameter 50 nm) on a Si substrate. Left: initial configuration, middle: pushing paths computed by the algorithm, right: final configuration. Image from Ref. [17].



(f) Two user-defined arrays of Rb atoms obtained from initial random configurations using a real-time control system and a moving optical tweezers. Image from Ref. [18].

Figure 1.2: (a) One of the earliest STM images of an atomic-precision nanomanipulation experiment and (b) a more recent one. (c) Clusters of nanoparticles positioned with STM, (d) clusters of colloidal particles positioned with optical tweezers. (e) Automatic nanomanipulation experiment with AFM on nanoparticles and (f) with optical tweezers on atoms.

configuration (see Fig. 1.2e). Similar methods have been developed also for optical tweezers [18] in order to obtain atomic arrays of arbitrary geometries (see Fig. 1.2f).

Research on the control of small particle clusters does not stop there. There are many important challenges that are still open. One of them is to control matter at the nanoscale with an external macroscopic field that does not act on one single particle or atom at a time, but on the whole cluster. In the literature, there are several examples of systems where the interaction between macroscopic fields and microscopic clusters were shown to lead to complex equilibrium or non-equilibrium shapes, such as light acting on metal nanoparticle clusters [21] or electric fields acting on two-dimensional (monolayer) atomic clusters [22, 23, 24]. However, these shapes, which are dictated by the physics of the interaction of the driving force with the system, are usually uncontrolled, in the sense that they are not chosen a priori by the experimentalist.

Another challenge lies in the ability to obtain refined control of nanostructure shapes in the presence of thermal fluctuations that activate the random diffusion of particles and atoms, leading to shape fluctuations [25, 24, 26]. In fact, nanomanipulation experiments are usually performed at temperatures such that the thermal energy k_BT in the system is much lower than the typical binding energy between particles J, in order to forbid surface diffusion. For example, if we look at Figs. 1.2a and 1.2b, we can see that, for atoms and small molecules, the experiments were carried out at very low temperatures, around 5 K, whereas the experiment with the metal nanoparticles of Fig. 1.2c was carried out at room temperature, because the binding energy in this system is much higher.

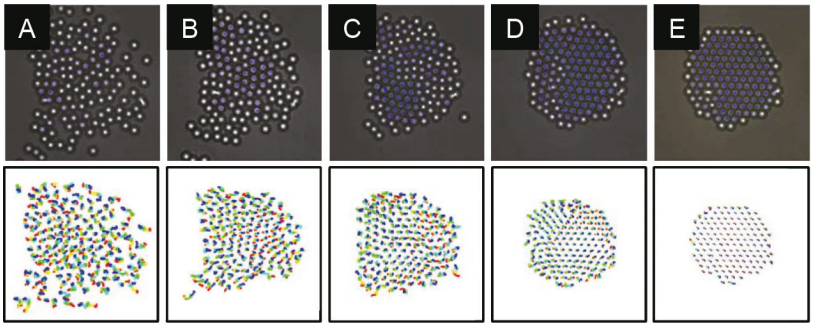
Some progress in these two directions has been made for colloids. We will now list a few relevant works where some control has been achieved on colloidal clusters at finite temperature using macroscopic fields. The reader interested in a more complete overview of the existing techniques for the control of colloidal assemblies can refer to the reviews of Refs. [27, 28].

Using external electric fields, several papers have reported the possibility of controlling the assembly and disassembly of colloidal crystals of few hundreds of particles [29] (see Fig. 1.3a), their crystallinity [32], and even the precise number of particles in the cluster [33]. These three works are particularly relevant to us because they use a *closed-loop* (or *feedback*) control approach, where the experimental apparatus is coupled to an automated controller that changes the external field as a function of the observed configuration of the system. This approach is a cornerstone of our contribution.

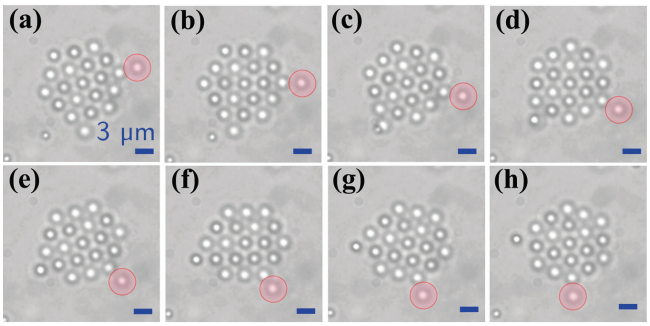
Using light (optical traps), the generation of static structures of colloidal particles and their translation or rotation in a controlled manner has been achieved [30] (see Fig. 1.3b). By applying a combination of rotating and static magnetic fields, the authors of Ref. [31] managed to assemble and actuate microscale pumps made of few magnetic particles in a microfluidic environment (see Fig. 1.3c).

These are just a few examples that highlight the growing interest in the control of the assembly of micro- and nano-objects at small scales through the use of various types of macroscopic fields, without directly manipulating individual particles [3]. However, to our knowledge, the control of the reshaping of an already formed cluster is still an open question. In particular, the idea of how external fields can be tuned in time to guide the stochastic evolution of a particle cluster from one shape to another has not yet been investigated.

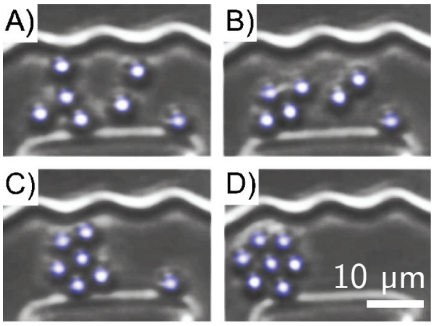
All the works cited above focus on two-dimensional clusters of atoms (i.e. monolayers) or particles. This is indeed the type of system that we have considered in this thesis, and for several reasons. The first is dictated by the choice of a closed-loop control approach, where changes to the external macroscopic field are made as a function of the observed shape of the cluster. Two-dimensional clusters on a surface are relatively easy to observe. There are many experiments where this has been done in detail, see, for example Refs. [34, 35] for 2d metal atomic clusters (also called monolayer islands), and Refs. [36, 37] for nano- and microscopic 2d



(a) Dynamic feedback controlled assembly of a colloidal crystal of $\sim$ 130 silica particles (diamter $\sim$ 3 µm). Experimental optical video microscopy images (top) and particle trajectories for the assembly process (bottom). Image from Ref. [29].



(b) Controlled rotation of colloidal clusters made of 19 silica beads (diamter $\sim$ 2 µm) with interferometric optical traps. Image from Ref. [30].



(c) Directed self-assembly of 7 magnetic colloidal particles (diameter 4.5µm) into a compact micropump in the presence of a rotating magnetic field. Image from Ref. [31].

Figure 1.3: Micromanipulation of colloidal clusters by means of macroscopic fields, using (a) an electric field, (b) optical traps, and (c) a magnetic field.

colloidal clusters. Another reason for this choice is that we want to exploit thermal fluctuations to guide the dynamics of the system towards a desired shape. A 2d cluster has a 1d interface, and this causes the fluctuations of the interface to be strong, as they become larger when the dimensionality is decreased [38].

Finally, there are well-established models for describing the evolution of a fluctuating 2d cluster under the effect of an external field. The one we have chosen is a discrete lattice model (explained in detail in Chapter 2), which has been used extensively for describing metal monolayer islands in the presence of an external electric field. This field can bias the diffusion of mobile atoms along the edge of the cluster (this phenomenon is known as *electromigration*), see, for example, Refs. [39, 24, 40]. This model also aims at describing other systems consisting of fluctuating particle clusters with biased edge diffusion, such as clusters of colloids or nanoparticles under the effect of a temperature gradient (i.e. *thermophoresis*), an electric field (i.e. *electrophoresis*) or a magnetic field.

Moreover, the discrete nature of this model has allowed us to apply a class of control algorithms that is relatively simple to implement and allows one to obtain accurate results for small (few-atoms) clusters.

1.2 Control of continuous systems

Interest in shape control of small-scale physical systems is not limited to discrete clusters, but is also directed toward spatially-extended continuous systems. The type of systems we have considered are those in which there is an evolving interface—of which we want to control the morphology—such as in the growth of a crystal, or in the phase separation of a binary mixture. Following the same lines as for clusters, in this thesis we have turned our attention toward a control strategy for continuous systems based on the effect of a global parameter, rather than on localized manipulation.

The idea that one can influence the morphology of an evolving physical interface by varying global parameters of the surrounding environment is not completely new. For instance, changes in the shapes of ice crystals (snowflakes), depending on ambient temperature and water vapour supersaturation, were first documented as early as the 1950s [41] and are still actively studied today, as the underlying physical mechanisms that drive the puzzling growth behaviours of snowflakes are not well understood [42].

For the past 20 years, Kenneth Libbrecht, a professor of physics at the California Institute of Technology, has been studying how ice crystals form and has refined his technique of obtaining what he calls "designer crystals" [43, 44]. These crystals are grown under controlled conditions in his lab, where he can create a specific growth pattern by adjusting the applied temperature, humidity, and other environmental factors by hand as a function of time. In this way, he can get plates, branches and other desired effects, practically *designing* the final morphology of the snowflake at his will. Some examples of the crystals he obtained are shown in Fig. 1.4. Libbrecht was also recently featured in an educational video where he shows his work and his experimental setup [45].

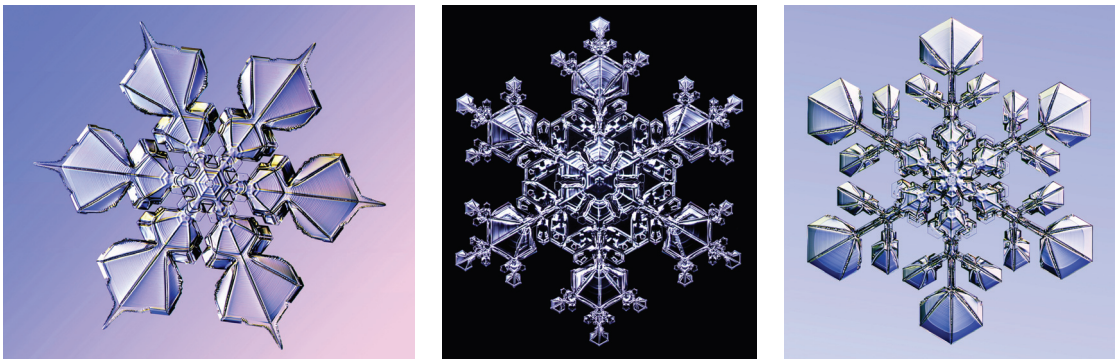
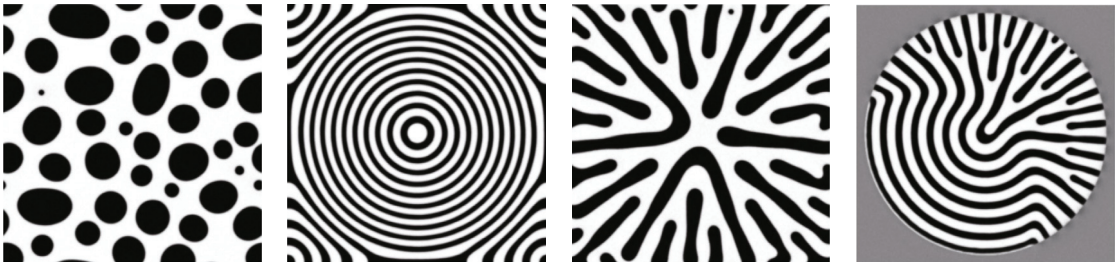


Figure 1.4: Three examples of Kenneth Libbrecht's designer crystals, obtained by manually varying the temperature and humidity of the environment during growth. Image from Ref. [43].

Libbrecht's designer crystals are essentially an artistic endeavour, which can be seen as a novel kind of ice sculpture, discarding the chisel in favour of using molecular self-assembly and the laws of crystal growth to create beautiful crystalline structures. Nevertheless, there are also several scientific publications where some kind of morphological control of an interface by means of a varying macroscopic parameter has been studied. One of the simplest approaches to this kind of control is that of slowly decreasing or increasing the temperature at a constant rate. Experimental and theoretical works have shown that, in a solution undergoing a phase separation (or *spinodal decomposition*), the heating or cooling rate can influence the characteristic length of the phase domains [46, 47] and even lead to surprising periodic oscillations of this length [48, 49].

Phase-separation models under an external field that favours one phase with respect to the other have also been extensively studied in the applied mathematics literature [50, 51, 52, 53]. Instead, we have chosen to control the parameter that allows for the appearance of the instability. A related approach to control phase separation has been conducted recently by Kurita and Tsukada [54, 55, 56]. They use a space- and time-dependent temperature field, called "phase-separation trigger". Some of the patterns that they were able to obtain with their method are shown in Fig. 1.5. We will report more in detail on this and other theoretical works that are relevant to this thesis in Chapter 4, which is devoted to continuous systems.



(a) (b) (c) (d)

Figure 1.5: Different patterns formed during simulated phase separation controlled by means of a space- and time-dependent temperature field. (a) A random droplet pattern, (b) concentric circles, (d) a dendritic pattern, and (e) a combination of concentric circle and dendritic patterns. Image from Ref. [54].

As stated before, we are not interested in space-dependent or spatially-localized control, hence, unlike the works of Kurita and Tsukada cited above, we chose to use a spatially-uniform control parameter that can vary only in time, similarly to Libbrecht's method for designing snowflakes. Like in the case of particle clusters, we have used a closed-loop approach, where the control parameter is changed over time as a function of the observed state of the system by an automated controller. Since in this case we are dealing with continuous interfaces with infinitely many degrees of freedom, we had to resort to advanced algorithms that use artificial neural networks to process relevant information about the shape of the system and find an approximate solution to the control problem.

1.3 Main contributions of this thesis

The core idea that we have explored in this thesis is the control of the morphology in dynamical systems. More precisely, we want to tackle the problem of reaching a target configuration, or shape, of an evolving physical system in finite time. We consider the target shape to be as arbitrary as possible.

To achieve this goal, we applied several techniques from control theory, that can be divided into two categories: *model-based* and *model-free* control methods.

In the model-based approach, complete knowledge of the governing laws (i.e. the dynamical model) of the system is exploited to compute an optimal control strategy. If this knowledge is not available, or is incomplete, then one can try to solve the control problem by interacting with the dynamical system itself and observing its response. In this case, the idea is to learn a control strategy by trying many times different actions and "reinforcing" those that maximize a certain reward signal. This second approach is called model-free, since it relies only on observing the

response of the system when acting on it, without the need for knowledge on the model that governs the system.

Although historically they developed mostly independently [57], the first approach having his roots in *optimal control* theory and the second one in *reinforcement learning*, nowadays these two classes of methods are closely interrelated, and can be seen as two complementary approaches to achieve the same goal.

We applied model-based control to the case of small two-dimensional clusters of particles evolving by thermally-activated diffusion under the effect of an external macroscopic field, such as an electric field or a temperature gradient, that plays the role of the control parameter. This is a discrete stochastic system with a total number of configurations, or states, that is fixed, if the number of particles in the cluster is conserved. As we will see in more detail in the next section, having a set of states that is discrete and finite allows us to apply control algorithms in a relatively simple form, called *tabular*, which takes advantage of the possibility of listing the states of the system one by one in a table.

We modelled the system using a well-known lattice model and we laid down the mathematical framework of the problem of shape control in small particle clusters. Within this model, the problem of reaching a target cluster shape can be rephrased as a first passage process in the space of configurations. Then, using *dynamic programming*, which is the standard model-based method to solve stochastic control problems, we computed optimal policies for tuning the external field in order to reach different target shapes in minimum time.

Taking advantage of the formalism of dynamic programming, we also derived some analytical results based on a high-temperature expansion. This expansion allowed us to investigate the physical time required to reach a target shape, both in the case where there is no external field (equilibrium dynamics) and in the presence of the field. Without external field, we found that compact shapes that are large enough exhibit an optimal temperature at which they are reached in minimum time. Then, we have computed the optimal strategy (also called *policy*) to set the external field as a function of the observed state of the system in order to minimize the time to reach a desired target shape. We found that this leads to a gain of time (with respect to the absence of the field or to the presence of an unoptimized, randomly varying field) that grows when increasing cluster size or decreasing temperature.

We have also found that the optimal way to set the external field is non-unique, and its degeneracy is mainly related to symmetries of the system. Furthermore, the optimal policy exhibits a discrete set of transitions when the temperature is varied. As the size of the cluster increases, a continuum density of temperature transitions emerges. These results are reported in Chapter 2.

With model-free control, we performed a computational analysis of an experiment-like situation, where an automated controller learns to manipulate a small cluster by interacting with it. We compared different learning methods and showed that, with this approach, it is possible to reach control performances that are close to the optimal ones calculated in Chapter 2. However, there are limitations, mainly related to the difficulty of learning at high temperature, where thermal noise is strong. The results of this analysis are reported in Chapter 3.

Finally, Chapter 4 focuses on morphological instabilities in continuous deterministic systems. Differently from the case of clusters, here control is not achieved by adding an external field, but by directly tuning a global parameter of the model that has the role of governing the stable or unstable character of the system, such as the temperature difference from the critical point in a mixture undergoing a phase transition. We were able to obtain promising results for the control of a simple one-dimensional model based on a generalisation of the Allen-Cahn equation, also known as time-dependent Ginzburg-Landau equation. In this case, we chose to focus on model-free methods. We could not apply tabular methods, since the set of states is continuous,

and had to resort to advanced approximation techniques based on neural networks, such as *deep Q-learning*, discussed in the next section. This method has allowed us to control the number of domains that appear in the system.

We also considered a more complicated two-dimensional phase-field model that can describe growth phenomena, such as the solidification of a crystal from a melt, and with this model we were able to perform shape classification for domains undergoing a fingering instability. This is a result that marks a first step toward morphological control of growth processes.

1.4 Control theory in a nutshell

In nature as well as in man-made products, dynamical systems—systems with a state that changes over time—can be found everywhere: from physical systems, such as a growing crystal, to social systems, such as economy, to life systems, such as population growth. The idea of influencing the behaviour of such systems in order to drive them to a desired state is at the heart of control theory [58].

In this section we will illustrate in a concise and simplified way the main ideas of control theory that are relevant to the work presented in this thesis. The reader interested in an in-depth treatment of these topics can refer to the book by Sutton and Barto [57].

As stated in the previous section, control methods can be divided into two main classes: model-based and model-free methods. We will now present the key concepts of these two approaches through the formal framework of Markov decision processes.

1.4.1 Markov decision processes

Markov decision processes, or MDPs, are a classical formalization of decision making problems. They rely on the concept of the agent-environment interface, where a decision maker, called the *agent*, can take actions that affect the state of a given dynamical system (real or simulated), called the *environment*. These interact continually, the agent selecting actions and the environment evolving accordingly, presenting new situations to the agent. The environment also gives rise to *rewards*, i.e. numerical values that the agent seeks to maximize over time through its choice of actions.

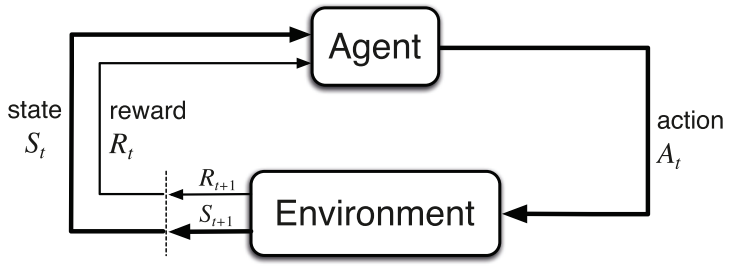


Figure 1.6: The agent-environment interface in a Markov decision process. Image from Ref. [57].

For simplicity, we restrict attention to discrete-time MDPs, where the agent and environment interact at each of a sequence of discrete time steps, $t = 0, 1, 2, 3, \ldots$. At each discrete time t, the agent receives a representation of the current state of the environment, $S_t \in \mathcal{S}$, and on that basis

selects an action, $A_t \in \mathcal{A}$[1]. One time step later, in part as a consequence of its action, the agent receives a numerical reward, $R_{t+1} \in \mathcal{R}$, and finds the environment in a new state, S_{t+1}, and so on[2]. We assume that the process is always stopped at some point, either because the environment reached a designated target state or because there is a time limit. The MDP thereby gives rise to a sequence of states and actions, called a *trajectory* or an *episode*, that looks like this:

$$S_0, A_0, S_1, A_1, ..., S_{t_f-1}, A_{t_f-1}, R_{t_f}, S_{t_f} \,, \tag{1.1}$$

where t_f is a final time. The sum of all the rewards is the *return* associated to a trajectory

$$G = R_1 + R_2 + ... + R_{t_f} \,. \tag{1.2}$$

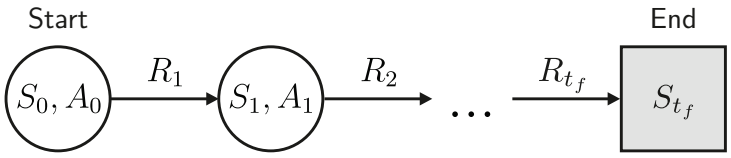


Figure 1.7: Schematic representation of a trajectory in state space.

The control strategy followed by the agent is encoded in a function π called a *policy*, which determines what action to take in each state. Policies are in general stochastic, and are written using the notation of conditional probability $\pi(a \,|\, s)$. The latter is a function that assigns a probability distribution over the possible actions a for each state s. A policy can also be deterministic, and in that case $\pi(a \,|\, s)$ is a delta function peaked on the only action a to be taken in state s.

The expected value of the return G depends on which policy π is used to choose the actions. Solving an MDP consists in finding, among all the possible policies, an optimal policy π_* which maximizes the expected return:

$$\pi_* \in \underset{\pi}{\operatorname{argmax}}\, \mathbb{E}_\pi \left[G\right] \,, \tag{1.3}$$

where $\operatorname{argmax}_x f(x)$ is the set of values of x at which $f(x)$ takes its maximal value and the notation $\mathbb{E}_\pi \left[G\right]$ denotes the expected value of the random variable G, given that the agent follows policy π. Note that there might be more than one optimal policy that maximizes G, hence the set membership symbol $\in$.

Finding a policy that satisfies Eq. (1.3) is the standard goal of control theory. The choice among methods that could achieve this goal depends crucially on our knowledge of the laws that govern the environment's dynamics. The latter are characterized by the function

$$p(s', r \mid s, a) = \Pr\left\{S_{t+1} = s', R_{t+1} = r \;\mid\; S_t = s, A_t = a\right\} \,, \tag{1.4}$$

for all $s', s \in \mathcal{S}$, $r \in \mathcal{R}$, and $a \in \mathcal{A}$. The function p defines the *dynamics* of the MDP and it gives the probability of transitioning to state s' and receiving r as a reward when taking action a from state s. Capital letters are used for random variables, whereas lower case letters are used for the values of random variables. As for the policy, the vertical bar in the middle comes again from

[1] In general, the set of possible actions can be different in each state, hence we should write $\mathcal{A}(s)$. However, to simplify notation, we assume the special case in which the action set is the same in all states and write it simply as $\mathcal{A}$.

[2] Following the notation of Ref. [57], we use R_{t+1} instead than R_t to denote the reward due to A_t because it emphasizes that the next reward and next state, R_{t+1} and S_{t+1}, are jointly determined.

the notation for conditional probability, but here it is used just to remind us that p specifies a probability distribution for each choice of s and a, i.e. $\sum_{s',r} p(s', r \mid s, a) = 1$, for all $s \in \mathcal{S}$, $a \in \mathcal{A}$.

Note that, in an MDP, the probability of each possible value for S_{t+1} and R_{t+1} depends only on the immediately preceding state and action, S_t and A_t, rather than on the whole sequence of states and actions encountered up to time t. This is the *Markov* property.

From the four-argument function p one can compute anything else one might want to know about the dynamics of the environment, such as the *state-transition probabilities*

$$p(s' \mid s, a) = \Pr\{S_{t+1} = s' \mid S_t = s, A_t = a\} = \sum_{r \in \mathcal{R}} p(s', r \mid s, a)\,, \tag{1.5}$$

and the expected reward for any state-action pair

$$\varrho(s, a) = \mathbb{E}\left[R_{t+1} \mid S_t = s, A_t = a\right] = \sum_{r \in \mathcal{R}} r \sum_{s' \in \mathcal{S}} p(s', r \mid s, a)\,. \tag{1.6}$$

Depending on our knowledge of the dynamics function p we can, at this point, differentiate between model-based and model-free control approaches. Model-based methods assume a complete knowledge of the dynamics and rely on *planning* an optimal trajectory as their primary component. In contrast, model-free techniques do not assume any knowledge of the dynamics—encoded in the transition probability Eq. (1.4)—and primarily rely on *learning* how to take actions in order to maximize the overall reward.

1.4.2 Model-based control

So far we have discussed the objective of solving an MDP quite informally. To go further, we need to generalize the definition of the return given in Eq. (1.2) to an arbitrary starting time. We now define[3] the return G_t as the sum of all the rewards observed after time t up to the final time step t_f:

$$G_t = R_{t+1} + R_{t+2} + R_{t+3} + ... + R_{t_f} = \sum_{k=t+1}^{t_f} R_k\,. \tag{1.7}$$

The expected value of the return when starting in a state s and then following a policy π is called the *value function* of state s under π and denoted $v_\pi(s)$. We can define v_π formally by

$$v_\pi(s) = \mathbb{E}_\pi\left[G_t \mid S_t = s\right]\,. \tag{1.8}$$

A fundamental property of the value function is that it satisfies a recursive relation. For any policy π and any state s, the following condition holds (see Appendix A.1 for details):

$$\begin{aligned} v_\pi(s) &= \mathbb{E}_\pi\left[G_t \mid S_t = s\right] = \mathbb{E}_\pi\left[R_{t+1} + G_{t+1} \mid S_t = s\right] \\ &= \sum_a \pi(a \mid s) \sum_{s',r} p(s', r \mid s, a)\Big[r + \mathbb{E}_\pi\left[G_{t+1} \mid S_{t+1} = s'\right]\Big] \\ &= \sum_a \pi(a \mid s) \sum_{s',r} p(s', r \mid s, a)\left[r + v_\pi(s')\right]\,, \end{aligned} \tag{1.9}$$

[3]To be even more general, it is possible to include in the definition of return a *discount factor* β, so that $G_t = R_{t+1} + \beta R_{t+2} + \beta^2 R_{t+3} + \dots$. The discount factor, which has to be between 0 and 1, determines how much the agent cares about the future. The closer it is to 0, the more the agent is "myopic" and is only concerned with maximizing immediate rewards. As β approaches 1, the agent becomes more farsighted. For the sake of simplicity, we restrict this discussion to the special case $\beta = 1$.

where the actions a, the next states s' and the rewards r are taken respectively from the sets of the possible actions $\mathcal{A}$, the set of the states of the system $\mathcal{S}$ and the set of the possible rewards $\mathcal{R}$. Equation (1.9) is called the *Bellman equation* for v_π, and it is a fundamental result both for model-based and model-free control. It expresses a relationship between the value of a state and the values of its successor states. We can visualize the meaning of this equation by thinking of looking ahead from a state s to its possible future states, as suggested by the diagram in Fig. 1.8.

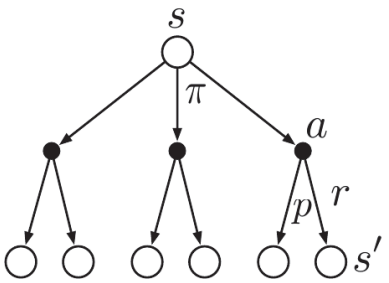


Figure 1.8: Diagram for v_π. Image from Ref. [57].

Each open circle represents a state and each solid circle an action. Starting from state s, the root node at the top, the agent could take any of some set of actions (three are shown in the diagram) based on its policy π. From each of these, the environment could respond with one of several next states, s' (two are shown in the figure), along with a reward, r, depending on its dynamics given by the function p. The Bellman equation (1.9) averages over all the possibilities, weighting each possibility by its probability of occurring. It states that the value of the starting state must equal the value of the expected next state, plus the reward expected along the way.

Policy evaluation

With Eq. (1.9) it is quite straightforward to derive a numerical scheme to compute the value function v_π for an arbitrary policy π. Consider a sequence of approximate value functions $v_0, v_1, v_2, \ldots$. The initial approximation, v_0, is chosen arbitrarily, and each successive approximation is obtained by using the Bellman equation for v_π (1.9) as an update rule:

$$\begin{aligned} v_{k+1}(s) &= \mathbb{E}_\pi \left[R_{t+1} + v_k(S_{t+1}) \mid S_t = s \right] \\ &= \sum_a \pi(a \mid s) \sum_{s',r} p(s', r \mid s, a) \left[r + v_k(s') \right] . \end{aligned} \tag{1.10}$$

Clearly, $v_k = v_\pi$ is a fixed point for this update rule because the Bellman equation for v_π assures us of equality in this case. The sequence $\{v_k\}$ can be shown in general to converge to v_π as $k \to \infty$ [57]. This algorithm is called *iterative policy evaluation*. One simple approach to check the convergence is to test the quantity $\max_s |v_{k+1}(s) - v_k(s)|$ at each iteration and to stop when it is sufficiently small. A sequential pseudocode to implement efficiently Eq. (1.10) is given in Appendix A.2.

Policy improvement

We have seen how to evaluate the performance of a policy π by computing its value function (with an uncertainty that is controlled by the convergence of the method discussed above). Now we will see how to find an optimal policy π_* which maximizes the expected return.

A policy π is defined to be better than or equal to a policy π' if its expected return is greater than or equal to that of π' for all states. In other words, $\pi \geq \pi'$ if and only if $v_\pi(s) \geq v_{\pi'}(s)$ for all states s in the system. There is always at least one policy that is better than or equal to all other policies, and this is an optimal policy [57]. This policy is not necessarily unique, but all optimal policies share the same value function, called the optimal value function, denoted v_*, and defined as

$$v_*(s) = \max_\pi v_\pi(s)\,. \tag{1.11}$$

Because v_* is the value function for a policy, it must satisfy the self-consistency condition given by the Bellman equation (1.9). Since it is the optimal value function, its consistency condition can be written in a special form without reference to any specific policy. This is the Bellman equation for v_*, or the *Bellman optimality equation*. Intuitively, it expresses the fact that the value of a state under an optimal policy must equal the expected return for the best action from that state:

$$v_*(s) = \max_a \sum_{s',r} p(s', r \mid s, a)\left[r + v_*(s')\right]\,. \tag{1.12}$$

Once one has v_*, it is relatively easy to determine an optimal policy. Indeed, for each state s, there will be one or more actions at which the maximum is obtained in the Bellman optimality equation. Any policy that chooses these actions in state s is an optimal policy. If we define the set of optimal actions in state s as

$$\mathcal{A}^*(s) = \operatorname*{argmax}_a \sum_{s',r} p(s', r \mid s, a)\left[r + v_*(s')\right]\,, \tag{1.13}$$

we can write such an optimal policy as

$$\pi_*(a \,|\, s) = I_{\mathcal{A}^*(s)}(a)\,, \tag{1.14}$$

where $I_{\mathcal{A}^*(s)}(a)$ is the indicator function of $\mathcal{A}^*(s)$, i.e. a function that maps the elements a belonging to $\mathcal{A}^*(s)$ to 1 and the others to 0.

For finite MDPs, i.e. when the sets of states, actions, and rewards ($\mathcal{S}$, $\mathcal{A}$, and $\mathcal{R}$) all have a finite number of elements, Eq. (1.12) has a unique solution. This equation is actually a system of $|\mathcal{S}|$ equations in $|\mathcal{S}|$ unknowns. Here, $|\mathcal{S}|$ is the number of states in the set of states $\mathcal{S}$, and the unknowns are the optimal values $v_*(s)$. If the dynamics p of the environment are known, then in principle one can directly solve this system of equations for v_* based on an exhaustive listing of all possible combinations of actions. When the number of states is high, however, this direct procedure can become difficult or even impossible.

The body of methods that have been developed to solve Eq. (1.12) belongs to the field of *dynamic programming* (DP). The term dynamic programming was originally coined in the 1950s by Richard Bellman[4] to describe the mathematical optimization of multistage decision processes [60]. Nowadays, it refers more specifically to a collection of algorithms that can be used to compute optimal policies given complete knowledge of a MDP [57]. For the sake of concision, we will report here only the DP method that we used in this work, called *value iteration*.

In a similar way as we did for policy evaluation, we can turn the Bellman optimality equation (1.12) into an update rule, to obtain

$$v_{k+1}(s) = \max_a \sum_{s',r} p(s', r \mid s, a)\left[r + v_k(s')\right]\,. \tag{1.15}$$

[4] A small curiosity about the origin of the term comes directly from Bellman's autobiography [59], where he says: "Let's take a word that has an absolutely precise meaning, namely dynamic, in the classical physical sense. It also has a very interesting property as an adjective, and that is it's impossible to use the word, dynamic, in a pejorative sense. Try thinking of some combination that will possibly give it a pejorative meaning. It's impossible. Thus, I thought dynamic programming was a good name."

Note that the value iteration update is identical to the iterative policy evaluation update (1.10), except that it requires a maximum to be taken over all actions at each iteration. Like policy evaluation, value iteration formally requires an infinite number of iterations to converge exactly to v_*. In practice, we stop once the value function changes by only a small amount after an iteration. A pseudocode for a program that implements Eq. (1.15) is given in Appendix A.2.

Tabular and approximate solution methods

An important remark should be made at this point. If the set of states $\mathcal{S}$ is *discrete*, finite and small enough for the value function to be represented as an array, or a *table*, then the control methods described above can be shown to converge to an exact solution [57]. That is, they can find with arbitrary precision the optimal value function and the associated optimal policy. This is called the *tabular* case, and the corresponding algorithms are called tabular methods.

If the set of states is *continuous*, or if it is discrete but there are infinitely many or far more states than there could possibly be entries in a table stored in a computer, then handling such an array becomes impossible. This issue proved to be relevant to the work presented in Chapter 4 of this thesis, where we dealt with continuous deterministic systems. In this case, there are several options to deal with the problem.

A first option is to maintain a model-based approach and fix the initial condition of the system. With a deterministic model and a known initial condition, it is no longer necessary to observe the state of the system to decide what action to take, since each sequence of actions corresponds to a trajectory in the space of states. This approach, called *open-loop*, simplifies the control problem, since it requires to compute only one trajectory of the dynamics at a time, which is easier than solving the continuous equivalent of the Bellman optimality equation for all states. We have not investigated this approach.

Another option is to use methods to approximate the continuous state space. Algorithms of this type belong to the class of *approximate solution methods*. This is, in fact, the direction we have chosen to control continuous systems. We have chosen to develop this approach within the frame of model-free methods. Later in this section, we will take a closer look at how these techniques work and introduce *deep Q-learning*, the approximate control algorithm we used for our work.

1.4.3 Model-free control

Unlike the previous section, with model-free control methods we do not assume complete knowledge of the dynamics of the environment[5]. These kind of methods require only experience—sample sequences of states, actions, and rewards from actual or simulated interaction with an environment. Of course, when dealing with a *simulated* environment, a model is still required to generate the dynamics, however, model-free control algorithms do not have access to the model, but only to sample observations of trajectories.

The simplest model-free control techniques are those of Monte Carlo (MC) methods, which are ways of solving the MDP based on averaging sample returns. For simplicity, we consider only *episodic* tasks, that is, we assume that experience is divided into episodes, and that all episodes eventually terminate at a final time t_f, either because the environment reached a target state (which in this case corresponds to a so-called *terminal* state), or because, even if the target state has not been reached, there is a maximum time limit at which the dynamics is stopped.

[5]More precisely, we do not assume any knowledge of the dynamics beyond its Markovian character. However, we have to assume complete knowledge of a class of states and actions that covers all possible observations and decisions.

Monte Carlo estimation of action values

We begin by considering MC methods for learning the state-value function for a given arbitrary policy. Recall that the value of a state is the expected return starting from that state. An obvious way to estimate it from experience, then, is simply to average the returns observed after visits to that state. As more returns are observed, the average will converge to the expected value.

In particular, suppose we wish to estimate $v_\pi(s)$, the value of a state s under policy π, given a set of episodes obtained by following π and passing through s. Each occurrence of state s in an episode is called a *visit* to s. Then, a straightforward way of estimating $v_\pi(s)$ is to average the returns of the visits to s.

However, in model-free methods, it is particularly useful to estimate *action* values (the value associated to each specific action in a state) rather than *state* values. Indeed, with a model, state values alone are sufficient to determine a policy, because the dynamics function p can be used to predict whichever action leads to the best combination of reward and next state, as in model-based methods (see Eq. (1.12)). Without a model, instead, the function p is not available and one must explicitly estimate the value of each action in order for the values to be useful in computing a policy.

Similarly to how we defined the state-value function $v_\pi(s)$ in Eq. (1.8), we can define the action-value function $q_\pi(s, a)$ as the expected return starting from s, taking the action a, and thereafter following policy π:

$$q_\pi(s, a) = \mathbb{E}_\pi \left[G_t \mid S_t = s, A_t = a \right] . \tag{1.16}$$

The policy evaluation problem for action values is to estimate $q_\pi(s, a)$. The MC method for this is essentially the same as just discussed for state values, except now we talk about visits to a state-action pair rather than to just a state. A state-action pair s, a is said to be visited in an episode if ever the state s is visited and action a is taken in it.

Suppose we have a sequence of returns $G^{(1)}, G^{(2)}, \ldots, G^{(n)}$, all starting from the same state-action pair s, a. Then, the MC estimate of the action-value function for the pair after n visits, that we denote with $Q^{(n)}$, is simply

$$Q^{(n)} = \frac{1}{n} \sum_{i=1}^{n} G^{(i)} . \tag{1.17}$$

There is a complication, though: many state-action pairs may never be visited. Indeed, if for example π is a deterministic policy, then in following π the agent will observe returns only for one of the actions from each state. With no returns to average, the estimates of the other actions will not improve with experience, and this is a problem because, to compare alternatives and learn an optimal policy, the agent needs to estimate the value of all the actions from each state. This is in essence the problem of *exploration* in model-free control tasks.

For policy evaluation to work for action values, we must assure continual exploration. One way to do this is by specifying that the episodes start in a state-action pair, and that every pair has a nonzero probability of being selected as the start. This guarantees that all state-action pairs will be visited an infinite number of times in the limit of an infinite number of episodes. This method is called *exploring starts*.

Monte Carlo control

For learning an optimal policy using MC estimates, the basic idea is to alternate steps of policy evaluation and policy improvement. The approximated action-value function Q_π is repeatedly altered to approximate more closely the action-value function for the current policy,

and simultaneously the policy π is improved with respect to the current action-value function, as suggested by the diagram in Fig. 1.9. After enough iterations of this scheme, both policy and value function will approach optimality.

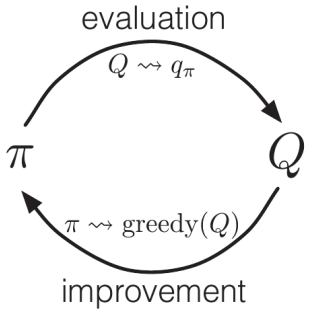


Figure 1.9: General scheme to obtain an optimal policy by alternating evaluation and improvement. Image from Ref. [57].

Policy evaluation is done as described in the previous section. Policy improvement is done by making the policy *greedy* with respect to the current estimate of the action-value function. For any action-value function q, the corresponding greedy policy is the one that, for each $s \in \mathcal{S}$, deterministically chooses an action with maximal action-value $\text{argmax}_a\, q_\pi(s,a)$.

This deterministic policy update rule, however, has the risk of not encouraging enough exploration. For a finite number of episodes, even if we allow for the assumption of exploring starts, there are still some state-actions pairs that may be selected too infrequently to be properly evaluated. To avoid this problem, one can consider *ε-greedy* policies, such a policy by definition chooses most of the time an action that has maximal estimated action value, but with probability ε it selects instead an action at random. That is

$$\pi(a \,|\, s) = \begin{cases} 1 - \varepsilon + \varepsilon/|\mathcal{A}| & \text{if } a = \arg\max_a q_\pi(s,a)\,, \\ \varepsilon/|\mathcal{A}| & \text{otherwise } (a \text{ is not optimal})\,. \end{cases} \tag{1.18}$$

The introduction of ε-greedy policies brings us to one of the central issues of model-free methods: the trade-off between exploration and exploitation. To obtain a high reward, the agent must prefer actions that it has tried in the past and found to be effective in producing reward. But to discover such actions, it has to try actions that it has not selected before. Hence, the agent has to *exploit* what it has already experienced in order to obtain reward, but it also has to *explore* in order to make better action selections in the future.

If we use a purely greedy policy ($\varepsilon = 0$), then we are only exploiting our current knowledge of the values of the actions. Conversely, if we use a completely random policy ($\varepsilon = 1$), then we are just exploring, because this enables us to improve our estimate of the values of the non-optimal actions. The agent must try a variety of actions and progressively favour those that appear to be best. In mathematical terms, this translates into a gradual decay of the value of ε as learning proceeds.

A simple strategy is to alternate between evaluation and improvement on an episode-by-episode basis. After each episode, the observed returns are used for policy evaluation, and then the policy is improved at all the states visited in the episode. A complete algorithm along these lines is given in pseudocode in Appendix A.2.

As an additional note, we would like to add that the control method presented here is not the only one based on Monte Carlo estimates of action values. Technically, this method is part of the

so-called *on-policy* methods, in the sense that it tries to evaluate and improve the same policy that is used to make decisions and generate experience. This is the main reason why we had to introduce the ε-greedy policy, in order to ensure that the agent still explores all available actions, progressively exploiting those that are estimated to be optimal. This is not the only approach to tackle the problem of exploration. There are methods, called *off-policy*, where the idea is to use two policies, one that is learned and becomes the optimal policy (and that does not have to be ε-greedy), and one that is more exploratory and is used to generate experience from interactions with the environment.

Let us now look at one model-free control technique that is slightly more sophisticated than the simple MC learning scheme. This method is called *Q-learning* and it incorporates ideas from the MC approach and the DP approach seen in Section 1.4.2.

Q-learning

The explanation of Q-learning that we report here is quite informal, we just want to give a general idea of how the method works. For a detailed treatment of the topic, the interested reader can refer to the original publication by Watkins [61] or to the book by Sutton and Barto [57].

Let us look at the MC estimate formula of Eq. (1.17). In order to apply it, we need to maintain, for each state-action pair s, a, the cumulative sum of returns $\sum_i G^{(i)}$, and this can be computationally costly. However, we can write an incremental implementation of that formula:

$$\begin{aligned} Q^{(n)} &= \frac{1}{n}\sum_{i=1}^{n} G^{(i)} = \frac{1}{n}\left[G^{(n)} + \sum_{i=1}^{n-1} G^{(i)}\right] = \\ &= \frac{1}{n}\left[G^{(n)} + (n-1)Q^{(n-1)}\right] = Q^{(n-1)} + \frac{1}{n}\left[G^{(n)} - Q^{(n-1)}\right] . \end{aligned} \tag{1.19}$$

This implementation of the MC estimate $Q^{(n)}$ requires memory only for the old estimate $Q^{(n-1)}$ and n. This update rule is of a form that occurs frequently in model-free methods. This form is

$$\mathit{NewEstimate} \leftarrow \mathit{OldEstimate} + \mathit{StepSize}\,[\mathit{Target} - \mathit{OldEstimate}] , \tag{1.20}$$

where the arrow represents an assignment. At every iteration (i.e. when a new return for the s, a pair is observed), the old estimate is updated by taking a step toward the "target", which is presumed to indicate a desirable direction in which to move, though it may be noisy. In the case above, the target is the n-th return.

Using Eq. (1.20), we can generalise Eq. (1.19) for the MC estimate of the action-value function to an arbitrary step size. For every state-action pair S_t, A_t observed at a time t during an episode, with the corresponding return G_t, the general update rule is

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\,[G_t - Q(S_t, A_t)] , \tag{1.21}$$

where α is a constant step-size parameter between 0 and 1, called *learning rate*. This formula is basically the same as Eq. (1.19), with the difference that the step-size parameter is now a small constant.

In order to determine the increment to $Q(S_t, A_t)$, MC methods must wait until the end of the episode, because only then G_t (i.e. the sum of rewards starting from t up to the final time t_f) is known. However, by making a simple assumption, we can drop this constraint and write a formula that allows us to obtain updates at each time step during the episode. The assumption is that we can estimate $G_t = R_{t+1} + \beta G_{t+1}$ with the quantity $R_{t+1} + \beta Q(S_{t+1}, A_{t+1})$, where we have introduced the discount factor β for generality (see footnote on page 11). In other words,

we are approximating the return starting from the next time step G_{t+1} with the estimate of the action-value function of the next state-action pair $Q(S_{t+1}, A_{t+1})$. By inserting this estimate in Eq. (1.21), we get

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha \left[R_{t+1} + \beta Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t) \right] . \tag{1.22}$$

At each transition from a state-action pair S_t, A_t, we form a new target and make an update using the observed reward R_{t+1} and the estimate $Q(S_{t+1}, A_{t+1})$. To obtain a control scheme based on this update rule, we just need to alternate between evaluation of the action-value function and improvement of the policy, as discussed earlier for MC methods. The resulting control algorithm is called *Sarsa*. Sarsa is an on-policy method.

We now introduce Q-learning, an off-policy method, which effectively uses two policies: one, non-optimal, to generate experience, and one that is instead optimal, for which the update rule estimates the action-value function q_*. The Q-learning algorithm is basically a modified version of Eq. (1.22):

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha \left[R_{t+1} + \beta \max_a Q(S_{t+1}, a) - Q(S_t, A_t) \right] . \tag{1.23}$$

The intuition behind this formula is that now the learned action-value function Q directly approximates q_*, the optimal action-value function, independent of the policy being followed to generate experience. The exploration policy has still to provide for sufficient exploration of state space, i.e. to ensure that all state-action pairs are visited frequently enough. Under this assumption, the estimate Q has been shown to converge with probability 1 to q_* [61]. Once we have q_*, an optimal deterministic policy is readily obtained by assigning the best action $\text{argmax}_a \, q_*(s, a)$ for each state s in $\mathcal{S}$. A pseudocode for a program that implements Q-learning is given in Appendix A.2.

Deep Q-learning

So far, we have assumed to be dealing with a finite and discrete set of states, thus having the possibility to apply tabular control algorithms. In this last part, we discuss approximate control methods, which allow to handle a continuous or infinitely large set of the states.

We have already mentioned that managing large state spaces in a tabular way is practically impossible. The problem is not just the computer memory needed to store large arrays, but the time and data needed to fill them. If the state space is too large (or infinite), almost every state encountered will never have been seen before. For an agent to make sensible decisions in such states it is necessary to generalize from previous encounters with different states that are in some sense similar to the current one.

The key issue when dealing with an infinite state space is that of generalization: experience with a limited subset of state space has to be usefully generalized to produce a good approximation over a much larger subset. One way to address this problem is to represent the action-value function not as a table but as a parametrized functional form $\hat{Q}(s, a, \boldsymbol{w})$ with weight vector $\boldsymbol{w}$. For example, $\hat{Q}$ might be a linear function in features of the state, with $\boldsymbol{w}$ the vector of feature weights. One may also resort to nonlinear approximation methods. As an example, $\hat{Q}$ might be the function computed by an artificial neural network, with $\boldsymbol{w}$ the vector of connection weights in all the layers.

The main difference from the tabular implementation of Q-learning is that, in the tabular case, the update rule for the action-value function of a pair s, a only affects the table entry $Q(s, a)$ of this pair, which is shifted a fraction of the way towards the target, while the estimated value of all other pairs is left unchanged. Now, in contrast, the update is performed on the weights

that parametrize the q function, and the update after the observation of a pair s, a generalizes so that the estimated values of many other states-action pairs are changed as well.

What we need now is an update rule for the weight vector $\boldsymbol{w}$ that makes the approximate action-value function $\hat{Q}(s, a, \boldsymbol{w})$ come closer to the true $q(s, a)$. A natural way to do this is by implementing a stochastic gradient-descent update rule in the space of weights that decreases the squared error

$$\text{Error} = \sum_{s,a} [q(s, a) - \hat{Q}(s, a, \boldsymbol{w})]^2 \,. \tag{1.24}$$

The word "stochastic" comes from the randomness of the order in which state-action pairs are selected. The direction of the update is computed at each observation of a pair. If we assume that we can update $\boldsymbol{w}$ each time that we observe a state-action pair S_t, A_t, and we call $\boldsymbol{w}_t$ the weight vector at each timestep, then such a rule is

$$\begin{aligned} \boldsymbol{w}_{t+1} &= \boldsymbol{w}_t - \frac{1}{2}\alpha \nabla_{\boldsymbol{w}} \left[q(S_t, A_t) - \hat{Q}(S_t, A_t, \boldsymbol{w}_t) \right]^2 \\ &= \boldsymbol{w}_t + \alpha \left[q(S_t, A_t) - \hat{Q}(S_t, A_t, \boldsymbol{w}_t) \right] \nabla_{\boldsymbol{w}} \hat{Q}(S_t, A_t, \boldsymbol{w}_t) \,, \end{aligned} \tag{1.25}$$

where α is again a positive step-size parameter, and $\nabla_{\boldsymbol{w}} \hat{Q}$ denotes the vector of partial derivatives of $\hat{Q}$ with respect of the components of $\boldsymbol{w}$. This equation can be rewritten in the more general form

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t + \alpha \left[Y_t - \hat{Q}(S_t, A_t, \boldsymbol{w}_t) \right] \nabla_{\boldsymbol{w}} \hat{Q}(S_t, A_t, \boldsymbol{w}_t) \,, \tag{1.26}$$

where Y_t represent the target of the update. Applying the Q-learning update rule Eq. (1.23) to the parameters $\boldsymbol{w}$ results in the estimated target

$$Y_t^{\mathrm{Q}} = R_{t+1} + \beta \max_a \hat{Q}(S_{t+1}, a, \boldsymbol{w}_t) \,. \tag{1.27}$$

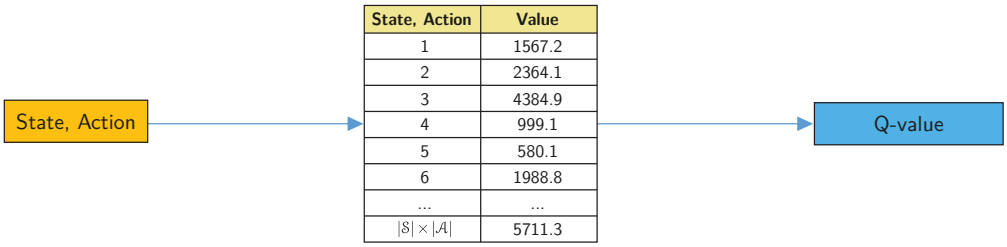
The method that we have used in Chapter 4 belongs to a class of algorithms combining Q-learning and deep neural networks, called *deep Q-learning*. This class of methods have been successfully applied to solve complex problems such as computer games, sometimes exceeding human performances [62, 63].

There are several possible ways of parametrizing the action-value function q using a neural network. Because q maps state-action pairs to scalar estimates of their action-value (also called the Q-value), the state and the action have been used as inputs to the neural network by some approaches [64, 65]. This type of architecture has, however, the drawback of having a computational cost that scales linearly with the number of actions.

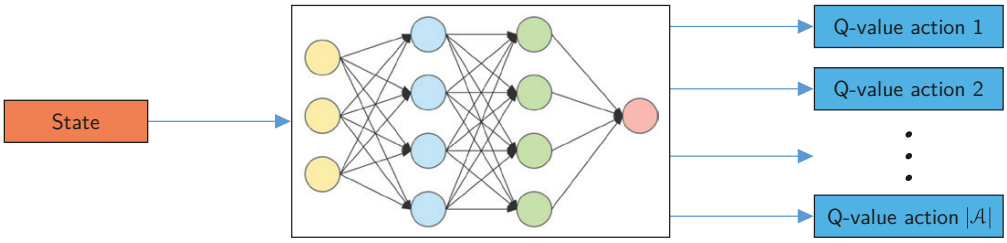
To address this issue, another type of architecture has been proposed [66], called *deep Q network* (DQN). In this architecture, there is a separate output unit for each possible action, and only the state representation is an input to the neural network. The outputs correspond to the predicted Q-values of the individual actions for the input state. More specifically, a DQN is a multi-layered (deep) neural network with weights $\boldsymbol{w}$, that for a given state s outputs a vector of action values $\hat{Q}(s, \cdot, \boldsymbol{w})$. The difference between the tabular representation of the action-value function and the approximation through a DQN is illustrated in Fig. 1.10.

Two important ingredients for an efficient DQN-based learning algorithm have been proposed by Mnih *et al.* [66]: the use of a *target network*, and the use of *experience replay*. The target network is another DQN which has exactly the same structure as $\hat{Q}$, except that it has different weights $\boldsymbol{w}^-$. The target network is used to form the target

$$Y_t^{\mathrm{DQN}} = R_{t+1} + \beta \max_a \hat{Q}(S_{t+1}, a, \boldsymbol{w}_t^-) \,, \tag{1.28}$$



(a) Tabular representation of the Q function.



(b) Approximation of the Q function through a DQN.

Figure 1.10: Differences in the management of the Q function according to the sizes of the state space and the action space. (a) When these spaces are of reasonable size, the Q function can be modelled by an array of dimensions $|\mathcal{S}| \times |\mathcal{A}|$ storing the values of the state-action pairs. (b) When the dimension of one of the two spaces (or both) is too large, it is possible to use an approximation function such as a DQN, which returns, for a given state, the set of values of the different associated state/action pairs.

and its parameters are are periodically updated to match $\boldsymbol{w}$, so that $\boldsymbol{w}_t^- = \boldsymbol{w}_t$ only at certain timesteps t, and kept fixed on all other steps. The idea behind this modification is that the standard Q-learning target Eq. (1.27) depends on the current estimate of q, which changes over time (this is known as a *non-stationary* target). This can make the algorithm unstable, contrary to supervised learning algorithms, for example, where the target is known and constant. Switching to a target that varies less frequently in time increases the stability of the algorithm.

The second ingredient is experience replay [67], where observed transitions $S_t, A_t \to R_{t+1}, S_{t+1}$ are stored for some time in a *buffer* memory and then sampled from this memory. This allows to refine the update of the network weights in Eq. (1.26) by replacing the gradient with a stochastic estimate of the gradient on a *mini-batch* (i.e. a small list) of transitions sampled uniformly from the buffer. As the memory size is finite, the oldest transitions are replaced by new ones once the memory is full. The target network and the experience replay dramatically improve the performance of the DQN method [66].

There is one more improvement that we need to consider in order to obtain the final method that we have used. The max operator used in standard Q-learning (Eq. (1.27)) and in DQN (Eq. (1.28)) uses the same values both to *select* and to *evaluate* an action. This makes it more likely to select overestimated values, resulting in overoptimistic value estimates. To prevent this, we can decouple the selection of the optimal action from its evaluation. In this method, called *double Q-learning* [68], two different value functions are learned by assigning experiences randomly to update one of the two value functions, resulting in two sets of weights, $\boldsymbol{w}$ and $\boldsymbol{w}'$. For each update, one set of weights is used to determine the greedy policy (i.e. the optimal action) and the other to determine its value.

For clarity, we can untangle the selection and the evaluation in Q-learning and rewrite the

target of Eq. (1.27) as

$$Y_t^{\mathrm{Q}} = R_{t+1} + \beta \hat{Q}(S_{t+1}, \operatorname*{argmax}_a \hat{Q}(S_{t+1}, a, \boldsymbol{w}_t), \boldsymbol{w}_t)\,. \tag{1.29}$$

The double Q-learning target can then be written as

$$Y_t^{\mathrm{DoubleQ}} = R_{t+1} + \beta \hat{Q}(S_{t+1}, \operatorname*{argmax}_a \hat{Q}(S_{t+1}, a, \boldsymbol{w}_t), \boldsymbol{w}'_t)\,, \tag{1.30}$$

where the weights $\boldsymbol{w}_t$ are used for the selection of the optimal action (i.e. the argmax operator) while the second set of weights $\boldsymbol{w}'_t$ is used to fairly evaluate the value of the policy. This second set of weights can be updated periodically by switching the roles of $\boldsymbol{w}$ and $\boldsymbol{w}'$.

Instead than introducing another DQN with weights $\boldsymbol{w}'$, van Hasselt *et al.* [69] propose to use the target network that we introduced above, with weights $\boldsymbol{w}^-$. The resulting algorithm is called *double DQN*, and it is the one that we have used to control continuous systems. Its update rule is the same as DQN, but replacing the target Y_t^{DQN} with

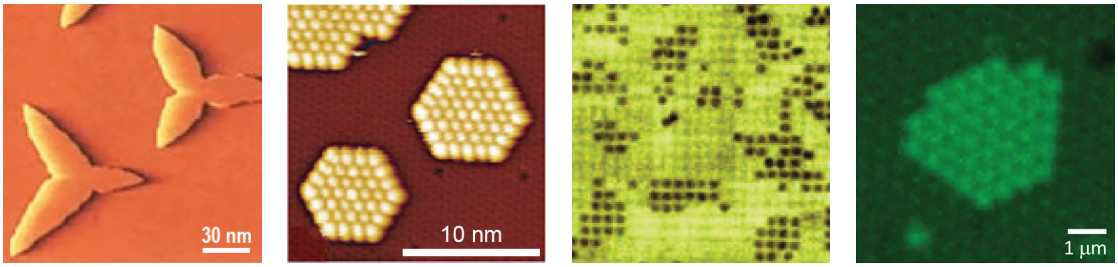
$$Y_t^{\mathrm{DoubleDQN}} = R_{t+1} + \beta \hat{Q}(S_{t+1}, \operatorname*{argmax}_a \hat{Q}(S_{t+1}, a, \boldsymbol{w}_t), \boldsymbol{w}_t^-)\,. \tag{1.31}$$

In comparison to double Q-learning Eq. (1.30), the weights of the second network $\boldsymbol{w}'_t$ are replaced with the weights of the target network $\boldsymbol{w}_t^-$ for the evaluation of the current greedy policy. The update to the target network stays unchanged from DQN, and remains a periodic copy of $\hat{Q}(s, \cdot, \boldsymbol{w})$.

CHAPTER 2

Model-based control and equilibrium dynamics of fluctuating 2d clusters

In this chapter, we consider the case of model-based control applied to discrete stochastic systems. Our aim is to control the morphology of a fluctuating few-particle monolayer cluster. More precisely, we investigate how an arbitrary target shape of the cluster can be reached in minimum time in the presence of thermal fluctuations and with or without a macroscopic external field. Some examples of 2d monolayer clusters made of Pt atoms interacting by metallic bonding, C_{60} nanopartices (fullerenes) bound by van der Waals interactions, and micron-sized PS colloids held together by depletion forces are shown in Fig. 2.1.



(a) Image from Ref. [34]. (b) Image from Ref. [70]. (c) Image from Ref. [71]. (d) Image from Ref. [37].

Figure 2.1: Examples of particle clusters at different scales. (a) Atomic monolayer island: Pt on Pt(111), (b) clusters of C_{60} nanoparticles (fullerenes), (c) cluster of polystyrene (PS) particles (diameter ~ 1 µm) on a square lattice and (d) on an hexagonal lattice.

A cluster subjected to thermal fluctuations is a discrete system that evolves over time, stochastically exploring different configurations, or shapes, due to the random hopping motion of particles. Thus, its dynamics can be seen as a *random walk* in the space of cluster configurations, which we identify with the states of the system. Our question is then recast into a *first-passage* problem on the state space of the cluster, which we solve numerically. We first study the equilibrium dynamics in the absence of external forces, and we obtain the striking result that large

compact target shapes exhibit an optimal temperature at which they are reached in minimum time.

Then, using the value iteration algorithm introduced in Section 1.4.2, we compute the closed-loop optimal policy that sets the external field as function of the cluster shape in order to minimize the time to reach a desired target configuration, at a fixed temperature. This leads to a gain in the time to reach the target that grows when increasing cluster size or decreasing temperature. We find that this gain can shift the optimal temperature to lower temperatures, or even create one when it is not present in the absence of the external field.

We also show that the optimal policy is not unique. This non-uniqueness appears to be mainly dictated by symmetry properties. In addition, we find that the optimal policy can undergo transitions when the temperature is varied. The non-uniqueness and the number of temperature transitions both increase drastically with the size of the cluster. Due to computational constraints, we limit our study to clusters consisting of up to 12 particles. Then, with the help of a high-temperature expansion, we discuss and interpret our findings and speculate on the generalization of our results for clusters larger than 12 particles.

We focus on the case of a monolayer cluster of particles on a square lattice with *edge diffusion*, i.e. the movement of particles at edge or kink sites, where the mobile particle maintains its proximity to the cluster throughout the process. Edge diffusion has been observed with STM in atomic monolayer islands of various types of metals. It is dominant at low enough temperatures, when detachment of atoms from the edge is negligible. Some examples of systems where edge diffusion has been observed include Cu on Cu(001) below 600 K [72] and on Cu(111) below 500 K [73], Pt on Pt(111) below 800 K [74], and Ag on Ag(111) below 450 K [75, 35]. This process also appears in colloidal clusters, in particular for ligand-coated Au nanoparticles of a few nanometers in diameter in drop-drying experiments [76, 36].

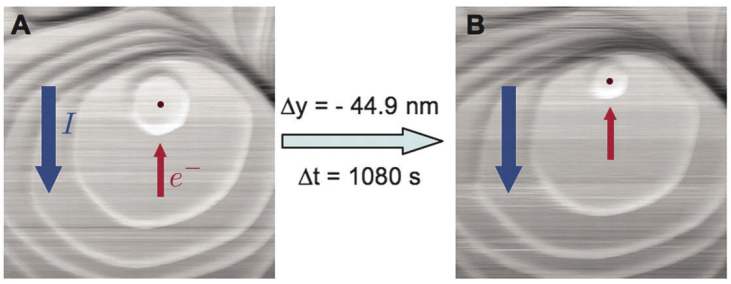
We consider the case of metal islands under electromigration as a first candidate of experimental system where our approach could be applied. Electromigration refers to the phenomenon of current-induced mass transport: when electrical current passes through a conductor, the current carriers can scatter on atoms at interfaces and transfer momentum to atoms in these regions, resulting in biased diffusion of mobile atoms. When mass transport occurs via edge diffusion, this ultimately results in an overall island displacement in the direction of the electron flow, as shown in Figs. 2.2a and 2.2b.

Experimental studies [35] have shown that this phenomenon can lead to a biased island displacement of tens of nanometers, proving that the position of a nanometric island on a surface can be significantly controlled by using an external electric field. On the other hand, kinetic Monte Carlo simulations and continuum models [24, 77, 23, 22] have shown that not only the position of the island can be influenced by the effects of electromigration, but also its shape, as shown in Fig. 2.2c. For islands that are large enough, these morphological changes can trigger a splitting instability, leading to island breakup. This instability appears for islands with an average radius greater than a critical value

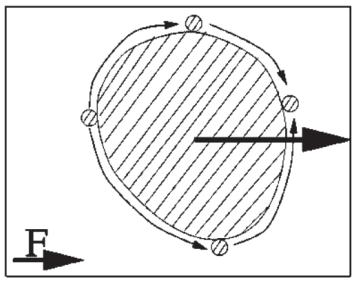
$$R_c = \left(\chi_c \frac{l^2 \tilde{\beta}}{F_0} \right)^{1/2} , \tag{2.1}$$

where $\chi_c \approx 10.65$ is a constant, l is the lattice constant, $\tilde{\beta}$ is the step stiffness, and F_0 is the magnitude of the external force. To get an idea of the order of magnitude of R_c, we can insert the values $l = 0.288$ nm, $\tilde{\beta} = 0.65$ eV/nm and $F_0 = 1.3 \times 10^{-2}$ meV/nm for Ag islands on Ag(111) [78, 35], and obtain $R_c \approx 210$ nm, corresponding to ~ 730 lattice constants.

In our study we do not observe this instability, since we focus on very small clusters of a few atoms consisting of at most 12 atoms. Nevertheless, even if these small islands are morphologically



(a) (500×500 nm^2) Biased displacement of a monolayer Ag island driven by electromigration at T = 318 K. An external electric field induces a current with downward direction, the island displacement is hence upward, in the same direction of the electron flow. Image from Ref. [35].



(b) Scheme of biased edge diffusion. Image from Ref. [24].

(c) Kinetic Monte Carlo simulation showing the emergence of the splitting instability, for islands that are above a critical size. Image from Ref. [24].

Figure 2.2: (a) STM image of electromigration-induced displacement of a metal monolayer island. (b) Schematics of edge diffusion biased by an external force. (c) A simulation of the morphological instability appearing for islands with an average radius greater than R_c.

stable, their thermal fluctuations can be used to reach a target shape. Our strategy is to use the external force when the resulting bias is sufficiently strong to reduce significantly the time to reach the target. When the force is too weak, we resort to a different strategy and let thermal fluctuations randomly drive the cluster towards the target shape. We then ask the question of finding if there is an optimal temperature where this occurs in minimum time.

Our model is inspired from the case of metal atomic islands under electromigration, but it is adapted to describe in general any fluctuating particle cluster with biased edge diffusion. In the case of colloidal or nanoparticle clusters, the biasing force could be imposed, for example, by means of a temperature gradient (thermophoresis) [79], an electric field (electrophoresis) [28] or a magnetic field [80].

It should be noted that the orders of magnitude involved in these different systems can vary significantly. One relevant quantity is $F_0 l/J$, where J is the energy of the bond between two particles. This dimensionless number represents the typical ratio between the work done by the force for a nearest-neighbor particle movement on the lattice and the energy to break a bond. The higher this ratio, the larger the effect of the force on the dynamics of the cluster. For electromigration of atomic islands, this quantity is of the order of 10^{-4} [35]. For nanoparticle clusters under thermophoresis, it can be higher, up to 10^{-1} for fullerenes [81]. For colloids, the bond energy J can be fine-tuned using a variety of techniques (e.g. depletion forces [37], surface functionalization [82], or optical lattices [12]). For example, using depletants, one can obtain J around a few units of thermal energy $k_B T$ [37]. Thermophoretic forces [79, 83, 84] for PS beads of radius 2.5 µm are $F_0 \approx 10\ k_B T/\mu$m [79]. Hence, micron-size colloids can lead to a ratio $F_0 l/J$

of the order of one.

Another relevant dimensionless number is the ratio between the thermal energy in the system and the binding energy k_BT/J. As we already discussed in Chapter 1, nanomanipulation experiments that rely on control of single particles are usually performed in the low-temperature regime $k_BT/J \ll 1$, to reduce thermally-induced diffusion. For our purposes, however, we want to use thermal fluctuations to our advantage, so we do not limit ourselves to this regime, but instead consider a wide range of temperatures.

2.1 Lattice model

We now present a two-dimensional lattice model to describe the dynamics of a fluctuating cluster under an external field and formulate our problem as a Markov decision process (MDP), following the formalism introduced in Section 1.4.1. We start by defining the dynamical model of our environment, i.e. the function $p(s', r \mid s, a)$ that gives the probability of transitioning to state s' and receiving reward r when being in state s and taking action a, along with the policy π, that in this framework corresponds to the strategy to set the external field in order to reach a target state. Then, we define the value function $v(s)$ and the associated Bellman equation. For this problem, the value of a state is minus the expected time to reach the target configuration, which is the quantity that we want to minimize.

2.1.1 Formalization as an MDP

We consider a small cluster on a two-dimensional square lattice with lattice parameter l and nearest-neighbor bonds J under an external force that biases diffusion, following the same lines as in previous modelling of monolayer clusters in the presence of electromigration [24, 85]. A state s is associated to each cluster configuration on the lattice. The state s can change to another state s' via the motion of a single particle to one of its nearest or next-nearest neighbor sites, for a total of 8 possible moves. Moves are allowed only if they do not break the cluster. As shown in Fig. 2.3a, this leads to edge diffusion dynamics.

We use a closed-loop control approach, so that the actions taken by the agent are chosen based on the observation of the current state s, i.e. the shape of the cluster. The actions correspond to the choice of the external force in each state s, and are encoded in the deterministic policy

$$\pi(\boldsymbol{a} \mid s) = \delta_{\boldsymbol{a}-\boldsymbol{\phi}(s)} = \begin{cases} 1 & \text{if } \boldsymbol{a} = \boldsymbol{\phi}(s)\,, \\ 0 & \text{otherwise}\,, \end{cases} \tag{2.2}$$

where $\boldsymbol{\phi}(s)$ is a function that assigns to state s the force provided by action $\boldsymbol{a}$[1]. In the case of a deterministic policy that is considered here, the function $\boldsymbol{\phi}(s)$ completely characterizes $\pi(\boldsymbol{a} \mid s)$, and, for convenience, we also refer to $\boldsymbol{\phi}$ as the "policy". For simplicity, we set the force to be always oriented along the $\hat{\boldsymbol{x}}$ direction, and allow only 3 possible values. The set of actions—i.e. the possible forces in each state—is then $\mathcal{A} = \{-F_0\hat{\boldsymbol{x}}, \boldsymbol{0}, F_0\hat{\boldsymbol{x}}\}$, with $F_0 > 0$.

The dynamics of the cluster is governed by a master equation that describes the time evolution of the probability density function $P_{\boldsymbol{\phi}}(s,t)$ to find the cluster in state s at time t, when the force in s is set according to $\boldsymbol{\phi}(s)$ [86]

$$\frac{\partial P_{\boldsymbol{\phi}}(s,t)}{\partial t} = \sum_{s' \in \mathcal{S}} \left[P_{\boldsymbol{\phi}}(s',t)\gamma_{\boldsymbol{\phi}}(s',s) - P_{\boldsymbol{\phi}}(s,t)\gamma_{\boldsymbol{\phi}}(s,s') \right] , \tag{2.3}$$

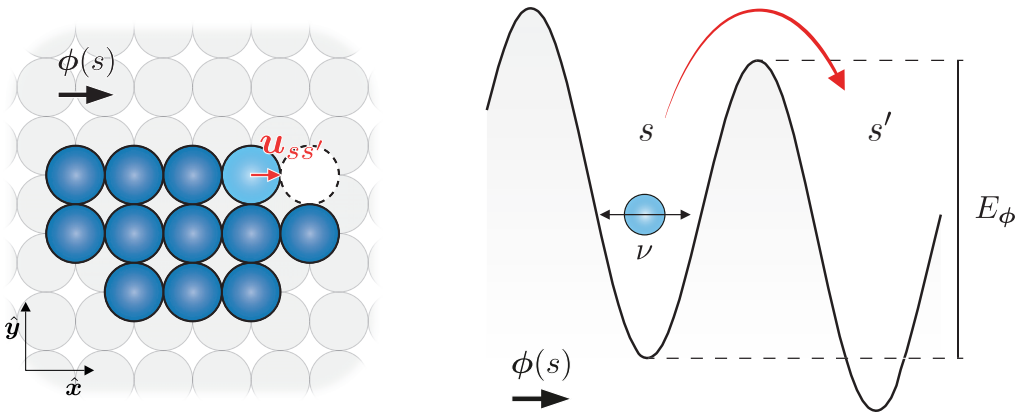
[1] To simplify the notation and avoid the use of a function that maps action indices to force values, we define actions directly as the values of the external force. With this definition, actions are therefore vectors.

where the sum runs over all system states $\mathcal{S}$ and $\gamma_{\boldsymbol{\phi}}(s, s')$ is the average transition rate from state s to state s', again when the force in s is given by $\boldsymbol{\phi}(s)$. The states of the system correspond to the possible configurations of the cluster. Transitions from one state to another are caused by atomic moves. As we will see in a moment, to each atomic jump along the edge corresponds exactly one state transition (for clusters of size larger than 2 atoms). We can therefore identify $\gamma_{\boldsymbol{\phi}}(s, s')$ with the atomic hopping rates. Because of the Markovian nature of the dynamics, these quantities are independent of the system history and thus an exclusive function of the properties of the two states s and s' involved.

Following usual models for biased diffusion based on transition state theory [87, 85, 24], the atomic hopping rate is assumed to take an Arrhenius form

$$\gamma_{\boldsymbol{\phi}}(s, s') = \nu \exp(-E_{\boldsymbol{\phi}}(s, s')/k_B T)\,, \tag{2.4}$$

where ν is the attempt frequency, a constant related to the vibration of the particle near its equilibrium position. Attempt frequencies for surface diffusion of metal atoms are typically of the order of 10^{12} - 10^{13} Hz [88]. The effect of the external field is taken into account as a direction-dependent bias in the hopping barriers $E_{\boldsymbol{\phi}}$. This can be visualized as a linear contribution that is added to the diffusion energy potential, which favours atomic movements in the direction of the force, as illustrated schematically in Fig. 2.3b.



(a) (b)

Figure 2.3: Lattice model for biased edge diffusion. (a) Hopping of a particle. In this example, $\boldsymbol{u}_{ss'} = +(1/2)\hat{\boldsymbol{x}}$ and $n_{ss'} = 2$. (b) Schematic representation of the diffusion energy potential.

The total energy barrier from state s to state s' is then written

$$E_{\boldsymbol{\phi}}(s, s') = n_{ss'} J - \boldsymbol{\phi}(s) \cdot \boldsymbol{u}_{ss'}\,, \tag{2.5}$$

where $n_{ss'}$ is the number of in-plane nearest neighbors of the moving atom in state s before hopping and $\boldsymbol{u}_{ss'}$ is half the displacement vector between the initial and the final state [24], as shown in Fig. 2.3a. More generally, $\boldsymbol{u}_{ss'}$ is the displacement vector between the equilibrium position of the particle before the move and the saddle point of the diffusion energy potential in the absence of a field. The first term of Eq. (2.5) is based on an intuitive picture where atoms have to break all the bonds with their neighbours in order to reach an excited state where they are able to move. More realistic and quantitatively accurate models are known for specific systems [89], however, the bond-breaking model of Eq. (2.5) is still the most commonly used in the literature.

Furthermore, more sophisticated microscopic descriptions of electromigration [90, 91] suggest that the migration force could depend on the configuration of the surface around the moving atom. The influence of these variations of the force on the dynamics of step edges were discussed [92]. Here, we use the simplest available description of electromigration with Eq. (2.5).

To gain computation time, we freeze atoms with $n_{ss'} = 4$. In addition, we use normalized units where $k_B = 1$, $J = 1$, and $l = 1$. As shown in Fig. 2.4, Eqs. (2.4) and (2.5) define the rates to go from a given state s to all states s' that can be reached with one atomic move, under the force $\boldsymbol{\phi}(s)$. These rates are uniquely defined, since we consider rotations and reflections of shapes as distinct cluster configurations, and therefore to each atomic move corresponds exactly one transition from s to s'. Also, remark that, for clusters with 3 or more atoms, there is a unique move that allows for the transition between two states s and s'.

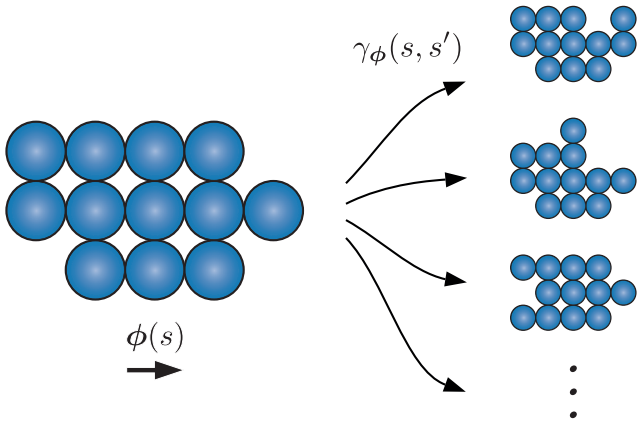


Figure 2.4: For a given configuration s and force $\boldsymbol{\phi}(s)$, the hopping rates $\gamma_{\boldsymbol{\phi}}(s, s')$ to reachable states s' are given by Eq. (2.4).

Our goal is to minimize the time to reach a given target cluster configuration $\bar{s}$. Thus, the reward in our MDP will be related to the physical time that passes in state s before moving to another state s'. The expectation value of this quantity is called the expected residence time $t_{\boldsymbol{\phi}}(s)$ and is given by the reciprocal of the sum of all possible hopping rates in state s (see Section 3.1 for derivation):

$$t_{\boldsymbol{\phi}}(s) = \frac{1}{\sum_{s' \in \mathcal{B}_s} \gamma_{\boldsymbol{\phi}}(s, s')}\,, \tag{2.6}$$

where the set $\mathcal{B}_s$ of neighboring states of s is composed of the states that can be reached from s via a single atomic move.

We are now able to formulate the dynamics of the MDP with the help of transition probabilities. For simplicity, we introduce the notation $p_{\boldsymbol{\phi}}(s, s')$ to indicate the state-transition probability to state s' starting from s when the force $\boldsymbol{\phi}(s)$ is set according to action $\boldsymbol{a} = \boldsymbol{\phi}(s)$. Hence, we have

$$p(s' \mid s, \boldsymbol{a} = \boldsymbol{\phi}(s)) = p_{\boldsymbol{\phi}}(s, s')\,. \tag{2.7}$$

This probability is simply given by the hopping rate $\gamma_{\boldsymbol{\phi}}(s, s')$ divided by the sum of all hopping rates in s:

$$p_{\boldsymbol{\phi}}(s, s') = \frac{\gamma_{\boldsymbol{\phi}}(s, s')}{\sum_{s' \in \mathcal{B}_s} \gamma_{\boldsymbol{\phi}}(s, s')} = \gamma_{\boldsymbol{\phi}}(s, s')\, t_{\boldsymbol{\phi}}(s)\,. \tag{2.8}$$

We define the reward associated to the state-action pair s, $\boldsymbol{a}$ to be a deterministic quantity. In the usual formalism of MDPs, the goal is to maximize the expected rewards (see Section 1.4.1).

However, our objective here is to minimize the time to reach the target shape. For this reason, the expected reward in this case is simply the opposite of the expected residence time:

$$\varrho(s, \boldsymbol{a} = \boldsymbol{\phi}(s)) = -t_{\boldsymbol{\phi}}(s)\,. \tag{2.9}$$

In summary, the transition probabilities that describe the dynamics of the environment take the form

$$p(s', r \mid s, \boldsymbol{a} = \boldsymbol{\phi}(s)) = p_{\boldsymbol{\phi}}(s, s')\,\delta_{r+t_{\boldsymbol{\phi}}(s)}\,. \tag{2.10}$$

Note that, by plugging this equation in the definition of the expected reward Eq. (1.6), we obtain Eq. (2.9).

2.1.2 Recursion relation

The time to reach a target shape $\bar{s}$ can be seen as a first passage time in a random walk on the space of cluster configurations. A similar concept has been adopted previously in the literature to describe the shape evolution of 2d clusters by edge diffusion. Specifically, it has been used to study the irreversible relaxation of clusters towards an equilibrium shape at low temperature [93], and the dominant kinetic pathways that allow surface diffusion of the cluster onto the substrate [94]. However, these works focus on special paths in the space of configurations of the cluster, while we are interested in describing all possible trajectories to be able to reach arbitrary target shapes. The space of cluster configurations can be represented by a graph, as shown in Fig. 2.5 for a tetramer (4-particle) cluster.

Under a given policy $\boldsymbol{\phi}$, the expected first passage time $\tau_{\boldsymbol{\phi}}(s, \bar{s})$ from state $s \neq \bar{s}$ to a given target $\bar{s}$ obeys a recursion relation, which can be directly obtained by identifying the value of a state as the opposite[2] of the expected first passage time from s to the target $\bar{s}$:

$$v_{\boldsymbol{\phi}}(s) = -\tau_{\boldsymbol{\phi}}(s, \bar{s})\,, \tag{2.11}$$

and by plugging the deterministic policy and the dynamical model given by Eqs. (2.2) and (2.8) into the Bellman equation (1.9). In this way, we obtain

$$\tau_{\boldsymbol{\phi}}(s, \bar{s}) = t_{\boldsymbol{\phi}}(s) + \sum_{s' \in \mathcal{B}_s} p_{\boldsymbol{\phi}}(s, s')\tau_{\boldsymbol{\phi}}(s', \bar{s})\,. \tag{2.12}$$

Equation (2.12) expresses a relationship between the expected first passage time to reach the target from a state and the expected first passage times of its successor states. Indeed, since the dynamics is Markovian, the first passage time is equal to the expected residence time $t_{\boldsymbol{\phi}}(s)$ in state s plus the first passage time from the neighboring state s' after the move [86]. This is expressed in Eq. (2.12) as a weighted sum of $\tau_{\boldsymbol{\phi}}(s', \bar{s})$, where the weights are the state-transition probabilities $p_{\boldsymbol{\phi}}(s, s')$. In addition, the system is already on the target when $s = \bar{s}$, and we therefore have

$$\tau_{\boldsymbol{\phi}}(\bar{s}, \bar{s}) = 0\,. \tag{2.13}$$

This relation acts as a boundary condition to solve Eq. (2.12).

We also define the expected return time to target, i.e. the expected time spent outside the target before returning to it when the system starts in the target itself

$$\tau^{\mathrm{r}}_{\boldsymbol{\phi}}(\bar{s}) = \sum_{s \in \mathcal{B}_{\bar{s}}} p_{\boldsymbol{\phi}}(\bar{s}, s)\tau_{\boldsymbol{\phi}}(s, \bar{s})\,, \tag{2.14}$$

[2] For the same reason that the expected reward of Eq. (2.9) is defined with a minus sign.

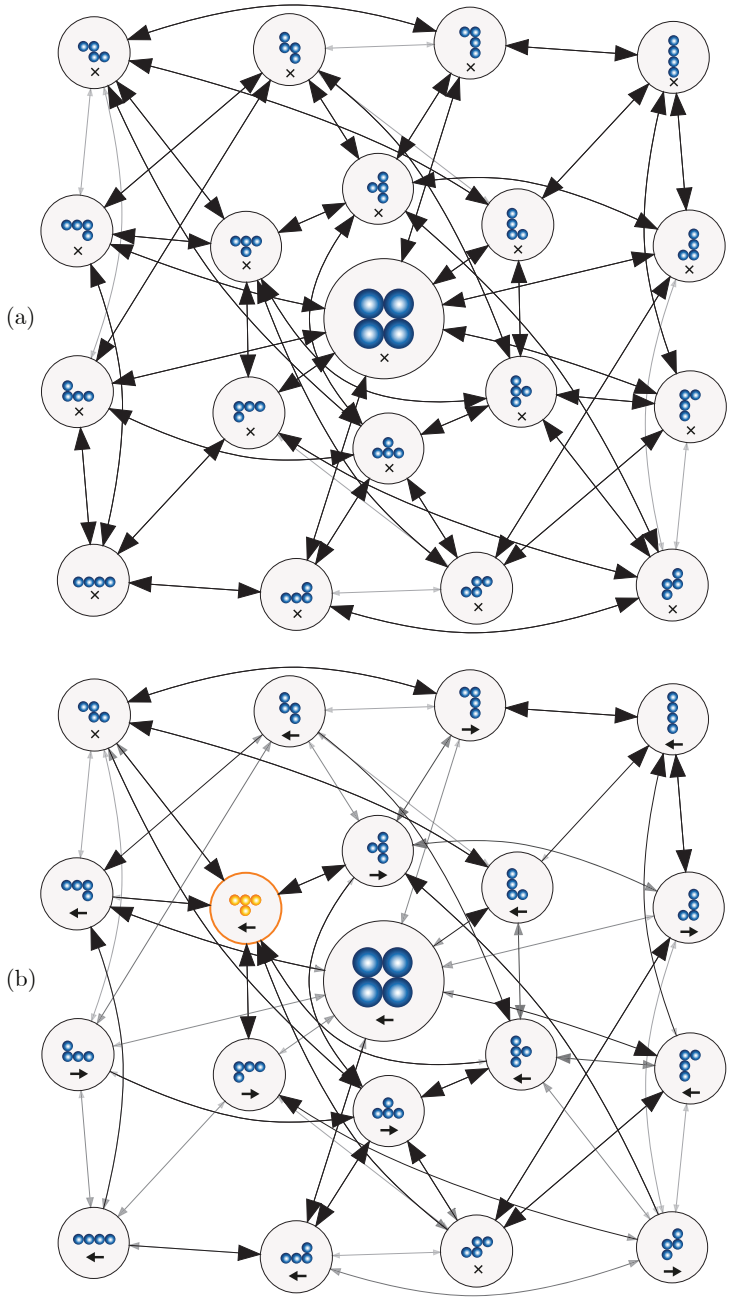


Figure 2.5: Graph of configurations of a tetramer (4-particle) cluster at $T = 0.24$ for (a) unbiased edge diffusion ($\boldsymbol{\phi}(s) = \mathbf{0}$ in all states) and (b) under an optimal policy $\boldsymbol{\phi}_*(s)$ to reach the target shape shown in orange, with $F_0 = 0.4$. The size of the nodes is proportional to the expected residence time on the state $t_{\boldsymbol{\phi}}(s)$, while the size and shade of the edges are proportional to the probability of transition $p_{\boldsymbol{\phi}}(s, s')$. The arrows in the nodes represent the actions, crosses correspond to $\boldsymbol{\phi}(s) = \mathbf{0}$.

This definition requires to extend the policy and to define an additional force on the target state itself. Due to the Markovian character of the dynamics, this does not affect the mean first passage time to target and the optimal policy in the other states outside the target. We choose to set the force on the target so as to minimize $\tau^{\mathrm{r}}_{\boldsymbol{\phi}}(\bar{s})$. This is done by adding to $\mathcal{S}$ an additional state $\hat{s}$ which is an artificial copy of the target state $\bar{s}$, with the same shape than the target but a residence time equal to zero $t_{\boldsymbol{\phi}}(\hat{s}) = 0$, as shown schematically in Fig. 2.6 for a trimer. The introduction of $\hat{s}$ in the set of states is useful also on a computational level, because it allows to obtain the expected return time to target simply as the opposite of the value of this state, since $v_{\boldsymbol{\phi}}(\hat{s}) = -\tau^{\mathrm{r}}_{\boldsymbol{\phi}}(\bar{s})$.

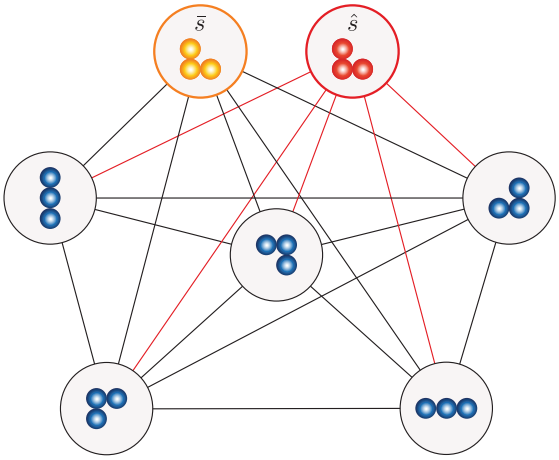


Figure 2.6: Schematic representation of the method to extend the policy to the target. An artificial copy $\hat{s}$ (in red) of the target state $\bar{s}$ (in orange) is added to the state space, with residence time $t_{\boldsymbol{\phi}}(\hat{s}) = 0$.

For the sake of concision, we will mainly focus on the analysis of $\tau^{\mathrm{r}}_{\boldsymbol{\phi}}(\bar{s})$ instead of $\tau_{\boldsymbol{\phi}}(s, \bar{s})$ which is different for each s. Since $\tau^{\mathrm{r}}_{\boldsymbol{\phi}}(\bar{s})$ appears as a linear combination of $\tau_{\boldsymbol{\phi}}(s, \bar{s})$ in Eq. (2.14), we expect that it exhibits similar properties as $\tau_{\boldsymbol{\phi}}(s, \bar{s})$. In addition, we will see below that $\tau^{\mathrm{r}}_{\boldsymbol{\phi}}(\bar{s})$ is a convenient quantity for the analytical investigation of the high-temperature regime.

2.2 High-temperature regime of the expected return time to target

A first glimpse of the behaviour of $\tau^{\mathrm{r}}_{\boldsymbol{\phi}}(\bar{s})$ can be gained in the high-temperature regime. Indeed, we obtain a simple result for the infinite-temperature limit $T \to \infty$. In addition, we perform a high-temperature expansion to get a correction to linear order in $1/T$ to this simple result.

Before moving to the high-temperature analysis, remark that transition state theory, which is the theoretical basis of Eq. (2.4) is derived in the limit $J \gg k_B T$. Although these rates are often used for $J \sim k_B T$, they are certainly not valid in the very high temperature limit $J \ll k_B T$. However, this limit provides useful insight into the physical behaviour of $\tau^{\mathrm{r}}_{\boldsymbol{\phi}}(\bar{s})$.

As a preamble, we start by rewriting Eq. (2.14) in a way that is convenient for the study of the high-temperature regime. Dividing the recursion relation Eq. (2.12) by $t_{\boldsymbol{\phi}}(s)$ and summing

over all states but the target state, we obtain

$$\bar{\sum_s} \frac{\tau_{\boldsymbol{\phi}}(s,\bar{s})}{t_{\boldsymbol{\phi}}(s)} = S_N - 1 + \bar{\sum_s} \sum_{s' \in \mathcal{B}_s} \frac{p_{\boldsymbol{\phi}}(s,s')}{t_{\boldsymbol{\phi}}(s)} \tau_{\boldsymbol{\phi}}(s',\bar{s})\,, \tag{2.15}$$

where the symbol $\bar{\sum}_s = \sum_{s \in \mathcal{S}\setminus\{\bar{s}\}}$ indicates the sum over all states in the system except for the target state $\bar{s}$, and S_N is the total number of states for a cluster with N particles.

All physical moves except for those coming from the target are summed over in the last term of the right hand side of Eq. (2.15). Hence, we can invert the indices s and s' in the sum if we subtract the contribution of the moves starting from the target, leading to

$$\sum_s \frac{\tau_{\boldsymbol{\phi}}(s,\bar{s})}{t_{\boldsymbol{\phi}}(s)} = S_N - 1 + \sum_s \sum_{s' \in \mathcal{B}_s} \frac{p_{\boldsymbol{\phi}}(s',s)}{t_{\boldsymbol{\phi}}(s')} \tau_{\boldsymbol{\phi}}(s,\bar{s}) - \sum_{s \in \mathcal{B}_{\bar{s}}} \frac{p_{\boldsymbol{\phi}}(\bar{s},s)}{t_{\boldsymbol{\phi}}(\bar{s})} \tau_{\boldsymbol{\phi}}(s,\bar{s})\,. \tag{2.16}$$

where we have dropped the exclusion of the target state (indicated by the bar) in the sum since $\tau_{\boldsymbol{\phi}}(\bar{s},\bar{s}) = 0$. The last term is proportional to the expected return time to target Eq. (2.14). We can therefore rewrite Eq. (2.16) as

$$\tau^{\mathrm{r}}_{\boldsymbol{\phi}}(\bar{s}) = t_{\boldsymbol{\phi}}(\bar{s})\,(S_N - 1) + t_{\boldsymbol{\phi}}(\bar{s}) \sum_s \tau_{\boldsymbol{\phi}}(s,\bar{s}) \left[\left(\sum_{s' \in \mathcal{B}_s} \frac{p_{\boldsymbol{\phi}}(s',s)}{t_{\boldsymbol{\phi}}(s')} \right) - \frac{1}{t_{\boldsymbol{\phi}}(s)} \right]. \tag{2.17}$$

Then, using Eq. (2.6) to write $t_{\boldsymbol{\phi}}$ and $p_{\boldsymbol{\phi}}$ as a function of $\gamma_{\boldsymbol{\phi}}$, we obtain

$$\tau^{\mathrm{r}}_{\boldsymbol{\phi}}(\bar{s}) = t_{\boldsymbol{\phi}}(\bar{s})\,(S_N - 1) + t_{\boldsymbol{\phi}}(\bar{s}) \sum_s \tau_{\boldsymbol{\phi}}(s,\bar{s}) \sum_{s' \in \mathcal{B}_s} \left(\gamma_{\boldsymbol{\phi}}(s',s) - \gamma_{\boldsymbol{\phi}}(s,s')\right). \tag{2.18}$$

This latter equation will be the starting point for the analysis of the high-temperature regime.

2.2.1 Infinite-temperature limit

In the limit of infinitely high temperatures $T \to \infty$, the rates (2.4) are independent of the initial and final state and on the force: $\gamma_{\boldsymbol{\phi}}(s,s') \to 1$. As a consequence, $\tau^{\mathrm{r}}_{\boldsymbol{\phi}}(\bar{s})$ is independent of the policy $\boldsymbol{\phi}$ at infinite temperature $\tau^{\mathrm{r}}_{\boldsymbol{\phi}}(\bar{s}) \to \tau^{\mathrm{r}}_{\infty}(\bar{s})$. From Eq. (2.6), we obtain

$$t_{\boldsymbol{\phi}}(s) \to \frac{1}{d_s}\,, \tag{2.19}$$

where d_s, called the degree of state s, is the number of possible atomic moves from s. Note that $d_s = |\mathcal{B}_s|$, where the notation $|\mathcal{Z}|$ denotes the cardinal of the set $\mathcal{Z}$. As a consequence, Eq. (2.18) implies

$$\tau^{\mathrm{r}}_{\boldsymbol{\phi}}(\bar{s}) \to \tau^{\mathrm{r}}_{\infty}(\bar{s}) = \frac{S_N - 1}{d_{\bar{s}}} \tag{2.20}$$

in the infinite-temperature limit. This remarkably simple result is known in the literature [95].

In addition, the general recursion relation Eq. (2.12) can be re-written as

$$1 = \sum_{s' \in \mathcal{B}_s} \gamma_{\boldsymbol{\phi}}(s,s')(\tau_{\boldsymbol{\phi}}(s,\bar{s}) - \tau_{\boldsymbol{\phi}}(s',\bar{s}))\,. \tag{2.21}$$

In the limit of infinite temperatures, this expression takes the form of the usual discretized Laplacian for the expected first passage times to the target $\tau_{\infty}(s,\bar{s})$

$$1 = \sum_{s' \in \mathcal{B}_s} (\tau_{\infty}(s,\bar{s}) - \tau_{\infty}(s',\bar{s}))\,. \tag{2.22}$$

This relation is valid everywhere but at the target state, where $\tau_\infty(\bar{s},\bar{s})=0$. The value of $\tau_\infty(s,\bar{s})$ depends in general on the precise structure of the dynamical graph.

A vast literature is dedicated to the study of the first passage times on random graphs [96, 97]. However, most studies are devoted to the case where $t_{\boldsymbol{\phi}}(s)=1$, which is different from our infinite-temperature limit, which leads to Eq. (2.19). The difference lies in the factor $1/d_s$ in $t_{\boldsymbol{\phi}}(s)$. A trace of this simple difference is that the mean first passage time (MFPT), which is the average of the first passage time over all initial states for a fixed target state, discussed in the literature with $t_{\boldsymbol{\phi}}(s)=1$, exhibits a lower bound for large S_N [98, 99]

$$\text{MFPT} \geq S_N \langle d_s \rangle / d_{\bar{s}}\,, \tag{2.23}$$

where $\langle d_s \rangle$ is the average of d_s over all states. Up to the factor $\langle d_s \rangle$, this lower bound is identical to the expression of the expected return time to target at infinite temperature $\tau^{\mathrm{r}}_\infty(\bar{s})$ in Eq. (2.20).

2.2.2 High-temperature expansion

We now perform an expansion to linear order in $1/T$ to obtain the first correction to Eq. (2.20). We start with the expansion of the rates, defined in Eq. (2.4)

$$\gamma_{\boldsymbol{\phi}}(s,s') \approx 1 + \frac{1}{T}(-n_{ss'} + \varphi(s)\, u_{ss'})\,, \tag{2.24}$$

where we recall that $n_{ss'}$ is the initial number of bonds of the atom which moves in the transition from state s to state s'. In addition, we have $u_{ss'}=1/2$, 0, 0 or $-1/2$ when the move is in the direction $+\hat{\boldsymbol{x}}$, $-\hat{\boldsymbol{y}}$, $+\hat{\boldsymbol{y}}$ or $-\hat{\boldsymbol{x}}$ respectively. Remark that, for the displacement vector, the following symmetry relation holds

$$u_{ss'} = -u_{s's}\,, \tag{2.25}$$

while there is no similar trivial symmetry for $n_{ss'}$. Moreover, $\varphi(s)=-F_0$, 0, or $+F_0$ is the scalar force in the state s given by the policy $\boldsymbol{\phi}$, so that

$$\boldsymbol{\phi}(s) = \varphi(s)\,\hat{\boldsymbol{x}}\,. \tag{2.26}$$

Inserting this expression in the expected return time Eq. (2.18), we obtain to linear order in $1/T$

$$\tau^{\mathrm{r}}_{\boldsymbol{\phi}}(\bar{s}) = \tau^{\mathrm{r}}_\infty(\bar{s})\left(1+\frac{M_{\boldsymbol{\phi}}(\bar{s})}{T}\right)\,, \tag{2.27}$$

where the normalized high-temperature slope reads

$$\begin{aligned} M_{\boldsymbol{\phi}}(\bar{s}) =& \frac{1}{d_{\bar{s}}}\sum_{s\in\mathcal{B}_{\bar{s}}}(n_{\bar{s}s}-\varphi(\bar{s})\,u_{\bar{s}s}) + \frac{1}{S_N-1}\sum_s \tau_\infty(s,\bar{s})\sum_{s'\in\mathcal{B}_s}(n_{ss'}-n_{s's}) \\ &+ \frac{1}{S_N-1}\sum_s \tau_\infty(s,\bar{s})\sum_{s'\in\mathcal{B}_s}(\varphi(s')\,u_{s's}-\varphi(s)\,u_{ss'})\,. \end{aligned} \tag{2.28}$$

We see the main advantage of an expansion based on Eq. (2.18): we do not need to expand $\tau_{\boldsymbol{\phi}}(s,\bar{s})$, which depends on T, and we only refer to $\tau_\infty(s,\bar{s})$, which is independent of T and of $\boldsymbol{\phi}$. However, we do not have an analytical expression for $\tau_\infty(s,\bar{s})$, and we therefore need to compute this quantity numerically. The normalized slope $M_{\boldsymbol{\phi}}(\bar{s})$ can be rewritten as (see Appendix A.4 for details)

$$\begin{aligned} \left(1-\frac{1}{S_N}\right)M_{\boldsymbol{\phi}}(\bar{s}) =& \Big\langle (1-\delta_{s\bar{s}})\langle n_{ss'}\rangle_{s'\in\mathcal{B}_s} - d_s g_n(s,\bar{s})\Big\rangle_{s\in\mathcal{S}} \\ &- \Big\langle \varphi(s)\left((1-\delta_{s\bar{s}})\langle u_{ss'}\rangle_{s'\in\mathcal{B}_s} - d_s g_u(s,\bar{s})\right)\Big\rangle_{s\in\mathcal{S}}\,, \end{aligned} \tag{2.29}$$

where $\delta_{ss'}$ is the two-arguments Kronecker delta and the notation

$$\langle q_s \rangle_{s \in \mathcal{Z}} = \frac{1}{|\mathcal{Z}|} \sum_{s \in \mathcal{Z}} q_s \tag{2.30}$$

indicates the average of q_s taken over the states s belonging to the set of states $\mathcal{Z}$ with cardinal $|\mathcal{Z}|$. In addition, we have defined the local covariances

$$g_n(s, \bar{s}) = \left\langle \left(n_{ss'} - \langle n_{ss''} \rangle_{s'' \in \mathcal{B}_s} \right) \left(\tau_\infty(s', \bar{s}) - \langle \tau_\infty(s'', \bar{s}) \rangle_{s'' \in \mathcal{B}_s} \right) \right\rangle_{s' \in \mathcal{B}_s} \tag{2.31}$$

$$g_u(s, \bar{s}) = \left\langle \left(u_{ss'} - \langle u_{ss''} \rangle_{s'' \in \mathcal{B}_s} \right) \left(\tau_\infty(s', \bar{s}) - \langle \tau_\infty(s'', \bar{s}) \rangle_{s'' \in \mathcal{B}_s} \right) \right\rangle_{s' \in \mathcal{B}_s} \tag{2.32}$$

that are averaged over the neighboring states of state s.

2.3 Equilibrium dynamics in the absence of forces

Before looking at the control problem with the optimization of the forces, let us study the expected return time to a target configuration without external bias, i.e. with the force set to zero $\boldsymbol{\phi}(s) = \mathbf{0}$ in all states, as illustrated in the dynamical graph of Fig. 2.5a. This case leads to standard equilibrium fluctuation dynamics that have been investigated thoroughly in the case of edge diffusion, in particular to determine diffusion coefficients for clusters of various sizes [25, 75, 26], and the relaxation of their shape towards equilibrium [100, 101].

Equilibrium dynamics of atomic steps and clusters has also been studied to derive time-correlation functions of fluctuating steps [75, 102] and to characterize the stability of fluctuating nanostructures over finite times and finite deviations through persistence exponents [103, 104]. Although persistence properties involve quantities that are related to first passage processes, there is to our knowledge no study of the first passage time to a fixed cluster configuration.

2.3.1 Numerical results from value iteration

We have evaluated the first passage time numerically: for a fixed $\bar{s}$, we iterate the evaluation of $\tau_{\boldsymbol{\phi}}(s, \bar{s})$ by substitution of its value in the right hand side of Eq. (2.12). This procedure corresponds to the method of iterative policy evaluation explained in Section 1.4.2, with the special policy $\boldsymbol{\phi}(s) = \mathbf{0}$ for all s in $\mathcal{S}$. We have implemented the iterative policy evaluation algorithm reported in Appendix A.2 in Python, with a convergence condition that depends on the size of the cluster (see Appendix A.5 for details).

Since we are using a tabular method, it is necessary that all states in the system are listed, and this can be done only for small clusters, which correspond to our focus in this work. Because we are forbidding the breaking of the cluster, once the number of particles N is fixed, the total number of cluster configurations S_N—or the cardinality of the set of states $\mathcal{S}$— correspond to the number of fixed polyominoes [105, 106] with N cells[3]. This number grows exponentially with N:

$$S_N \sim \frac{c\lambda^N}{N}\,, \tag{2.33}$$

with $\lambda \approx 4.0626$ and $c \approx 0.3169$ [107]. We have performed simulations with $N \leq 12$, where $S_{12} \approx 5 \times 10^5$ states.

[3]In Appendix A.6 we explain the method we used to uniquely encode each polyomino into a numerical identifier, so that we have a list of states that can be easily handled by the computer.

The resulting expected return time to target with zero force $\tau_0^{\mathrm{r}}(\bar{s})$ is shown in Fig. 2.7 as a function of $1/T$, for three targets of different size. For small clusters, $\tau_0^{\mathrm{r}}(\bar{s})$ increases monotonously as the temperature is decreased. This is expected as thermally activated hopping diffusion events become slower at low temperatures. However, the expected return time to target exhibits a minimum as a function of temperature for clusters that are larger and more compact.

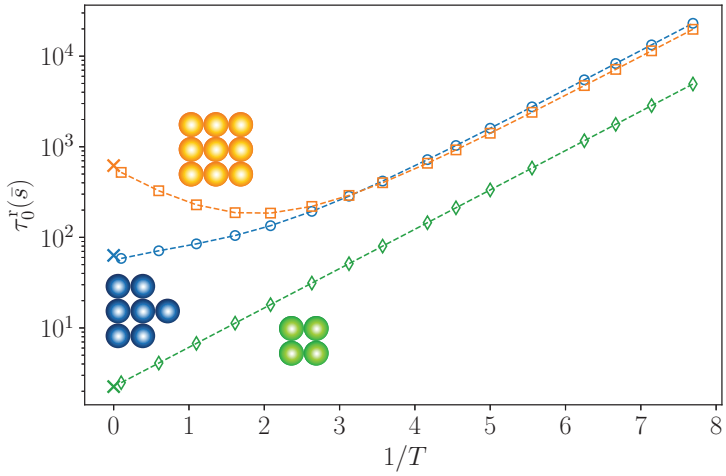


Figure 2.7: Expected return time to target in the absence of forces $\tau_0^{\mathrm{r}}(\bar{s})$ as a function of $1/T$. The $\times$ symbols correspond to the analytical expression Eq. (2.20) of $\tau_\infty^{\mathrm{r}}(\bar{s})$ for $T \to \infty$.

As shown in Fig. 2.8a, a similar minimum is found in the time $\tau_0(s, \bar{s})$ to reach the target starting from any state s. This striking result implies that some targets exhibit an optimal temperature at which the target can be reached in minimum time.

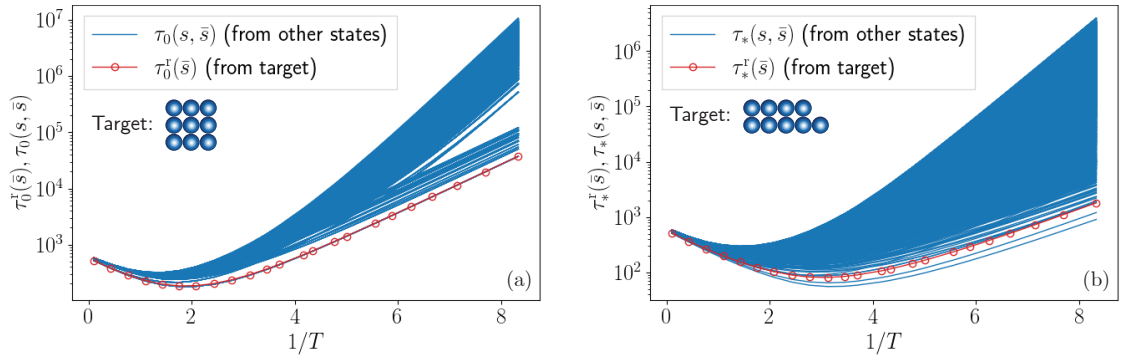


Figure 2.8: Expected time to reach the target as a function of $1/T$, starting from the target state itself (return time), and from all the other states in the system. (a) Zero-force case and (b) under an optimal policy $\boldsymbol{\phi}_*$, with $F_0 = 0.4$. The number of blue curves in these graphs is equal to the number of states minus one (the target), i.e. $S_9 - 1 = 9909$.

2.3.2 High-temperature behaviour

The presence of a minimum is associated to a change of slope of $\tau_0^{\mathrm{r}}(\bar{s})$ as a function of $1/T$ at high temperatures. We therefore study the high-temperature behaviour in more details. We have seen in the previous section that, in the infinite-temperature limit $T \to \infty$, all moves have the same rate, therefore, the expected return time to target is independent of the policy, and we find Eq. (2.20). When the temperature is decreased, this is no longer true, and the moves become sensitive to the energy.

In the absence of an external force $\boldsymbol{\phi}(s) = \mathbf{0}$, the system will eventually reach equilibrium, characterized by a probability $P_0(s)$ of being in any state s that does not change over time $\partial P_0(s)/\partial t = 0$. From the master equation (2.3), this condition leads to

$$\sum_{s' \in \mathbb{S}} P_0(s', t)\gamma_0(s', s) = \sum_{s' \in \mathbb{S}} P_0(s, t)\gamma_0(s, s') \,. \tag{2.34}$$

The stationary distribution $P_0(s)$ of a system at equilibrium is given by the Boltzmann distribution [86]

$$P_0(s) = \frac{1}{Z} \exp\left(-\frac{H(s)}{k_B T}\right) , \tag{2.35}$$

where Z is the partition function

$$Z = \sum_{s \in \mathbb{S}} \exp\left(-\frac{H(s)}{k_B T}\right) , \tag{2.36}$$

and $H(s)$ is the energy of state s, which can be expressed by the formula[4]

$$H(s) = L_s \frac{J}{2} \,, \tag{2.39}$$

where L_s is the perimeter of the cluster. This equation has a simple intuitive interpretation. Dividing a cluster into two by cutting it along a straight line of length L requires a total energy of JL. The cut produces $2L$ new unit edges, each contributing to the system with a certain energy E_{edge}, for a total energy increase of $2LE_{\mathrm{edge}}$. Imposing equality between the energy introduced into the system and the energy increase, we obtain $E_{\mathrm{edge}} = J/2$, which expresses the fact that every edge of the cluster has an energy of $J/2$.

[4]The energy of an island configuration s can be linked to the Hamiltonian of an equivalent Ising model on a 2d square lattice, without external field [108]

$$H_{\mathrm{Ising}} = -\frac{J}{4} \sum_{i,j} \varsigma_i \varsigma_j \,, \tag{2.37}$$

where the variable $\varsigma_i \in \{1, -1\}$ represents the spin of lattice site i, and the sum runs over pairs of adjacent spins, i.e. nearest neighbours. Then, by defining the variable $n_i = (\varsigma_i + 1)/2$ which is either 1 or 0, depending on whether the site i is occupied by an atom, we obtain

$$\begin{aligned} H &= -\frac{J}{4} \sum_{i,j} \left[4 n_i n_j - 2(n_i + n_j) + 1\right] \\ &= \frac{J}{2} \sum_{i,j} \left[n_i(1 - n_j) + n_j(1 - n_i)\right] - N_{\mathrm{sites}} \frac{J}{2} \,, \end{aligned} \tag{2.38}$$

where N_{sites} is the total number of sites on the lattice. The first term on the right hand side is equal to 1 if and only if one of the two neighbouring sites i, j is occupied and the other is empty, while in the other cases is 0. Hence, this term is equal to the perimeter L_s of the atomic island in the configuration s of which we are calculating the energy. The second term is a constant.

For our system defined by the rates in Eq. (2.4), there is actually a stronger equilibrium condition than Eq. (2.34) that can be applied [109], and it is that of *detailed balance*:

$$P_0(s',t)\gamma_0(s',s) = P_0(s,t)\gamma_0(s,s')\,. \tag{2.40}$$

This condition expresses the fact that, at equilibrium, the average frequency of occurrence of an elementary process (i.e. an atomic movement) is equal to the average frequency of occurrence of its reverse process. Plugging the steady state distribution Eq. (2.35) in Eq. (2.40), we obtain the following relation for the rate of a process and the rate of its reverse process

$$\gamma_0(s,s') = \gamma_0(s',s)\exp\left(\frac{H(s)-H(s')}{k_BT}\right)\,. \tag{2.41}$$

Hence, a move from s to s' that leads to a decrease of energy $H(s') < H(s)$ has a larger rate than the reverse process $\gamma_0(s,s') > \gamma_0(s',s)$. As a consequence, the cluster goes faster towards states with lower energy.

As the temperature is decreased, the difference between the rate of a move and its reverse becomes larger, and the time to reach a target decreases if the target has a lower energy. However, this trend is only describing relative variations of the time to reach different targets. When decreasing the temperature, there is also a global slowing-down of the dynamics because of the Arrhenius dependence of the rates on temperature in Eq. (2.4). The decrease or increase of first passage times to the target—or equivalently of $\tau^{\mathrm{r}}_{\boldsymbol{\phi}}(\bar{s})$—depends on the competition between these two effects: relative energy effect vs global slowing down.

The global slowing down is always dominant at low temperatures, and $\tau^{\mathrm{r}}_{\boldsymbol{\phi}}$ therefore always increases when the temperature is decreased at low temperatures. However, the global slowing down can be dominated by the relative energy effect at high temperature. In this case, $\tau^{\mathrm{r}}_{\boldsymbol{\phi}}$ decreases with decreasing temperature at high temperatures. We then expect a minimum of $\tau^{\mathrm{r}}_{\boldsymbol{\phi}}$ at some finite (non-zero) temperature.

This competition can be analysed quantitatively with the help of the high-temperature expansion that we carried out in the previous section. In the case of zero force, Eq. (2.27) reads

$$\tau^{\mathrm{r}}_0(\bar{s}) = \left(1+\frac{M_0(\bar{s})}{T}\right)\tau^{\mathrm{r}}_\infty(\bar{s})\,, \tag{2.42}$$

where the high-temperature slope is readily obtained by setting $\varphi(s)=0$ in Eq. (2.29):

$$M_0(\bar{s}) = \frac{1}{1-S_N^{-1}}\left\langle (1-\delta_{s\bar{s}})\langle n_{ss'}\rangle_{s'\in\mathcal{B}_s} - d_s g_n(s,\bar{s})\right\rangle_{s\in\mathcal{S}}\,. \tag{2.43}$$

The existence of a minimum is then associated to a negative value of the high-temperature slope M_0. In Fig. 2.9a, we see that the expression Eq. (2.43) is in good agreement with the value $M_0^{\mathrm{sim}}(\bar{s})$ found with a fit of the numerical solution obtained from iterative evaluation (see Appendix A.7 for details). Small deviations are caused by the freezing of 4-neighbors atoms in simulations.

2.3.3 Approximate expression of M_0

In Eq. (2.43), the first term $\langle (1-\delta_{s\bar{s}})\langle n_{ss'}\rangle_{s'\in\mathcal{B}_s}\rangle_{s\in\mathcal{S}}$ is the average over all possible states s except the target $\bar{s}$ of the average number of bonds $n_{ss'}$ that are broken to perform a move from s. This term essentially accounts for the global slowing down of the dynamics when the temperature is decreased. Indeed, the higher this average number of bonds to be broken for atom motion, the

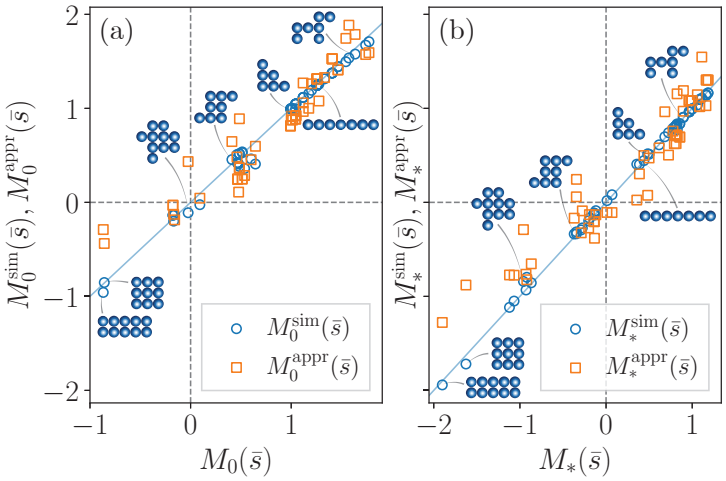


Figure 2.9: Estimates of the high-temperature slope for (a) zero force and (b) optimal force. $M_0^{\rm sim}(\bar{s})$ and $M_*^{\rm sim}(\bar{s})$ refer to the slope extracted from a parabolic fit at high temperatures ($T = 5, 10, 20$) of the expected return time to target $\tau_{\boldsymbol{\phi}}^{\rm r}(\bar{s})$ obtained from iterative numerical methods. These quantities and the approximations $M_0^{\rm appr}(\bar{s})$ and $M_*^{\rm appr}(\bar{s})$ defined in Eqs. (2.51) and (2.58) are plotted against the analytical expressions of Eqs. (2.43) and (2.55).

slower the dynamics. The number d_s of possible moves from a given state s grows approximately linearly with N (since the cluster has N atoms, and each atom has at maximum 7 allowed moves, the average number d_s of possible moves is smaller than $7N$), as shown in Fig. 2.10. In contrast, the total number of states S_N grows exponentially with N. As a consequence, the contribution of the moves that start from the target is negligible when N is large enough. Hence, the first term in Eq. (2.43) can be approximated with the average $\langle\langle n_{ss'}\rangle_{s'\in\mathcal{B}_s}\rangle_{s\in\mathcal{S}}$ of $n_{ss'}$ over all states. Since $n_{ss'}$ is bounded ($1 \leq n_{ss'} \leq 4$), this average converges quickly to a value ρ_0 around 1.64 as N is increased. A fit of

$$\rho_0(N) = \langle\langle n_{ss'}\rangle_{s'\in\mathcal{B}_s}\rangle_{s\in\mathcal{S}} \tag{2.44}$$

for $N > 2$, as provided in Fig. 2.11, shows that this convergence is approximately exponential

$$\rho_0(N) = \rho_0 - \tilde{\rho}_0 \exp(-\frac{N}{N_0}), \tag{2.45}$$

with $\tilde{\rho}_0 \approx 0.89$ and $N_0 \approx 5.4$.

The second term $\langle d_s g_n(s,\bar{s})\rangle_{s\in\mathcal{S}}$ in Eq. (2.43) accounts for the correlations between the number of bonds that are broken and the first passage time to target at infinite temperature when starting from a given state s. We therefore interpret this second term as an indication for the relative trend of the system to go faster towards low-energy states. Indeed, since $\tau_\infty(s,\bar{s})$ decreases when s approaches the target $\bar{s}$ where $\tau_\infty(s,\bar{s}) = 0$, this second term indicates if moves from states that are closer to the target require more or less bonds to break on average.

We introduce the concept of rings, following the study of first-passage times in random graphs in Ref. [97]. The index of the rings m is defined as the minimum number of moves to reach the target. A ring $\mathcal{R}_m$ is the ensemble of states with the same ring index m. Note that our definitions are such that $\mathcal{R}_1 = \mathcal{B}_{\bar{s}}$, the neighboring states of the target $\bar{s}$. The first passage time to target at infinite temperature $\tau_\infty(s,\bar{s})$ is known to increase quickly up to an asymptotic value as m increases [97]. The asymptotic value is reached after a few rings. A plot of the mean first passage

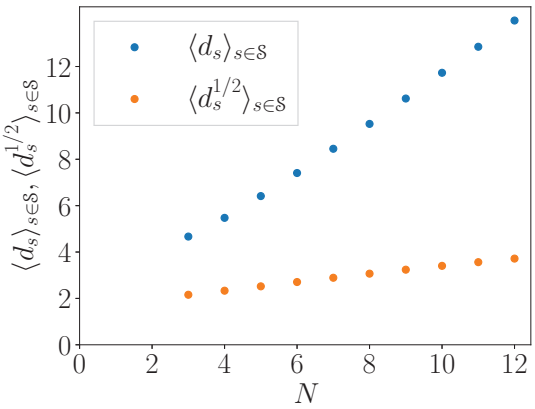


Figure 2.10: Mean of the degrees of the states d_s and their square roots over all states, for clusters of increasing size.

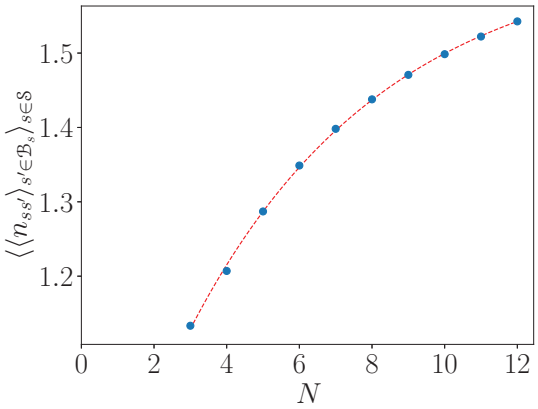


Figure 2.11: Mean of $\langle n_{ss'} \rangle_{s'\in\mathcal{B}_s}$ over all states, for clusters of increasing size. The points are fitted with a decaying exponential, that gives an asymptotic value $\rho_0 \approx 1.64$.

time to target as a function of the ring index is reported in Fig. 2.12. We therefore simply assume that the second term of Eq. (2.43) is dominated by the contributions related the large variation of $\tau_\infty(s, \bar{s})$ between the target state $\bar{s}$ where $\tau_\infty(\bar{s}, \bar{s}) = 0$ and the first ring $\mathcal{B}_{\bar{s}}$.

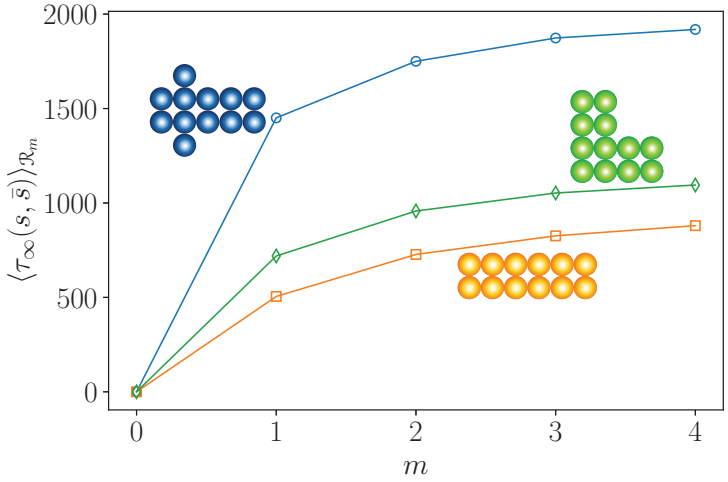


Figure 2.12: Mean on rings of the first passage time to target at infinite temperature as a function of the ring index m for three different targets with $N = 12$.

When evaluating the term $\langle d_s g_n(s, \bar{s}) \rangle_{s\in\mathcal{S}}$ in Eq. (2.43) we therefore need to sum only over

the moves from states in $\mathcal{B}_{\bar{s}}$ to the target $\bar{s}$. Hence, an approximation of this term reads

$$\begin{aligned}\langle d_s g_n(s,\bar{s})\rangle_{s\in\mathcal{S}} &= \frac{1}{S_N}\sum_{s\in\mathcal{S}} d_s g_n(s,\bar{s}) \approx \frac{1}{S_N}\sum_{s\in\mathcal{B}_{\bar{s}}} d_s g_n(s,\bar{s}) = \frac{d_{\bar{s}}}{S_N}\langle d_s g_n(s,\bar{s})\rangle_{s\in\mathcal{B}_{\bar{s}}} \\ &= \frac{d_{\bar{s}}}{S_N}\Big\langle \sum_{s'\in\mathcal{B}_s}\Big(n_{ss'} - \langle n_{ss''}\rangle_{s''\in\mathcal{B}_s}\Big)\Big(\tau_\infty(s',\bar{s}) - \langle\tau_\infty(s'',\bar{s})\rangle_{s''\in\mathcal{B}_s}\Big)\Big\rangle_{s\in\mathcal{B}_{\bar{s}}} \qquad (2.46)\\ &\approx -\frac{d_{\bar{s}}}{S_N}\langle(n_{s\bar{s}} - \langle n_{ss''}\rangle_{s''\in\mathcal{B}_s})\langle\tau_\infty(s'',\bar{s})\rangle_{s''\in\mathcal{B}_s}\rangle_{s\in\mathcal{B}_{\bar{s}}}\,,\end{aligned}$$

where in the last line we have retained in the sum only the contributions of $s'=\bar{s}$, i.e. the moves from $\mathcal{B}_{\bar{s}}$ leading to the target, and we have used the boundary condition $\tau_\infty(\bar{s},\bar{s})=0$. Then, using the high-temperature recursion relation Eq. (2.22) (see Eq. (A. 15) in appendix for details), we obtain an approximation of $\langle\tau_\infty(s'',\bar{s})\rangle_{s''\in\mathcal{B}_s}$ as

$$\langle\tau_\infty(s'')\rangle_{s''\in\mathcal{B}_s} = \tau_\infty(s,\bar{s}) - \frac{1}{d_s} \approx \tau_\infty(s,\bar{s})\,, \qquad (2.47)$$

where we have used $1/d_s \ll \tau_\infty(s,\bar{s})$, which is valid for N large enough. In summary, we obtain

$$\langle d_s g_n(s,\bar{s})\rangle_{s\in\mathcal{S}} \approx -\frac{d_{\bar{s}}}{S_N}\langle(n_{s\bar{s}} - \langle n_{ss''}\rangle_{s''\in\mathcal{B}_s})\,\tau_\infty(s,\bar{s})\rangle_{s\in\mathcal{B}_{\bar{s}}}\,. \qquad (2.48)$$

Finally, we simply assume the statistical independence of the two terms in the product in the brackets of Eq. (2.48), leading to

$$\langle d_s g_n(s,\bar{s})\rangle_{s\in\mathcal{S}} \approx -\frac{d_{\bar{s}}}{S_N}\langle n_{s\bar{s}} - \langle n_{ss''}\rangle_{s''\in\mathcal{B}_s}\rangle_{s\in\mathcal{B}_{\bar{s}}}\langle\tau_\infty(s,\bar{s})\rangle_{s\in\mathcal{B}_{\bar{s}}}\,. \qquad (2.49)$$

From Eq. (2.20), we have $\langle\tau_\infty(s,\bar{s})\rangle_{s\in\mathcal{B}_{\bar{s}}} = \tau_\infty^{\mathrm{r}}(\bar{s}) = (S_N-1)/d_{\bar{s}}$. Then, taking the limit $S_N \gg 1$, we finally obtain

$$\langle d_s g_n(s,\bar{s})\rangle_{s\in\mathcal{S}} \approx -\langle n_{s\bar{s}} - \langle n_{ss''}\rangle_{s''\in\mathcal{B}_s}\rangle_{s\in\mathcal{B}_{\bar{s}}}. \qquad (2.50)$$

This approximate relation is shown to be accurate in Fig. 2.13 up to a multiplicative factor $\rho_1 \approx 1.60$. When the target shape is compact, then $n_{s\bar{s}}$ is smaller than the average of the $n_{ss'}$ on the neighbors s' of the states s of the first ring. Hence, compact shapes lead to a positive contribution to $\langle d_s g_n(s,\bar{s})\rangle_{s\in\mathcal{S}}$, and a negative contribution to M_0. As a consequence, compact shapes are expected to lead to a minimum in the temperature dependence of $\tau_0^{\mathrm{r}}(\bar{s})$. Furthermore, compact shapes are also those which exhibit a lower energy, as we can see from the Hamiltonian of Eq. (2.39). Therefore, a minimum will be obtained for shapes with a low energy.

Combining the approximations of both contributions to Eq. (2.43), we obtain

$$M_0(\bar{s}) \approx M_0^{\mathrm{appr}}(\bar{s}) = \rho_0(N) + \rho_1\langle n_{s\bar{s}} - \langle n_{ss''}\rangle_{s''\in\mathcal{B}_s}\rangle_{s\in\mathcal{B}_{\bar{s}}}\,. \qquad (2.51)$$

As shown in Fig. 2.9a, $M_0^{\mathrm{appr}}(\bar{s})$ provides a fair approximation to $M_0(\bar{s})$. The sign of $M_0^{\mathrm{appr}}(\bar{s})$ can serve as a simple guide to the possible presence of a minimum as a function of T, i.e. an optimal temperature, and also makes explicit the link between the minimum and the compactness of the target. For example, for a linear one-atom-thick target, only the two atoms at the tips can move, so that $\langle n_{s\bar{s}}\rangle_{s\in\mathcal{B}_{\bar{s}}} = 1$ and an inspection of the possible moves shows that $\langle\langle n_{ss''}\rangle_{s''\in\mathcal{B}_s}\rangle_{s\in\mathcal{B}_{\bar{s}}} = 6/5$, as illustrated in Fig. 2.14. This leads to $M_0^{\mathrm{appr}}(\bar{s}) = \rho_0(N) - \rho_1/5 \approx 1.08 > 0$ for $N=7$, in agreement with $M_0^{\mathrm{sim}}(\bar{s}) \approx 1.04$ found by iterative evaluation.

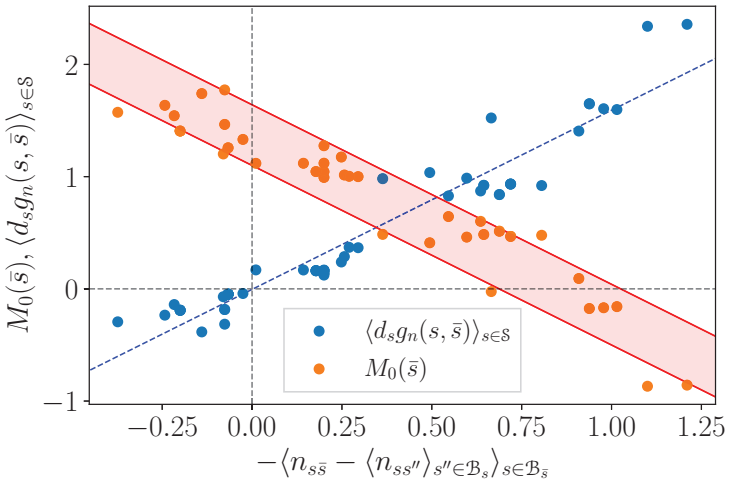


Figure 2.13: $M_0(\bar{s})$ and mean of $d_s g_n(s,\bar{s})$ over all states as a function of the quantity $-\langle n_{s\bar{s}} - \langle n_{ss''}\rangle_{s''\in\mathcal{B}_s}\rangle_{s\in\mathcal{B}_{\bar{s}}}$ (the mean is taken over the neighboring states of the target $\bar{s}$). The blue dashed line is a fit of $\langle d_s g_n(s,\bar{s})\rangle_{s\in\mathcal{S}}$, that gives a slope $\rho_1 \approx 1.60$. This value is used to draw the two red lines, which correspond to the approximate relation $M_0^{\text{appr}}(\bar{s}) = \rho_0(N) - \rho_1\langle n_{s\bar{s}} - \langle n_{ss''}\rangle_{s''\in\mathcal{B}_s}\rangle_{s\in\mathcal{B}_{\bar{s}}}$, with $\rho_0(N=3) = 1.1$ and $\rho_0(N=\infty) = 1.64$ (the minimum and the asymptotic value of $\langle\langle n_{ss'}\rangle_{s'\in\mathcal{B}_s}\rangle_{s\in\mathcal{S}}$, see Fig. 2.11).

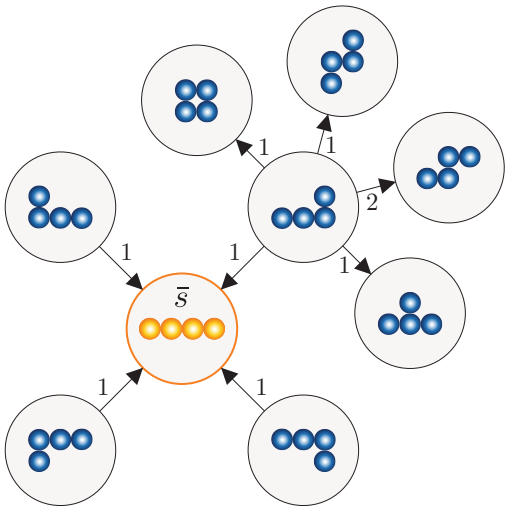


Figure 2.14: The numbers on the edges represent the number of broken bonds for the transition. The four neighboring states of the target all have $n_{s\bar{s}} = 1$, so that $\langle n_{s\bar{s}}\rangle_{s\in\mathcal{B}_{\bar{s}}} = 1$. Then, for any of these four states, we find $\langle n_{ss''}\rangle_{s''\in\mathcal{B}_s} = 6/5$, hence $\langle\langle n_{ss''}\rangle_{s''\in\mathcal{B}_s}\rangle_{s\in\mathcal{B}_{\bar{s}}} = 6/5$.

In contrast, in the limit of large compact square or rectangular target shapes, most of the moves are related to the atoms along the edges, which are the majority compared to the four atoms at the corners. Hence $\langle n_{s\bar{s}}\rangle_{s\in\mathcal{B}_{\bar{s}}} = 1$ and $\langle\langle n_{ss''}\rangle_{s''\in\mathcal{B}_s}\rangle_{s\in\mathcal{B}_{\bar{s}}} \to 3$. With these numbers, we obtain $M_0^{\text{appr}}(\bar{s}) = -1.57 < 0$ leading to a minimum.

Since the presence of the minimum depends only on the generic competition between the

relative energy effect and global slowing down, we speculate that it should not depend on the details of the kinetics, such as edge or surface diffusion, or dislocation-mediated dynamics, etc. (as long as the number of atoms is conserved).

2.3.4 Random policy

Beyond the zero-force policy, another natural candidate for a reference policy that does not contain information on the system is to pick a force at random. There are several possible ways to implement this idea. A first choice is to set the force to a randomly chosen value in each state once for all. Another one is to switch the force randomly at each move. We have chosen this latter procedure.

Note that Eq. (2.29) suggests that the high-temperature behaviour should be independent of the policy $\varphi(s)$ for a random policy. Indeed $\langle\varphi(s)\rangle = 0$ and $\varphi(s)$ is not correlated with the other quantities. This suggests that the normalised high-temperature slope is the same for zero-force and random policies.

In Fig. 2.15 we show a plot of the expected return time to target for three different targets. We can see that the two policies are not only identical in the high-temperature regime, but also similar in the full temperature range considered.

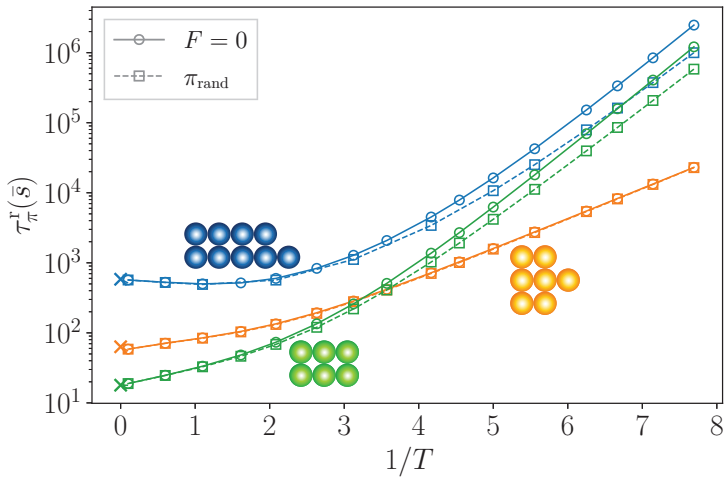


Figure 2.15: Comparison of the expected return time to target $\tau_\pi^{\text{r}}(\bar{s})$ for the zero-force case and under a random policy π_{rand}, with $F_0 = 0.4$ (i.e. a policy that chooses a random force with equal probability in each state) for the three targets shown in the figure, as a function of $1/T$.

2.4 Optimal policy in the presence of forces

Our goal now is to determine the optimal policy $\boldsymbol{\phi}_*$ that maximizes the value function of our MDP. Since we associate state values to the opposite of expected first passage times to reach the target $\bar{s}$ (see Eq. (2.11)), this corresponds to finding $\boldsymbol{\phi}_*$ that minimizes $\tau_{\boldsymbol{\phi}}(s, \bar{s})$, and the resulting optimal first passage time $\tau_*(s, \bar{s}) = \min_{\boldsymbol{\phi}(s)} \tau_{\boldsymbol{\phi}}(s, \bar{s})$ for non-zero forces.

Such a problem can be solved using well-known dynamic programming algorithms [57, 110]. We have implemented the value iteration method described in Section 1.4.2 in Python. We

substitute the optimal policy in Eq. (2.12) to obtain the Bellmann optimality equation for the optimal first passage time to target

$$\tau_*(s,\bar{s}) = \min_{\boldsymbol{\phi}(s)}\Big[t_{\boldsymbol{\phi}}(s) + \sum_{s'\in\mathcal{B}_s} p_{\boldsymbol{\phi}}(s,s')\tau_*(s',\bar{s})\Big]\,. \tag{2.52}$$

Then, as in the zero-force case, we iterate Eq. (2.52), but now a minimization over the possible values of the force in state s must be taken at each iteration.

An example of optimal policy is shown in Fig. 2.5b. If we compare this graph with that of unbiased dynamics of Fig. 2.5a, we see that the optimal policy tries to avoid the compact state in the center, which is the slowest, by decreasing the transition probabilities leading to it.

As an important observation, the force can drive the cluster towards any target shape even if the symmetries of the target are not compatible with those of the force. In other words, since in a closed-loop approach the control agent can *see* the cluster during evolution, it can drive the dynamics toward a shape that does not have the left-right symmetry, even if this left-right symmetry is a symmetry of the action set $\mathcal{A} = \{-F_0\hat{\boldsymbol{x}}, \mathbf{0}, F_0\hat{\boldsymbol{x}}\}$. The blue target of Fig. 2.16 is an example of target shape without the left-right symmetry.

In Fig. 2.16 we show the resulting optimal return time to target $\tau^{\mathrm{r}}_*(\bar{s})$ for the three targets that we already considered for the zero-force case in Fig. 2.7. The first important observation is that the optimal policy significantly decreases the expected return time to target, especially at low temperatures and for bigger targets. At $T\to\infty$, the curves for $\tau^{\mathrm{r}}_*(\bar{s})$ and $\tau^{\mathrm{r}}_0(\bar{s})$ both approach the theoretical value of $\tau^{\mathrm{r}}_\infty(\bar{s})$ given by Eq. (2.20). This is expected, since at infinite temperature all transition rates are equal and the external force has no effect on the dynamics.

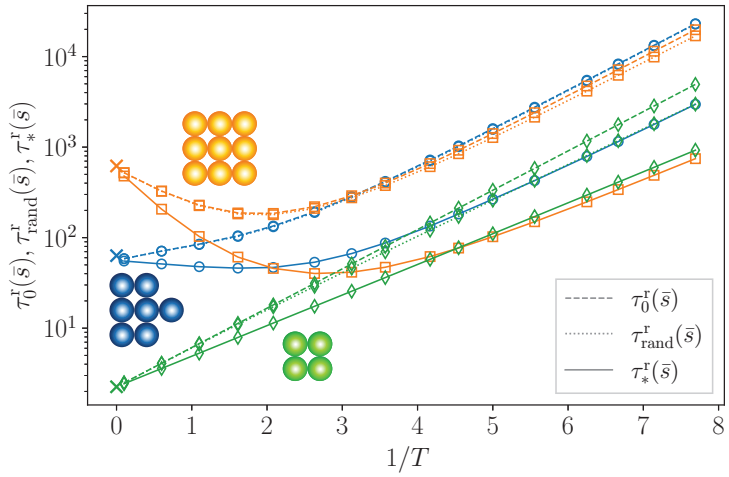


Figure 2.16: Expected return time to target as a function of $1/T$. Zero-force case $\tau^{\mathrm{r}}_0(\bar{s})$, random-force case $\tau^{\mathrm{r}}_{\mathrm{rand}}(\bar{s})$ and under an optimal policy $\tau^{\mathrm{r}}_*(\bar{s})$, with $F_0 = 0.4$. The $\times$ symbols correspond to $\tau^{\mathrm{r}}_\infty(\bar{s})$.

Furthermore, we can see that the optimal policy can shift the minimum of the return time to target to a lower optimal temperature (this is the case for the 9-atoms square target), or even create one, if it was not present in the absence of force (this is the case for the 7-atoms target).

The decrease in the expected return time to target can be better visualized through the optimization gain, defined as the ratio between $\tau^{\mathrm{r}}_0(\bar{s})$ and $\tau^{\mathrm{r}}_*(\bar{s})$ (in Section 2.3.4 we show that using the random-force policy as a reference leads to similar results).

The gain due to the application of the optimal policy is reported in Fig. 2.17, for different magnitudes of the external force F_0 and for different values of N. As seen from Fig. 2.17a, the gain increases not only when the force is increased, but also when the temperature is decreased. This temperature dependence is intuitively expected since the relative change between different rates due to the force increases monotonously when the temperature is decreased.

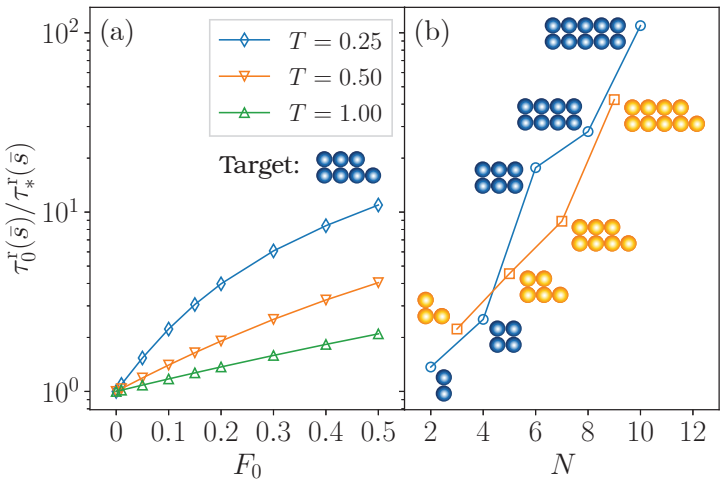


Figure 2.17: Gain $\tau_0^{\mathrm{r}}(\bar{s})/\tau_*^{\mathrm{r}}(\bar{s})$ in the return time to target due to the optimization of the forces. (a) As a function of the force magnitude F_0, for a fixed target at different T. (b) For similar targets of increasing size, with $T = 0.24$ and $F_0 = 0.4$.

In addition, the gain increases when the size of the cluster increases, as shown in Fig. 2.17b. A naive explanation for this trend is that an increase of N leads to an increase of the number of states S_N, and therefore to an increase of the number of ways to tune the policy $\boldsymbol{\phi}$ in order to minimize $\tau_{\boldsymbol{\phi}}(s, \bar{s})$. Again, the high-temperature expansion provides further hints on the dependence in N.

To linear order, the optimal return time to target reads (see Eq. (2.27))

$$\tau_*^{\mathrm{r}}(\bar{s}) = \left(1 + \frac{M_*(\bar{s})}{T}\right) \tau_\infty^{\mathrm{r}}(\bar{s})\,, \tag{2.53}$$

where the normalized slope for the optimal policy is calculated as follows.

From Eq. (2.29), we see that $\tau_{\boldsymbol{\phi}}^{\mathrm{r}}(\bar{s})$ is linear in the forces $\varphi(s)$. Hence, we obtain the optimal policy $\boldsymbol{\phi}_*(s) = \varphi_*(x)\hat{\boldsymbol{x}}$ from the sign of the prefactor of $\varphi(s)$ in Eq. (2.29):

$$\begin{aligned} \varphi_*(s) &= -F_0 \,\mathrm{sign}\left[d_s g_u(s,\bar{s}) - \langle u_{ss'}\rangle_{s'\in\mathcal{B}_s}\right]\,, \\ \varphi_*(\bar{s}) &= -F_0 \,\mathrm{sign}\left[g_u(\bar{s})\right]\,. \end{aligned} \tag{2.54}$$

If the terms in the squared brackets vanish, then the contribution of the force term is at least second order in $1/T$, and should therefore be negligible for temperatures that are high enough.

Using Eq. (2.54) in Eq. (2.29), we obtain the high-temperature correction to the optimal

expected return time to target

$$
\begin{aligned}
M_*(\bar{s}) &= \frac{1}{1-S_N^{-1}} \Big\{ \Big\langle (1-\delta_{s\bar{s}}) \langle n_{ss'} \rangle_{s' \in \mathcal{B}_s} - d_s g_n(s,\bar{s}) \Big\rangle_{s \in \mathcal{S}} \\
&\quad - F_0 \Big\langle \Big| (1-\delta_{s\bar{s}}) \langle u_{ss'} \rangle_{s' \in \mathcal{B}_s} - d_s g_u(s,\bar{s}) \Big| \Big\rangle_{s \in \mathcal{S}} \Big\} \\
&= M_0(\bar{s}) - \frac{F_0}{1-S_N^{-1}} \Big\langle \Big| (1-\delta_{s\bar{s}}) \langle u_{ss'} \rangle_{s' \in \mathcal{B}_s} - d_s g_u(s,\bar{s}) \Big| \Big\rangle_{s \in \mathcal{S}} .
\end{aligned}
\tag{2.55}
$$

As we did for the zero-force case, we now derive an approximate expression for the high-temperature slope M_*.

2.4.1 Approximate expression of M_*

Heuristically, we expect that the presence of an absolute value in the second term of Eq. (2.55) forbids the cancellation of contributions with randomly different signs, leading to a behaviour which is qualitatively different from that of M_0. Indeed, we speculate that the average is not dominated by the largest terms coming from the strong change of $\tau_\infty(s,\bar{s})$ between the target and its first neighbors, but by the typical values of $|d_s g_u(s,\bar{s})|$ in all states (as we will see later, the terms $\langle u_{ss'} \rangle_{s' \in \mathcal{B}_s}$ are small and their contribution to the average is negligible).

Assuming for the sake of simplicity complete uncorrelation between $u_{ss'}$ and $\tau_\infty(s',\bar{s})$, the covariance $|d_s g_u(s,\bar{s})|$ can be approximated as the product of the standard deviations σ_u and $\sigma_{\tau_\infty}(\bar{s})$ of $u_{ss'}$ and $\tau_\infty(s,\bar{s})$ in $\mathcal{B}_s$, times the absolute value of a sum of d_s random signs that account for the possible signs of the products. This sum is approximated by the standard formula for the expectation value of the absolute distance after d_s steps as $(2d_s/\pi)^{1/2}$ [111, 112]. The standard deviations are defined as

$$
\begin{aligned}
\sigma_u &= \Big\langle \big\langle (u_{ss'} - \langle u_{ss''} \rangle_{s'' \in \mathcal{B}_s})^2 \big\rangle_{s'' \in \mathcal{B}_s}^{1/2} \Big\rangle_{s \in \mathcal{S}} , \\
\sigma_{\tau_\infty}(\bar{s}) &= \Big\langle \big\langle (\tau_\infty(s',\bar{s}) - \langle \tau_\infty(s'',\bar{s}) \rangle_{s'' \in \mathcal{B}_s})^2 \big\rangle_{s' \in \mathcal{B}_s}^{1/2} \Big\rangle_{s \in \mathcal{S}} .
\end{aligned}
\tag{2.56}
$$

As a consequence, we have

$$
\langle |d_s g_u(s,\bar{s})| \rangle_{s \in \mathcal{S}} \approx \left(\frac{2}{\pi} \right)^{1/2} \langle d_s^{1/2} \rangle_{s \in \mathcal{S}} \, \sigma_u \, \sigma_{\tau_\infty}(\bar{s}) . \tag{2.57}
$$

As shown in Fig. 2.18, this expression provides a fair approximation to $\langle |d_s g_u(s,\bar{s})| \rangle_{s \in \mathcal{S}}$. The contributions of the averages $\langle u_{ss'} \rangle_{s' \in \mathcal{B}_s}$ to the term $\langle |(1-\delta_{s\bar{s}}) \langle u_{ss'} \rangle_{s' \in \mathcal{B}_s} - d_s g_u(s,\bar{s})| \rangle_{s \in \mathcal{S}}$ is negligible. This is also shown in Fig. 2.18, where the difference between $\langle |d_s g_u(s,\bar{s})| \rangle_{s \in \mathcal{S}}$ and $\langle |d_s g_u(s,\bar{s}) - \langle u_{ss'} \rangle_{s' \in \mathcal{B}_s}| \rangle_{s \in \mathcal{S}}$ is seen to be small.

Therefore, by using Eq. (2.57) to approximate the average of Eq. (2.55) and since $S_N \gg 1$, we obtain

$$
M_*(\bar{s}) \approx M_*^{\mathrm{appr}}(\bar{s}) = M_0(\bar{s}) - \left(\frac{2}{\pi} \right)^{1/2} \langle d_s^{1/2} \rangle_{s \in \mathcal{S}} \, \sigma_u \, \sigma_{\tau_\infty}(\bar{s}) \, F_0 . \tag{2.58}
$$

This formula can serve as a basis to analyse the dependence of $M_*(\bar{s})$ with N. Using the approximate expression Eq. (2.51), M_0 is seen to be composed of two contributions. The first one is $\rho_0(N)$, which is bounded and converges exponentially with N as already discussed in Section 2.3.3. The second contribution $\langle n_{s\bar{s}} - \langle n_{ss''} \rangle_{s'' \in \mathcal{B}_s} \rangle_{s \in \mathcal{B}_{\bar{s}}}$ is bounded because $1 \leq n_{ss'} \leq 4$.

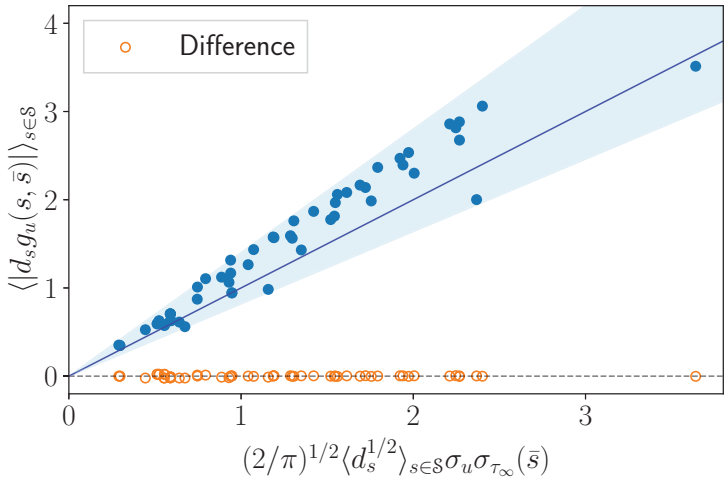


Figure 2.18: Blue dots: mean of the absolute value of the covariance $|d_s g_u(s,\bar{s})|$ over all states as a function of its approximation Eq. (2.57). The orange dots represent the difference between $\langle |d_s g_u(s,\bar{s})| \rangle_{s\in\mathcal{S}}$ and $\langle |d_s g_u(s,\bar{s}) - \langle u_{ss'}\rangle_{s'\in\mathcal{B}_s}| \rangle_{s\in\mathcal{S}}$.

Hence, from Eq. (2.51), the approximation of $M_0(\bar{s})$ is bounded. Let us now consider the second term of Eq. (2.58), which is proportional to F_0. This term is proportional to the three factors σ_u, $\sigma_{\tau_\infty}(\bar{s})$, and $\langle d_s^{1/2}\rangle_{s\in\mathcal{S}}$, that we discuss separately in the following.

In figure Fig. 2.19 we plot σ_u as a function of N for $3 \leq N \leq 11$ and show that it converges to a constant roughly equal to 0.44 (the points have been fitted with a decaying exponential). This is expected, as this quantity is bounded, because $-1/2 \leq -u_{ss'} \leq 1/2$. As seen from Fig. 2.20, $\sigma_{\tau_\infty}(\bar{s})$ grows with N. This plot does not allow us to conclude if $\sigma_{\tau_\infty}(\bar{s})$ never exceeds some finite upper bound, or if this quantity can grow without bound when N increases. However, since the average of $\tau_\infty(\bar{s})$ in the first ring is equal to the mean return time to target $\tau_\infty^{\mathrm{r}}(\bar{s})$, which grows exponentially with N from Eq. (2.20), it is tempting to speculate that $\sigma_{\tau_\infty}(\bar{s})$ grows without bound.

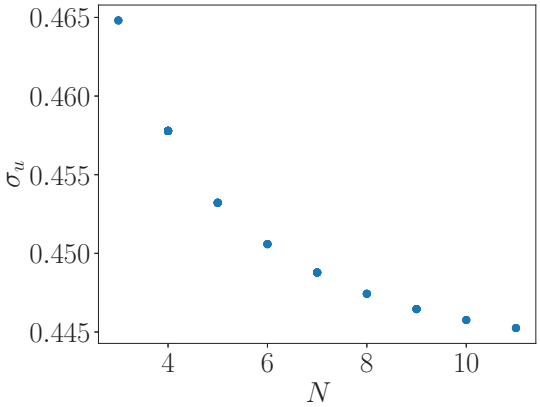


Figure 2.19: Standard deviation σ_u as a function of N. A fit with a decaying exponential gives an asymptotic value of ~ 0.44.

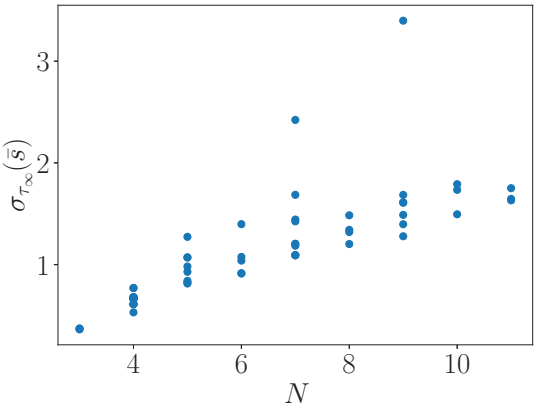


Figure 2.20: Standard deviation $\sigma_{\tau_\infty}(\bar{s})$ for different targets as a function of N.

The last factor to be considered in the last term of Eq. (2.58) is $\langle d_s^{1/2} \rangle_{s \in \mathcal{S}}$. As seen in Fig. 2.10, this term also grows with N. Since d_s can be roughly estimated to be proportional to the length of the periphery of the cluster, we expect that d_s grows at least like $N^{1/2}$, as the periphery of compact clusters, and at most like N, as the periphery of ramified clusters. Hence, we expect that $\langle d_s^{1/2} \rangle_{s \in \mathcal{S}}$ grows without bound as N to some finite power. Multiplying the three factors, we therefore expect the second contribution in Eq. (2.58) to grow without bound as N increases.

Hence, the contribution proportional to F_0 in Eq. (2.58) grows without bound with N and should always dominate over the term $M_0(\bar{s})$ for large N. Thus when N is large enough, we speculate that $M_*(\bar{s})$ should be negative and an optimal temperature should be generically present. This trend is confirmed by Fig. 2.9a and Fig. 2.21. However, we cannot conclude on the behaviour of very ramified shapes for large N, because ramification leads to an increase of the slope $M_*(\bar{s})$ that competes with the decrease expected from an increase of N. In addition, the iterative solution of Eq. (2.52) does not converge in this regime.

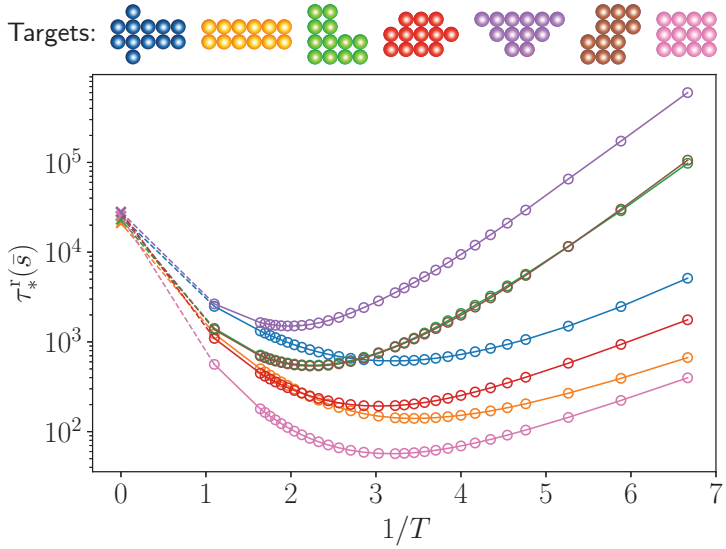


Figure 2.21: Optimal return time to target $\tau_*^{\mathrm{r}}(\bar{s})$ as a function of $1/T$ for targets with $N = 12$, with $F_0 = 0.4$. All the curves exhibit a minimum, corresponding to an optimal temperature. $\times$ correspond to the analytical expression of $\tau_\infty^{\mathrm{r}}(\bar{s})$ for $T \to \infty$.

2.4.2 Non-uniqueness of the optimal policy

As we already mentioned in Section 1.4.1, there can be more than one optimal policy that is solution of an MDP. In Fig. 2.22 we show a graph of the dynamics of the trimer ($N = 3$) cluster, with the optimal policy to reach the horizontal bar target shape in orange, computed with value iteration. Indeed, we can see that there are two states, the vertical bar (top left node) and the target itself, where the force can be set equivalently to the right or to the left, or, in other words, where the optimal action is degenerate (the numerical procedure to check degeneracy of the actions is reported in Appendix A.8). The non-uniqueness of the optimal policy is not exclusive to trimers, but appears for clusters of all the sizes that we have investigated, with $2 \leq N \leq 12$. We think that this feature is related to target symmetries, in the way that we will now explain.

Let us consider an optimal policy for a given target $\bar{s}$, which associates a force $\boldsymbol{\phi}_*(s)$ to each

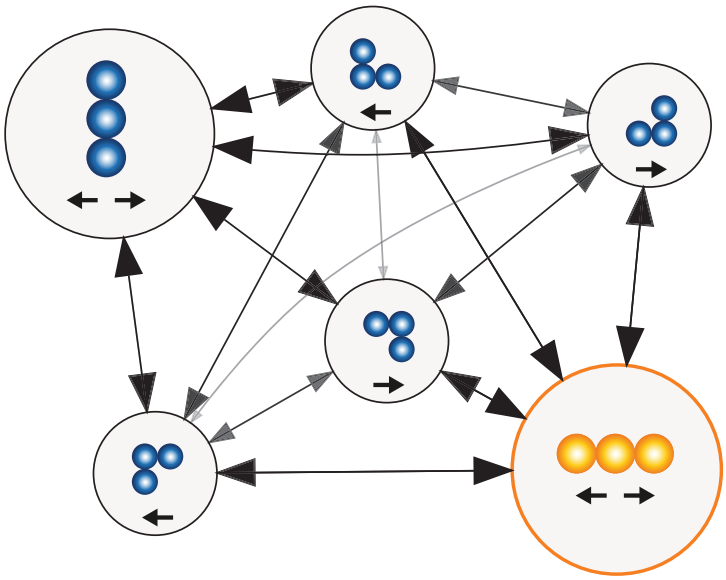


Figure 2.22: Trimer dynamics with the optimal policy computed at $T = 0.61$ to reach the orange target. In the target and in the upper left state, the optimal action is degenerate, and the force can be equivalently set to the right or left.

state s. Performing one of the transformations $x \to -x$, $y \to -y$, or $(x, y) \to (-x, -y)$ on all the states and forces, we obtain another optimal policy. If the target $\bar{s}$ is transformed in itself, then the new policy is again an optimal policy for $\bar{s}$. In addition, if a state s is transformed in itself and the force in s is transformed into its opposite, then the force $\boldsymbol{\phi}_*(s)$ and its opposite $-\boldsymbol{\phi}_*(s)$ are both optimal in state s.

Let us focus on the transformations $x \to -x$ or $(x, y) \to (-x, -y)$ that invert the x direction. Let us call f one of these transformations, and let us consider a target $\bar{s}$ which is invariant under the transformation f, i.e. which obeys $f\bar{s} = \bar{s}$. The states can then be classified in two groups: the group of *invariant* states such that $fs = s$, and the *variant* states such that $fs \neq s$. Let us first consider the variant states. A common property of the two symmetry transformations is that f^2 is the identity, i.e. $ffs = s$. A a consequence, the variant states appear in pairs (s, fs), one member of the pair being transformed into the other by f. In all the simulation results that we have obtained so far, the forces of an optimal policy are opposite within these pairs, i.e. $\boldsymbol{\phi}_*(s) = -\boldsymbol{\phi}_*(fs)$. We will therefore assume this property for this discussion (note however that we do not have a proof of its general validity, and we therefore do not know if there are policies that do not obey this rule). Two examples illustrating this property of the optimal policy are shown in Fig. 2.22 for a trimer, and in Fig. 2.23 for a tetramer. In particular, in Fig. 2.23, the states of the system are arranged such that the invariant states lie along a central axis of symmetry of the graph, while the variant states lie on opposite sides.

Under the f transformation that inverts the x direction, the optimal force transforms to its opposite $f\boldsymbol{\phi}_*(s) = -\boldsymbol{\phi}_*(s)$. Thus, for a variant state $f\boldsymbol{\phi}_*(s) = -\boldsymbol{\phi}_*(s) = \boldsymbol{\phi}_*(fs)$. This relation indicates that, under the f transformation, the policy is invariant in the variant states. In contrast, the policy is reversed by the transformation f in all invariant states, which obey $f\boldsymbol{\phi}(s) = -\boldsymbol{\phi}(s)$. As a summary, applying the f transformation, we obtain another optimal policy which is not changed in the variant states, and is reversed in all invariant states. In other words, reverting simultaneously all the forces in the invariant states of an optimal policy leads to another optimal

policy. However, due to the Markovian character of the dynamics, it is clear that each excursion of the dynamics into the variant part of the states has a symmetric counterpart with the same transition probabilities. This is again clearly visible in Fig. 2.23, where the two halves of the graph have an identical structure, both in terms of nodes connectivity and weights of the edges (which are proportional to the transition probabilities). As a consequence, choosing the forces in an invariant state to have a value or its opposite will lead to the same statistical properties of the first passage times. As a conclusion from this discussion, under the assumption that the optimal forces are opposite within pairs of variant states, we expect that the forces in each invariant state can be reverted independently from the others.

Hence, there are two possibilities for the force in an invariant state s. First possibility, the force vanishes. Indeed, it is clear that $\boldsymbol{\phi}_*(s) = 0$ is equal to its opposite. Second possibility, the force is non-zero. Then, both $\boldsymbol{\phi}_*(s) = F_0\hat{\boldsymbol{x}}$ and $\boldsymbol{\phi}_*(s) = -F_0\hat{\boldsymbol{x}}$ are solutions, and the state is said to be *degenerate*. As a summary, if a target $\bar{s}$ and state s are both invariant under a symmetry transformation and if the force in s is reversed under this transformation, then the state s has either a vanishing force, or a degenerate force.

Indeed, we can see in Fig. 2.23 that all the invariant states under the $x \to -x$ transformation (which lie along the axis of the graph) are degenerate. Note that, since we are extending the policy to the target itself, the optimal action is degenerate also on the target state.

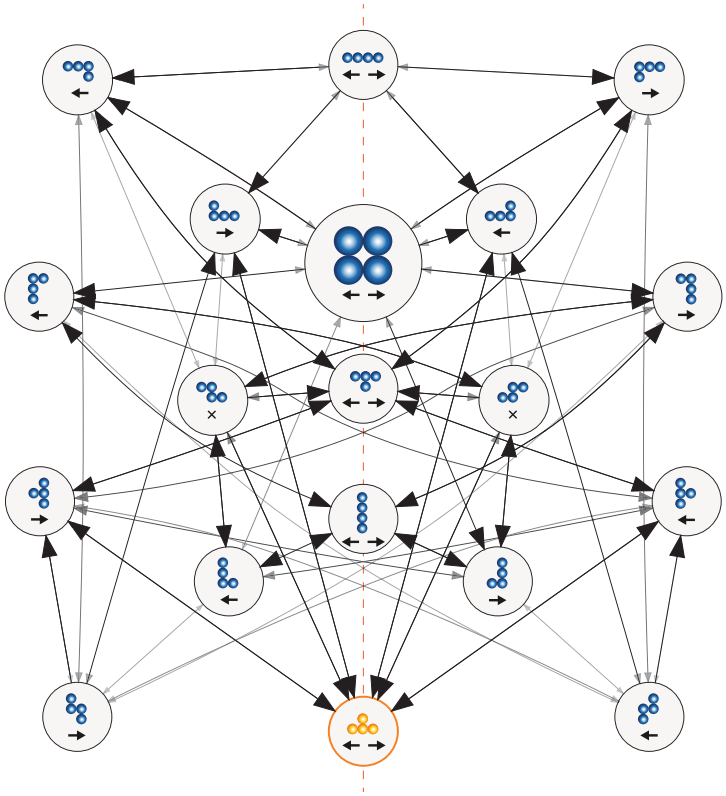


Figure 2.23: Dynamical graph of a tetramer cluster at $T = 0.24$, with the optimal policy to reach the orange target. $F_0 = 0.4$.

For the most trivial case of a dimer ($N = 2$) with only two states ($S_2 = 2$) that are both invariant under the $x \to -x$ transformation, the policies are therefore trivially degenerate. The

dimer case can actually be solved analytically.

Analytical solution of the dimer

There are only two states of the dimer cluster: the vertical and the horizontal bar, as shown in Fig. 2.24. Let us pick the latter as our target $\bar{s}$ (the calculation is identical if we choose the vertical bar) and let us calculate the optimal policy $\boldsymbol{\phi}(s)$ in the other state.

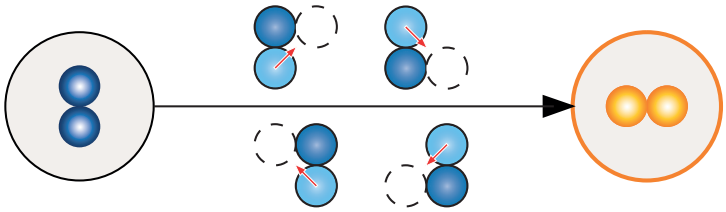


Figure 2.24: Graph of configurations of a dimer, with the 4 possible atomic moves to reach the orange target. Top two moves have $u_{ss'} = 1/2$ while the two bottom ones have $u_{ss'} = -1/2$.

Since on the target $\tau_{\boldsymbol{\phi}}(\bar{s}, \bar{s}) = 0$, the expected time to reach the target starting from the vertical bar is simply the residence time $\tau_{\boldsymbol{\phi}}(s, \bar{s}) = t_{\boldsymbol{\phi}}(s)$. There are 4 possible atomic moves when going from s to $\bar{s}$, which are also shown in Fig. 2.24. This is a specific property of dimers. Indeed, larger clusters always exhibit a unique transition from one state to another[5]. Two moves are in the positive horizontal direction and have $u_{ss'} = 1/2$ while the other two are in the negative direction and have $u_{ss'} = -1/2$, hence

$$\begin{aligned} \tau_{\boldsymbol{\phi}}(s, \bar{s}) = \tau^{\mathrm{r}}_{\boldsymbol{\phi}}(\bar{s}) = t_{\boldsymbol{\phi}}(s) &= \left\{ 2\exp\left(-\frac{1-\varphi(s)/2}{T}\right) + 2\exp\left(-\frac{1+\varphi(s)/2}{T}\right) \right\}^{-1} \\ &= \left\{ 2\exp\left(-\frac{1}{T}\right)\left[\exp\left(\frac{\varphi(s)}{2T}\right) + \exp\left(\frac{-\varphi(s)}{2T}\right)\right] \right\}^{-1} = \frac{\exp\left(\dfrac{1}{T}\right)}{4\cosh\left(\dfrac{\varphi(s)}{2T}\right)}, \end{aligned} \tag{2.59}$$

where we recall that $\varphi(s)$ is the scalar force in state s, so that $\boldsymbol{\phi}(s) = \varphi(s)\,\hat{\boldsymbol{x}}$.

Since the hyperbolic cosine is a symmetric function, both choices $\varphi(s) = \pm F_0$ result in the same expected time to target $\tau_{\boldsymbol{\phi}}(s, \bar{s})$. Moreover, these choices are both optimal, because $\cosh(\pm F_0/(2T)) > \cosh(0) = 1$, for $F_0 > 0$. Thus, for the dimer, we have two equivalent optimal policies $\boldsymbol{\phi}_*(s) = \pm F_0 \hat{\boldsymbol{x}}$.

2.4.3 Degeneracy in bigger clusters

To investigate the relation between degeneracy of the optimal actions and symmetries in more details, we define five mutually exclusive symmetry classes of states. One example of each class is shown in the first line of Fig. 2.25. The first class $\mathcal{V}$ includes clusters with mirror symmetry with respect to a vertical y axis, or invariance under the $x \to -x$ transformation. The second class $\mathcal{H}$ includes clusters with mirror symmetry with respect to a horizontal x axis, or invariance

[5]Because of the presence of multiple atomic moves for the transition between two states, our previously defined notation $\gamma_{\boldsymbol{\phi}}(s, s')$, which identifies both to the atomic rates Eq. (2.4) and to the transition rates Eqs. (2.6) and (2.8) is not suitable for dimers. In order to avoid the introduction of a new notation, we simply avoid the use of $\gamma_{\boldsymbol{\phi}}(s, s')$ for the analysis of dimers, and use these notations in the case where it is valid, i.e. for $N \geq 3$.

under the $y \to -y$ transformation. The third class $\mathcal{I}$ corresponds to clusters with space inversion symmetry, or invariance under the transformation $(x, y) \to (-x, -y)$. The fourth class $\mathcal{A}$ includes clusters that are separately invariant under the $x \to -x$ transformation and under the $y \to -y$ transformation. The fifth class $\mathcal{N}$ corresponds to cluster with none of the above symmetries. These five classes cover all cluster configurations and have no overlap, in the sense that each shape belongs to only one class, corresponding to the class with the highest symmetry where it can be included.

Let us consider how states belonging to these five symmetry classes and the associated optimal force are transformed when the entire dynamical graph of the system and the force are transformed under one of the three transformations $x \to -x$, $y \to -y$, or $(x, y) \to (-x, -y)$. This is illustrated in the table of Fig. 2.25. Among the 15 cases in the table, four cases have a transformed shape which is identical to the initial one, with a force that is flipped. These 4 cases, which correspond either to a vanishing or degenerate optimal force, have a background colored in yellow in the table.

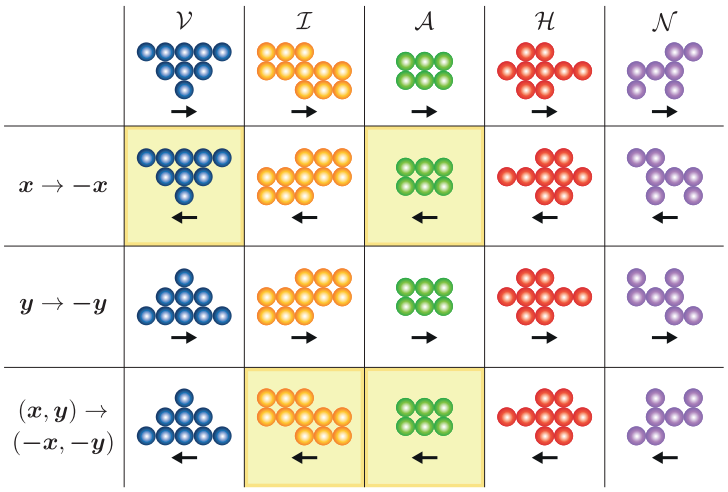


Figure 2.25: The five symmetry classes and their transformations.

More precisely, we expect the following symmetry rules. For a $\mathcal{V}$ target, all states that are invariant with respect to the $x \to -x$ transformation, i.e. the $\mathcal{V}$ and $\mathcal{A}$ states, will be either degenerate or with a zero-force optimal action. Instead, for a $\mathcal{I}$ target, all states that are invariant with respect to the $(x, y) \to (-x, -y)$ transformation, i.e. the $\mathcal{I}$ and $\mathcal{A}$ states, will be either degenerate or with a zero-force optimal action. Finally, for a target in the $\mathcal{A}$ symmetry class, all states that are invariant with respect to either the $x \to -x$, or the $(x, y) \to (-x, -y)$ transformation, i.e. the $\mathcal{V}$, $\mathcal{I}$ and $\mathcal{A}$ states, will be either degenerate or with a zero-force optimal action.

We have checked the validity of the above symmetry rules for all possible targets with $N \leq 7$ ($S_7 = 760$): states belonging the above mentioned classes have either a vanishing or a degenerate optimal force. Due to computational limitations, we were able to check only some arbitrarily selected targets with $8 \leq N \leq 12$. Again, all states obeying the symmetry rules mentioned above exhibit a vanishing or degenerate optimal force.

However, when a state do not obey these symmetry rules, we have no indication about its possible degeneracy. Indeed, other "hidden" symmetries of the graph could come into play. We found only 10 cases of degenerate states outside these symmetry rules, and they all correspond to the tetramer ($N = 4$, $S_N = 19$) cases. Two examples of such targets are shown in Fig. 2.26. The

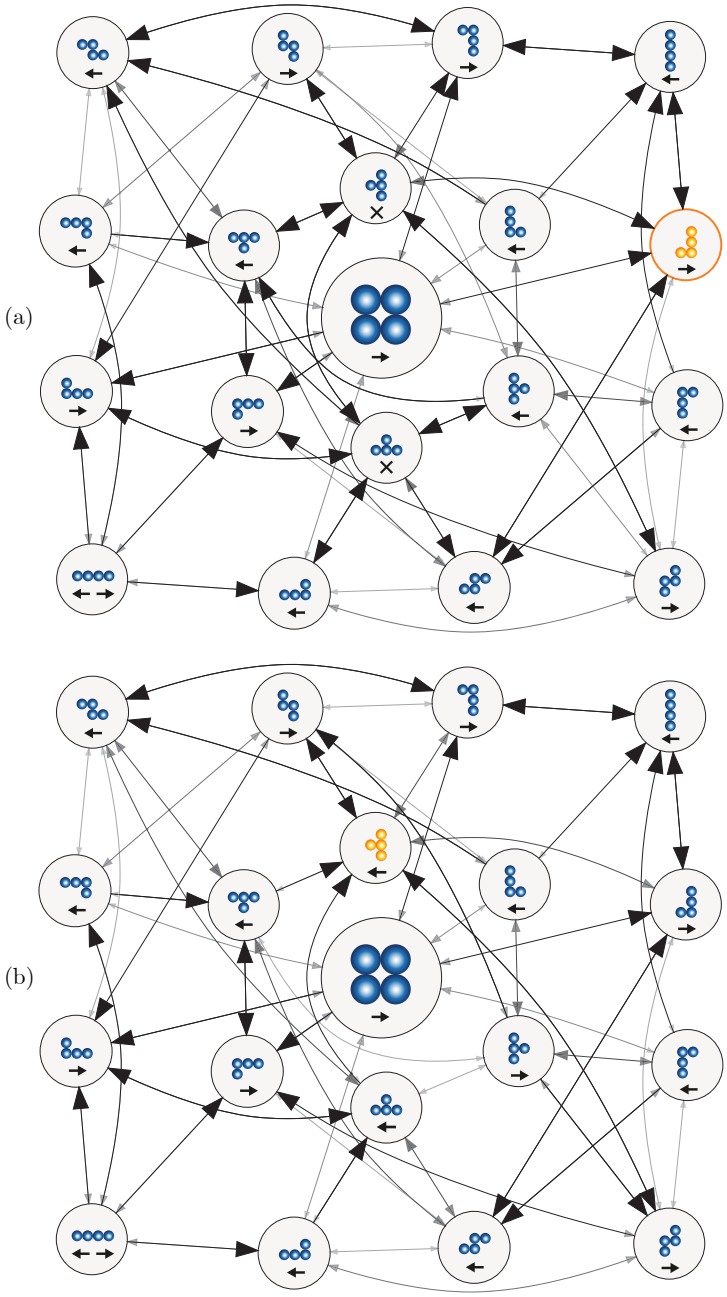


Figure 2.26: Dynamical graphs of a tetramer cluster at $T = 0.24$, with the optimal policies to reach the two orange targets, for $F_0 = 0.4$. The target in (a) belongs to the $\mathcal{N}$ symmetry class, while the one in (b) belongs to the $\mathcal{H}$ symmetry class. However, in both cases, the bottom left state shows a degenerate optimal action.

other 8 correspond to all the possible rotations and reflections of these two that belong to the $\mathcal{N}$ or $\mathcal{H}$ symmetry class. They are the only cases for $N \leq 7$, however we do not know if there are other cases for $8 \leq N \leq 12$, since we could not check all possible targets in this range.

For clusters with $N \geq 4$, the dynamical graph is too large to be displayed. Therefore, to study degeneracy in bigger clusters, we focus on the frequency of appearance of degenerate states. We define $D_N(\bar{s})$ as the number of degenerate states in the optimal policy of a target $\bar{s}$ of size N. The fraction of degenerate states $D_N(\bar{s})/S_N$ for targets of size $4 \leq N \leq 12$ is plotted in Fig. 2.27a for a fixed temperature $T = 0.24$. In Fig. 2.27a, we have reported all targets with $4 \leq N \leq 7$, and some arbitrary selected targets with $8 \leq N \leq 12$. Beyond the exceptions at $N = 4$, the fraction $D_N(\bar{s})/S_N$ vanishes for the $\mathcal{H}$ and $\mathcal{N}$ classes. In contrast, the targets belonging to the other classes of symmetry exhibit a non-zero fraction of degenerate states. We also observe that the variation of the values of the fraction of degenerate states for different targets with the same N is small as compared to the variation with N. We therefore conclude that the most relevant parameter for the variation of the fraction of degenerate states is N.

The fraction $D_N(\bar{s})/S_N$ for targets that belong to $\mathcal{V}$, $\mathcal{I}$, or $\mathcal{A}$ symmetry classes is found to decay exponentially with N. Let us define P_N, the number of polyominoes of size N belonging to one of the three symmetry classes $\mathcal{V}$, $\mathcal{I}$ or $\mathcal{A}$. In Fig. 2.27b, the fraction of symmetric polyominoes P_N/S_N (for $4 \leq N \leq 14$) is found to exhibit the same exponential decay as Redelmeier's results from Ref. [113] up to $N = 24$ (and extended to $N = 28$ by Oliveira e Silva [114]). Note that the fractions of symmetric polyominoes computed by Redelmeier are larger than ours because they take into account more symmetry classes, however the scaling of this fraction with respect to N is the same $\sim \mu^N$, where $\mu \approx 0.51$ (the fit was done by considering odd and even values of N separately). The fraction of degenerate states for the $\mathcal{V}$, $\mathcal{I}$, and $\mathcal{A}$ classes of targets in Fig. 2.27a also behaves according to the same scaling $D_N(\bar{s})/S_N \sim \mu^N$. A value $\mu < 1$ indicates that $D_N(\bar{s})/S_N \to 0$ for large N. However, the number of degenerate states in a given class $D_N(\bar{s}) \sim (\lambda\mu)^N$ with $\lambda\mu \approx 2.07 > 1$ grows exponentially with N.

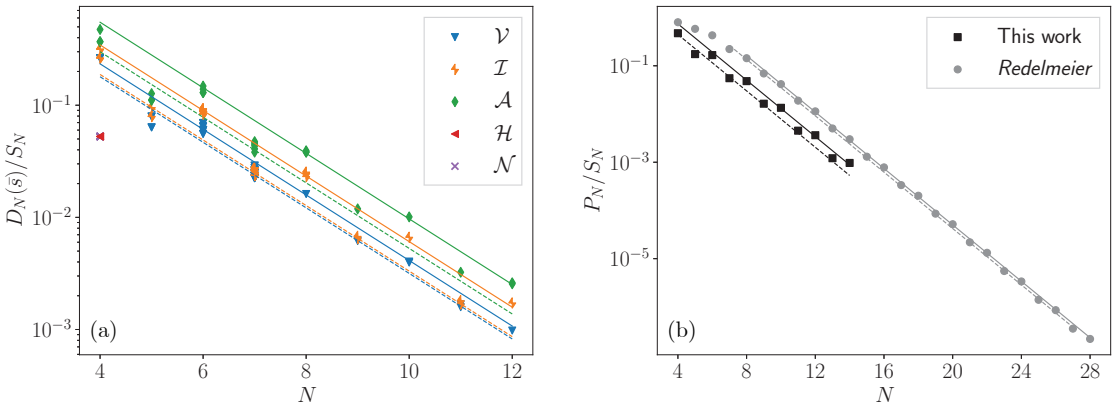


Figure 2.27: (a) Fraction of degenerate states for several targets at a fixed temperature $T = 0.24$. The lines correspond to the scaling $D_N(\bar{s})/S_N \sim \mu^N$, with $\mu \approx 0.51$ obtained fitting Redelmeier's data in the (b) fraction of symmetric polyominoes with area N. Our results consider only $\mathcal{V}$, $\mathcal{I}$ and $\mathcal{A}$ symmetry classes, while Redelmeier's results include more classes.

As discussed above, states that obey the symmetry rules are also compatible with a zero force. Thus, the number of degenerate states is not fixed by symmetry and can vary with temperature. However, an analysis reported in Appendix A.9 shows that $D_N(\bar{s})/S_N$ only varies weakly with temperature. Hence, the results discussed above on the variation of $D_N(\bar{s})/S_N$ with N at $T = 0.24$

are expected to hold at other temperatures.

As a summary, two conclusions can be drawn from this analysis of degenerate states. First, degeneracies in the optimal actions are mostly associated to symmetries of the target state. Second, although the number of degenerate states increases exponentially, the fraction of degenerate states vanishes as N increases.

2.4.4 Transitions of the optimal policy with the temperature

The second interesting feature is that $\boldsymbol{\phi}_*$ is, in general, not constant as a function of temperature. As opposed to the dimer case, where the policy is independent from the temperature, transitions in the policy are often observed. An example of policy transition for the trimer is shown in Fig. 2.28. In the state at the center of the graph, the optimal action flips from right to left at a critical temperature $T = T_c$, with $0.66 < T_c < 0.67$. The optimal policies above and below the transition at $T \leq 0.66$ and $T \geq 0.67$ are represented in Fig. 2.28.

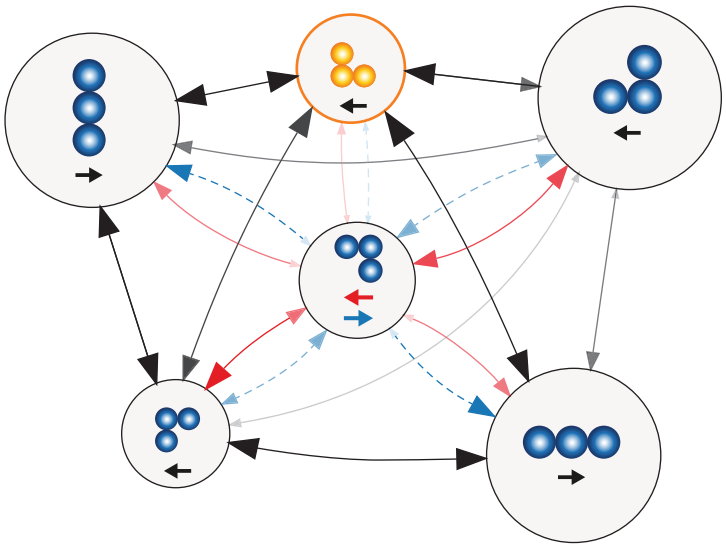


Figure 2.28: Trimer dynamics with two optimal policies for reaching the target in orange at different temperatures: $T = 0.66$ (in blue) and $T = 0.67$ (in red). In the state in the center, the optimal policy changes between low and high temperature.

The mean return time to target $\tau^{\mathrm{r}}_*(\bar{s})$ for the trimer target of Fig. 2.28 is reported in Fig. 2.29. The transition at T_c is seen to be continuous for $\tau^{\mathrm{r}}_*(\bar{s})$, but discontinuous for its first derivative with respect to the inverse temperature. The continuity of $\tau^{\mathrm{r}}_*(\bar{s})$ means that the low-temperature and the high-temperature policies have the same performance at the transition. However, the derivatives are properties of the policies themselves, and are therefore expected to be different on the left and right sides of the transition. The first passage times to the target from the other 5 states outside the target exhibit the same features and are reported in Appendix A.10.

As an additional remark, the discontinuity of the derivative is always negative. Indeed, let us denote the high-temperature and low-temperature optimal policies on both sides of the transition as $\boldsymbol{\phi}_{HT}$ and $\boldsymbol{\phi}_{LT}$. The discontinuity corresponds to a crossing of the functions $\tau^{\mathrm{r}}_{\boldsymbol{\phi}_{LT}}(\bar{s})$ and $\tau^{\mathrm{r}}_{\boldsymbol{\phi}_{LT}}(\bar{s})$ that must be optimal at high and low temperatures respectively. As shown in Fig. 2.30, the policy $\boldsymbol{\phi}_{LT}$ on the low-temperature side must have a smaller slope than $\boldsymbol{\phi}_{HT}$ as a function of the inverse temperature. As a consequence, the jump of the derivative of $\tau^{\mathrm{r}}_*(\bar{s})$ is always negative.

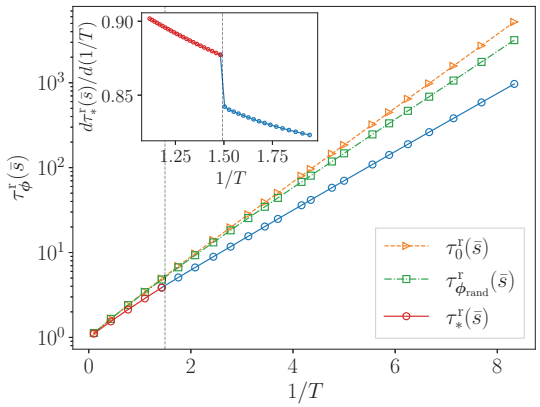


Figure 2.29: Expected return time to target as a function of $1/T$ for the optimal policy $\boldsymbol{\phi}_*$, the random policy $\boldsymbol{\phi}_{\mathrm{rand}}$ and the zero-force for the trimer target of Fig. 2.28. Inset: first derivative of the optimal return time near the transition temperature T_c (vertical line).

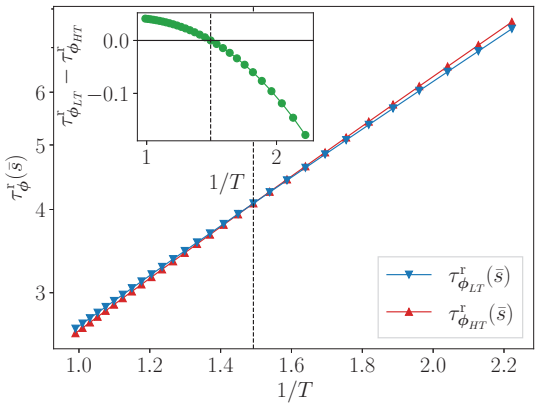


Figure 2.30: Expected return time to target for the high- and low-temperature optimal policies. Inset: difference between the two.

The increase of the cluster size N leads to a strong increase of the number of transitions of the optimal policy. As N reaches 12, the number of transitions is so large that identifying each single transition is clearly not a meaningful approach. As we did for degeneracy in the previous section, we cannot rely on a visual analysis of dynamical graphs when looking at bigger clusters. We therefore resort to a statistical analysis of the transitions. We define the density of change

$$\rho_N(T, \bar{s}) = \frac{\Delta S_N(T, \bar{s})}{S_N \Delta T}, \tag{2.60}$$

where $\Delta S_N(T, \bar{s})$ is the number of states where the optimal action has changed between T and $T + \Delta T$, for a target $\bar{s}$ of size N. In Fig. 2.31, we show $\rho_N(T, \bar{s})$ with $\Delta T = 0.02$ for several targets of size 7 and 12.

The first important feature that emerges from this plot is that $\rho_N(T, \bar{s})$ is of the order of one, and therefore $\Delta S_N(T, \bar{s})$ grows exponentially like $S_N \Delta T$ when N increases. Second, $\rho_N(T, \bar{s})$ becomes smoother for $N = 12$, suggesting that the density of transitions tends to a well defined smooth function of T as N increases. To analyse this behavior more systematically, we have computed the roughness of the function $\rho_N(T, \bar{s})$ for $7 \leq N \leq 12$ at different values of ΔT, ranging from 0.005 to 0.1. We use a simple measure of the roughness, defined as the number of local extrema (obtained by counting the sign changes of the slope) divided by the total number of points in the curve.

A plot of the roughness of the density of change (averaged over several targets for each value of N) as a function of ΔT is reported in Fig. 2.32a. We can then set a threshold for the roughness above which the density of transitions can be considered to be smooth. For each N, there is a critical value of ΔT, that we call $(\Delta T)_{\mathrm{sm}}$, above which $\rho_N(T, \bar{s})$ becomes smooth. In Fig. 2.32b we plot the trend of $(\Delta T)_{\mathrm{sm}}$ as a function of N for four thresholds of the roughness. Even for the most restrictive threshold of 5%, the critical value of ΔT above which the density of change becomes smooth is consistently decreasing to zero as N increases, confirming the trend first seen in Fig. 2.31.

It is tempting to speculate that the emergence of a kind of continuum limit for a system as

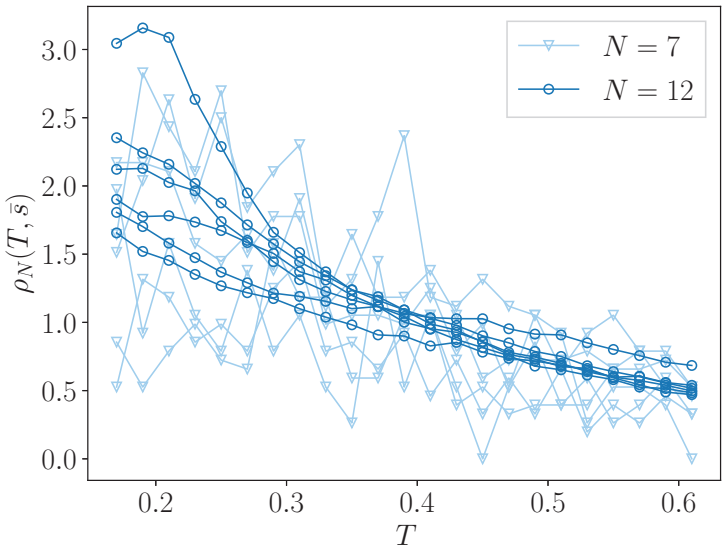


Figure 2.31: Density of change $\rho_N(T, \bar{s})$ as a function of T for several targets of size 7 and 12 (each curve corresponds to a different target). The step in the temperature is $\Delta T = 0.02$.

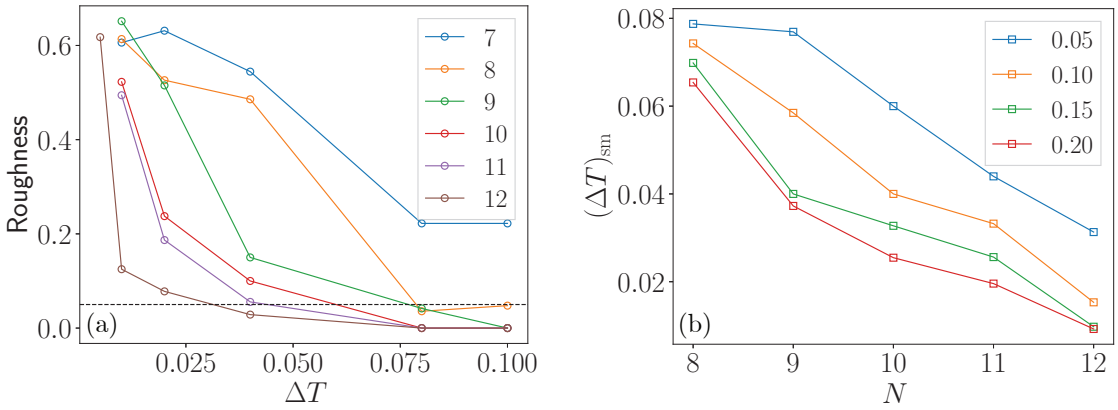


Figure 2.32: Roughness of the temperature transition density $\rho_N(T, \bar{s})$. (a) Average roughness as a function of ΔT for different values of N. The dashed horizontal line corresponds to the lowest value of the threshold, at 0.05. (averages are performed over 10 targets for $N = 7$, to 4 targets for $N = 10$. For $\Delta T = 0.005$ and $N = 12$, the average is performed on 2 targets and in a smaller temperature range $0.21 \leq T \leq 0.3$). (b) Critical value $(\Delta T)_{\rm sm}$ for different thresholds of roughness.

small as $N = 12$ particles relies on the double-exponential increase of the number of policies $3^{S_N} \sim 3^{c\lambda^N/N}$ which reaches quickly very large numbers ($\sim 10^{10^5}$ for $N = 12$). However, at this point, we have no theoretical understanding of the convergence of the density of transitions to a possible continuum limit.

2.5 Discussion and perspectives

We have studied the first passage time to a target shape at equilibrium for a monolayer cluster evolving via random motion of atoms along the periphery of the cluster.

We have seen how thermal fluctuations—which are usually considered to be detrimental—can be used to reach desired nano-cluster shapes. We found that there is an optimal temperature where compact target shapes that are large enough are reached in minimum time. In the presence of a feedback-controlled (i.e. closed loop) macroscopic field that biases diffusion, the time to reach the target shape can be decreased by orders of magnitude, with a gain that increases with decreasing temperature and with increasing cluster size.

For the majority of metals where edge diffusion is observed (e.g., Ag and Cu), the edge diffusion barrier $\sim J$ or kink energies $\sim J/2$ suggest that $J \approx 0.2$ and 0.3 eV [115, 116, 117, 118, 75]. For the square 9-atom target depicted in Fig. 2.16 the optimal temperature corresponds to $J/k_BT \approx 2$. Choosing $J = 0.2$ eV we obtain an optimal temperature of about 10^3 K, which is too high to be observed in usual experiments. Indeed, such a high temperature should lead to detachment of atoms from the cluster. As a consequence, the expected first passage time to target should decrease with temperature in the usual experimental temperature range. In addition, using the typical order of magnitude of $F_0 l/J \approx 10^{-4}$ for electromigration of atomic islands, we can see from Fig. 2.17a that this value would be too small to allow for an appreciable control of few-atoms clusters. However, since the gain in the time to reach the target grows quickly with the cluster size N, control might be possible to achieve for clusters that are larger—but not much larger— than the biggest cluster ($N = 12$) that we have investigated.

Moreover, lower energy barriers can exist in more complicated metallic systems. For example, in Ag/NiAl(110) the edge diffusion barriers have been estimated with STM observations and ab-initio calculations to be 0.29 eV in one direction and 0.13 eV in the other (perpendicular) one [119, 120], however we cannot say whether our results would be applicable to such a system since our model has a higher 4-fold symmetry.

In the case of colloids, a theoretical investigation [76] compared to experimental observations of the drop drying of ligand-coated Au nanoparticles in solution [36] suggests that some amount of edge diffusion can be observed. However, most experiments on colloids clusters report mass transport dominated by attachment-detachment of particles at the edges, such in the case of silica or PS microparticles on single crystalline colloidal substrates [71]. Our analysis could also be extended to this regime when considering the detachment-diffusion-reattachment events within vacancy clusters, which preserve the total volume [16, 121, 122]. In contrast to the case of atomic clusters, colloid clusters should allow for the observation of control via an external force. As we have seen in the beginning of this chapter, for colloidal systems the ratio $F_0 l/J$ can be tuned up to values of order one, which allows for control.

The method that we have presented is versatile and can be easily be adapted to fit modified objectives. One obvious extension is to vary the temperature in time. We have not investigated this direction, but a simple example to show how this could be used is the case in which the experimental goal is to keep the system as much as possible in the target state, once it is reached. In this case, a simple strategy is to quench the system with a low temperature once the target is observed, in order to slow down the dynamics as much as possible. Another way to achieve this goal without varying the temperature is to set the force in the target state so as to maximize the residence time in the target.

In our model, we decided to forbid cluster breaking by particle detachment. We made this choice mainly for technical reasons: in this way, the number N of particles in the cluster is conserved and the possible configurations can be straightforwardly calculated as the fixed polyominoes with N cells. However, even when the experimental conditions are such that edge

diffusion is the dominant process of mass transport, detachment of particles from the edge can always occur. Furthermore, we have seen that in a colloidal system there are ways to increase the magnitude of the external force, and this could lead to the splitting instability discussed in the introduction of this chapter, since the critical radius of the instability decreases when increasing the force (see Eq. (2.1)). One possible direction is to use the force to avoid cluster breaking. This could be achieved by adding the atomic moves that would lead to breaking in the list of possible moves and to assign them an arbitrary negative reward. In this way, the value iteration algorithm converges to an optimal policy that reaches the target in minimum time, while at the same time trying to avoid breaking. The balance between the two objectives can be controlled by the magnitude of the breaking reward.

In order to apply our approach to the quantitative description of specific physical systems, it would be important to refine the physical model to account, for example, for other lattices and for accurate values of hopping rates. Moreover, we have taken into account only particle-particle lateral bonds, discarding particle-substrate interactions. In the simple case where the bond between the particles and the substrate is assumed to be equal to J, we just have to multiply all rates of Eq. (2.4) by $\exp(-J/k_BT)$ to consider the presence of the extra bond, which results in a rescaling of all the timescales by $\exp(J/k_BT)$. In Appendix A.11 we show that this procedure leads to results that are qualitatively identical to those reported above, but with a decrease of the depth of the minimum in the mean return time to target.

CHAPTER 3

Model-free control of fluctuating 2d clusters

The work we present in this chapter is closely related to that of Chapter 2 on model-based control. Using the same lattice model introduced in Section 2.1, we explore the control of 2d fluctuating clusters using model-free techniques. We simulate the dynamics of the environment with a kinetic Monte Carlo method, to which we couple the two model-free tabular control algorithms introduced in Section 1.4.3.

The results of this chapter complement those of the previous one and can be seen as a first step towards an experimental realisation of our methodology, where the control agent would only have access to observations of the physical environment and not to the full dynamical model.

As we mentioned in Section 1.4.3, a key issue in model-free control methods is that of exploring the space of states and actions of the environment. The usual way to approach this problem is to consider ε-*greedy* policies such as the one defined in Eq. (1.18), which select with a probability ε a random action instead of the optimal one. Unlike in the previous chapter where we used deterministic policies, we therefore use here a stochastic ε-*greedy* policy $\pi(\boldsymbol{a} \,|\, s)$. For consistency of notation, we still use the function $\boldsymbol{\phi}(s)$, which sets the force provided by action $\boldsymbol{a}$ on state s, but here this function is not sufficient to completely characterise the policy. Hence, within this chapter, we will refer to π as the "policy".

3.1 Kinetic Monte Carlo implementation of the dynamics

In computational physics, the term Monte Carlo (MC) refers to a broad class of algorithms that rely on the use of random numbers to obtain numerical results[1]. They first emerged in the 1950s as computers came into use, and they offer a powerful way to evaluate equilibrium proprieties of physical systems. In the 1960s, a different type of MC algorithm began to be developed to dynamically evolve systems from one state to another. The terminology for this approach settled in as kinetic Monte Carlo (KMC). The popularity and range of applications of this method has continued to grow and KMC is now a common tool in material science and statistical physics. In the rest of this section, we describe this well-known method.

[1]The name alludes to the famous casino in Monte Carlo, where random numbers are extracted by the croupiers (for exclusively non-scientific purposes).

A particle cluster undergoing thermal fluctuations is characterised by occasional transitions from one state to another, with long periods of relative inactivity between these transitions. This is sometimes called an *infrequent-event* system. Each state s corresponds to a single minimum in the energy landscape of the system, and the long time between transitions arises because the system must overcome an energy barrier that is larger than the thermal energy k_BT to get from one minimum to another, as we already discussed in the previous chapter when presenting our lattice model, and illustrated in Fig. 2.3b.

As the particles of the cluster vibrate erratically around their equilibrium position, assuming they have not escaped over an energy barrier yet, the system is considered to remain state s. Adjacent to state s there are other potential minima, corresponding to the states $\mathcal{B}_s$ that are reachable from s, each separated from s by an energy barrier $E_{\boldsymbol{\phi}}(s, s')$, defined in Eq. (2.5). When a particle overcomes a potential barrier, the system has been taken to a new state s'. The key property of an infrequent-event system is that, when it is in a particular potential minimum, it stays there for a long time (relative to the typical time of one vibrational period), and hence it "forgets" how it got there. This allows us to define escape rates $\gamma_{\boldsymbol{\phi}}(s, s')$ to adjacent states s' that do not depend on the previous states visited by the system. This characteristic, that the transition rates for exiting state s are independent of the history prior to entering state s, is the defining property of a Markovian system.

Let us define $P^{\text{surv}}_{\boldsymbol{\phi}}(s, t)$ as the probability that the system has not yet escaped from state s under a force $\boldsymbol{\phi}(s)$—i.e. that state s *survives*—up to a time t. The escape of the system from a state is a process with exponential decay statistics (analogous, for example, to nuclear decay), and therefore obeys a Poisson probability distribution. This can be easily seen if we consider that $P^{\text{surv}}_{\boldsymbol{\phi}}(s, t)$ is equal to the probability that no particle movement has occurred up to time t, thus, for a small time increment dt

$$P^{\text{surv}}_{\boldsymbol{\phi}}(s, t + dt) = P^{\text{surv}}_{\boldsymbol{\phi}}(s, t)(1 - \Gamma^{\text{tot}}_{\boldsymbol{\phi}}(s)\, dt) \quad \xrightarrow[dt\to 0]{} \quad P^{\text{surv}}_{\boldsymbol{\phi}}(s, t) = \exp(-\Gamma^{\text{tot}}_{\boldsymbol{\phi}}(s)\, t)\,, \tag{3.1}$$

where $\Gamma^{\text{tot}}_{\boldsymbol{\phi}}(s) = \sum_{s'\in\mathcal{B}_s} \gamma_{\boldsymbol{\phi}}(s, s')$ is the total escape rate from the state.

We are interested in the probability distribution function $P^{\text{first}}_{\boldsymbol{\phi}}(s, t)$ for the time of *first escape* from state s, which can be obtained by considering that the integral of $P^{\text{first}}_{\boldsymbol{\phi}}(s, t)$ to some time t' gives the probability that the system has escaped by time t', which must equate to $1 - P^{\text{surv}}_{\boldsymbol{\phi}}(s, t')$. Hence

$$\int_0^{t'} P^{\text{first}}_{\boldsymbol{\phi}}(s, t)dt = 1-P^{\text{surv}}_{\boldsymbol{\phi}}(s, t') = 1-\exp(-\Gamma^{\text{tot}}_{\boldsymbol{\phi}}(s)t') \quad \Rightarrow \quad P^{\text{first}}_{\boldsymbol{\phi}}(s, t) = \Gamma^{\text{tot}}_{\boldsymbol{\phi}}(s)\exp(-\Gamma^{\text{tot}}_{\boldsymbol{\phi}}(s)t)\,. \tag{3.2}$$

The average time that the system spends in state s before escaping to any other state corresponds to the expected residence time that we already introduced in Section 2.1, and it is just the first moment of this distribution:

$$t_{\boldsymbol{\phi}}(s) = \int_0^\infty t\, P^{\text{first}}_{\boldsymbol{\phi}}(s, t)\, dt = \frac{1}{\Gamma^{\text{tot}}_{\boldsymbol{\phi}}(s)}\,. \tag{3.3}$$

When a transition from state s occurs, we have to determine which state s' is reached. The probability to perform a transition to state s' is $p_{\boldsymbol{\phi}}(s, s') = \gamma_{\boldsymbol{\phi}}(ss')t_{\boldsymbol{\phi}}(s)$, as discussed in Section 2.1. The procedure to choose s' is shown schematically in Fig. 3.1. We imagine that for each of the $|\mathcal{B}_s|$ moves[2] we have an object with a length equal to the rate $\gamma_{\boldsymbol{\phi}}(s, s')$ for the pathway associated to that move. We put these objects end to end, giving a total length $\Gamma^{\text{tot}}_{\boldsymbol{\phi}}(s)$.

[2]Recall that, for $N \geq 2$, each atomic move corresponds to a transition from state s to a different state s', so there are as many possible moves as there are neighboring states $\mathcal{B}_s$.

We then choose a single random position along the length of this stack of objects by extracting a random number between 0 and $\Gamma_{\phi}^{\mathrm{tot}}(s)$. This random position will "point" to one of the objects, and this is the energetic pathway that we choose for the system to follow. This procedure gives a probability of choosing a particular particle move that is proportional to the rate of the move.

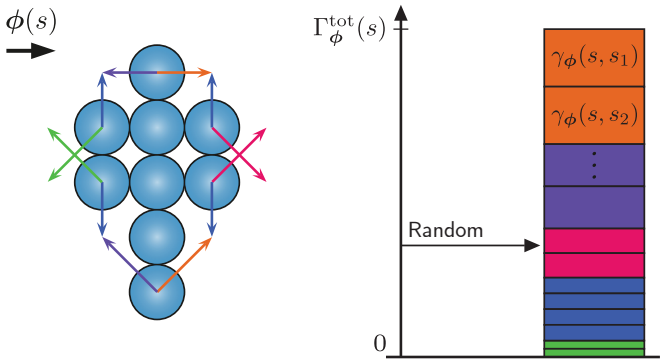


Figure 3.1: Illustration of the procedure for picking a move to advance the system to the next state in the standard KMC algorithm. To each escape rate in a given state s is associated an object (a box for this illustration) with a length proportional to the rate. These objects are "stacked" in a list that adds up to $\Gamma_{\phi}^{\mathrm{tot}}(s)$. Then, the extraction of a random number between 0 and $\Gamma_{\phi}^{\mathrm{tot}}(s)$ is used to select a move in the list. In this example, the highest rates correspond to atoms with 1 bond moving in the direction of the force (in orange), while the lowest ones correspond to atoms with 2 bonds moving opposite to the force (in green).

To advance the simulation time, we need to draw a time of first escape $t_{\phi}^{\mathrm{draw}}(s)$ from the distribution $P_{\phi}^{\mathrm{first}}(s,t)$, i.e. generate an exponentially-distributed random number. This can be done in the following way. We consider a random number u, with a uniform distribution $p(u) = 1$, and $0 < u \leq 1$. Then, we impose the equality of the probability densities $P_{\phi}^{\mathrm{first}}(s,t)\,dt = -p(u)\,du$, leading to the following variable change:

$$\frac{du}{dt} = -P_{\phi}^{\mathrm{first}}(s,t) \quad \Rightarrow \quad u = \exp(-\Gamma_{\phi}^{\mathrm{tot}}(s)\,t)\,. \tag{3.4}$$

Hence, in the KMC algorithm, we first extract a uniformly-distributed random number u between 0 and 1, and then form the time advancement

$$t_{\phi}^{\mathrm{draw}}(s) = -\frac{\log(u)}{\Gamma_{\phi}^{\mathrm{tot}}(s)}\,. \tag{3.5}$$

A time drawn in this way is an appropriate realization for the time of first escape for a process with total escape rate $\Gamma_{\phi}^{\mathrm{tot}}(s)$. Note that the time advance is independent from the event that is chosen. The time to escape depends only on the total escape rate.

Once the move is selected and performed, the system is in the new state. Then, the list of possible moves and the associated rates are updated, and the procedure is repeated. This KMC scheme is often referred to as the BKL algorithm (or the *n-fold* algorithm), due to the 1975 paper by Bortz, Kalos and Lebowitz [123].

As a technical note, it should be added that the above scheme is valid whenever the transition rates between states are constant in time. If, however, one is dealing with time-varying rates, then more complicated time-dependent algorithms should be considered [124]. In our model, the

external force varies in the course of time. However, the external field is assumed to be switched instantaneously just after the transition from state s to state s'. As a consequence, the rates are constant when the system is in a given state, and the standard KMC algorithm can be used. In any experiment, the observation of the transition would take a finite time t_{obs}, the processing of this information and the decision to switch the external field takes a finite time t_{proc}, and the switching of the external force also takes a finite time t_{switch}. Our assumptions are that these three times are much smaller than the residence times $t_{\phi}(s)$.

3.2 Equilibrium dynamics

With KMC, it is straightforward to simulate the equilibrium dynamics of a fluctuating cluster. We use the same rules described in Section 2.1, that is, we forbid atom detachment and cluster breaking, and freeze atoms with 4 nearest neighbours. We have seen in Section 2.3.2 that, at equilibrium, the probability $P_0(s)$ of being in a state s is given by the Boltzmann distribution of Eq. (2.35), with the energy of the island configuration s given by the simple formula $H(s) = L_s J/2$, where L_s is the length of the perimeter of the island. Hence, we expect that, in the absence of external forces, the fraction of time that the system spends on a given state s is given by $P_0(s)$.

We have implemented the KMC algorithm described above in Python, and we have performed several simulations of the dynamics of a 7-atom cluster at different temperatures. Each simulation was done with 10^6 steps. During the simulations, we kept track of the time that the system spent on five selected states by summing the stochastic time advancements $t_{\phi}^{\text{draw}}(s)$ on these states. The ratio between these times $t_{\phi}^{\text{draw}}(s)$ over the total simulation time (i.e. the total sum of $t_{\phi}^{\text{draw}}(s)$ for all states) is shown in Fig. 3.2. In this figure, we also show the theoretical equilibrium probabilities $P_0(s)$ for states with $L_s = 12$, $L_s = 14$ and $L_s = 16$.

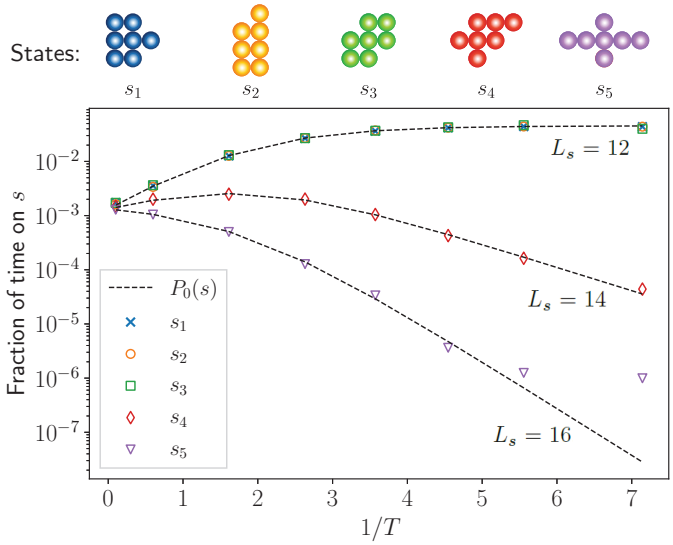


Figure 3.2: Fraction of time spent on a state s over the total time, for 8 independent simulations with 10^6 steps, for 5 different 7-atom states. These fractions correspond to the probabilities $P_0(s)$.

The three states s_1, s_2 and s_3 all have the same perimeter $L_s = 12$, and hence the same occupation probability $P_0(s)$. The states s_4 and s_5, instead, have respectively $L_s = 14$ and $L_s = 16$, which correspond to higher energies, and thus to lower probabilities. Overall, we

can see that, a part from some deviations at low temperature for state s_5, that are likely due to low statistics, the KMC simulations are correctly reproducing the equilibrium occupation probabilities.

It should be noted that freezing the motion of atoms with 4 nearest neighbours actually violates the equilibrium condition of detailed balance given by Eq. (2.40), because in this way there are some processes that have no reverse process, like the move shown in Fig. 3.3, which brings a particle with 2 nearest neighbours (unfrozen) to a position with 4 nearest neighbours (frozen). However, we can see from Fig. 3.2 that this does not affect significantly the equilibrium dynamics of the cluster, which still obey the expected probability distributions.

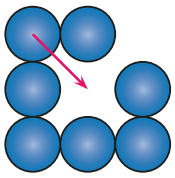


Figure 3.3: A process that violates detailed balance. When freezing the motion of atoms with 4 nearest neighbours, this move has no reverse move.

3.3 Performance of model-free control

Following the framework described in Section 1.4.1, we have implemented, in Python, two model-free tabular methods for the control of the dynamics of a fluctuating particle cluster in the presence of an external field. The control agent and the environment—i.e. the KMC simulation of the cluster—are interfaced according to the scheme represented in Fig. 3.4. At every step t of the simulation, the agent "observes" the state S_t of the cluster (i.e. its shape) and receives a representation of such shape in a form of a numerical identifier, as explained in Appendix A.6. It is useful to uniquely encode each state of the environment into a number, so that the states can be easily stored and managed by the computer in a table.

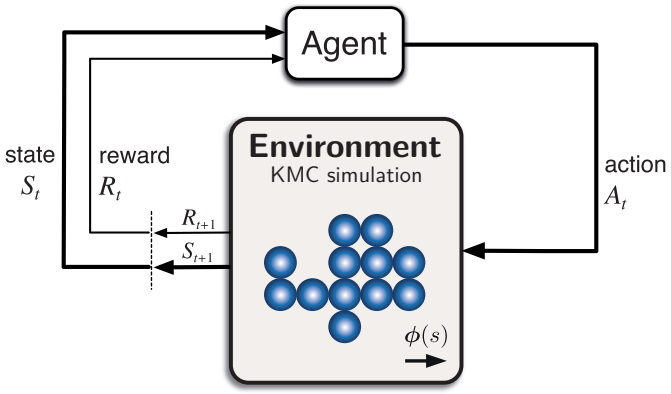


Figure 3.4: The agent-environment interface for the control of a fluctuating particle cluster in the presence of an external field.

Using the same approach as in the previous chapter for model-based control, we define the action set as the three possible values $\mathcal{A} = \{-F_0\hat{\boldsymbol{x}}, \mathbf{0}, F_0\hat{\boldsymbol{x}}\}$ of the external force. Based on the

observation of state $S_t = s$, the agent chooses an action $A_t = \boldsymbol{a} \in \mathcal{A}$, which corresponds to setting the external force $\boldsymbol{a} = \boldsymbol{\phi}(s)$ to one of the three values.

In contrast to what we did in the previous chapter, here we choose the reward to be a stochastic quantity. We define the reward R_{t+1} associated to state $S_t = s$ and action $A_t = \boldsymbol{a} = \boldsymbol{\phi}(s)$ as the opposite of the stochastic realisation of the time of first escape from state s given by Eq. (3.5):

$$R_{t+1} \leftarrow -t_{\boldsymbol{\phi}}^{\mathrm{draw}}(S_t = s), \tag{3.6}$$

where the symbol $\leftarrow$ correspond to an assignment statement. The minus sign is due again to the fact the our goal is to minimize the time to reach a target configuration, while the control agent seeks to maximize the rewards.

We use an *ε-greedy* stochastic policy $\pi(\boldsymbol{a} \,|\, s)$, as defined in Eq. (1.18), with the parameter ε, determining its exploratory character, which decays linearly during the learning process.

3.3.1 Convergence scheme

Our goal is to learn optimal policies to minimize the time to reach a target cluster configuration $\bar{s}$. As in the previous chapter, we use the expected return time to target $\tau_\pi^{\mathrm{r}}(\bar{s})$ to evaluate the performance of a policy π. However, since we are using model-free control methods here, we do not assume any knowledge of the master equation of the system, and only rely on observations of the environment in order to estimate $\tau_\pi^{\mathrm{r}}(\bar{s})$. Specifically, we run a KMC simulation starting from the target state $\bar{s}$ and with the force that is changed during the dynamics according to the policy to be evaluated π, and obtain a number of samples n_{sam} of the return time to target. The estimate of $\tau_\pi^{\mathrm{r}}(\bar{s})$ is obtained simply by averaging over the samples.

We also assume no knowledge of the optimal return time to target $\tau_*^{\mathrm{r}}(\bar{s})$ computed by model-based methods. Thus, we cannot use this quantity to decide when to stop the learning task, but must resort to a convergence condition based on the evolution of the algorithm's performance with respect to the zero-force return time $\tau_0^{\mathrm{r}}(\bar{s})$. The convergence condition we used is given in pseudocode in Fig. 3.5.

The general idea of this convergence scheme is based on controlling the "improvement" of the policy $1 - \tau_\pi^{\mathrm{r}}(\bar{s})/\tau_0^{\mathrm{r}}(\bar{s})$, which is similar to the concept of gain we used in Chapter 2, except that this factor is, in principle, bounded between 0 and 1 (because we expect a learned policy to produce a return time to target that is smaller or, at worst, similar to the zero-force return time $\tau_\pi^{\mathrm{r}}(\bar{s}) \lesssim \tau_0^{\mathrm{r}}(\bar{s})$). First, we obtain an estimate of the zero-force return time to target $\tau_0^{\mathrm{r}}(\bar{s})$, which will be used as a reference. Next, we perform an initial learning run on n_{epi} episodes and estimate the return time of the learned policy $\tau_\pi^{\mathrm{r}}(\bar{s})$. We then calculate the improvement factor and evaluate whether it is better than the best improvement factor we had during previous runs (this factor is initially set to 0), within a small tolerance I_{tol}. If it is, we update the best improvement factor and reset a counter which keeps track of how many runs we did without improvement, otherwise, we increase this non-improvement counter.

The cycle is then repeated but doubling the number of learning episodes, then tripling it, and so on. Whenever the counter of runs without improvement reaches a value n_{runs}, we consider that the learning algorithm would not benefit from a further increase in the number of episodes, and stop the cycle.

Each time the selected learning algorithm is called, the learning run is executed with a value of ε of the *ε-greedy* policy that starts from an initial value ε_i and then is linearly decreased over the course of learning, reaching a final value ε_f at the last episode of the run, as illustrated schematically in Fig. 3.6. This process is similar to the concept of *simulated annealing* [125], which is often used in molecular dynamics or equilibrium Monte Carlo simulations to find the ground state of a system. In simulated annealing, one or more thermal cycles formed by a temperature

Figure 3.5: Pseudocode of the general convergence scheme for the learning task

Parameters:
Number of samples n_{sam} to evaluate the return time to target of a policy
Number of runs n_{runs} that must pass without improvement before learning is stopped
Number of episodes of the first learning run n_{epi}
Small number I_{tol} representing the tolerance on the improvement check

Run KMC simulation to estimate $\tau_0^{\text{r}}(\bar{s})$ of the zero-force policy with n_{sam}
$k \leftarrow 1$
$Counter \leftarrow 0$
$Impr_* \leftarrow 0$
While $Counter < n_{\text{runs}}$ **:**
  Run learning algorithm on $n_{\text{epi}} \times k$ episodes, which outputs a policy π
  Run KMC simulation to estimate $\tau_\pi^{\text{r}}(\bar{s})$ of the learned policy π with n_{sam}
  $Impr \leftarrow 1 - \tau_\pi^{\text{r}}(\bar{s})/\tau_0^{\text{r}}(\bar{s})$
  If $Impr > Impr_* \times (1 + I_{\text{tol}})$ **:**
    $Impr_* \leftarrow Impr$
    $Counter \leftarrow 0$
  Else:
    $Counter \leftarrow Counter + 1$
  $k \leftarrow k + 1$

Output: $\pi \approx \pi_*$

rise followed by slow cooling allow the system to explore the energy landscape, progressively relaxing towards low-energy states and avoiding getting stuck in local energy minima. In our case, ε is the parameter that characterizes the degree of exploration of the policy, and thus has, in a sense, the same role as the temperature in simulated annealing.

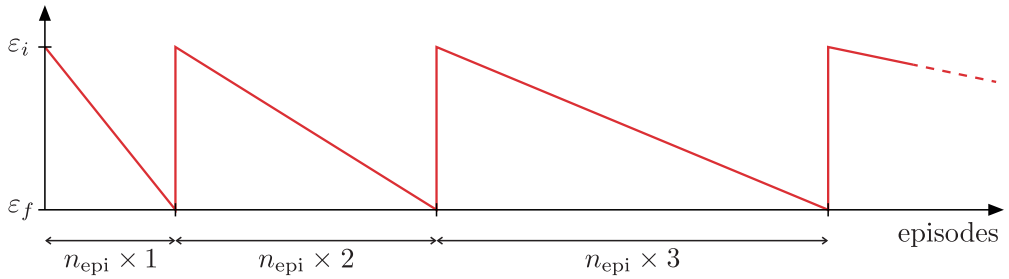


Figure 3.6: Schematic representation of the evolution of ε over the course of the whole learning task. At the beginning of each learning run, ε is reset to ε_i, and then decayed linearly, reaching ε_f at the end of the run. At the end of each run, the performance of the policy is evaluated.

We implemented the two tabular control algorithms described in Section 1.4.3 and given in pseudocode in Appendix A.2: Monte Carlo control and Q-learning. For both algorithms, we set the discount factor $\beta = 1$ and used the following parameters for the convergence scheme: $n_{\text{sam}} = 2000$, $n_{\text{runs}} = 3$, $n_{\text{epi}} = 1000$, $I_{\text{tol}} = 0.02$. For Monte Carlo, we used $\varepsilon_i = 0.9$ and $\varepsilon_f = 0$, while for Q-learning we used $\varepsilon_i = 1$, $\varepsilon_f = 0.1$ and set the learning rate $\alpha = 0.05$. We have also set the maximum number of KMC moves in a single episode to 1000, after which the episode is terminated even if the target has not been reached. These values were obtained empirically.

We now present the results for these two control algorithms compared to the optimal performance obtained with the model-based approach in Chapter 2.

3.3.2 Results for Monte Carlo and Q-learning algorithms

In Figs. 3.7 and 3.8 we show the evaluated return time to target as a function of inverse temperature for some selected targets with $3 \leq N \leq 10$. The return time corresponding to the policies learned with Monte Carlo and Q-learning is plotted together with the optimal return time and the zero-force return time computed with the dynamic programming method of value iteration.

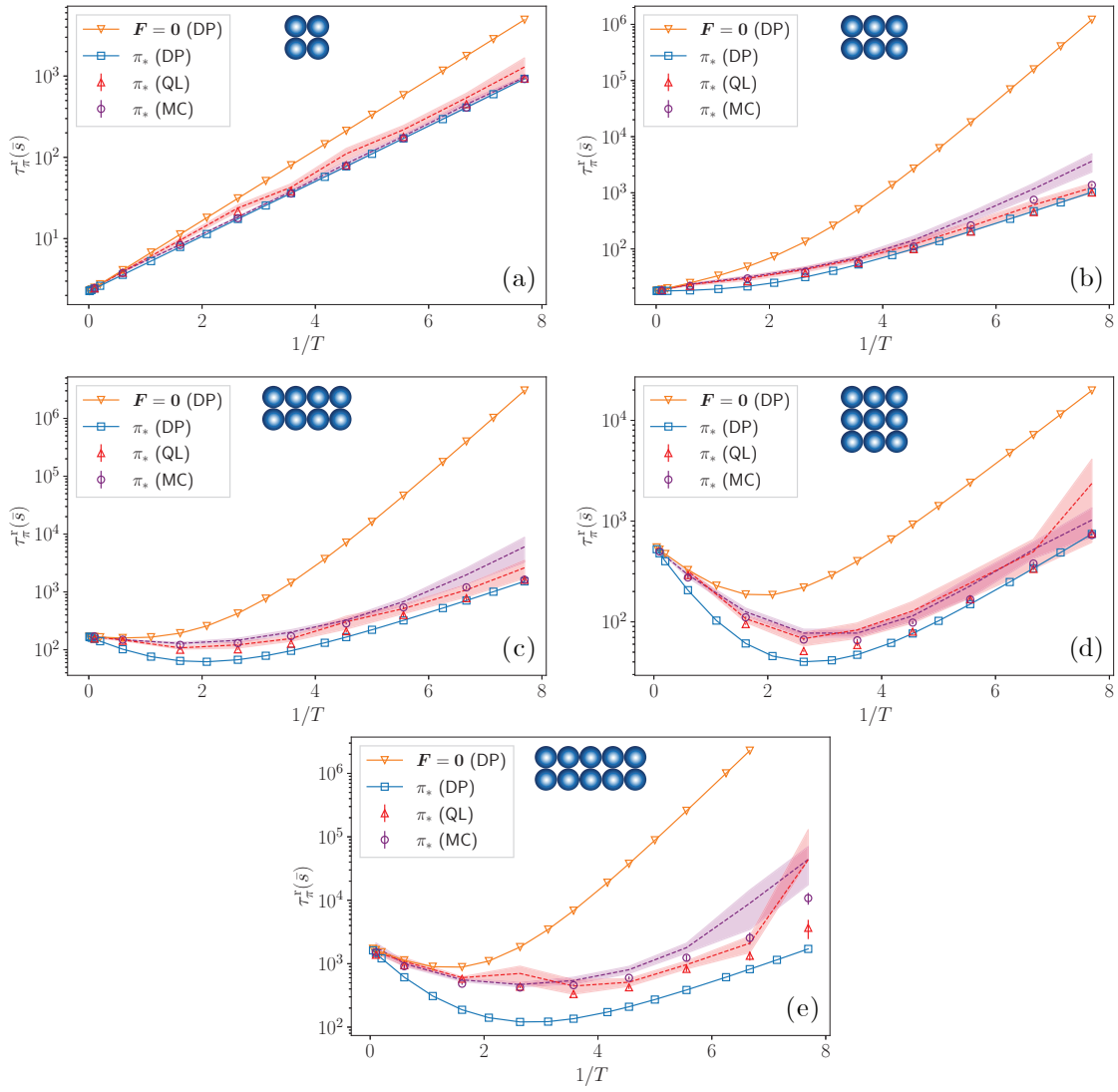


Figure 3.7: Mean return time to target $\tau_\pi^{\mathrm{r}}(\bar{s})$ as a function of $1/T$ for some selected compact targets. The Monte Carlo (MC) and Q-learning (QL) optimal policies are plotted together with the zero-force and optimal policies computed with dynamic programming (DP).

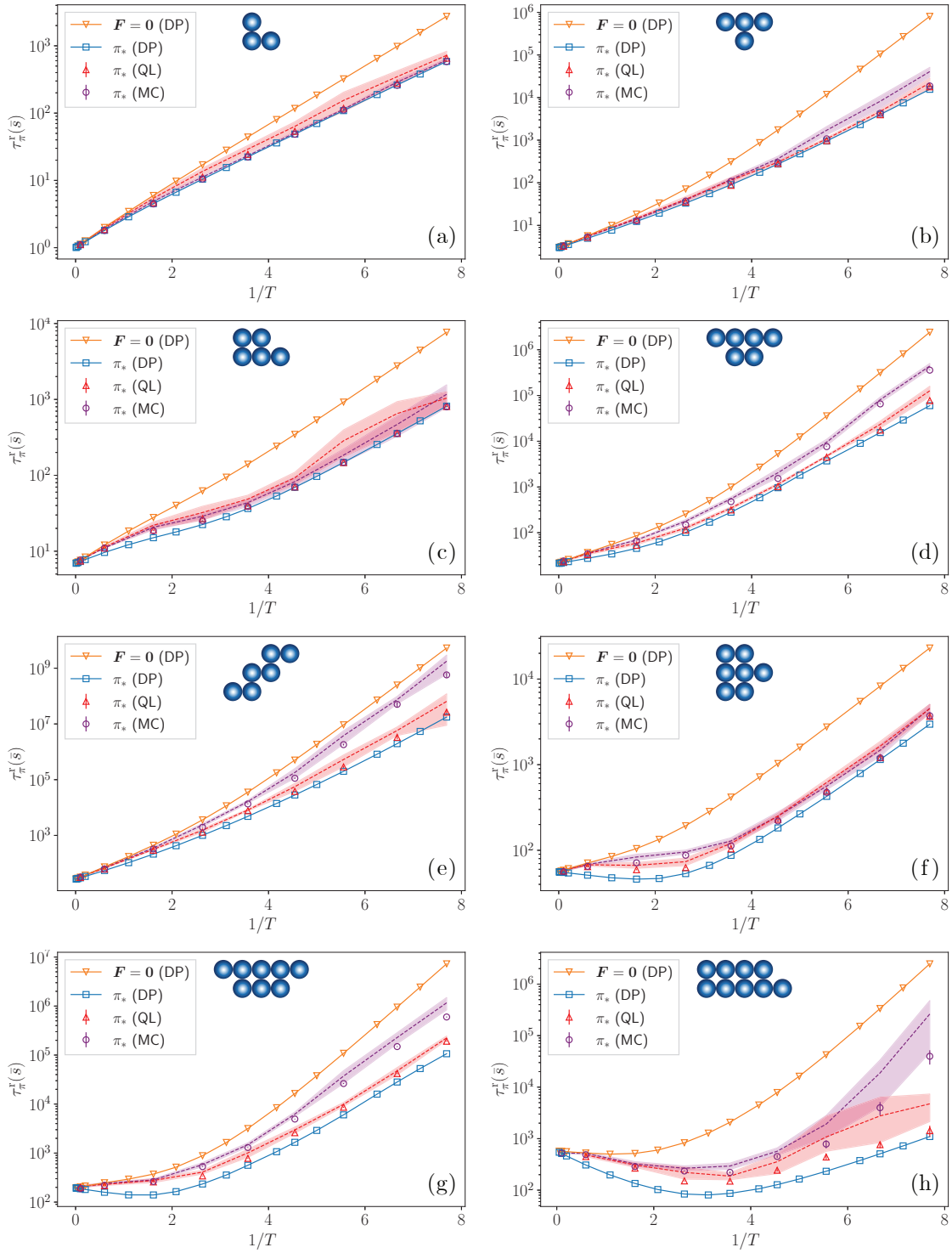


Figure 3.8: Mean return time to target $\tau_\pi^{\mathrm{r}}(\bar{s})$ as a function of $1/T$ for some selected less compact targets. The Monte Carlo (MC) and Q-learning (QL) optimal policies are plotted together with the zero-force and optimal policies computed with dynamic programming (DP).

For each temperature considered, we performed 10 independent learning tasks for each learning method, following the convergence scheme described above. The markers in figure correspond to the return time of the best policy obtained out of 10 (which, for plotting purposes, was re-evaluated after the learning with $n_{\text{sam}} = 10^4$, to have better statistics). This best policy is used as a basis for defining the performance. The dashed lines indicate the mean of the 10 return times obtained with the learning tasks, and the shaded area represents the standard error of the mean (defined as the standard deviation divided by the square root of the sample size, which is equal to 10).

In Fig. 3.7, we grouped the most compact (i.e. low-energy) targets among those selected for this analysis. We can see that, for the 4-particle square in Fig. 3.7a, the performances of the best policies learned with both Monte Carlo (MC) and Q-learning (QL) are essentially the same as the optimal case obtained with the model-based method (DP). For the $N = 6$ and $N = 8$ rectangles in Figs. 3.7b and 3.7c, some differences between the two methods begin to appear, with Q-learning performing slightly better than Monte Carlo, especially at low temperatures. This behaviour becomes even more pronounced for the $N = 10$ rectangular target in Fig. 3.7e. Interestingly, in the case of the 9-particle square of Fig. 3.7d, Q-learning performs better than Monte Carlo at intermediate to high temperatures

In Fig. 3.8, instead, we show the results for less compact (i.e. high-energy) targets. Again, for small clusters made up of 3, 4 or 5 particles (Figs. 3.8a, 3.8b and 3.8c), the performances of the two methods are similar, and the best learned policies produce a mean return time to target that is very close to the one computed with dynamic programming. For all the other cases, with $N \geq 6$, instead, Q-learning performs better than Monte Carlo.

It is interesting to compare the 6-particle compact target in Fig. 3.7b with the two less compact targets of the same size in Figs. 3.8d and 3.8e. We can notice that, as the shape of the target becomes less compact, the gap in performance between Q-learning and Monte Carlo becomes larger. For the least compact of the three (Fig. 3.8e), the mean return time to target obtained with Monte Carlo is closer to the zero-force return time than to the theoretical optimal return time computed with dynamic programming. The same considerations hold when comparing the 8-particle compact target in Fig. 3.7c and the less compact one in Fig. 3.8g.

For the 7-particle target in Fig. 3.8f, the advantage of Q-learning is again more pronounced towards high temperatures (like the $N = 9$ square in Fig. 3.7d), with both methods performing very well at low temperatures. For the $N = 8$ and $N = 9$ targets in Figs. 3.8g and 3.8h, Q-learning outperforms Monte Carlo over almost the entire temperature range.

In general, we can conclude that the two model-free control methods are able to learn to drive a fluctuating cluster of particles towards a desired target shape using a macroscopic field, at least for the targets that we have considered. At high temperatures, both methods appear to have difficulties in learning an optimal policy, probably due to the difficulty in observing the effects of actions when the system is subject to large thermal fluctuations (or, in other words, when the transition rates are too similar). At low temperatures, learning is generally good, especially with Q-learning, but the standard error bars become wider, showing that there is greater dispersion of the performances of the learned policies. This suggests that, in this temperature range, it is more difficult to efficiently explore the space of states, and the learned policies are highly variable. Overall, Q-learning performs better than Monte Carlo, especially for bigger and less compact clusters.

As a technical note, the computational times required to converge to an optimal policy are comparable for the two methods.

3.3.3 Performance as a function of the observation time

In the previous section, we saw that it is possible for the two model-free algorithms to learn optimal policies by observing and interacting with the simulated environment. These policies have a performance that is comparable to the theoretical values obtained using the model-based iterative evaluation method introduced in Chapter 2.

To learn these policies, the control algorithms require observing the dynamics of the simulated cluster for an extended period of time. To show how the performance of the learning algorithms evolves as a function of observation time, we executed several learning runs on fixed numbers of episodes, with no convergence condition. At the end of each run, we tested the learned policy on $n_{\text{samples}} = 10000$ and plot the resulting mean return time to target as a function of the total time of the learning run, obtained as a sum of all the stochastic time increments given by Eq. (3.5) during all the episodes of the run. The results of this analysis are reported in Fig. 3.9, for the 7-particle target of Fig. 3.8f at $T = 0.22$ (left) and $T = 0.62$ (right).

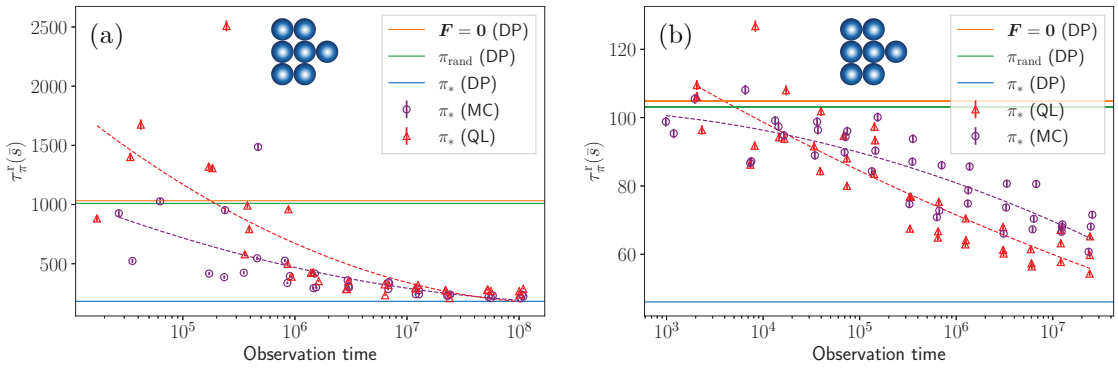


Figure 3.9: Mean return time to target $\tau_\pi^{\text{r}}(\bar{s})$ as a function of the observation time for the 7-particle target of Fig. 3.8f, at (a) $T = 0.22$ and (b) $T = 0.62$. The dashed lines correspond to a parabolic fit of the respective simulation points and are just a guide to the eye to show the general decreasing trend of the evaluated mean return times.

We can see that, for the lower temperature, the mean return time to target converges to a value very close to that obtained with dynamic programming at an observation time of about 10^7 dimensionless units. For the higher temperature, the mean return time to target decreases more slowly with the observation time, but, in this case as well, at values of the order of 10^7 dimensionless units the two algorithms have achieved a performance that is much closer to that of the dynamic programming optimal policy than to that of the zero-force or the random policy. But what does this number correspond to, in physical units?

We have argued in Section 2.5 that colloidal clusters are a candidate system that should allow for control via an external force, since they have a ratio $F_0 l/J$ that can be tuned up to values of order one. For colloidal clusters of ligand-coated Au nanoparticles in solution, in which some amount of edge diffusion can be observed, a theoretical study [76] suggests that the energy barrier E_e for the edge diffusion of a particle along a straight edge is of the order of $0.1 \sim 0.3$ eV, for particles of diameter ranging from 4.9 to 8 nm at room temperature (which roughly corresponds to $1/T = E_e/k_B T_{\text{room}} \simeq 4 \sim 10$ in our simulations). This corresponds to an edge-diffusion hopping rate of the order of $10^5 \sim 10^6$ s^{-1}. Hence, in such a system, we could predict that a learning algorithm would need an observation time of a few tens or hundreds of seconds in order to reach an appreciable performance.

However, we doubt that there are experimental techniques that allow for cluster observation,

processing, and real-time switching of the external field in timescales shorter than milliseconds. The edge-diffusion rates derived from Ref. [76], therefore, correspond to hopping processes that are probably too fast to be observable in a closed-loop control experiment. Since interactions in colloidal clusters are highly tunable, we hypothesize that it should be possible, in principle, to design a system with hopping rates that are of the order of 1 kHz, and thus adapted to a control experiment. If we assume that the realization of such a system is possible, then in this case the control algorithm would need a few hours of observation to learn how to control the cluster.

3.4 Discussion and perspectives

Using kinetic Monte Carlo, we have implemented the stochastic dynamics of a fluctuating particle cluster described by the lattice model introduced in Chapter 2. This has allowed us to define a simulated environment to which we coupled the two model-free tabular control methods described in Chapter 1, in order to test their performance. We found that Monte Carlo and Q-learning algorithms can learn optimal control policies to drive the cluster towards some selected target shapes using a macroscopic field. The two methods perform similarly for small ($N \leq 5$) clusters, while Q-learning has better results for bigger clusters, especially when the target shape is not very compact.

What we have presented here is not intended to be an exhaustive analysis of the performance of these model-free control algorithms on a stochastic system. There is certainly still room for technical improvement in the choice of algorithm parameters and in the design of the convergence scheme. For example, we have noticed that the value of the learning rate α for Q-learning can have a strong impact on the performance of the algorithm. For this application, we found empirically that the value $\alpha = 0.05$ produces satisfactory results, however a systematic analysis could probably allow us to further optimize the performance of Q-learning. It is also possible to decay this parameter during the learning process (as we did for ε) and it has been shown that this approach can improve the performance of Q-learning [126, 127]. We have not explored this direction.

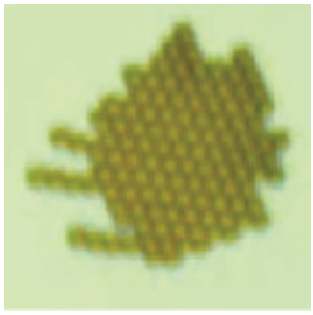
Instead, our analysis should be regarded as a proof of concept, suggesting that our control approach should, in principle, be experimentally applicable. We believe that a promising candidate for the experimental implementation of our method is given by colloidal particle clusters. As we discussed in the previous section, it appears that in a system of ligand-coated Au nanoparticles in solution, the edge diffusion processes are too rapid to be controlled in real time. However, we are confident that experimental physicists with expertise in colloids will be able to fine-tune particle interactions and design experimental potentials that allow for edge diffusion with timescales on the order of kHz, which should allow for real-time observation and control.

An interesting research perspective, would also be to look at diffusion inside monolayer vacancy clusters. This case has been studied extensively in the literature, both experimentally and theoretically [128, 129, 130, 24, 131]. In this scenario, the volume is preserved even in the regime of detachment-diffusion-reattachment of particles at the edges [16, 121, 122]. The diffusion-deattachment-reattachment regime has been observed in many experiments on colloidal clusters, such as silica or PS microparticles on single crystalline colloidal substrates [71]. However, we have not found detailed experimental studies of the dynamics of colloid vacancy clusters in the literature.

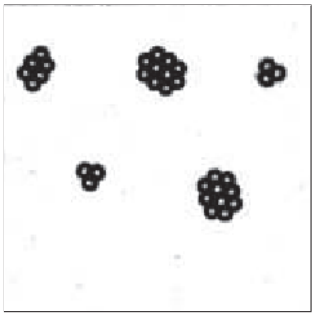
Finally, we want to emphasize on the fact that our approach is not limited to the case of biased edge diffusion, but can readily be extended to any type of evolution rules that preserve the volume of the cluster, such as dislocation-mediated cluster rearrangements in colloids [132] and metal nanoclusters [133, 134], or configurational changes of adsorbed molecules and polymers [135].

Provided that the range of accessible cluster size is limited (to have a finite state space), then our approach could also be extended to systems with various types of particle-particle and particle-field interactions, such as clusters of magnetic beads manipulated by rotating magnetic fields [136, 80], acoustically-bound bubble crystals manipulated by a sound field (Bjerknes forces) [137, 138, 139], light-driven nanoparticles [21], active crystals of self-propelled colloidal particles controlled by a combination of light and magnetic field [140], active Janus particles [141, 142], microbial aggregates [143], or floating granular rafts [144]. See Fig. 3.10 for some examples.

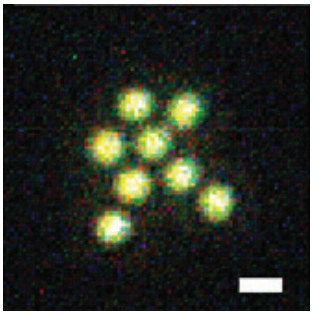
The control of bigger clusters, with a number of states that is too large to be handled by tabular methods, is not out of the question, but rather represents another exciting perspective for our study. In this case, one possibility is to use approximate control methods, as discussed in Section 1.4.



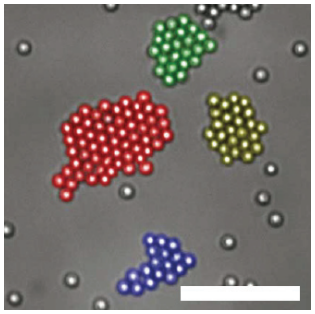
(a) ($\sim 50 \times 50$ µm^2) Image from Ref. [136].



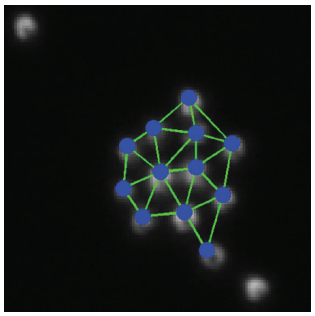
(b) ($\sim$ 500 $\times$ 500 µm^2) Image from Ref. [137].



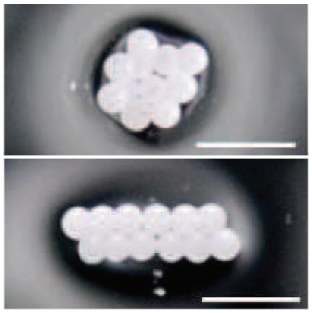
(c) (Scale bar indicates 600 nm) Image from Ref. [21].



(d) (Scale bar indicates 10 µm) Image from Ref. [140].



(e) ($\sim 15 \times 15$ µm^2) Image from Ref. [141].



(f) (Scale bar indicates 1 cm) Image from Ref. [144].

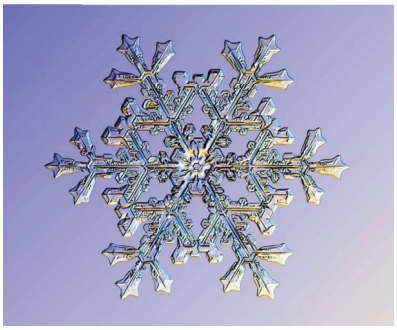
Figure 3.10: Monolayer particle clusters in different physical systems. These systems constitute examples to which our model could be extended. (a) Cluster of magnetic beads, (b) acoustically-bound bubble crystals, (c) cluster of light-driven nanoparticles, (d) colloidal active crystals, (e) aggregate of Janus particles, with its geometric definition superimposed, (f) two granular rafts floating on an oil-water interface.

CHAPTER 4

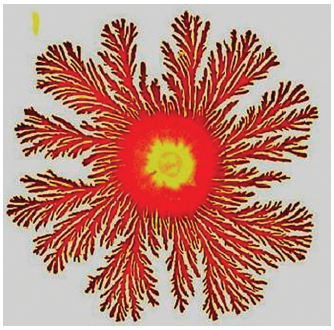
Approximate control and classification of morphological instabilities

In this last chapter, we focus on the control of shapes in continuous deterministic systems. For this aim, we rely on model-free learning methods coupled with approximation techniques. We wish to achieve control on morphologies that can emerge from the dynamics of evolving interfaces that separate domains—or phases—in spatially extended physical systems.

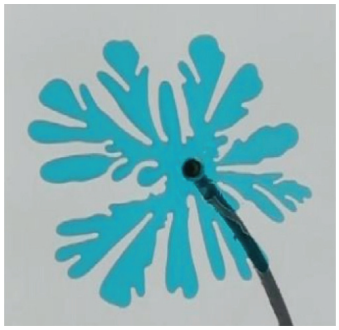
A rich variety of morphologies can arise from the non-equilibrium dynamics of continuous physical systems [145, 146], such as hydrodynamic instabilities [147], growth processes [148, 42, 102], and phase-separation phenomena [149, 150]. Some examples of complex interface morphologies that appear in such systems are shown in Figs. 4.1 and 4.2.



(a) A snowflake. Image from Ref. [43].



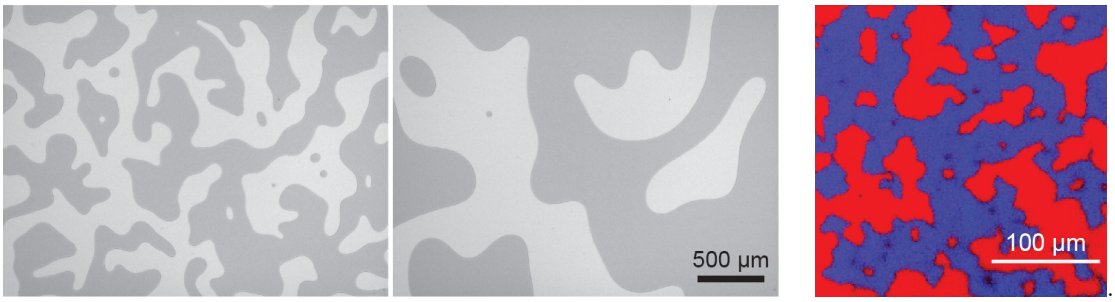
(b) A branching pattern formed by a colony of the *P. dendritiformis* bacteria. Image from Ref. [151].



(c) Viscous fingering in a Hele-Shaw cell. Image from Ref. [152].

Figure 4.1: Complex shapes appearing in three different growth processes.

In the same spirit in which we studied the control of small discrete stochastic systems in Chapters 2 and 3, we propose here a control strategy for continuous systems based on the tuning of a global parameter (that is time-dependent ant space-independent) rather than on localized manipulation. Again we use a closed-loop control approach, where changes to the control



(a) Ordering of twisted nematic liquid crystals (TNLC) during a relaxation towards equilibrium. Different shades of grey correspond to twists of opposite handedness in the nematic director field. Time increases from left to right, final time: 170 seconds. Image from Ref. [153].

(b) Killing-mediated phase separation in a dense *V. cholerae* bacterial population. The colors corresponds to two different strains. Image from Ref. [154].

Figure 4.2: Complex shapes forming during phase separation.

parameter are made by a model-free agent depending on the observation of the state of the system.

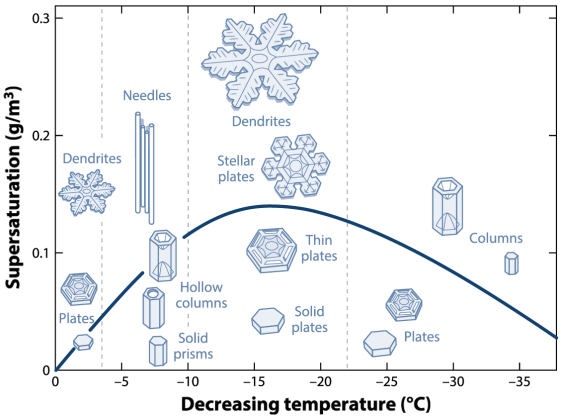
However, there are major differences with respect to the cluster control problem. The first is that in this case we have to deal with a continuous set of states, and this forces us to use approximation techniques. We have considered two approximation methods. One method is based on deep neural networks (DNN), which allowed us to obtain promising results on the control of an interface in a one-dimensional system. Another approach is the curvature scale space (CSS) method, which relies on a reduction of the dimensionality of the description of the interface, with which we were able to compare two-dimensional shapes.

The second difference is that we are dealing with deterministic dynamics. A well known method to control deterministic systems is optimal control. In this model-based method, one finds an open-loop policy (i.e. an evolution of the control parameter as a function of time that does not rely on the observation of the state of the system during the dynamics) that takes the system from a specific initial condition to a specific target state [155]. Here instead, we look for a closed-loop model-free strategy that is able to take the system from an arbitrary initial condition to given target shape, and that requires observation of the system at all times.

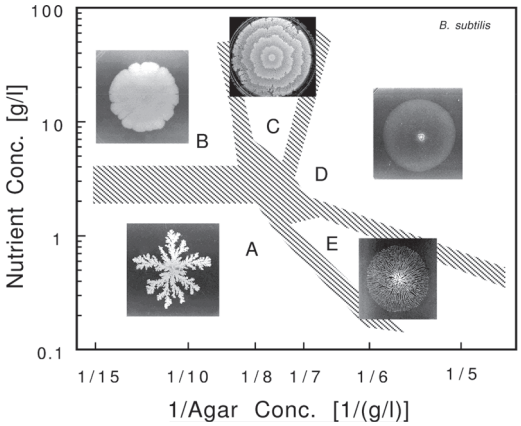
Many studies have already reported that different environmental conditions in evolving nonequilibrium physical systems can lead to a wide diversity of morphologies. This fact has long been known for snow crystals, which can vary from columns to thin slabs, sometimes branched, hollow, or faceted, depending on different growth conditions—in particular temperature and water vapour supersaturation. This is shown in the diagram of Fig. 4.3a, called the Nakaya diagram after the Japanese physicist Ukichiro Nakaya, who pioneered the study of snowflake morphologies in the 1950s.

Similar morphological diversity also exist for other dynamical systems, for example in the growth of bacterial colonies [156, 157]. In this case, the properties of the patterns depend on two main factors: the concentration of nutrients, which influences the growth rate of the colony, and the concentration of *agar*, which determines the hardness of the substrate, and therefore, the mobility of the bacteria (see Fig. 4.3b).

In the following sections, we will introduce diffuse interface models, which are the mathematical framework we used to model this kind of extended systems in which there is an interface that evolves over time, and then we will discuss more specifically the two dynamical models we



(a) Diagram describing qualitatively the growth of snow crystals as a function of temperature and water vapor supersaturation. Image from Ref. [42].



(b) Morphological diagram of *B. subtilis* colony patterns obtained when varying hardness of agar surface and nutrient concentration. Image from Ref. [156].

Figure 4.3: Morphological diagrams showing the different shapes that can arise by varying ambient conditions during (a) ice growth and (b) growth of a bacterial colony.

considered: the Allen-Cahn equation, typically used to describe phase transitions or reaction-diffusion processes (see Fig. 4.2) and model C, also known as the *phase-field* model, a very versatile tool suitable for describing diverse growth phenomena, such as solidification [158, 159], growth of atomic steps [160], or viscous fingering [161] (see Fig. 4.1). Model C is also formally similar to reaction-diffusion models [162] that are ubiquitous models for patter-forming systems that can describe a wide variety of systems, from nonlinear waves to the growth of bacterial colonies [163]. After that, we will present the methods and results concerning the control and classification of nonequilibrium morphologies that emerge from the dynamics of these models.

4.1 Diffuse-interface models

The dynamics of an interface is traditionally modelled by using the so-called *sharp-interface* description, which provides an explicit law for its local normal velocity v_n. A typical evolution law is [164]

$$v_n = -\kappa\,, \tag{4.1}$$

where κ is the mean curvature of the interface. Equation (4.1) is known as *mean curvature flow*, or *motion by curvature*.

This equation describes for example the evolution of a physical interface between a crystalline phase and another phase. We can define the equilibrium conditions as the physical conditions under which a flat interface does not grow or recede. In equilibrium conditions, the free energy—which is proportional to the interface area—decreases. The way the interface area decreases is described by Eq. (4.1). For an interface which is flat on average, small interface protuberances—which have a positive κ, and small concavities—which have a negative κ, both decay leading to a smoothing of the interface. Motion by curvature also induces the shrinking of finite-size crystals which exhibit a positive average curvature via dissolution or melting. Similarly, it induces

the shrinking of voids or cavities inside crystals which exhibit a finite average negative interface curvature via growth.

However, this kind of formulation can have strong limitations. The main drawback is that the numerical simulation of such models turns out to be more difficult than diffuse-interface models. The most challenging aspect is the complex interactions between interfaces that can undergo topological changes during the evolution, such as merging and pinch-off. Such situations are often addressed by applying somewhat arbitrary criteria for describing the dynamics when interface merging or pinch-off occurs, and manually adjusting the topology. It is noteworthy that numerical codes for sharp-interface models are often very lengthy and complex.

A way to avoid some of these problems is to use a different modelling paradigm, called phase-field method. This technique was introduced by Fix [165, 166] and Langer [167] in the 1980s and it deals with the problem of tracking topological changes of the interface by introducing an auxiliary continuous field, the *phase field*, that has the role of an *order parameter*. This field takes two distinct constant values (for instance $+1$ and -1) in the bulk of each phase, smoothly interpolating between both values across a thin boundary layer around the interface, which is then diffuse with a finite width, as depicted in Fig. 4.4. A discrete location of the interface may be defined as the collection of all points where the phase field takes a certain value (e.g. 0).

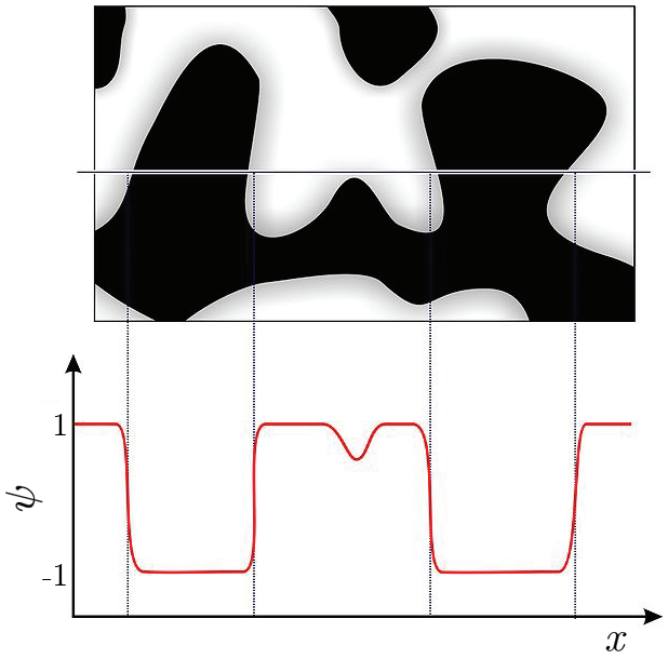


Figure 4.4: A two-phase microstructure and the order parameter ψ profile is shown on a line across the domain. Gradual change of order parameter from one phase to another shows diffuse nature of the interface. Image from Wikimedia Commons.

Historically, diffuse-interface models have first been developed as models for phase transitions. For example, from the perspective of condensed matter physics, the phase field ψ may be seen as an order parameter describing the degree of crystallinity or atomic order in a phase. In these models, the finite width of the interface is the true physical interface width. Later in the 1990s, it was realized that they define a class of models that are very useful for numerical simulations, they were called phase-field models. However, the physical width of the interface is often much smaller

than the width that makes the simulations fast and efficient. The idea then emerged that the width could be chosen arbitrarily as long as it is small enough as compared to the other characteristic lengthscales of the problem, to ensure that the evolution of the interface obeys well-defined sharp-interface equations. This property of being small enough was established mathematically by means of some asymptotic analysis, called the "sharp" or "thin" interface limits, where the interface width is a small parameter. The conditions of validity of these asymptotics can actually be fulfilled even if the interface width is much larger than the physical interface width. Hence, in phase-field models, the interface width is often just a numerical tool to ensure convergence to a given sharp-interface model.

Traditional phase-field models are connected to thermodynamics by a phenomenological free-energy functional $\mathcal{F}$ written in terms of the phase field ψ and possibly other fields (e.g. temperature or material concentration). In its simpler form, $\mathcal{F}$ depends only on ψ and reads [168]

$$\mathcal{F} = \int_\Omega \left\{ \frac{1}{2}(\nabla\psi)^2 + f(\psi) \right\} d\boldsymbol{x}\,, \tag{4.2}$$

where the bounded domain Ω is the volume of the system and the function $f(\psi)$ is the free-energy density, a double well having two global minima for values of ψ corresponding to the two stable phases of the system. Equation (4.2) is often referred to as the Ginzburg-Landau free energy [169].

The next step is to write a partial differential equation for the rate of change of the phase field. This is done through dissipative minimization of this free energy. We will focus here on the case where the order parameter associated to ψ does not evolve constrained to a conservation law, or, in other words, when it is non-conserved. This is the case, for example, of magnetic domain growth, order/disorder transitions, or isothermal solidification of a pure material (in the absence of a density jump). The simplest dissipative dynamical evolution for a non-conserved order parameter is given by [170]

$$\frac{\partial\psi}{\partial t} = -\frac{\delta\mathcal{F}}{\delta\psi} = \nabla^2\psi - \frac{\partial f}{\partial\psi}\,, \tag{4.3}$$

where $\delta\mathcal{F}/\delta\psi$ is the functional derivative.

A phenomenological interpretation of Eqs. (4.2) and (4.3) is the following. The system evolves toward a more stable state by reducing its free energy $\mathcal{F}$. To decrease $\mathcal{F}$, the gradient term in the integral of Eq. (4.2), which is the energy associated to the interface, makes the phase-field profile to spread out, i.e. to widen the transition region. On the other hand, the double-well potential $f(\psi)$ makes the bulk phases stable, and hence has the effect to sharpen the transient region. The diffuse interface maintains a stable width by a balance between these two opposite effects. Once the stable diffuse interface is formed, the two terms start to cooperate to decrease the total volume (or area, in the 2d case) of the diffuse interfacial region, where $\nabla\psi$ is not vanishing. This is corresponding to the motion by curvature contribution in the classical sharp-interface model given by Eq. (4.1).

We will now proceed to formulate two models based on Eq. (4.3). The first one is the Allen-Cahn equation, which is obtained by considering a simple free-energy density that depends only on the phase field ψ. To this model, we will then add a global control parameter that we will use to control pattern formation. The second one is the so-called model C, which instead is derived by plugging in Eq. (4.3) a more arbitrary double-well potential function for the free-energy density, that depends not only on ψ but also on a second field. This field is associated to another varying quantity in the system, for example temperature or concentration, and it has its own evolution equation, which is coupled to Eq. (4.3).

4.2 Allen-Cahn equation

The Allen-Cahn equation was first considered as a model for matter of a non-uniform composition by Van der Waals [171] in 1893. Allen and Cahn [172] observed in 1978 that the interface between the two phases obeys the mean curvature flow. This equation, sometimes called the time-dependent Ginzburg-Landau equation, describes the universal dynamics of the phase-separation process of a binary system close to the critical point of the transition. The most iconic and perhaps simple example of such a system is the 2d Ising model of ferromagnetism, but problems of this kind are found in, essentially, all branches of science and at very different scales [173] (see Fig. 4.2 for two examples).

The basic assumption of this model is that, close to a continuous transition point denoted by the critical value of the temperature θ_c (or any equivalent thermodynamic quantity playing the role of the temperature in a phase transition), the mean value of the order parameter associated to ψ is small, since at a temperature $\theta \geq \theta_c$ the system is completely disordered, and the free-energy density $f(\psi)$ can be expressed as a Taylor series expansion about the disordered phase. Then, $f(\psi)$ has the form of a symmetric double-well function (see, for example, Ref. [170] or Ref. [109] for a full derivation)

$$f(\psi) = -\frac{\varepsilon}{2}\psi^2 + \frac{1}{4}\psi^4 , \tag{4.4}$$

where ε is a small parameter ($|\varepsilon| \ll 1$) proportional to the temperature difference from the critical point $\varepsilon = (\theta_c - \theta)/\theta_c$. Hence, the parameter ε changes sign when the temperature of the system is above or below the critical temperature. Note that, for simplicity, Eq. (4.4) is normalised so as to remove prefactors and obtain a dimensionless form.

The free-energy potential Eq. (4.4) is illustrated in Fig. 4.5. When $\theta > \theta_c$, there is only one minimum at $\psi = 0$. Minimizing this potential thus leads globally to a zero-value of the order parameter associated to ψ. For $\theta < \theta_c$, instead, two symmetric minima are present, and two phases associated to non-zero values of ψ can coexist.

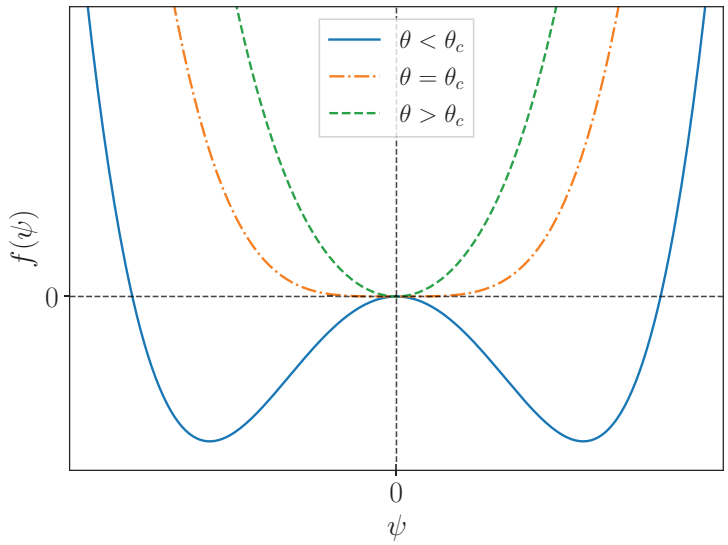


Figure 4.5: Sketch of the free energy density of a simple binary mixture or Ising model. Two stable phases arise continuously from one for $\theta < \theta_c$.

Inserting Eq. (4.4) into the evolution law for the phase field Eq. (4.3), we obtain the Allen-Cahn equation

$$\frac{\partial \psi}{\partial t} = \nabla^2 \psi + \varepsilon \psi - \psi^3 \,. \tag{4.5}$$

It can be shown that this equation converges to the sharp-interface model of mean curvature flow Eq. (4.1) when the interface width goes to zero [174, 168], as we discussed heuristically in the previous section. In the case of the coexistence of two phases, Eq. (4.5) typically leads to morphologies related to the progressive increase of the size of the phase domains, a phenomenon called *coarsening* [175], as shown in Fig. 4.6 for a two-dimensional system.

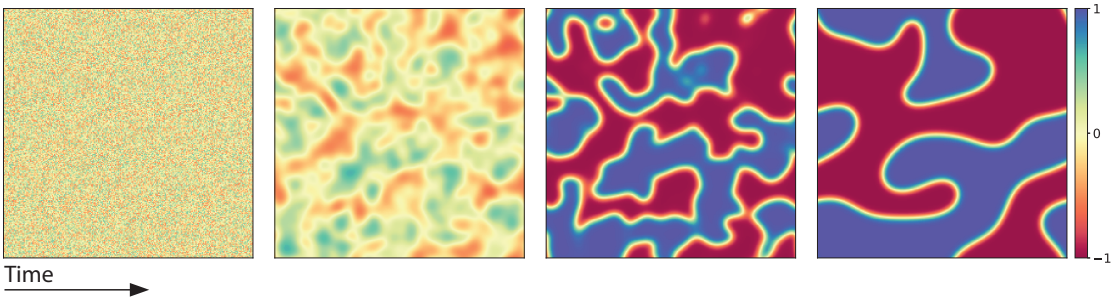


Figure 4.6: Coarsening dynamics in the two-dimensional Allen-Cahn equation with $\varepsilon = 1$.

In one dimension, the zero-dimensional diffuse interfaces, which are called *kinks*, have no curvature. Therefore, there is no motion by curvature. However, coarsening is also obtained, as shown in Fig. 4.7. This coarsening in one dimension is caused by an attractive interaction between kinks, that leads to the shrinking of the smallest domains. Since this attractive interaction decays exponentially with the distance, the resulting coarsening process is very slow, and gives rise to a logarithmic growth of the average domain size with time.

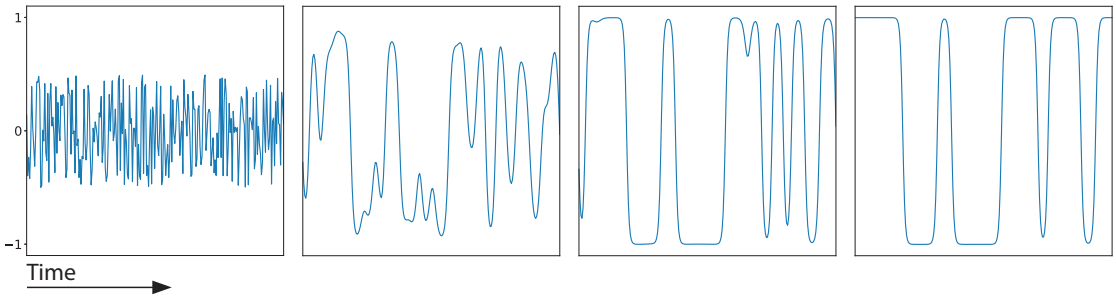


Figure 4.7: Coarsening dynamics in the one-dimensional Allen-Cahn equation with $\varepsilon = 1$.

4.2.1 Adding the control parameter

We are interested in controlling the morphology of the solutions of Eq. (4.5) in one dimension, by dynamically varying the temperature θ of the system close to the critical point. In order to do this, we substitute the ε parameter by the quantity $\varepsilon C(\varepsilon t)$, where C is a time-dependent control function of order one. This substitution corresponds to both small and slow temperature variations.

By rescaling time, space, and the phase field respectively by $t \to t/\varepsilon$, $x \to x/\varepsilon^{1/2}$, and $\psi \to \varepsilon^{1/2}\psi$, we obtain the normalized, critically-controlled Allen-Cahn equation

$$\frac{\partial \psi}{\partial t} = \nabla^2 \psi + C(t)\psi - \psi^3 \,. \tag{4.6}$$

$C(t)$ is the control function that the agent will be able to change in order to reach a target morphology. For the sake of simplicity, and in analogy to what we did in the case of discrete clusters in the previous chapters, we choose three possible values of C—the actions—that can be used by the control algorithm. The action set is then $\mathcal{A} = \{-1, 0, 1\}$.

4.2.2 Related approaches in the literature

Extensive work has been done on control of Allen-Cahn type systems, especially in the applied mathematics and control theory communities. A widespread approach is that of *distributed* control, in which a control function C is added to Eq. (4.5), resulting in

$$\frac{\partial \psi}{\partial t} = \nabla^2 \psi + \varepsilon\psi - \psi^3 + C \,. \tag{4.7}$$

The idea is then to define an *objective* functional to be minimised, of the form

$$\mathcal{J}(\psi, C) = \frac{1}{2}\int_0^L \left[\psi(t_f, x) - \bar{\psi}(x)\right]^2 dx + \frac{1}{2}\int_0^L \int_0^{t_f} C^2 \, dx \, dt \,, \tag{4.8}$$

where $\bar{\psi}$ is the desired target pattern and t_f is a fixed final time. The first term in Eq. (4.8) accounts for the control task itself, while the second one aims to limit the variations of C. Finding the control function C that minimises the objective functional $\mathcal{J}$ is a typical optimal control (model-based) task. There are many publications devoted to this kind of problems (see, for example, Refs. [52, 53, 176]), however they differ from our approach in many ways.

The most significant difference is that distributed control corresponds to an external force *added* to the system[1], whereas what we propose here is to control the system by dynamically varying the parameter associated with the distance from the critical point of the phase separation (e.g. the temperature θ near θ_c). The control problem that we aim to tackle, embodied in the critically-controlled Allen-Cahn equation (4.6), is fundamentally different from the one of Eq. (4.7). In our case, the control function C *multiplies* the phase field ψ (in mathematics this known as a *bilinear* form in the variables C, ψ), and this makes the associated optimal control problem a much more challenging task [177]. In addition, the added external force breaks the $\psi \to -\psi$ symmetry, while the multiplicative control parameter does not.

Another major difference lies in the fact that we aim to control pattern formation starting from a family of random initial conditions, whereas the works cited above assume a fixed initial state. This constraint also greatly complicates the model-based approach. Finally, most work on this type of problems uses a control function $C(x, t)$ that is space-dependent, whereas we are interested in a function that is uniform in space and varies only in time.

Some recent work by Kurita and Tsukada [54, 55, 56] explores an approach to controlled pattern formation that is closer to ours. Their model is based on a phenomenological equation for phase-separation dynamics under an inhomogeneous temperature proposed by Jaiswal *et al.* [178]. The resulting model equation reads

$$\frac{\partial \psi}{\partial t} = \nabla^2 \left[C(\boldsymbol{x}, t)\psi + \psi^3 - \nabla^2 \psi\right] \,, \tag{4.9}$$

[1]Incidentally, this is instead the approach we used for the control of clusters in previous chapters, where we considered the presence of an external field.

where $C(\boldsymbol{x}, t)$ is a control function called "phase-separation trigger". This function can be either $+1$ or -1 and locally triggers the phase separation. As we can see from Eq. (4.9), their approach is indeed similar to ours. However, their model describes the dynamics of a *conserved* order parameter, and thus is based on a different equation (the Cahn-Hilliard equation), rather than the Allen-Cahn equation. Moreover, they use a space-dependent control function, and most importantly they do not attempt to reach a desired target pattern. What they do is to systematically explore the morphologies that emerge when phase-separation triggers with different spatial distribution and speed are considered (see Fig. 1.5).

Other work exploring an approach similar to ours has been conducted by Golovin *et al.* In particular, in Ref. [179], they present a closed-loop control approach for pattern-forming systems described by the 2d Swift-Hohenberg equation. This equation is used to model a large class of nonlinear systems that exhibit spatiotemporal pattern formation, such as the Rayleigh-Bénard convection in fluids [180]. In a rescaled form, it can be written as

$$\frac{\partial \psi}{\partial t} = C\psi - (1 + \nabla^2)^2 \psi + a\psi^2 - b\psi^3 \,, \tag{4.10}$$

where C is the critical parameter which is proportional to the distance from the instability threshold, a is a parameter that characterizes the competition between stripes and hexagonal patterns, and $b = \pm 1$. The case $b = +1$ corresponds to a regime in which Eq. (4.10) describes relaxational dynamics leading to the formation of stripes or hexagonal patterns, while the case $b = -1$ leads to a blowup and does not describe pattern formation dynamics.

In Ref. [179], the authors investigate two different closed-loop control approaches of Eq. (4.10). The first is to control the competition between the formation of stripes and hexagons by tuning the parameter a, in the $b = +1$ regime, and the second is to suppress the blowup by controlling the critical parameter C, in the $b = -1$ regime. In the latter, the spatially-constant critical parameter C is varied over time as a function of ψ, in an approach that is similar to ours for the Allen-Cahn equation. However, they choose a specific functional form $C = C_0 - p \max_x |\psi|$, with $p > 0$, to prevent blowup. This is different from out approach where wo do not assume any specific functional form, and we wish to reach a specific target pattern in finite time.

4.2.3 Linear stability analysis

The constant profile $\psi(x, t) = 0 = \psi_0$ is a stationary (time independent) solution of the controlled Allen-Cahn equation. Depending on the value of the control parameter C, the stability of ψ with respect to a small perturbation $\delta\psi(x, t)$ varies. Indeed, by plugging the perturbed profile $\psi = \psi_0 + \delta\psi$ into Eq. (4.6), we obtain the linear equation

$$\frac{\partial \delta\psi}{\partial t} \approx \nabla^2 \delta\psi + C(t)\delta\psi \,, \tag{4.11}$$

where we neglected the nonlinear term because of its small amplitude. Let us consider a small harmonic perturbation of the form $\delta\psi = A \exp(i(\omega t - qx))$ with $A \ll 1$ in Eq. (4.11). With C independent of time, we obtain the linear stability equation

$$i\omega = C - q^2 \,. \tag{4.12}$$

The quantity $\mathrm{Re}\,(i\omega)$ corresponds to the growth rate (or decay, depending on its sign) of the perturbation. If the perturbation grows with time, it is said to be unstable. The behaviour of the growth rate for the three possible values of C is illustrated in Fig. 4.8. When $C \leq 0$, any perturbation decreases spontaneously, and the profile ψ is therefore stable. Higher wavenumbers

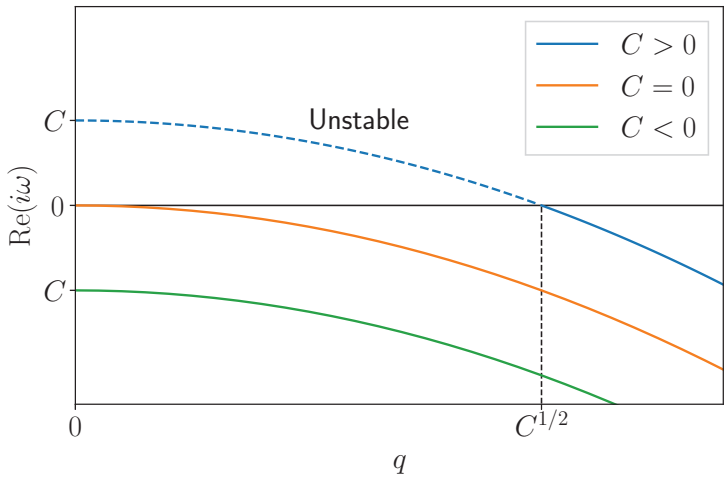


Figure 4.8: Growth rate of the perturbation $\delta\psi$ for different values of the control parameter C.

q, corresponding to smaller wavelengths, decay faster. When $C > 0$, the range of values of the wavenumber $0 \leq |q| \leq C^{1/2}$ has a positive growth rate, corresponding to an unstable perturbation.

This analysis also allows us to justify the restricted choice of C values used for control $\{-1, 0, 1\}$. Indeed, having more values of C, or even allowing C to vary continuously within an interval, would not give the learning agent a qualitatively different control over the stable or unstable character of the perturbation.

For $C > 0$, the critical wave number $q_c = C^{1/2}$ defines a characteristic length of the system

$$x_c = \frac{2\pi}{q_c} = \frac{2\pi}{C^{1/2}}\,, \tag{4.13}$$

which is related to the minimum size of features in the profiles that appear during the dynamics. Similarly, we can define a characteristic time associated with the largest growth rate

$$t_c = \frac{1}{\max\left(\mathrm{Re}\left(i\omega\right)\right)} = \frac{1}{C}\,, \tag{4.14}$$

which corresponds to the typical timescale associated with the development of features in the profiles of the controlled Allen-Cahn equation. Since in our control problem the only allowed positive value of C is 1, we simply obtain $x_c = 2\pi$ and $t_c = 1$.

4.2.4 Simulation parameters

To simulate the dynamics of Eq. (4.6) we have used a pseudo-spectral method called exponential time differencing (ETD), explained in detail in Ref. [181]. In particular, we have implemented the "second-order Runge-Kutta ETD" (ETDRK2) integration scheme in Python.

The Allen-Cahn equation is symmetric with respect to the phase field ψ, i.e. invariant to the $\psi \to -\psi$ transformation. It is also invariant to spatial translation $x \to (x + x_0)$ type transformations) and reflection symmetry ($x \to -x$). To reflect the translational symmetry and for simplicity, we naturally choose to simulate the dynamics of this equation in a space with periodic boundary conditions. We have used a periodic box of length L discretized into uniformly spaced bins of size dx.

The characteristic time of the variations of the control parameter C, that we call t_{act}, will be of the order of the physical characteristic time t_c of Eq. (4.14) associated with the instabilities of the equation. The step for the time integration dt, instead, will be smaller than this characteristic time, in order to correctly resolve the dynamics. For our simulations, we used $dt = 1/8$.

On the other hand, the characteristic spatial scale x_c of Eq. (4.13) gives us the guidelines for choosing a size L of the simulation box that is much larger than t_c, so that we have enough space to observe the emergence of features in the profiles. Instead, the space bins need to be smaller than x_c, to have sufficient spacial resolution. In the rest of this work, we used $L = 2^6$ and $dx = 1/2$.

In order to explore various dynamic trajectories during learning, we choose to study the dynamics starting from initial conditions composed of a small number of periodic sinusoidal modes of random amplitude and phase shift

$$\psi(x) = \frac{1}{n_{\max} - n_{\min} + 1} \sum_{j=n_{\min}}^{n_{\max}} A_j \cos\left(q_j x + \varphi_j\right), \tag{4.15}$$

with $q_j = j2\pi/L$. We discard the mode $j = 0$ in the decomposition, which is associated to a constant, to preserve the $\psi \to -\psi$ invariance of the dynamics. We choose $n_{\min} = 1$, so the first mode has a wavelength equal to the dimension of the box L, and the last mode $L/n_{\max}$. To restrict the number of modes and ensure that the last mode is sufficiently resolved, we choose $L/n_{\max} > x_c$. In the following, we will use $n_{\max} = 5$, i.e. a minimal wavelength of about $2x_c$.

Depending on the type of physical system studied, constraints may impose relations between the initial amplitudes A_j (for example, equipartition implies that $|A_j|^2(q_j^2 + 1) \sim k_B T/2$ from Eq. (4.2) if the system is prepared at equilibrium in the high-temperature phase). In order not to lose generality, we choose instead random values of the amplitudes A_j and phase shifts φ_j, taken uniformly in the respective intervals $0 \leq A_j \leq 1$ and $0 \leq \varphi_j \leq 2\pi$.

4.2.5 Attractor and dynamics in the intermediate-time regime

The typical dynamics of the critically-controlled Allen-Cahn equation for the three possible values of the control parameter C and in the intermediate time regime, i.e. when the maximum duration of the dynamics does not exceed a few tens of times the characteristic time t_c, is shown in Fig. 4.9.

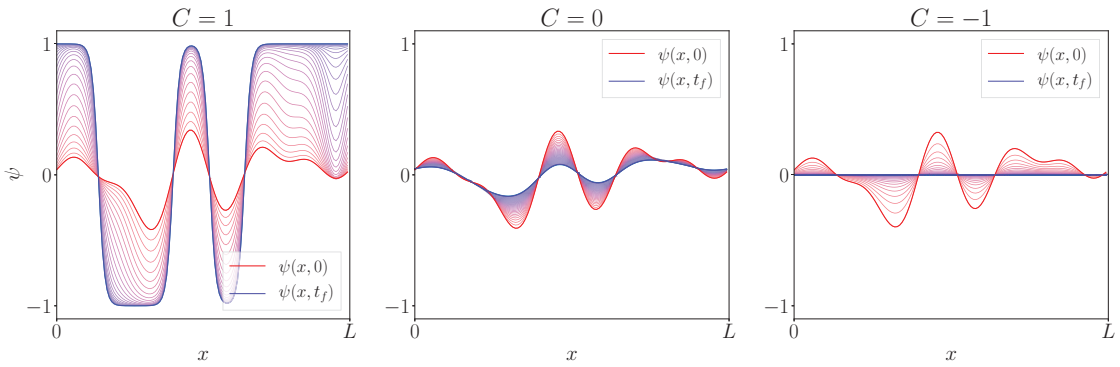


Figure 4.9: Typical dynamics associated with the three allowed values of the control parameter C, starting from the same random initial condition (in red). The value $C = 1$ leads to the growth of the profile amplitude and the appearance of positive and negative phase domains. The values $C = 0$ and $C = -1$ lead to a decrease of the profile amplitude, since they stabilize the solution $\psi = 0$. The final time is $t_f = 8t_c$.

Using the random initial condition defined in Eq. (4.15), the value $C = 1$ leads to the growth of the profile amplitude and the appearance of domains of varying size, represented by plateaus at $\psi = \pm C^{1/2} = \pm 1$ values, corresponding to the minima of the potential (see Fig. 4.5). The domains are continuously separated by structures called *kinks* when they are increasing, and *anti-kinks* when they are decreasing. The width of these structures is of the order of x_c.

Once the kinks and anti-kinks have been formed, the dynamics become slower. For a set of n pairs of kink/anti-kink centred on the positions $\left[(x_1^{\mathrm{k}}, x_1^{\mathrm{ak}}), \dots (x_n^{\mathrm{k}}, x_n^{\mathrm{ak}})\right]$, the profile can be approximated by Kawasaki and Otha's multi-kink ansatz [182]

$$\psi(x) \approx C^{1/2}\left[-1 + \sum_{i=1}^{n} \tanh\left(\left(\frac{C}{2}\right)^{1/2}(x - x_i^{\mathrm{k}})\right) - \tanh\left(\left(\frac{C}{2}\right)^{1/2}(x - x_i^{\mathrm{ak}})\right)\right]. \qquad (4.16)$$

However, from time to time, a kink and an anti-kink coalesce and annihilate, leading to a slow increase of the average domain size. The phase domains grow until all the kinks and anti-kinks disappear and give rise to a single domain, i.e. a flat profile either at $\psi = 1$ or $\psi = -1$, depending on the initial condition.

In contrast, the values $C = 0$ and $C = -1$ lead to a decrease of the profile amplitude, which is faster for the case $C = -1$, until converging asymptotically to a flat profile at $\psi = 0$, as expected from the linear stability analysis that we performed earlier (see Fig. 4.8).

The set of solutions of Eq. (4.16) defines a powerful *attractor* of the dynamics of the Allen-Cahn equation. Even for controlled dynamics, where the value of parameter C can be varied several times during evolution, we observe that the shape of the profile tends rapidly toward one of the morphologies based on kinks and plateaus given by Eq. (4.16). This strong morphological constraint greatly limits the control of the Allen-Cahn equation. We will now focus on tasks that consist of reaching specific target solutions belonging to the attractor, starting from random initial conditions.

Note however that, during the transient relaxation between $C = 1$ and $C = 0$ or -1, we can also reach shapes that exhibit a kink and plateau profile with intermediate values of the amplitude between 0 and $C^{1/2}$, as shown in Fig. 4.10. Since they can generically be reached during the dynamics when C is varied, we consider these profiles with an intermediate plateau height as being part of the attractor of the controlled Allen-Cahn equation. Moreover, due to their non-stationary character, and to the fact that they do not exist as solution of the standard Allen-Cahn equation, these profiles are interesting candidates for target states.

4.3 Control of the 1d Allen-Cahn equation

For our control task, we have used the double DQN algorithm described in Section 1.4.3. Our environment is given by the evolving critically-controlled 1d Allen-Cahn equation (4.6), integrated using the ETDRK2 scheme and with the space and time discretisation parameters dx and dt discussed in Section 4.2.4. The agent can change the control parameter $C(t)$ at regular time intervals t_{act}, by choosing among the three discrete actions $\mathcal{A} = \{-1, 0, 1\}$.

Since we are using a DQN, only the state representation is an input to the neural network. In this way, the agent only receives information about the current shape of the profile, while it receives nothing about the value of $C(t)$. Following the approach of Ref. [66], we then represent the state of the environment at time t by a set of n_{prof} stacked profiles of the Allen-Cahn equation at successive times, separated by a sampling interval t_{samp}. The result is an augmented state containing the information of the last n_{prof} profiles and, in a sense, is analogous to adding the information about temporal derivatives [62].

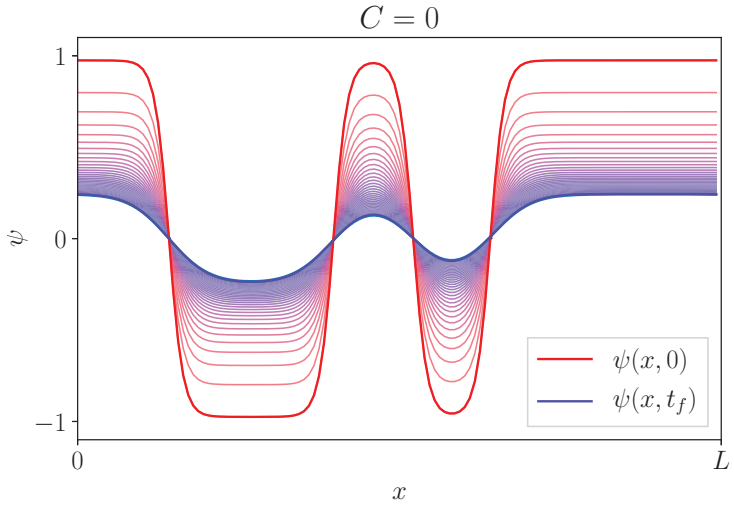


Figure 4.10: Starting from an initial profile (in red) with plateaus at amplitudes $\pm C^{1/2} = \pm 1$, it is possible to reach shapes with plateaus at intermediate amplitudes by setting $C = 0$ or $C = -1$ and letting the profile relax towards $\psi = 0$. Here, we have set $C = 0$ and a final time $t_f = 8t_c$.

All the temporal parameters of the integration and of the learning algorithm depend on the characteristic time of the instability t_c. In particular, we want $dt \lesssim t_c$ (to resolve the dynamics), $t_{\text{act}} \gtrsim t_c$ (for the instability to have time to develop), $t_{\text{samp}} \gtrsim t_c$ (so that the state contains enough information about the development of the instability). In practice, during the time interval t_{act} separating two actions, the equation of the dynamics is integrated t_{act}/dt times, and $t_{\text{act}}/t_{\text{samp}}$ uniformly spaced profiles are taken to constitute a state. This splitting is illustrated in Fig. 4.11 in the case $dt = t_c/2$, $t_{\text{act}} = 2\,t_c$, and when the state is composed of $n_{\text{prof}} = 2$ profiles separated by $t_{\text{samp}} = 2\,dt$. To reduce the size of a state, we also choose to coarsen its representation by skipping one spatial value of the profile every two when forming a state.

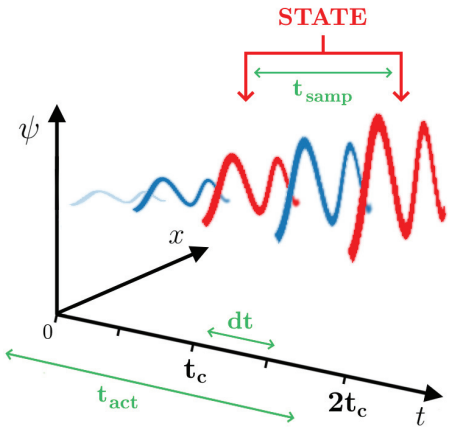


Figure 4.11: Illustration of the relations linking the different characteristic times involved in a learning episode and in the definition of the state of the environment. For this diagram, we show the case $t_c = 2\,dt$, $t_{\text{act}} = 2\,t_c$. The state is composed of 2 profiles separated by $t_{\text{samp}} = 2\,dt$.

For the architecture of the DQN used to approximate the Q-function, we took inspiration from Ref. [62]. This architecture is depicted in Fig. 4.12. The neural network takes as input the data array representing a state, with dimensions $L/(2\,dx) \times n_{\text{prof}}$, where the factor 2 comes from the fact that we skip one point every two in the profile, and the second dimension (analogous to the RGB channel of an image) is the dimension along which the profiles are stacked. Then, there are three *hidden* layers.

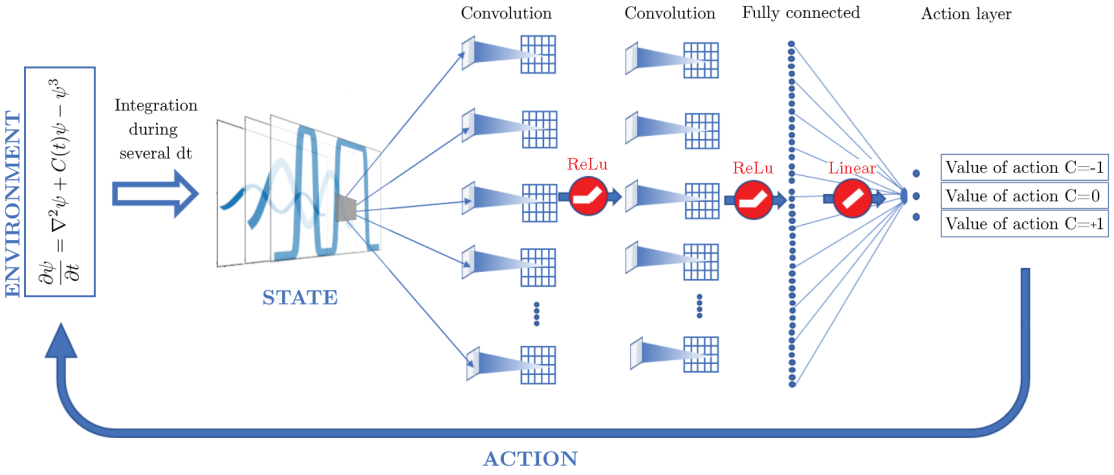


Figure 4.12: Architecture of the DQN used to approximate the Q-function. The neural network is composed of two successive convolution layers followed by a dense layer. Image modified from Patel *et al.* [183].

The first hidden layer is a convolutional layer. A convolutional layer contains a set of filters, called *kernels*, the parameters of which are to be learned throughout the training. The size of the kernels is usually smaller than the actual input. Each kernel convolves with the input and creates an *activation map*, as illustrated in Fig. 4.13. The kernel is slid across the height and width of the input and the scalar product between every element of the kernel and the input is calculated at every spatial position. The *stride* of a kernel defines the step size of the kernel when sliding through the input array. The output of the convolutional layer is generated by stacking the activation maps of each kernel in an array. Each activation map corresponds to a so-called *channel* of the output. This type of layer is common in the field of image processing, where convolutions are used, for example, for edge detection or image segmentation. The use of convolutional layers greatly reduces the number of trainable parameters and thus facilitates learning [184].

For our application, we have used a convolutional layer with a kernel of dimension $5 \times 1 \times n_{\text{prof}}$ and stride 1. We have used a circular padding at the boundaries due to the periodicity of the simulation box. The output of this layer has 8 output channels, thus, the weights of 8 different kernels are trained to produce this output (one for each output channel). The second one is another convolutional layer with 8 output channels and a kernel of dimension $5 \times 1 \times 8$ and stride 1. Between these layers there are nonlinear $ReLU$ activation functions. The last one is a fully connected (or *dense*) layer, which gives as output the estimates of the Q-values associated with the available actions, i.e. the three values of C.

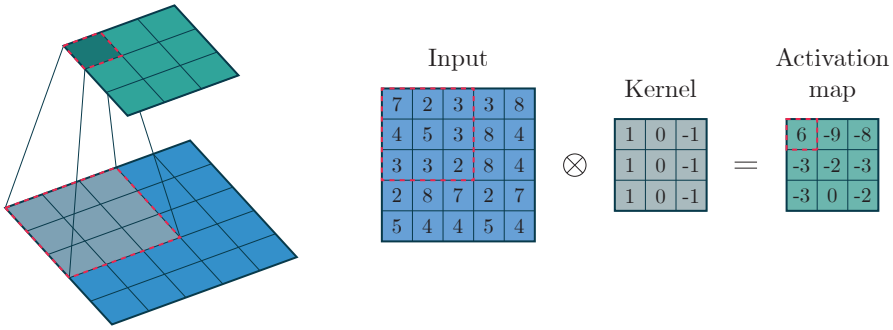


Figure 4.13: Illustration of the convolution process. The first item of the activation map is calculated by convolving the kernel with the portion in the input array marked with the red dashed outline. The full activation map is generated by repeating this process for each portion of the input data. The $\otimes$ symbol denotes the scalar product.

4.3.1 Definition of reward

For the discrete control problem of clusters discussed in Chapters 2 and 3, we have defined a reward that assigns, at every transition from a state to another, the opposite of the expected physical time of the transition (i.e. the residence time on the starting state). We did this choice because our aim was to minimise the time to reach a target configuration. For the continuous control problem of the Allen-Cahn equation that we are treating in this chapter, instead, we choose a different approach. We have seen in Section 4.2.5 that the dynamics of this equation are strongly attracted towards the kinks-and-plateaus morphologies given by Eq. (4.16). Trying to minimize the time to reach a target, then, would be too constraining. What we plan to do for this problem is instead to set a maximum time for the duration of an episode and let the algorithm learn how to reach the target at anytime before the episode is over.

To enforce such a learning task, the most naive form of a reward function assigns an arbitrary positive value to the target state, and assigns 0 to other states. From this simple definition, the algorithm must learn that, to maximize its rewards, it has to reach the target state before the end of the episode. Since all intermediate states give the same zero reward, the time it takes the algorithm to reach the target does not matter, as long as it is less than the duration of the episode. This type of reward function formulation has the advantage of letting the algorithm freely explore intermediate states without negatively affecting its reward sum. However, it also means that the algorithm has no direct feedback on the relevance of its choices leading to these intermediate states, since their values are all identical (i.e. zero). This is particularly problematic when it is difficult to reach the target state from most of the initial conditions. One way to solve this problem is to provide small intermediate rewards when the system is not far from the target state, in order to guide learning towards it. But what does "far" mean, in this context?

In the control of problem of clusters addressed in the previous chapters, we had a discrete space of states. In that case, the notion of "distance" from the target could also be seen in a discrete fashion. For example, one way to define the distance of a cluster shape from the target is to use the concept of rings, introduced in Section 2.3.3. Another possibility is to consider an overlap function, that outputs the number of particles that are in the "wrong" place with respect to the target. However, when the space of states is continuous, we need to think about the distance from the target as a continuous function.

Inspired by the approach of Ref. [185], we define the reward associated to state s with the

empirical deterministic function

$$\varrho(s) = \begin{cases} 100 & \text{if } \min_{\psi \in s} d(\psi) \leq \delta\,, \\ \exp\left(-\frac{4}{\delta^2} \min_{\psi \in s} d(\psi)^2\right) & \text{otherwise}\,, \end{cases} \tag{4.17}$$

where the notation $\min_{\psi \in s}$ means that we are taking the minimum over the profiles ψ belonging to the state s, δ is a parameter, and $d(\psi)$ is a continuous function representing the distance of the profile ψ from the target, and that will be defined later. Note that, since we are using a DQN to approximate the Q-function, we only feed to the learning agent a representation of the state of the environment, and not a state-action pair, like in tabular Q-learning (see Fig. 1.10). For this reason, we dropped the dependence on the action in the definition of the reward.

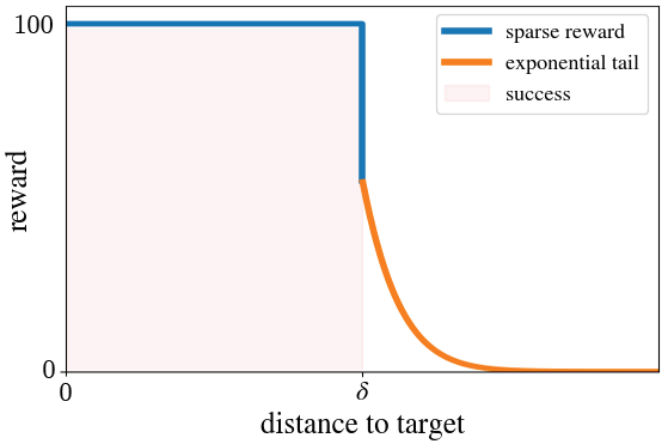


Figure 4.14: Schematic illustration of the reward function defined in Eq. (4.17). The blue part corresponds to a "sparse" reward, in the sense that it is reached only by the successful episodes, while the orange part is a decreasing exponential that guides the algorithm towards the target.

The idea behind Eq. (4.17) is to assign a positive reward $r = 100$ for states that satisfy the condition $\min_{\psi \in s} d(\psi) \leq \delta$, and hence are close to the target, within a treshold δ. The episodes that visit such a state are considered as *successful* episodes. All the other episodes that do not directly lead to the target, will still generate a certain amount of reward due to the exponential "tail" of the reward function, as illustrated in in Fig. 4.14.

We now investigate the learning capabilities of the double DQN algorithm for three different tasks, associated with different formulations of the target itself via the distance function. A scheme of the complete algorithm that we have used for the control of the Allen-Cahn equation is reported in pseudocode in Appendix A.3.

4.3.2 Results

We have considered three different types of targets. The first one is a "fixed" target, and it is the most restrictive one. It is defined by the exact correspondence between the profile ψ and the target profile $\bar{\psi}$, within the threshold δ. This definition of target is illustrated in Fig. 4.15(a) and can be seen as a zero-parameter family of profiles, in the sense that it corresponds to a single point in the configurational space of the Allen-Cahn equation.

The second one is a "translated" target and it is given by the set of spatial translations of a given profile $\bar{\psi}$. In this case, we are imposing an exact match of the *relative* positions of the kinks

and anti-kinks, as well as the amplitude of the plateaus. In other words, what matters is that the target morphology is reached, but it does not matter at what absolute position within the simulation box. This target is depicted in Fig. 4.15(b) and corresponds to a 1-parameter family of profiles, where the parameter is the horizontal shift of the whole profile.

Finally, we have considered a third target, that we have called "energy-based", because its definition involves the use of an energy functional, as we will see later. This target, illustrated in Fig. 4.15(c), is the least restrictive one. In order to reach this target, the algorithm only needs to match a desired number n_{k} of kink/anti-kink pairs, as well as the amplitude of the plateaus. It can be seen as a $2n_{\mathrm{k}}$-parameter family of profiles, where the $2n_{\mathrm{k}}$ parameters correspond to the horizontal shifts of the kinks and anti-kinks.

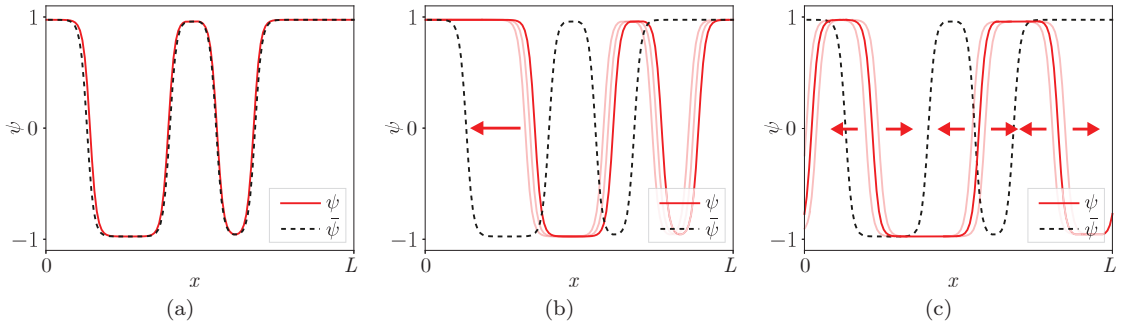


Figure 4.15: Schematic representation of the different definitions of targets. The three cases show a perfect match (within the tolerance δ) between a profile ψ (in red) and a target $\bar{\psi}$ (in black, dashed) for the three targets: (a) fixed, (b) translated, (c) energy-based. Red arrows represent the degrees of freedom.

Before addressing the problem of achieving a target morphology based on kinks and antikinks, let us begin with a test of the algorithm's validity on a trivial problem: taking an initially flat profile of arbitrary amplitude between 0 and 1 to a flat target profile $\bar{\psi}(x) = 0.8$. In this case, the gradient $\nabla^2\psi$ disappears from Eq. (4.6), and an initially flat profile remains flat. The solution can then be represented by scalar functions without spatial dependence.

For this test, we use the fixed target, defined by the distance function

$$d_{\mathrm{fixed}}(\psi, \bar{\psi}) = \left[\frac{\int_0^L \left(\psi(x,t) - \bar{\psi}(x)\right)^2 dx}{\int_0^L \bar{\psi}(x)^2 dx} \right]^{1/2} . \tag{4.18}$$

With this definition of target and the initial and target profile defined above, it is possible to reach a success rate of 100%, where the success rate is defined as the percentage of initial conditions that reach the target.

We impose the initial condition to be a flat profile at a random amplitude chosen uniformly between 0 and 1. The amplitude of the target profile $\bar{\psi}(x) = 0.8$ is chosen so that it does not match the amplitude of the stationary solution associated with any of the values of C available to the algorithm ($C = 0$ and $C = -1$ lead to a zero stationary solution, and $C = 1$ leads to a solution of amplitude ± 1). For this specific analysis, we use the values $t_{\mathrm{act}} = t_c = 1$ and $t_f = 32\, t_c = 32$. The tolerance of the reward function is set to $\delta = 0.04$.

In Fig. 4.16 we report the evolution of the success rate during the learning process on 1000 episodes. Since the convergence of the algorithm varies from one training to another, this quantity

is averaged over 20 independent trainings. In the plot, we also add the success rates of two untrained random policies: in red, one that chooses C from the complete set of actions $\{-1, 0, 1\}$, and in blue, one that chooses only between the two values $\{0, 1\}$. The rate obtained with the first policy is $\sim 76\%$, and this confirms that the algorithm is indeed learning to control the environment, since it reaches an average of $\sim 99\%$, after about 800 episodes. The random policy choosing between 0 and 1 obtains a success rate close to 100%. This result is not trivial because the time t_{act} is here equal to the characteristic time of the dynamics t_c, so that the height of the plateau has time for partial relaxation towards the stationary values $\psi = 0$ or $\psi = 1$. If t_{act} was much smaller than t_c, then the effect of the random policy would be an average value of $C = 0.5$ for the $\{0, 1\}$ random policy and $C = 0$ for the $\{-1, 0, 1\}$ random policy, leading to an effective stationary plateau height at $0.5^{1/2} \approx 0.7$ or close to zero respectively. Since the value $\psi = 0.5^{1/2} \approx 0.7$ is closer to the target value $\bar{\psi} = 0.8$, this suggests that the $\{0, 1\}$ policy is more efficient.

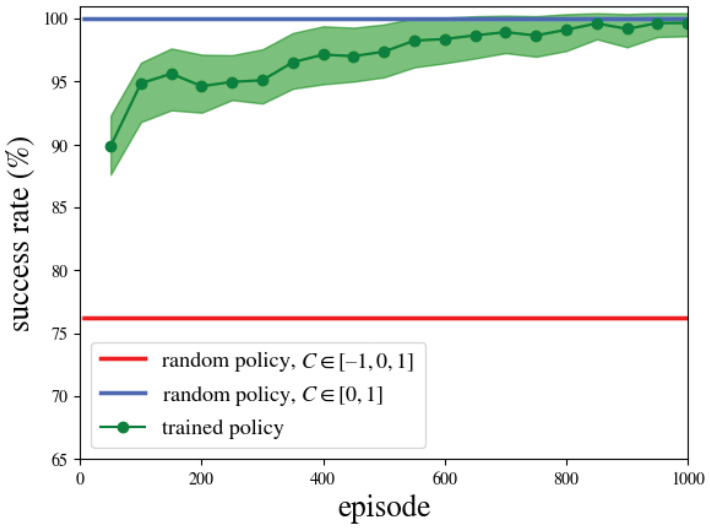


Figure 4.16: Evolution of the success rate over the course of learning. The average values over 20 learnings are represented by solid lines, and the standard deviations by the coloured surfaces.

Unlike the case of clusters, here we cannot resort to a simple graph representation of the states of the system to visualize the learned policy. To have an idea of which actions are chosen by the algorithm we have to resort to other ways. In Fig. 4.17 we plot what is the first action chosen, on average, by the agent for different amplitudes of the initial flat profile. An intuitive interpretation of this plot is as follows.

For amplitudes that are much lower than the final value $\bar{\psi} = 0.8$, the 20 policies always chose $C = 1$ as first action, which has the effect of increasing the amplitude of the profile. Around the value 0.8, we are in a special case: we are setting the initial condition on the target itself (within the threshold of the distance function) and we want the algorithm to find a way back to it, somewhat like in the case of clusters when looking at the mean return time to target. Since no value of C allows the profile to remain stationary, the algorithm must first move the profile out of the target morphology and then come back to it with a subsequent action. This leads policies to propose different values of C as a first action in the neighbourhood of 0.8. Finally, for initial amplitudes close to 1, most policies try to decrease the amplitude with the value $C = 0$, and

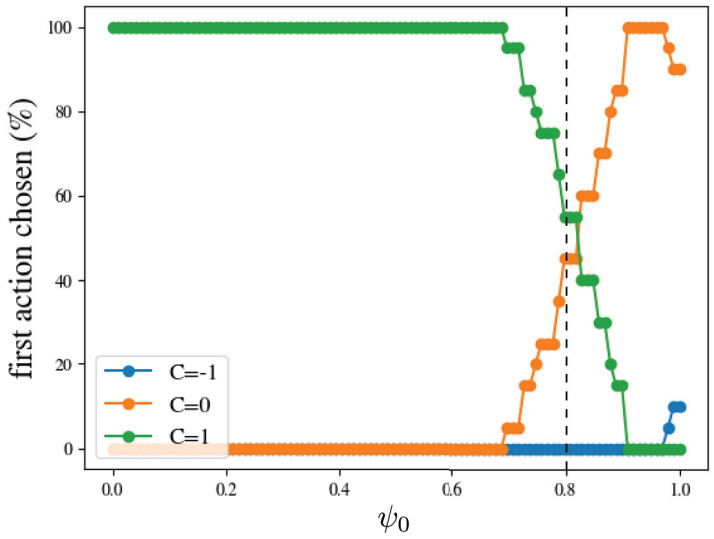


Figure 4.17: First action chosen by the algorithm, averaged over the policies obtained after 20 learnings, depending on the amplitude ψ_0 of the initial flat profile. The vertical dashed line corresponds to the amplitude of the target profile $\bar{\psi}$.

some even use the value $C = -1$, to decrease it more quickly.

Having confirmed that the learning algorithm works correctly, we now proceed to present the results for the three different families of targets.

1. Fixed target

For this task, we train the algorithm to bring an initial profile randomly chosen among the family defined by Eq. (4.15) towards a fixed target profile typical of the attractor of the Allen-Cahn equation, for episodes of length $t_f = 32$. We use the distance function defined above in Eq. (4.18), with a tolerance on the reward which is set again to $\delta = 0.04$. We choose for simplicity a target profile composed of one centred kink/anti-kink pair, separating plateaus of amplitudes equal to ± 0.8. We use Eq. (4.16) to design the target profile

$$\bar{\psi}(x) = 0.8 \left[-1 + \tanh\left(\left(\frac{0.8}{2} \right)^{1/2} \left(x - \frac{L}{4} \right) \right) - \tanh\left(\left(\frac{0.8}{2} \right)^{1/2} \left(x - \frac{3L}{4} \right) \right) \right]. \tag{4.19}$$

The evolution of the success rate during learning is shown in Fig. 4.18, averaged over 10 independent learning tasks. Here, the success rates of both the two random policies and the trained one are extremely low, between 0.15% and 0.25%. We believe that these poor results are due to the fact that very few random initial conditions are intrinsically able to reach the fixed target. In an attempt to understand the reasons leading to this phenomenon and the resulting low performance of the learning algorithm on this task, we develop here a qualitative argument.

In Fig. 4.19 we show the evolution of an initial profile (in red) composed of periodic sinusoidal modes, as given by Eq. (4.15), toward the symmetric single-period target of Eq. (4.19) (in dashed black line). When the initial condition contains only one centred mode, with a wavelength

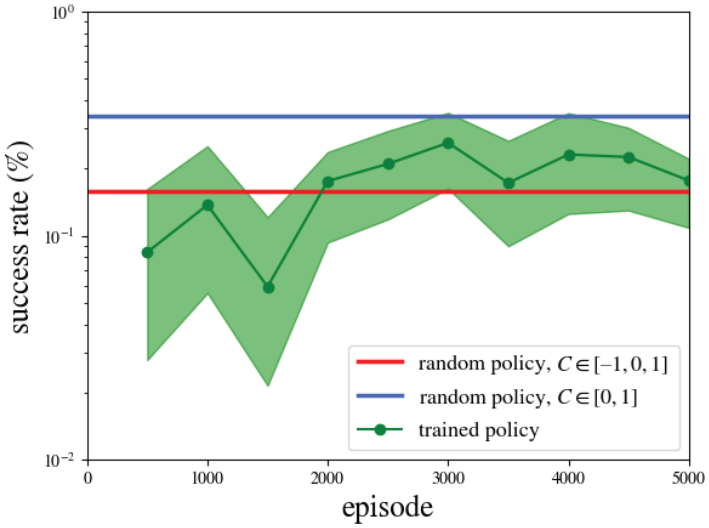


Figure 4.18: Evolution of the success rate over the course of learning for the fixed target, with $t_{\text{act}} = 1$. The average values over 10 learnings are represented by solid lines, and the standard deviations by the coloured area.

corresponding to the size L of the simulation box

$$\psi(x, 0) = -A \cos\left(\frac{2\pi}{L}x + \varphi_1\right) , \tag{4.20}$$

with $A > 0$ and $\varphi_1 = 0$, then the dynamics tends naturally towards a final morphology that has one kink and one anti-kink aligned with the target profile $\bar{\psi}(x)$, when the value $C = 1$ is kept constant, as shown in Fig. 4.19a. This final profile is then easily brought by the algorithm to the desired target amplitude 0.8 by switching C to 0 or -1, as shown in Fig. 4.10. When the phase shift of the first mode φ_1 is non-zero, then the initial profile is shifted relative to the target, and this leads to a similarly shifted final profile (this case is not shown in Fig. 4.19). Therefore, we expect the value of the phase shift to have a significant impact on the success rate of the algorithm.

When more than one mode is included in the initial condition, then the final morphology can vary greatly, depending on the relative amplitude of the modes, as shown in Figs. 4.19b and 4.19c.

In order to investigate these effects on the trained policies, we have performed a systematic analysis of the influence of the phase shift of the first mode of the initial condition on the success rate. The results of this analysis are shown in Fig. 4.20. The black line corresponds to the case where the algorithm is trained with initial conditions that have only one mode, as in Eq. (4.20). For a centred initial condition (i.e. $\varphi_1 = 0$), the algorithm reaches a success rate of 100%, as expected. This is also the case for values of of the phase shift φ_1 higher than 0, up to an offset of about 1% of the size L of the domain. This is a consequence of the tolerance threshold δ in the definition of the reward. For values of φ_1 higher than this threshold, the success rate drops to 0%, which suggests that the algorithm is not able to learn to shift the profiles by varying the C parameter.

The blue and orange lines correspond instead to the success rates obtained when the initial conditions are respectively composed of the first 2 modes and all 5 modes, for which the phase

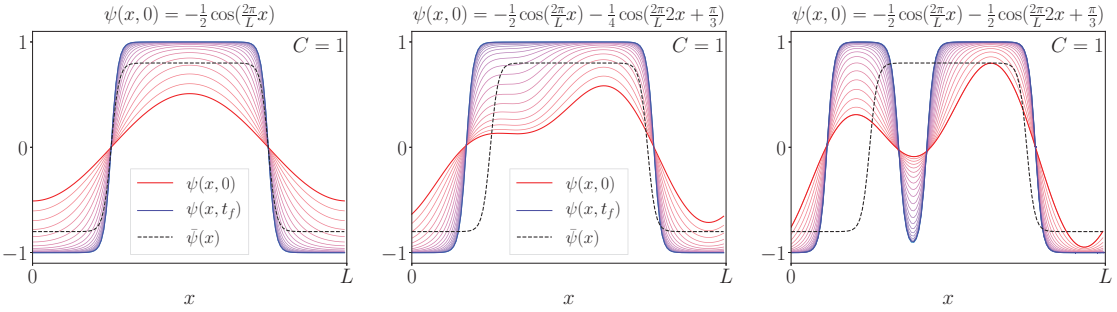


Figure 4.19: Illustration of the phenomenon of interaction of modes and the impact of the dominant mode in the morphology, for $C = 1$ and $t_f = 5t_c$. (a) The dynamics of the largest mode, alone, leads to a symmetrical final profile with the same period of the initial condition. (b) Disruptive addition of the second mode. By interaction, the final profile is no longer symmetric and is slightly shifted with respect to the initial position of the first mode. (c) As the amplitude of the second mode increases, its contribution to the dynamics of the system becomes more important, and the final profile can have the same period as the initial condition (the legend has been omitted for visualization purposes).

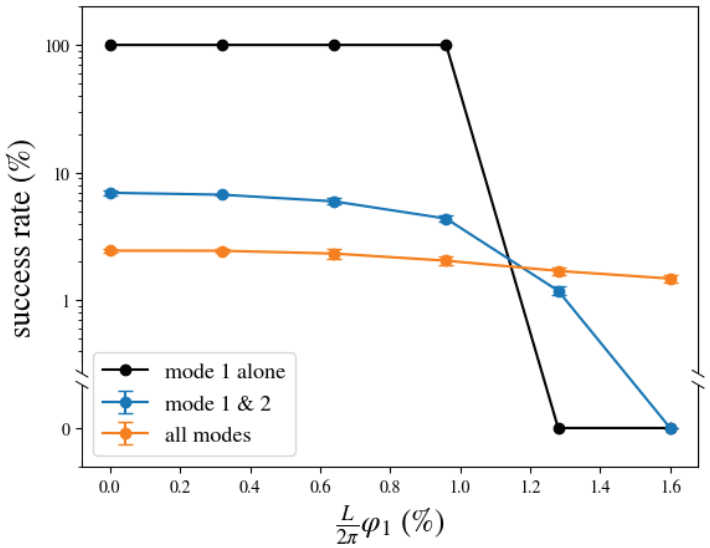


Figure 4.20: Evolution of the success rate of trained policies with different types of initial conditions as a function of the phase shift of the first mode φ_1. The black, blue and orange curves correspond respectively to initial conditions with only the first mode, the first two modes, and all modes. Each point is averaged over 10 trainings.

shifts of the modes with indices greater than 1 are chosen randomly, while the amplitude of the first mode is kept at 1/2. In this case, a success rate of 100% is never achieved, even when the first mode is aligned with the target ($\varphi_1 = 0$). However, non-zero success rates are obtained for some shifts beyond the tolerance threshold. We think that this phenomenon is due to the

interactions between the modes, as illustrated in Fig. 4.19b, which shows that when a second mode of small amplitude is added to the initial condition, the final profile can have a single wider, shifted plateau. The exact width and the shift of this plateau depend on the values of the amplitude and phase shift of the second mode. This can lead an initial condition to reach the desired target even if the first mode is shifted from the target beyond the distance threshold δ.

The success rates obtained when the initial condition is composed of all 5 modes are lower than ones for the first 2 modes, but are non-zero for larger values of the initial offset. This can be interpreted, in light of the previous considerations, as the fact that when the number of modes that can interact is large, these interactions greatly influence the dynamics, and the position of the first mode is no longer as important as when fewer modes are present in the initial condition.

In conclusion, the analysis presented here, and in particular the results shown in Fig. 4.20, suggests that it is difficult to shift an initial condition with a dominant mode that is not aligned with the target profile just by acting on the control parameter C. Although the target is part of the attractor of the Allen-Cahn equation, in practice only a restricted number of initial conditions are intrinsically able to reach a desired fixed morphology, and it seems that the learning algorithm is not able to increase significantly this number. Hence, imposing a fixed target is perhaps too restrictive for the DQN agent, which cannot access a large enough statistical sample to learn an optimal policy that is better than a random one, assuming such a policy exists.

2. Translated target

We now consider the case of a translated target. This target can be defined by extending the definition of Eq. (4.18) for a fixed target to the set of spatial translations $\{\bar{\psi}\}$ of a given target profile $\bar{\psi}$.

We use the reward function defined in Eq. (4.17), but this time we define the following distance function

$$d_{\text{transl}}\left(\psi, \{\bar{\psi}\}\right) = \min_{\Delta x} d_{\text{fixed}}\left(\psi, \bar{\psi}(x-\Delta x)\right) = \min_{\Delta x}\left[\frac{\int_0^L \left(\psi(x,t)-\bar{\psi}(x-\Delta x)\right)^2 dx}{\int_0^L \bar{\psi}(x)^2 dx}\right]^{1/2}, \quad (4.21)$$

with $0 \leq \Delta x \leq L$. In this case, for a target profile in the form of Eq. (4.16), this formulation imposes an exact match (within the tolerance threshold δ) of the relative positions of the kinks and anti-kinks, and of the amplitude of the plateaus.

For the learning task, we set a random initial condition, including all 5 modes of Eq. (4.15), and consider the same target profile $\bar{\psi}$ given by Eq. (4.19) that we have used for the fixed target, consisting in a single kink/anti-kink pair (see Fig. 4.19).

In Fig. 4.21 we show the evolution of the success rate, again averaged over 10 independent tasks, for episodes of length $t_f = 32$. Changing the target to a set of shifted profiles rather than considering a fixed one effectively improves the number of initial conditions potentially leading to success. Indeed, we can see that the success rate of the completely random policy (red line) is close to 5%. In this case, the optimal policy learned by the algorithm approaches 15% success rate, and also outperforms the random policy that can choose only the actions $\{0, 1\}$ (blue line), contrary to the case of the fixed target.

The performance shown in Fig. 4.21 was obtained with a time interval between two actions t_{act} equal to the timescale associated to the development of the kinks and anti-kinks t_c, i.e. $t_{\text{act}} = t_c = 1$. We have conducted an analysis of the effect of the value of t_{act} on the success rate of the algorithm. The results of this analysis are reported in Fig. 4.22. Between $t_{\text{act}} = 1$ and $t_{\text{act}} = 4$, the success rate varies weakly, and shows a small peak for $t_{\text{act}} = 4$. For values $t_{\text{act}} > 4$, the success rate drops, suggesting that, in this range, the number of actions during an episode (which has maximum duration $t_f = 32$) is not sufficient to achieve an effective control.

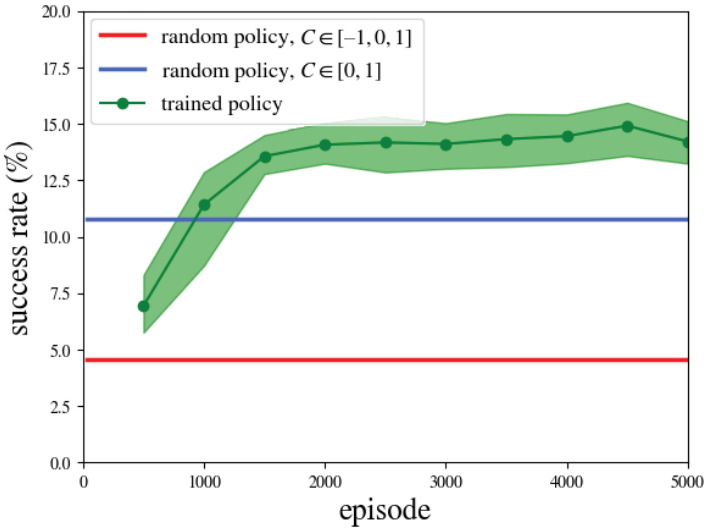


Figure 4.21: Evolution of the success rate over the course of learning for the translated target, with $t_{\text{act}} = 1$. The average values over 10 learnings are represented by solid lines, and the standard deviations by the green area.

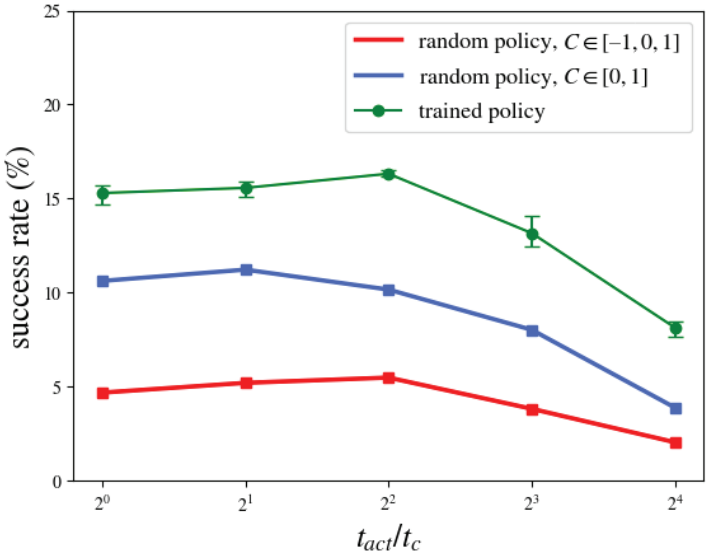
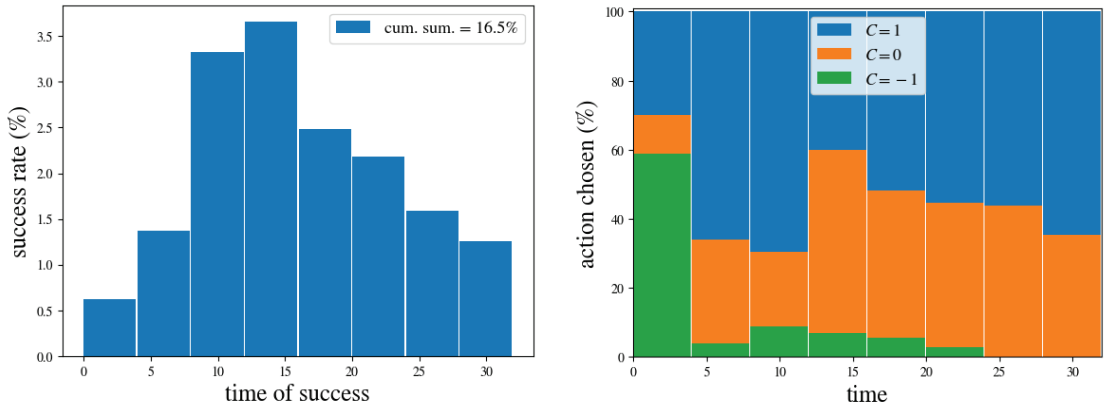


Figure 4.22: Success rate as a function of the ratio t_{act}/t_c, for the trained policy and the two considered random policies.

To try to visualize and interpret the learned optimal policies, we show in Fig. 4.23 two histograms, obtained with the value $t_{\text{act}} = 4$ at which the algorithm performs better. Figure 4.23a shows at what time, on average, the algorithm achieves a success during an episode, while Fig. 4.23b shows in what proportions each action is chosen, on average, at different timesteps during an episode. We can see that the algorithm primarily uses the value $C = -1$ as the first

action until $t \approx 4$, and then switches to using mostly $C = 1$ until $t \approx 12$, before finally using $C = 1$ and $C = 0$ in similar proportions, almost without using $C = -1$.

We can give a heuristic interpretation of this behaviour in the following way. Since the action $C = -1$ has the effect of reducing the amplitude of the profile and rapidly decreasing the small-wavelength modes (see Fig. 4.9), we can imagine that an efficient sequence of actions could be to use the value $C = -1$ at the beginning, when the first mode of the initial condition (which is morphologically close to the target) is not dominant, in order to reduce the relative amplitudes of the other modes. Then, once the mode $n = 1$ has become dominant, use the value $C = 1$ to restore the amplitude of the profile, and finally oscillate C between 0 and 1 to make the profile reach the desired amplitude of the target.



(a) (b)

Figure 4.23: Histograms (a) of average occurrences of successes and (b) proportions of choices of different actions during an episode. The interval between actions is $t_{\text{act}} = 4$. Averages are taken over 10^5 episodes.

To complete this analysis, we now test to what extent the optimal policies based on a closed-loop approach (i.e. depending on the current state of the system) can be approximated by stochastic open-loop policies, whose behaviour depends only on time and no longer on states.

We start considering an open-loop policy that chooses the actions as a function of time based on the average probabilities extracted from the histogram in Fig. 4.23b. The performance of this policy is reported in Fig. 4.24a. The final success rate is about 12%, compared to the 16.5% of the closed-loop trained policy (Fig. 4.23a). This result seems to suggest that the state of the system contains indeed important information that the agent is able to exploit and that does not allow a simple open-loop policy to reach the same performance of the closed-loop one. But what if we try to add our human intuition to design an efficient open-loop policy?

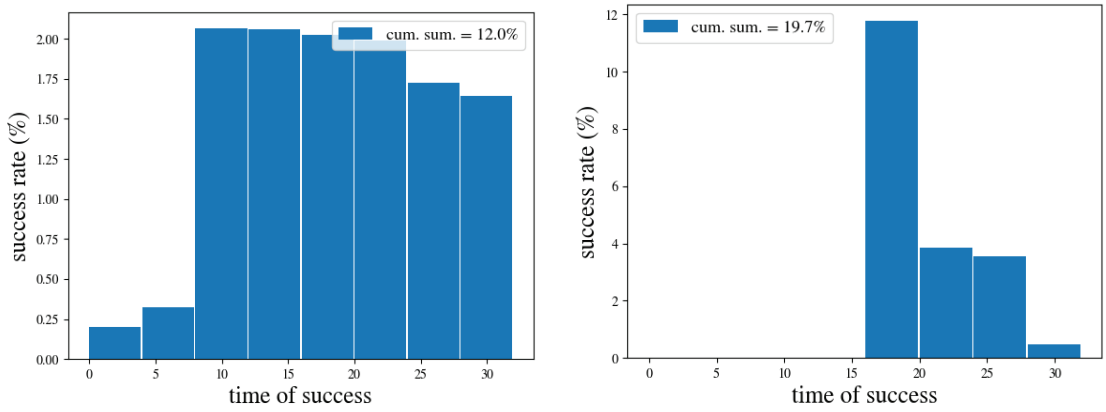
We define an *ad hoc* policy, that varies $C(t)$ as a function of time in the following manner

$$C(t) = \begin{cases} 0 & \text{if } 0 \leq t \leq 16 \text{ or } 20 \leq t \leq 24 \text{ or } 28 \leq t \leq 32\,, \\ 1 & \text{otherwise}\,. \end{cases} \tag{4.22}$$

Such a policy corresponds to keeping $C = 0$ constant during the first half of the episode, until $t = 16$, to make the first mode of the profile dominant over the other modes, since it is the one with the lowest decay rate. Then, it alternates between $C = 1$ and $C = 0$, to restore the amplitude of the profile and oscillate it close to the desired target value. We choose to keep

$C = 0$ at the beginning, rather than -1, because both have the effect of reducing the amplitude of high-frequency modes, but $C = -1$ leads more quickly to a profile very close to 0, for which it is harder to restore the amplitude to a value of order 1.

The performance of this policy is shown in Fig. 4.24b. We can see that it achieves a total success rate of $\sim 20\%$, higher than the 16.5% success rate obtained under the same conditions by the trained closed-loop policy (see Fig. 4.23a). The histogram shows a peak responsible for the majority of successes which appears immediately after $t = 16$, when the C value is reset to 1, showing that the $C = 0$ value effectively increases the relative amplitude of the first mode and thus smoothes the profiles enough to match the target morphology.



(a) (b)

Figure 4.24: Histograms of the average occurrences of successes during an episode for (a) the stochastic policy based on the histogram of Fig. 4.23b and (b) the *ad hoc* policy of Eq. (4.22).

Despite its simplicity, the policy of Eq. (4.22) provides better performance than the policy learned by the algorithm. It is possible that by extending the training time and carefully adjusting various parameters of the neural network and the learning algorithm, the agent could achieve a better or at least equal performance to that of this policy. However, we have not investigated in this direction further.

The conclusions we can draw from the analysis performed in this part are as follows. By extending the control task to achieving a morphology defined only by the number and relative position of kinks and anti-kinks (and not their absolute position), the control algorithm based on the double DQN succeeds in controlling the critically-controlled Allen-Cahn equation, with a success rate of $\sim 15\%$, which is significantly higher than the $\sim 5\%$ achieved by the untrained random policy. Furthermore, by inspecting the probabilities of choosing actions as a function of time, it is possible to manually design an open-loop policy that obtains performances even higher than those of the DQN agent.

3. Energy-based target

Finally, we report here the results for the least restrictive definition of target, out of the three we considered. This definition corresponds to imposing the number of kink/anti-kink pairs to be reached (and the amplitude of the plateaus). In order to define the corresponding distance function, we proceed as follows.

We have seen in Section 4.2 that, for a constant control parameter C, the 1d Allen-Cahn equation is associated to an energy functional of expression

$$\mathcal{F}(\psi, C) = \int_0^L \left\{ \frac{1}{2}(\nabla\psi)^2 - \frac{C}{2}\psi^2 + \frac{1}{4}\psi^4 \right\} dx \,. \tag{4.23}$$

If the profile ψ comes from the attractor of Eq. (4.16), the only contribution of ψ to $\mathcal{F}(\psi, C)$ comes from the kinks and the anti-kinks, since the plateaus do not bring energy. We call the contribution of a single kink or anti-kink $\mathcal{F}_{\rm k}(C)$. Note that the energy contribution of a kink and and that of an anti-kink are the same.

Let $n_{\rm k}$ be the number of kink/anti-kink pairs targeted and $C^{1/2}$ the desired amplitude of the plateaus. We then define the distance function

$$d_{\rm energy}(\psi, n_{\rm k}, C) = \left[\frac{\left(\mathcal{F}(\psi, C) - 2n_{\rm k}\mathcal{F}_{\rm k}(C)\right)^2 + \gamma(2n_k - n_0)^2}{\mathcal{F}_{\rm k}(C)^2} \right]^{1/2} . \tag{4.24}$$

The first term of the numerator acts as a quadratic distance between the energy of the current profile and the energy of the desired profile, of approximate value $2n_{\rm k}\mathcal{F}_{\rm k}(C)$. Alone, this contribution is not a strong enough constraint on the energy to impose the desired morphology. The second term adds an additional constraint on the number of zeros in the ψ profile, denoted n_0, which must be equal to the number of kink/anti-kink pairs of the desired morphology. The prefactor γ is used to adjust the magnitude of this second term. We wish to use this target for an arbitrary height of the plateaus in the target profile, as in the previous cases discussed above.

We start again by a random initial condition, including all 5 modes of Eq. (4.15). As a target, we have decided to impose the number of kink/anti-kink pairs $n_{\rm k} = 2$, with an amplitude 0.8, and a tolerance threshold in the reward function $\delta = 0.01$. We show in Fig. 4.25 the evolution of the success rate as a function of the episodes for this learning task.

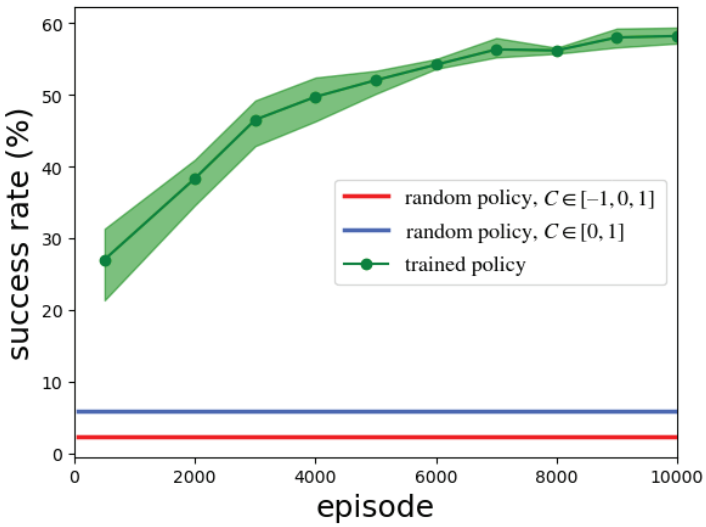


Figure 4.25: Evolution of the success rate over the course of learning for the energy-based target, with $t_{\rm act} = 2$. The average values over 10 learnings are represented by solid lines, and the standard deviations by the green area.

The final time of the episodes is again equal to $t_f = 32$, and we set, following an empirical analysis, the time interval between actions to $t_{\text{act}} = 2t_c = 2$. We can see that he performance of the algorithm here far exceeds that of the random policies and presents particularly tight standard deviations, showing stability of the learning process.

We now present a qualitative analysis of the policy learned by the algorithm. In Figs. 4.26a and 4.26b we report the histograms of the success rate and the average action chosen to reach the $n_{\text{k}} = 2$ target defined above. The histogram of success rates is particularly peaked between $t = 12$ and $t = 14$, and shows periodic peaks of decreasing amplitude at later times. The histogram of the action choices shows that the value $C = 1$ is systematically chosen as the first action, and then the histogram has a rather periodic structure of period 8, which is higher than the period 4 of the peaks in the histogram of success rates. The fact that we can observe a structured temporal pattern in the policy is a surprising feature, since physical time is not part of the state definition and thus not observable by the the agent.

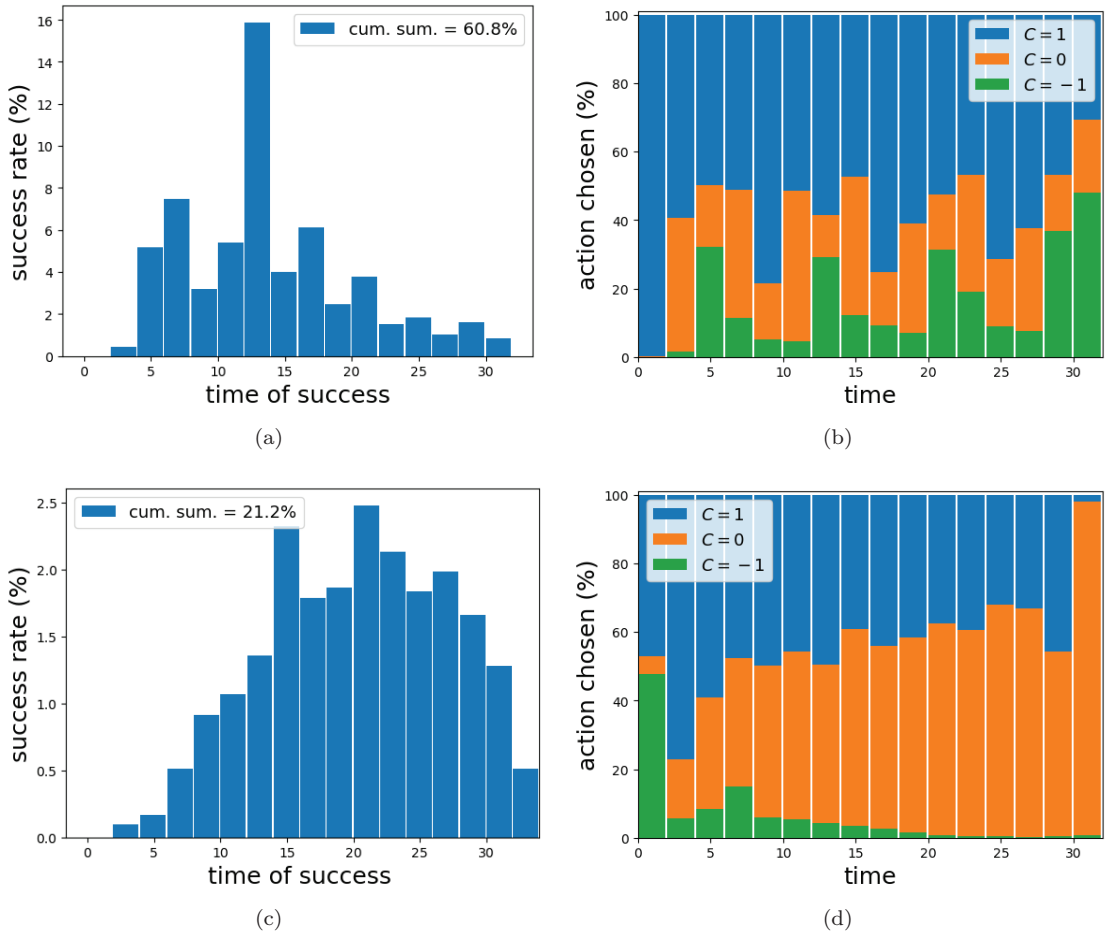


Figure 4.26: Histograms of average occurrences of successes (left) and proportions of choices of different actions during an episode (right), for $t_{\text{act}} = 2$. (a, b) Optimal policy to reach a $n_{\text{k}} = 2$ target. (c, d) Optimal policy to reach a $n_{\text{k}} = 1$ target. Averages are taken over 10^5 episodes.

Finally, in Figs. 4.26c and 4.26d we report the results for the policy trained to reach a target with only one pair of kink/anti-kink, i.e. $n_{\mathrm{k}} = 1$. It is interesting to see that the structure of the histogram in Fig. 4.26d for the average choice of action has some similarities with the histogram in Fig. 4.23b that was obtained for a translated target with one pair of kink/anti-kink. In the first action, the value $C = -1$ is again chosen with high probability, followed by the subsequent action $C = 1$. At later times, the values $C = 1$ and $C = 0$ are also chosen here with similar probabilities, which corresponds qualitatively to an adjustment of the amplitude of the plateaus.

4.3.3 Discussion and perspectives

In this section, we have investigated the morphological control of the Allen-Cahn equation, a universal differential equation that can model a spatially extended physical system close to an instability threshold. We have developed ways to obtain closed-loop and open-loop control policies that can dynamically change the value of a control parameter associated with the distance from the critical point of the instability. To our knowledge, no approach has been developed for controlling this type of system in the literature.

We noticed that the morphologies of the Allen-Cahn equation are strongly restricted to a particular set of functions, that we call the attractor. We then studied the learning capabilities of a double DQN-based algorithm on tasks of different complexity. By imposing target morphologies based on sufficiently large families of the attractor functions, we showed that the control algorithm is able to learn non-trivial strategies by exploiting the intrinsic dynamics of the system. We also proposed ways to interpret the strategies found by the algorithm, which is not an easy task since we are dealing with a continuous system and using an agent that relies on neural networks, which are notoriously difficult to interpret. In some simple cases, these policies can be used to design open-loop policies with performance that can exceed that of the learning algorithm.

Various research directions could be explored with the methods developed in this work. For example, one could consider extending the model to the 2d case, which would allow the study of more complex shapes, or the inclusion of stochastic noise in the Allen-Cahn equation, which might help the dynamics break out of the restrictive morphologies of the attractor. An interesting perspective would be to apply the learning algorithm developed here to the control of the discrete model described in Chapters 2 and 3, for particle clusters of area greater than 12. These clusters correspond to a space of states that is too large to be treated with a tabular approach, and thus require an approximation method.

4.4 Model C

In the first part of this chapter, we derived a model for phase separation in an isothermal system, and then added to this model a control parameter associated with small changes in temperature near the critical point of phase separation. In many cases of practical interest, however, treating temperature isothermally—or even uniformly—is not a good approximation. The Allen-Cahn dynamics of Eq. (4.5) can be augmented to consider non-isothermal evolution by allowing the constant temperature θ to be replaced by a field $\theta(\boldsymbol{x}, t)$.

We will now outline the main ideas for deriving the model C, that can describe growth phenomena. Note that we will consider the case of growth of a solid phase in an undercooled melt (see Fig. 4.1a), so we will refer to $\theta(\boldsymbol{x}, t)$ as the temperature field [170, 159]. However, this is a very versatile model that can be adapted to a wide variety of growth processes driven by very different fields, such as the supersaturation-driven growth of a precipitate, and in this case the driving field would represent material concentration [160], or the growth of a bubble against a viscous fluid in a Hele-Shaw cell (see Fig. 4.1c), where the field would represent pressure [186, 187].

Similar models have also been used to describe the growth of bacterial colonies (see Fig. 4.1b). The driving field, in this case, would be associated to nutrient concentration [188, 189].

The basic equations of model C can be derived from a single phenomenological free-energy functional, when expressed in the so-called "variational form", introduced by Langer [167]. In this formulation, the model describes the evolution of two independent fields: the phase field ψ and the dimensionless enthalpy U, defined as

$$U(\theta, \psi) = \theta - \theta_{\rm eq} - \frac{h(\psi)}{2}\,, \tag{4.25}$$

where $\theta_{\rm eq}$ is the temperature at which the two phases are at equilibrium (i.e. the melting temperature), and $h(\psi)$ is a function that describes the generation of excess heat produced by the formation of the solid phase. The free-energy functional reads

$$\mathcal{F}_{\rm VF} = \int_\Omega \left\{ \frac{1}{2}(\nabla\psi)^2 + f(\psi) + \lambda b \frac{(\theta - \theta_{\rm eq})^2}{2} \right\} d\boldsymbol{x}\,, \tag{4.26}$$

where $f(\psi)$ is the usual symmetric double-well function, and b and λ are constants. Then, the equations of motion for ψ and U, in dimensionless units, are

$$\begin{aligned} \frac{\partial\psi}{\partial t} &= -\frac{\delta\mathcal{F}_{\rm VF}}{\delta\psi}\,, \\ \frac{\partial U}{\partial t} &= \frac{1}{b\lambda}\nabla^2 \frac{\delta\mathcal{F}_{\rm VF}}{\delta U}\,. \end{aligned} \tag{4.27}$$

We will use here another formulation, proposed by Karma and Rappel in Ref. [159], called the "isothermal variational formulation" and defined by the equations

$$\begin{aligned} \frac{\partial\psi}{\partial t} &= -\frac{\delta\mathcal{F}_{\rm IVF}}{\delta\psi}\,, \\ \frac{\partial\theta}{\partial t} &= \nabla^2\theta - \frac{1}{2}\frac{\partial h(\psi)}{\partial t}\,, \end{aligned} \tag{4.28}$$

where

$$\mathcal{F}_{\rm IVF} = \int_\Omega \left\{ \frac{1}{2}(\nabla\psi)^2 + f(\psi) + \lambda g(\psi)(\theta - \theta_{\rm eq}) \right\} d\boldsymbol{x}\,. \tag{4.29}$$

The function $g(\psi)$ is defined in such a way that the first and second derivative of g both vanish at the minima of $f(\psi)$. In this case, the model equations (4.28) are not derived from a single free-energy potential. This formulation has the advantage that it allows one to choose the functions $g(\psi)$ and $h(\psi)$ independently.

Note that Eq. (4.29) can be seen as a modification of the simple free-energy functional of Eq. (4.2) that we have used to derive the Allen-Cahn equation, with the free energy $f(\psi)$ that has been replaced by the temperature-dependent function

$$\tilde{f}(\psi, \theta) = f(\psi) + \lambda g(\psi)(\theta - \theta_{\rm eq})\,. \tag{4.30}$$

The free-energy potential defined in Eq. (4.30) has the form of a double-well, with the relative height of the minima associated to the two phases that now depends on the temperature, as we can see from Fig. 4.27.

In the following, we will use the definitions of the functions f, g and h given by Karma and Rappel in Ref. [159] for the crystallization of an undercooled melt

$$f(\psi) = -\frac{1}{2}\psi^2 + \frac{1}{4}\psi^4, \qquad g(\psi) = \psi - \frac{2}{3}\psi^3 + \frac{1}{5}\psi^5, \qquad h(\psi) = \psi\,. \tag{4.31}$$

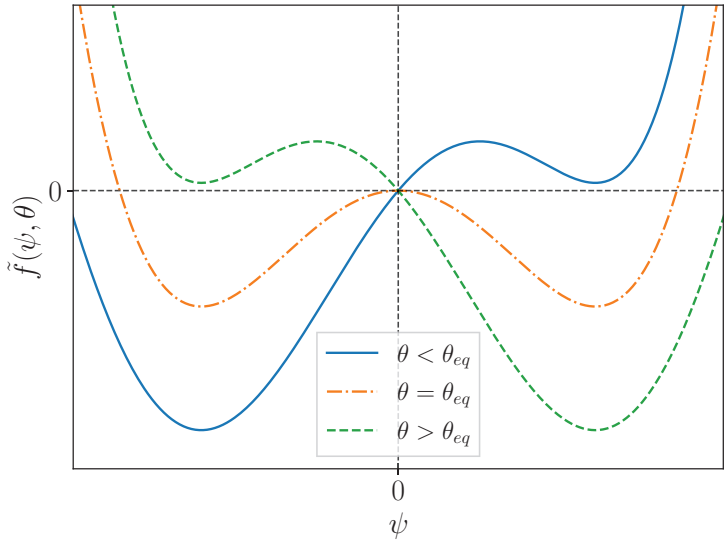


Figure 4.27: Sketch of the free energy in Eq. (4.30) above, below and at the melting temperature θ_{eq}. The relative height of the two minima is temperature-dependent.

Inserting these functions in Eqs. (4.28) and (4.29), we obtain

$$\begin{aligned} \frac{\partial \psi}{\partial t} &= \nabla^2 \psi + \psi - \psi^3 + \lambda(\theta - \theta_{\text{eq}})(1 - \psi^2)^2 \,, \\ \frac{\partial \theta}{\partial t} &= \nabla^2 \theta - \frac{1}{2} \frac{\partial \psi}{\partial t} \,. \end{aligned} \tag{4.32}$$

Note that there are two parameters in this model that we can change to modify (and potentially control) the evolving morphology of the system: the coupling parameter λ and the equilibrium temperature θ_{eq}. What we will do with this model is to consider different values of these parameters in order to explore the diverse morphologies of the growing system. We will then apply a geometric analysis tool to classify patterns.

4.4.1 Morphological instabilities and pattern formation

We will now present a simplified explanation of the interfacial instability which drives the pattern-forming process during solidification. The understanding of this mechanism is due largely to Mullins and Sekerka [190] who were the first, in 1963, to perform systematic linear-stability analyses and to point out the underlying kinetic nature of the process. We will propose a hand-waving argument, the reader interested in a formal analysis of this phenomenon can find the corresponding equations in Ref. [191], for example.

Consider a flat advancing interface of a solid growing in an undercooled melt and assume that a weak perturbation is present. If the perturbation has a wavelength that is above a certain critical value, then it will be self-amplifying, so that small bulges that form randomly on the solidification front will grow rapidly into thin fingers.

Intuitively, the process works as follows. When a liquid freezes, it releases latent heat. For example, ice and water can both exist at 0°C, but the water can freeze into ice only after it has given up latent heat. The rate of freezing depends on how quickly heat can be conducted away from the advancing edge of the solid. This in turn depends on how steeply the temperature

drops from that in the liquid close to the solidification front to that in the liquid further away, as expressed by Fourier's law

$$\boldsymbol{q} = -k\nabla T, \tag{4.33}$$

where $\boldsymbol{q}$ is the heat flux density and k is the material's thermal conductivity. In other words, the steeper the temperature gradient ∇T, the higher the rate of heat transfer.

If a small bulge is formed at random on an otherwise flat solidification front, the temperature gradient becomes steeper around the bulge than elsewhere, because the temperature drops over a shorter distance, as depicted in Fig. 4.28. As a consequence, latent heat is carried away from the bulge more rapidly than it is to either side, and the bulge grows faster. This in turn sharpens the tip and speeds up its growth even more.

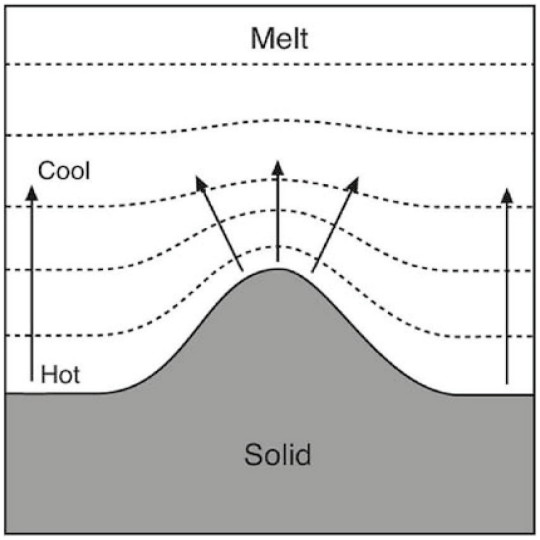


Figure 4.28: The Mullins-Sekerka instability makes protrusions at the surface of a solidifying material unstable. Because the temperature gradient (shown here as dashed contours of equal temperature) is steeper at the tip of the protrusion, heat is conducted away faster and so solidification proceeds more rapidly here. Image from Ref. [192].

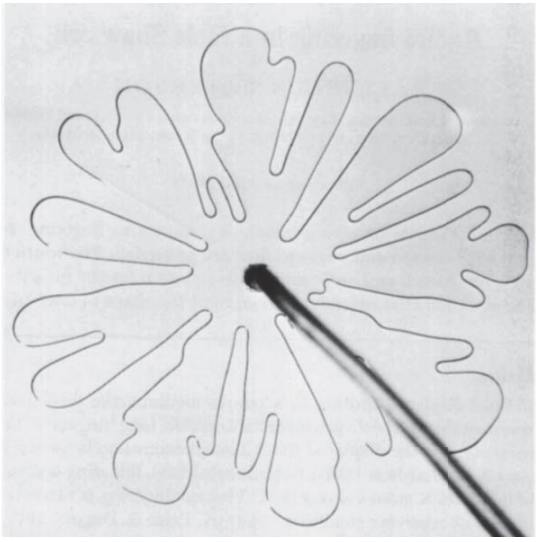


Figure 4.29: The Saffman-Taylor instability is clearly visible in the pattern which occurs after the injection of air into a Hele-Shaw cell filled with glycerine. Image from Ref. [193].

This instability tends to amplify any irregularity on the solid front into a growing finger, no matter how small it is. But there is another factor that sets a minimum limit to the width of the fingers. The interface between the solid and the liquid has a surface tension. The existence of surface tension means that an interface costs energy: the bigger the surface area, the higher the energetic cost. Surface tension is therefore a driving force that tends to reduce surface area. Hence, it tends to "pull" the solidification front back to its flat configuration. This stabilizing effect is similar to that of motion by curvature discussed in Section 4.1. Thanks to this smoothing effect, surface tension suppresses bulges smaller than a certain size. This means that the Mullins-Sekerka instability produces a characteristic branch-tip width, set by the competition between the narrowing of tips caused by positive feedback and their cost in surface energy. In other words, the front develops fingers with a typical wavelength, determined by a balance of opposing factors.

This process is entirely analogous to the morphological instability that occurs when a low-viscosity fluid, such as air, forces its way under pressure into a viscous medium, such as glycerine, as a branching bubble (see Figs. 4.1c and 4.29). The origin of this kind of branching instability, called viscous fingering, was identified in 1958 by Saffman and Taylor [194]. The edge of the air bubble moves forward into the glycerine because the pressure in the air just behind the interface is greater than that in the glycerine just in front of it. The speed at which the interface advances depends on how steep this pressure gradient is. This is analogous to the role of the temperature gradient in the Mullins-Sekerka instability.

This kind of interfacial instabilities can be reproduced by the model C defined in Eq. (4.32). The typical lengthscale of the features emerging from the instability will depend on the value of the parameters λ and $\theta_{\rm eq}$ of the model. We have simulated the dynamics using the implicit Euler (IE) integration scheme [195], which we implemented with the C programming language for high computational efficiency. In Fig. 4.30 we show some of the different patterns that the model C can produce, starting from an initial condition of a circular domain in the center of a periodic simulation box, with low-amplitude random noise added on the interface in order to trigger the instability.

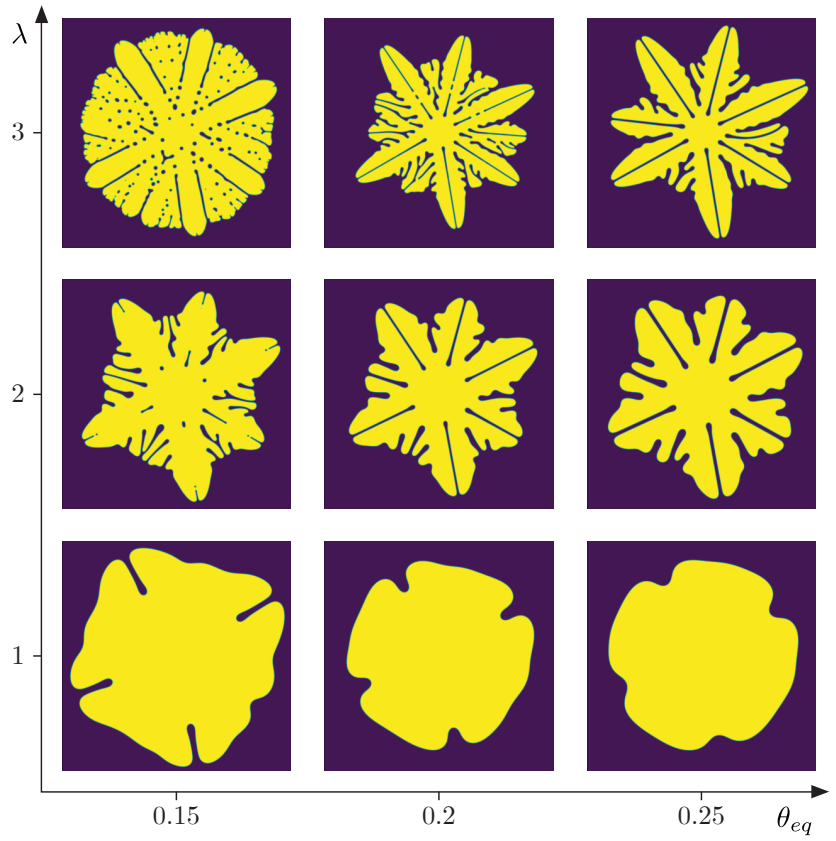


Figure 4.30: Some of the different morphologies that the model C of Eq. (4.32) can produce, for different values of the parameters λ and $\theta_{\rm eq}$. Yellow and purple domains correspond to $\psi = 1$ and $\psi = -1$, respectively.

The initial conditions correspond to a constant temperature larger than $\theta_{\rm eq}$. Periodic boundary

conditions are used in the simulation box. Such finite-size systems are analogous to closed systems (e.g. a cryostat for temperature or a closed cavity inside a porous material for solution growth). Due to this finite size, growth is not permanent and stops at some point. Morphological diagrams like the one shown in Fig. 4.30 are well known and have been studied thoroughly in the literature, see, for example, Ref. [196].

4.5 Matching of similar 2d patterns

In the previous section where we studied the morphologies emerging from the 1d Allen-Cahn equation, we made use of approximation methods based on artificial neural networks. Here, we are once again dealing with a continuous interface with an infinite number of degrees of freedom and an approximation strategy is needed. Instead of following the same lines as in the previous section, we choose to explore a different route for approximation based on dimensional reduction. In order to do so, we have used the curvature scale-space (CSS) method which allows one to represent a one-dimensional interface by a set of a few points on a square. Here, we leave the coupling of this approximation to a control algorithm as a perspective, and we focus only on the validation of the CSS, which is proved via the ability of identifying similarities between different shapes.

We produce a dataset of 200 different shapes obtained by independent simulations of the dynamics of model C, with a space discretization bin $dx = 1$, timestep $dt = 1$, a square simulation box of side $L = 400$, and a total simulation time $t_f = 3000$. The parameters θ_{eq} and λ are fixed respectively to 0.4 and 3.5, but each simulation has a slightly different initial condition. In particular, we change the *seed* to produce the random noise added to the interface of the initial condition, a circular domain of radius 10 in the center of a square simulation box. Some examples of shapes obtained by this procedure are given in Fig. 4.31, where we show only the interface identified by the level set $\psi = 0$, obtained through a post-simulation analysis done with Python.

Shape matching methods usually rely on a reduced representation of the original shape, so that the important characteristics of the morphology are preserved. The word "important" has different meanings for different applications. In our case, we are interested in matching shapes that exhibit similar local features, rather than global similarities. For example, the structure of the most pronounced protruding fingers, rather than their exact position along the interface.

The first step in doing shape classification is shape description, in which a *descriptor* vector (also called a *feature* vector) is generated from a given shape. The goal of description is to characterize the shape using its feature vector. The required properties of a shape description scheme are invariance to translation, homothetic scaling, and rotation. This is required because these three transformations, by definition, do not change the shape of the object.

4.5.1 Curvature scale-space (CSS) representation

The reader interested in a complete survey of existing approaches for shape-based feature extraction can refer to Ref. [197]. For our purpose, we use a method called curvature scale-space (CSS) representation, originally proposed in 1986 by Mokhtarian [198, 199]. We choose this method because, in addition to being robust with respect to scale, position and orientation changes of the objects, it is also robust with respect to noise and local shape deformations.

The CSS representation is a reduced description of a planar curve that is invariant under dilatations and rotations and is well-adapted to the description of fingers and invaginations. The method is explained in detail in Refs. [198, 200], here we briefly summarize the main ideas to obtain a CSS representation and to use it to classify similar shapes.

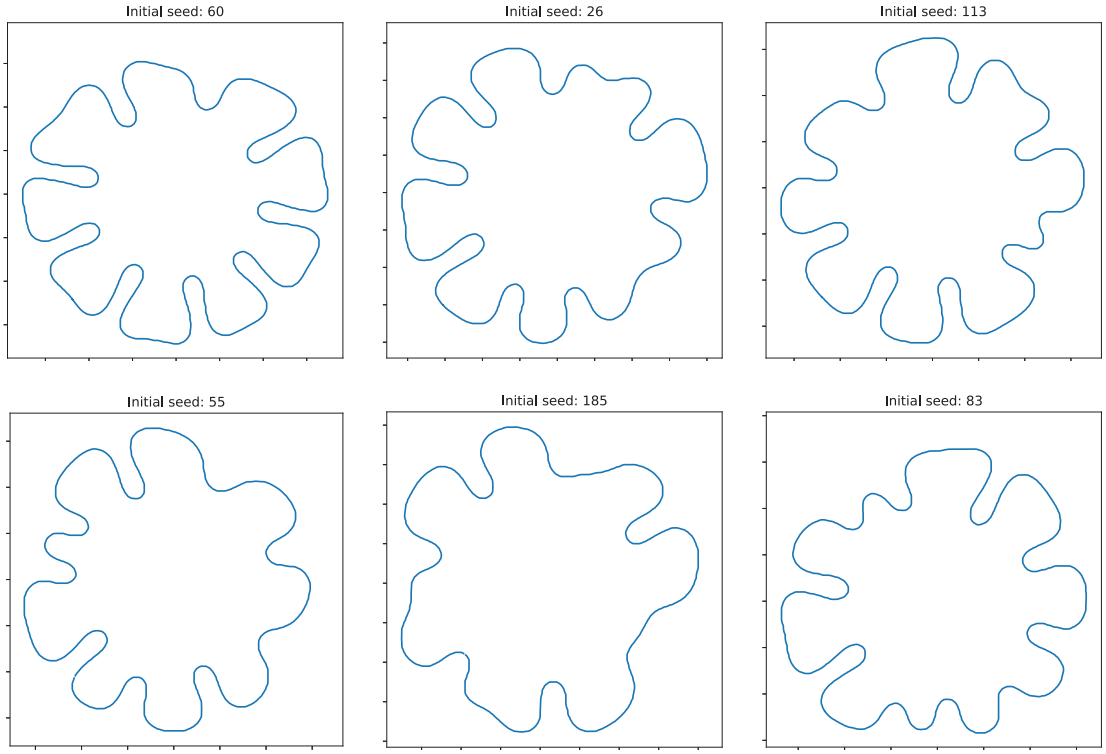


Figure 4.31: Six examples of shapes extracted from the dataset of 200 shapes that we have used for shape matching. Every shape corresponds to a different seed of the random number generator used to add a noise on the interface of the initial circular domain. Only the level set $\psi = 0$ is shown and not the full phase field.

This method is based on computing the curvature of the shape at varying levels of detail, that is, for varying degrees of *smoothing* of the curve. The first step, hence, consists in obtaining a parametrization of the curve which makes it possible to calculate the curvature. Such a parametrization is made possible by considering an arclength variable u along the curve Γ and expressing the curve in terms of two functions $x(u)$ and $y(u)$:

$$\Gamma(u) = \{x(u),\, y(u)\}\,, \tag{4.34}$$

where u is the normalised arclength, ranging over the interval $[0, 1]$. For a closed curve, $x(u)$ and $y(u)$ are periodic functions.

Then, in order to obtain versions of the the curve at varying levels of detail, functions $x(t)$ and $y(t)$ are convolved with a one-dimensional Gaussian kernel $g(u, \sigma)$ of width σ:

$$g(u, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{u^2}{2\sigma^2}\right)\,. \tag{4.35}$$

A *smoothed* version $\Gamma_\sigma(u)$ of Γ is defined by

$$\Gamma_\sigma(u) = \{x(u) * g(u, \sigma),\, y(u) * g(u, \sigma)\}\,, \tag{4.36}$$

where the symbol $*$ denotes the convolution operation

$$x(u) * g(u, \sigma) = \int_{-\infty}^{\infty} x(z)\, g(u - z, \sigma)\, dz\,. \tag{4.37}$$

The locations of the points of zero curvature κ along the curve are determined, as shown in Fig. 4.32, at different levels of smoothing (or *scale*).

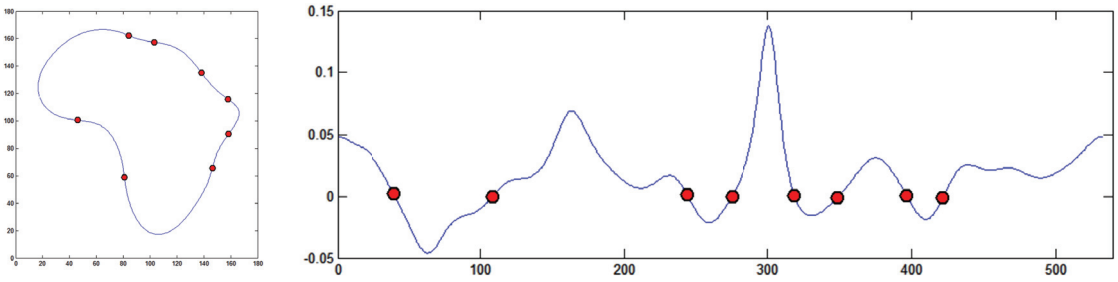


Figure 4.32: The curvature of the contour on the left is plotted on the right, as a function of the arclength parameter. The red dots correspond to the zero-curvature points.

As σ increases, the curve $\Gamma_\sigma(u)$ shrinks and becomes smoother, and the number of zero-curvature points on it decreases. Finally, when σ is sufficiently high, $\Gamma_\sigma(u)$ will be a convex curve with no curvature zero-crossings, as shown in Fig. 4.33.

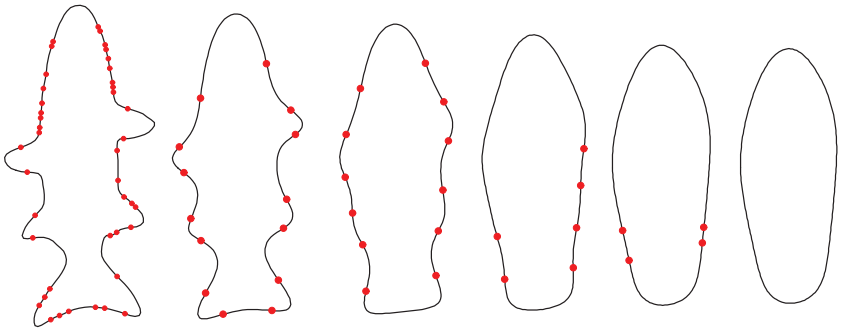


Figure 4.33: Shrinkage and smoothing of the curve and decreasing of the number of curvature zero-crossings during the evolution, from left: $\sigma = 1, 4, 7, 10, 12, 14$. Image from Ref. [201].

If the locations of the zero-curvature points of every $\Gamma_\sigma(u)$ during the smoothing procedure are determined, then the resulting points can be displayed in the (u, σ) plane. The result of this process is the CSS representation of the curve (see Fig. 4.34). More precisely, the CSS representation is the set of zero-curvature points $\{(u, \sigma) \mid \kappa(u, \sigma) = 0\}$.

As seen in Fig. 4.33, there are two curvature zero-crossings on every concave or convex part of the shape, and as the curve becomes smoother, these points approach each other and create a curve in the CSS representation of the shape, reported in Fig. 4.34.

When the protrusion or concavity disappears because of the smoothing process, the two points join and give rise to the maximum of the curve. The height of this curve then reflects the depth and size of a convex or a concave part of the initial shape. In other words, a curve maximum in the CSS representation corresponds to a local feature of the shape.

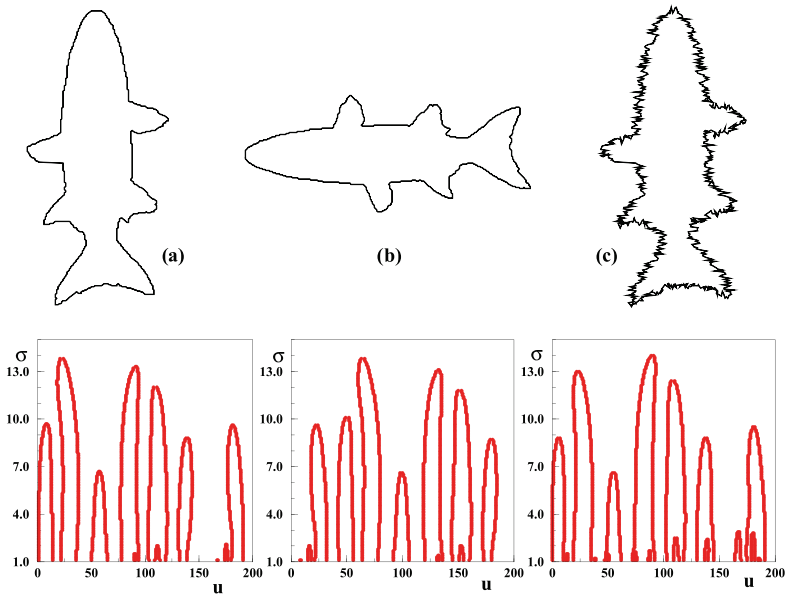


Figure 4.34: (a) A boundary and its CSS diagram. (b) Change in orientation causes a shift in the diagram. (c) Noise creates small curves at small σ. Image from Ref. [201].

To do shape matching, only the maxima of the CSS representation are retained. Small contours of the representation are related to noise or small ripples of the curve, hence, in order to simplify and speed up matching, small maxima are usually not included. This representation has the advantage of being compact, in the sense that a shape is represented by a relatively small number of points in the (u, σ) plane, and this allows to use a matching algorithm to compare two representations which is simple and fast.

We have used the algorithm described in Ref. [202], which compares two CSS representations and assigns a matching value as the measure of similarity between the shapes. The comparison procedure is based on superimposing the two representations, trying to match first the CSS maxima at higher sigma values, and then the others, and then calculating a matching value related to the distances between corresponding maxima in the two representations. The results of this analysis on the dataset of 200 shapes produced with model C are reported in the following section.

4.5.2 Results

In Fig. 4.35 we show the two best matches out of the dataset of 200 shapes (similar to those of Fig. 4.31), as found by the matching algorithm, together with their corresponding CSS maxima (on the left). As we can see, the similarity of the two shapes is reflected in the structure of their CSS maxima, especially at high values of σ, which correspond to the most prominent features.

A physical law for smoothing

The smoothing process used to obtain the CSS representation is based on the convolution with a Gaussian kernel, as explained above. However, we saw at the beginning of Section 4.1 a physical law that naturally leads to the smoothing of the interface, the mean curvature flow Eq. (4.1).

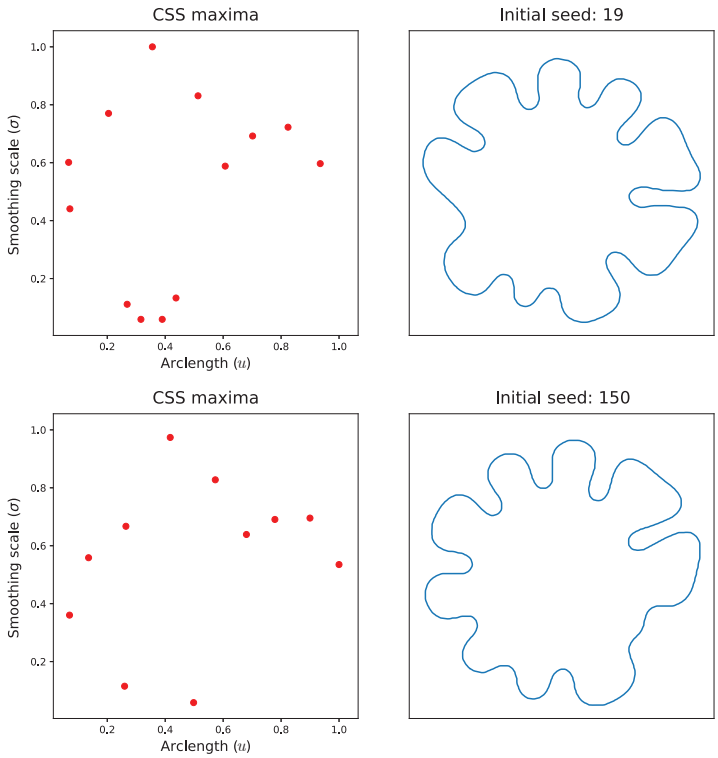


Figure 4.35: The two most similar shapes (on the right) and their CSS diagram (on the left) out of the dataset of 200 shapes, as found by the CSS matching algorithm.

Thus, we tried to use this evolution law to replace the convolution with a Gaussian kernel and obtain a CSS representation.

We have seen that the Allen-Cahn equation (4.5) converges to the sharp-interface model of mean curvature flow when the interface width is much smaller than the radius of curvature. Hence, it is sufficient to decouple the two fields ψ and θ in model C by setting the coupling parameter λ to zero in order to obtain dynamics governed by the mean curvature flow model, as shown in Fig. 4.36. In this case, the smoothing parameter is replaced by the physical integration time.

Figure 4.36: Simulation of model C with the coupling parameter $\lambda = 0$. The initial condition (on the left) corresponds to a typical shape obtained with $\lambda > 0$. The interface smooths out according to motion by curvature dynamics. The other parameters are: $dx = 0.6$, $dt = 0.01$, $L = 250$.

In Fig. 4.37 we show a comparison between the CSS representation obtained with mean curvature flow (left) and with the usual Gaussian smoothing (right), for the same shape, which is shown to the side. We can see that, in fact, smoothing done with the mean curvature flow produces a CSS representation that is complete and could potentially be used to do shape matching. In addition, this representation is very different from the one obtained with Gaussian smoothing, in terms of the relative heights of the maxima and the topology of the contours.

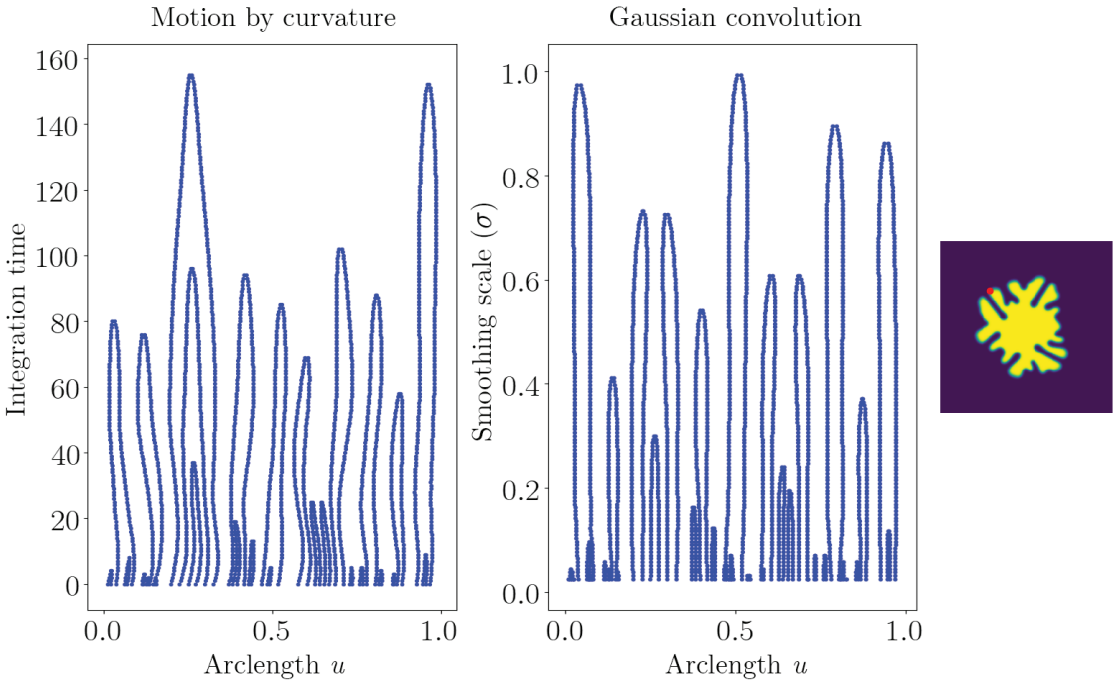


Figure 4.37: Comparison of the CSS representation obtained with motion by curvature and with the standard Gaussian convolution, for the contour on the right. The red dot corresponds to the starting point of the arclength parameter $u = 0$.

Although the mean curvature flow can be used to obtain a CSS representation of a shape, this method is much more computationally demanding than the classic Gaussian smoothing, because it requires the temporal integration of an equation in two dimensions and because the interface corresponding to the level set $\psi = 0$ must be extracted at each smoothing step. This also has the disadvantage that we have to keep track of the reference point $u = 0$, which is not the case with the smoothing with the Gaussian kernel.

4.5.3 Discussion and perspectives

We have seen how model C dynamics can reproduce the morphological instabilities typical of the growth of a crystal from a melt (Mullins-Sekerka instability) or the expansion of a bubble in a Hele-Shaw cell (Saffman-Taylor instability). By randomizing the initial conditions of the model, we have constructed a dataset of 200 different shapes, and we have then used a well-known method of shape analysis called CSS representation to match the two most similar shapes in the whole dataset.

The CSS representation could also be seen as a reduced and approximate representation of a

state resulting from the dynamics of model C. An interesting perspective of this work, then, would be to couple this type of representation to a learning algorithm similar to the one presented in the previous section for the 1d Allen-Cahn equation and attempt to control model C morphologies.

We also tried to use the mean curvature flow equation 4.1 as a smoothing process to obtain a CSS representation, and found that, although this technique can indeed be used, it cannot replace the Gaussian smoothing provided by the classical CSS method due to its low computational efficiency. However, an advantage of motion by curvature is that it could be implemented directly in an experiment. For example, an experiment could be switched from a diffusion-limited growth regime described by model C which produces fingers, to an equilibrium interface-kinetics regime described by the Allen-Cahn equation which leads to motion by curvature. The CSS representation could then be obtained from image analysis. Furthermore, the analogy between convolutional smoothing and motion by curvature raised some interesting questions about how these two reduced representations differ for the same a interface shape.

CHAPTER 5

Conclusion

In this thesis, we presented a theoretical study of shape control in low-dimensional physical systems involving stochastic or deterministic dynamics. We studied the problem of achieving an arbitrary morphology in finite time by varying a global (macroscopic) control parameter during the evolution of the system. The control approach that we have considered relies on the observation of the system during the dynamics. Changes to the global parameter are made as a function of the state of the system. This approach is called closed-loop.

The first case study we addressed is the control of small two-dimensional clusters evolving by thermally-activated edge diffusion, with or without the presence of an external macroscopic field. Resorting on a well-established lattice model, we have shown that, in the absence of an external field, there is an optimal temperature for which the time to reach a specific cluster shape is minimum. We studied which factors most influence this behaviour and predicted that the presence of an optimal temperature depends on the compactness of the cluster.

In the presence of an external field that biases edge diffusion, we computed the optimal closed-loop strategy to set the field as a function of the observed state of the system and found that the time to reach the target shape can be significantly decreased, with a gain that grows when increasing cluster size or decreasing temperature.

Using this discrete model, we also performed a computational analysis of an experimental situation, in which we used model-free (i.e. reinforcement learning) control algorithms, which do not require any knowledge of the governing laws of the system. Based on the results of this and the previous analysis, we identified and proposed a number of physical systems in which our findings could be tested. These systems correspond primarily to colloidal clusters under the effect of an external field, e.g. an electric or magnetic field or a temperature gradient.

The second model we considered is a one-dimensional continuous model based on a generalisation of the Allen-Cahn equation, typically used to describe phase transitions. In this case, we chose to use as a control variable the parameter of the model that is associated with the distance from the critical point of the phase transition, e.g. the temperature when the system is close to the critical temperature. Within this framework, we have investigated the capabilities of an advanced model-free control method based on deep neural networks, and found that this technique is able to learn non-trivial strategies to control the number of phase domains in the system.

We have also considered another continuous model, in two dimensions, that can describe a wide variety of growth phenomena. With this model we have generated a dataset of two-dimensional closed shapes and successfully tested the capabilities of a shape-matching algorithm based on the analysis of the curvature of the boundary. The promising results obtained with this analysis are preparatory to the morphological control of growth phenomena.

During the course of this thesis project, several internship students collaborated with us, making important contributions to our research. In particular:

- Baptiste Filoche worked on motion by curvature as a smoothing dynamics to obtain a curvature scale space (CSS) representation of a 2d shape. He developed and implemented in C a method for the contour extraction of an interface described by a 2d phase field, and wrote a Python script to obtain the CSS representation based on motion by curvature. Parts of his results are reported and commented in Section 4.5.2.

- Alexandre Mass studied a technique to discretise a continuous state of spaces called *tile coding*. He implemented this method in Python and tested it on a famous reinforcement learning task, known as the "mountain car problem". His contribution was important to better understand this approximation technique, which we initially considered for our control problem in continuous systems.

- Christopher Greenberg Bonilla worked on the extension of the discrete lattice model described in Section 2.1 to cluster breaking, a topic which is briefly discussed in Section 2.5. He used a modified version of the Python implementation of the model that also allows for the breaking of the cluster, and obtained some interesting preliminary results about the frequency of breaking as a function of the temperature.

- Jingyu Wang and Qianlong He focused on an analytical problem which is closely related to the control of the Allen-Cahn equation presented in Section 4.2.1. This problem consists in controlling a nonlinear equation of the form $\partial\psi/\partial t = C(t)\psi - \psi^3$, which is the universal amplitude equation for a *pitchfork bifurcation*. Differently from what we did in this thesis, where we used a model-free approach, they explored the possibility of using model-based control on this equation.

- Jules Vanaret worked on the control of the Allen-Cahn equation using a DQN agent. All the results reported in Section 4.3 were obtained with Python codes written by him. His contribution was crucial, as without his knowledge related to deep neural networks we would not have been able to deal with such a problem. Jules wrote all the codes to implement the DQN agent, the one to integrate the dynamics of the Allen-Cahn equation, and a series of scripts to analyse the data obtained.

We report now a list of the main Python and C codes that have been written for the simulations and the data analysis that have been performed during this work (when not specified, the language used is Python):

- Implementation of the discrete lattice model for fluctuating clusters
- Implementation of value iteration
- Implementation of iterative policy evaluation
- Implementation of Monte Carlo control
- Implementation of Q-learning
- KMC code to simulate the dynamics of fluctuating clusters
- Code to generate all possible configurations of a polyomino and classify them by symmetry

- Code to plot a dynamical graph of a cluster
- Code to count number of degenerate states
- Code to count temperature transitions of a policy
- Code to check approximations of M_0 and M_{ϕ}
- Implementation of Implicit Euler scheme (in C) to simulate the dynamics of model C
- Implementation of ETDRK2 scheme (in C and Python) to simulate the dynamics of the Allen-Cahn equation
- Implementation of the CSS extraction and matching algorithm (in C and Python)
- Implementation of a DQN-based learning algorithm (written by Jules Vanaret)
- Code to extract the profile from a 2d phase field and obtain a CSS representation based on motion by curvature (written in C and Python by Baptiste Filoche)

In all these codes an extensive use of the Python libraries `numpy` and `matplotlib` has been made. The DQN-based algorithm has been implemented using the GPU-accelerated deep learning library `pytorch`. To plot dynamical graphs of the clusters we have used the `pygraphviz` library.

The analysis on model-based control and equilibrium dynamics of particle clusters presented in Chapter 2 has been reported in a paper, which is currently under review. During the course of this PhD thesis we also spent some time finalising another paper, which reports the results of my master internship on the influence of elastic strain on solid-state dewetting. This paper has been published in Physical Review Letters in January 2022:

F. Boccardo, F. Rovaris, A. Tripathi, F. Montalenti, and O. Pierre-Louis. "Stress-Induced Acceleration and Ordering in Solid-State Dewetting". *Physical Review Letters* 128, 026101 (2022).

This work has been a pioneering experience in a hybrid and strongly emerging field of research, namely that of the application of control theory techniques to physical models. As physicists, we had to roll up our sleeves and learn the basics of model-free and model-based control methods, techniques that belong mainly to the fields of computer science and applied mathematics, and for which we were not well prepared at first. This initial effort, however, paid off, as it allowed us to see well-known models (in particular the lattice model of clusters and the Allen-Cahn equation) from new perspectives, and to ask questions that had not yet been asked about these models.

There are many exciting perspectives that lie ahead. Some can be readily explored with the methods and tools we have developed during this thesis project. For example, the problem of the breaking in particle clusters and its effect on the dynamics of the system and on the performance of control methods. The codes to implement the lattice model and the related model-based and model-free learning algorithms are already set up to account for breaking of the cluster. Others, however, require some modifications, such as controlling lattice models that describe the physics of vacancy clusters, or clusters of particles that interact with more complex rules than the simple bond-breaking model that we have considered (e.g. magnetic or electrostatic interactions).

Another short-term perspective is to identify the right system in which we can experimentally test our findings on cluster control. For this task, we hope that the strong expertise accumulated in colloid science on the fine-tuning of particle-particle interactions and visualisation techniques will allow one to design a suitable experimental system to achieve control of the shape of monolayer clusters.

There are also long-term perspectives, such as considering clusters larger than 12 particles, a limitation that has been imposed by the use of tabular methods. To tackle this problem, one possible option is to use a model-free DQN agent, such as the one we used to control the Allen-Cahn equation, and to couple it to KMC simulations.

Furthermore, our results also opens novel theoretical questions. A first direction is the analytical investigation of the first passage times in the graph of states of cluster configurations.

The large body of literature that has been developed for the analysis of first passage times on random graphs could be useful for this analysis. Another question is to understand more rigorously the properties of the optimal policies to control cluster shapes such as degeneracies and temperature transitions. In addition, very little is known about the mathematical properties of the models discussed in Chapter 4 in the presence of control.

To conclude, we hope that this work will foster new experimental and theoretical directions in the study of shape control in low-dimensional physical systems.

Appendix

Appendix

A.1 Detailed derivation of Bellman equation

In this appendix we report a detailed derivation of Eq. (1.9). We start from the definition of the value function Eq. (1.8):

$$\begin{aligned} v_\pi(s) &= \mathbb{E}_\pi \left[G_t \mid S_t = s \right] = \mathbb{E}_\pi \left[R_{t+1} + G_{t+1} \mid S_t = s \right] \\ &= \mathbb{E}_\pi \left[R_{t+1} \mid S_t = s \right] + \mathbb{E}_\pi \left[G_{t+1} \mid S_t = s \right] . \end{aligned} \tag{A. 1}$$

The first term on the right hand side is simply the expectation value of the reward received when being in state s and following policy π, which we can write explicitly as

$$\mathbb{E}_\pi \left[R_{t+1} \mid S_t = s \right] = \sum_a \pi(a \,|\, s) \sum_{s',r} p(s', r \mid s, a)\, r \,. \tag{A. 2}$$

We can then use the law of total (or iterated) expectation to rewrite the second term. The law of total expectation states that, if X is a random variable whose expected value $\mathbb{E}[X]$ is defined, and Y and Z are random variables on the same probability space, then

$$\mathbb{E}\left[X \mid Z \right] = \mathbb{E}\Big[\mathbb{E}\left[X \mid Y, Z \right] \mid Z \Big] . \tag{A. 3}$$

Applying this rule with $X = G_{t+1}$, $Y = S_{t+1}$, and $Z = S_t$, we obtain

$$\mathbb{E}_\pi \left[G_{t+1} \mid S_t \right] = \mathbb{E}_\pi \Big[\mathbb{E}_\pi \left[G_{t+1} \mid S_{t+1}, S_t \right] \mid S_t \Big] . \tag{A. 4}$$

Since we are assuming that the process is Markovian, G_{t+1}, which is the sum of all the future rewards that the agent receives starting from state S_{t+1}, is independent from the previous state S_t, hence

$$\mathbb{E}_\pi \Big[\mathbb{E}_\pi \left[G_{t+1} \mid S_{t+1}, S_t \right] \mid S_t \Big] = \mathbb{E}_\pi \Big[\mathbb{E}_\pi \left[G_{t+1} \mid S_{t+1} \right] \mid S_t \Big] . \tag{A. 5}$$

Finally, Eq. (A. 1) can be rewritten as

$$\begin{aligned} v_\pi(s) &= \mathbb{E}_\pi\Big[R_{t+1} + \mathbb{E}_\pi\left[G_{t+1} \mid S_{t+1}\right] \mid S_t = s\Big] \\ &= \sum_a \pi(a \mid s) \sum_{s',r} p(s', r \mid s, a)\Big[r + \mathbb{E}_\pi\left[G_{t+1} \mid S_{t+1} = s'\right]\Big] \\ &= \sum_a \pi(a \mid s) \sum_{s',r} p(s', r \mid s, a)\left[r + v_\pi(s')\right] , \end{aligned} \tag{A. 6}$$

which is identical to Eq. (1.9).

A.2 Tabular algorithms

Here we report, in pseudocode, the tabular algorithms for estimating value functions and learning optimal policies that we have used throughout this thesis:

- Iterative policy evaluation in Fig. A.1,
- Value iteration in Fig. A.2,
- Monte Carlo control (with exploring starts and decaying ε) in Fig. A.3,
- Q-learning (with exploring starts and decaying ε) in Fig. A.4.

Note that, in Monte Carlo and Q-learning, the episodes are obtained from KMC simulations, and terminated when reaching the target state or when reaching a maximum number of steps (which has been set to 1000 for the results reported in Chapter 3).

Figure A.1: Iterative policy evaluation

Input: π, the policy to be evaluated
Parameters: Small threshold $\theta > 0$ determining estimation accuracy, discount factor β
Initialize: $V(s)$, for all $s \in \mathcal{S}$, arbitrary except that $V(\bar{s}) = 0$, where $\bar{s}$ is the target state

While $\Delta > \theta$ **:**
  $\Delta \leftarrow 0$
  Loop for each $s \in \mathcal{S}$ **:**
    $v \leftarrow V(s)$
    $V(s) \leftarrow \sum_a \pi(a \mid s) \sum_{s',r} p(s', r \mid s, a)\left[r + \beta V(s')\right]$
    $\Delta \leftarrow \max(\Delta, |v - V(s)|)$

Output: $V \approx v_\pi$

Figure A.2: Value iteration

Parameters: Small threshold $\theta > 0$ determining estimation accuracy, discount factor β
Initialize: $V(s)$ arbitrarily, for all $s \in \mathcal{S}$, arbitrary except that $V(\bar{s}) = 0$, where $\bar{s}$ is the target state

While $\Delta > \theta$ **:**
  $\Delta \leftarrow 0$
  Loop for each $s \in \mathcal{S}$ **:**
    $v \leftarrow V(s)$
    $V(s) \leftarrow \max_a \sum_{s',r} p(s', r \mid s, a)\left[r + \beta V(s')\right]$
    $\Delta \leftarrow \max(\Delta, |v - V(s)|)$

Loop for each $s \in \mathcal{S}$ **:**
  $\mathcal{A}^* \leftarrow \operatorname{argmax}_a \sum_{s',r} p(s', r \mid s, a)\left[r + \beta V(s')\right]$
  $\pi(a \mid s) \leftarrow I_{\mathcal{A}^*}(a)$

Output: $\pi \approx \pi_*$, $V \approx v_*$

Figure A.3: Monte Carlo with exploring starts and decaying ε

Parameters:
Initial value ε_i and final value ε_f for ε
Number of learning episodes n_{epi}
Discount factor β

Initialize:
$\pi \leftarrow$ an arbitrary stochastic policy
$Q(s,a)$ arbitrarily, for all $s \in \mathcal{S}$, $a \in \mathcal{A}$
$Returns(s,a) \leftarrow$ empty list, for all $s \in \mathcal{S}$, $a \in \mathcal{A}$
Decay factor $\delta\varepsilon = (\varepsilon_f - \varepsilon_i)/n_{\text{epi}}$

Loop for each episode **:**
Choose $S_0 \in \mathcal{S}$, $A_0 \in \mathcal{A}$ randomly such that all pairs have probability > 0
Generate an episode starting from S_0, A_0 following π: $S_0, A_0, R_1, \dots, S_{t_f-1}, A_{t_f-1}, R_{t_f}$
$G \leftarrow 0$
Decay ε linearly: $\varepsilon \leftarrow \varepsilon - \delta\varepsilon$
Loop for each step of episode, $t = t_{f-1}, t_{f-2}, \dots, 0$ **:**
$G \leftarrow \beta G + R_{t+1}$
Unless the pair S_t, A_t appears in $S_0, A_0, S_1, A_1, \dots, S_{t-1}, A_{t-1}$:
Append G to $Returns(S_t, A_t)$
$Q(S_t, A_t) \leftarrow \text{average}(Returns(S_t, A_t))$
$\mathcal{A}^* \leftarrow \text{argmax}_a\, Q(S_t, a)$
For all $a \in \mathcal{A}$: $\pi(a \mid S_t) \leftarrow \begin{cases} 1 - \varepsilon + \varepsilon/|\mathcal{A}| & \text{if } a = \mathcal{A}^* \\ \varepsilon/|\mathcal{A}| & \text{otherwise} \end{cases}$

Output: $\pi \approx \pi_*$

Figure A.4: Q-learning with exploring starts and decaying ε

Parameters:
Initial value ε_i and final value ε_f for ε
Number of learning episodes n_{epi}
Learning rate α
Discount factor β

Initialize:
$\pi_{\text{exp}} \leftarrow$ an arbitrary stochastic policy (used for exploration)
$Q(s,a)$ arbitrarily, for all $s \in \mathcal{S}$, $a \in \mathcal{A}$
Decay factor $\delta\varepsilon = (\varepsilon_f - \varepsilon_i)/n_{\text{epi}}$

Loop for each episode **:**
Choose initial state S randomly such that all states have probability > 0
Loop for each step of episode, up to termination **:**
Choose action A using exploration policy π_{exp} derived from Q (e.g. ε-greedy)
Take action A, observe R, S'
$Q(S,A) \leftarrow Q(S,A) + \alpha\,[R + \beta \max_a Q(S',a) - Q(S,A)]$
$S \leftarrow S'$
Decay ε linearly: $\varepsilon \leftarrow \varepsilon - \delta\varepsilon$

Loop for each $s \in \mathcal{S}$ **:**
$\mathcal{A}^* \leftarrow \text{argmax}_a\, Q(s,a)$
$\pi(a \mid s) \leftarrow I_{\mathcal{A}^*}(a)$

Output: $\pi \approx \pi_*$

A.3 Double DQN algorithm

In Fig. A.5 we report, in pseudocode, the double deep Q-learning algorithm that we have used for the control of the Allen-Cahn equation. The value of the parameter t_{act} and the reward function $\varrho(s)$ have been changed depending on the three different targets considered (see main text). The other parameters, instead, have been fixed to the following values for all the three targets: $n_{\text{epi}} = 10^5$, $n_{\text{batch}} = 64$, $n_{\text{net}} = 3000$, $\varepsilon_i = 1$, $\varepsilon_f = 0.1$, $\alpha = 10^{-3}$, $\beta = 0.9$.

The $Mask$ variable is used to differentiate the successful episodes from the unsuccessful ones. The update rule for the exponential decay of ε has been established empirically. The episodes are terminated when reaching the target state or when reaching a maximum number of steps.

Figure A.5: Double deep Q-learning applied to the control of the Allen-Cahn equation

Parameters:
- Time between two actions t_{act}
- Reward function $\varrho(s)$
- Number of learning episodes n_{epi}
- Size n_{batch} of the learning mini-batch
- Number of episodes between setting the two DQNs alike n_{net}
- Initial value ε_i and final value ε_f for ε
- Learning rate α
- Discount factor β

Initialize:
- Memory for experience replay $\mathcal{M}$
- Online DQN $Q(s, a, \boldsymbol{w})$ and target DQN $\hat{Q}(s, a\boldsymbol{w}^-)$, with $\boldsymbol{w} = \boldsymbol{w}^-$

$n \leftarrow 0$

Loop for each episode **:**
- $t \leftarrow 0$
- $\varepsilon \leftarrow \varepsilon_i$
- Choose a random initial condition and initialize state S_0
- **Loop** for each step of episode, up to termination **:**
  - Choose action A_t using exploration policy derived from $Q(s, a, \boldsymbol{w})$ (e.g. ε-greedy)
  - Set $C = A_t$ in Allen-Cahn equation and integrate the dynamics during t_{act}
  - Observe new state $S_{t+t_{\text{act}}}$ and calculate reward: $R_{t+t_{\text{act}}} \leftarrow \varrho(S_{t+t_{\text{act}}})$
  - **If** $R_{t+t_{\text{act}}}$ is 100 **:** $Mask \leftarrow 0$
  - **Else:** $Mask \leftarrow 1$
  - Store transition $(S_t, A_t, R_{t+t_{\text{act}}}, A_{t+t_{\text{act}}}, Mask)$ in memory buffer $\mathcal{M}$
  - Sample random mini-batch of transitions $(S_j, A_j, R_{j+t_{\text{act}}}, S_{j+t_{\text{act}}}, Mask)$ of size n_{batch} from $\mathcal{M}$
  - $Y_j^Q \leftarrow R_j + Mask\ \beta\, \hat{Q}\left[S_{j+t_{\text{act}}}, \text{argmax}_a\, Q(S_{j+t_{\text{act}}}, a, \boldsymbol{w}_t), \boldsymbol{w}_t^-\right]$
  - Perform gradient descent on $\left[Y_j^Q - Q(S_j, A_j, \boldsymbol{w}_t)\right]^2$ with respect to $\boldsymbol{w}_t$
  - Decay ε exponentially: $\varepsilon \leftarrow \varepsilon_f + (\varepsilon_i - \varepsilon_f)\exp(-3n/n_{\text{epi}})$
  - Every n_{net} episodes, update target network: $\boldsymbol{w}_t^- \leftarrow \boldsymbol{w}_t$
  - $t \leftarrow t + t_{\text{act}}$
- $n \leftarrow n + 1$

Output: Trained DQN $Q(s, a, \boldsymbol{w})$

A.4 Detailed derivation of $M_{\boldsymbol{\phi}}$

In this appendix we show in detail how to manipulate and rewrite Eq. (2.28) in order to obtain Eq. (2.29). Equation (2.28) reads

$$\begin{aligned} M_{\boldsymbol{\phi}}(\bar{s}) =& \frac{1}{d_{\bar{s}}} \sum_{s\in\mathcal{B}_{\bar{s}}} (n_{\bar{s}s} - \varphi(\bar{s})\, u_{\bar{s}s}) + \frac{1}{S_N - 1} \sum_s \tau_\infty(s,\bar{s}) \sum_{s'\in\mathcal{B}_s} (n_{ss'} - n_{s's}) \\ &+ \frac{1}{S_N - 1} \sum_s \tau_\infty(s,\bar{s}) \sum_{s'\in\mathcal{B}_s} (\varphi(s')\, u_{s's} - \varphi(s)\, u_{ss'})\,. \end{aligned} \tag{A. 7}$$

The first term can be rewritten as

$$\frac{1}{d_{\bar{s}}} \sum_{s\in\mathcal{B}_{\bar{s}}} (n_{\bar{s}s} - \varphi(\bar{s})\, u_{\bar{s}s}) = \langle n_{\bar{s}s}\rangle_{s\in\mathcal{B}_{\bar{s}}} - \varphi(\bar{s})\langle u_{\bar{s}s}\rangle_{s\in\mathcal{B}_{\bar{s}}}\,. \tag{A. 8}$$

Then, since when summing over all states $s \in \mathcal{S}$ and all states $s' \in \mathcal{B}_s$ the indexes s and s' can be exchanged, we can rewrite the second and third term as

$$\begin{aligned} &\frac{1}{S_N - 1} \sum_s \tau_\infty(s,\bar{s}) \sum_{s'\in\mathcal{B}_s} (n_{ss'} - n_{s's}) + \frac{1}{S_N - 1} \sum_s \tau_\infty(s,\bar{s}) \sum_{s'\in\mathcal{B}_s} (\varphi(s')\, u_{s's} - \varphi(s)\, u_{ss'}) = \\ &= \frac{1}{S_N - 1} \sum_s \sum_{s'\in\mathcal{B}_s} \tau_\infty(s,\bar{s})\, [n_{ss'} - n_{s's} + \varphi(s')u_{s's} - \varphi(s)u_{ss'}] \\ &= \frac{1}{S_N - 1} \sum_s \sum_{s'\in\mathcal{B}_s} [\tau_\infty(s,\bar{s})n_{ss'} - \tau_\infty(s',\bar{s})n_{ss'} + \tau_\infty(s',\bar{s})\varphi(s)u_{ss'} - \tau_\infty(s,\bar{s})\varphi(s)u_{ss'}] \\ &= \frac{1}{S_N - 1} \left\{ \sum_s \sum_{s'\in\mathcal{B}_s} n_{ss'}\, [\tau_\infty(s,\bar{s}) - \tau_\infty(s',\bar{s})] + \sum_s \varphi(s) \sum_{s'\in\mathcal{B}_s} u_{ss'}\, [\tau_\infty(s',\bar{s}) - \tau_\infty(s,\bar{s})] \right\}. \end{aligned} \tag{A. 9}$$

For any quantity q_s, the two following formulas hold

$$\begin{aligned} \sum_{s'\in\mathcal{B}_s} q_s &= d_s \langle q_s\rangle_{s'\in\mathcal{B}_s}\,, \\ \frac{1}{S_N - 1} \sum_s q_s &= \frac{1}{S_N - 1} \left[q_{\bar{s}} + \bar{\sum_s} q_s \right] = \frac{1}{S_N - 1} q_{\bar{s}} + \langle q_s\rangle_{s\in\bar{\mathcal{S}}}\,, \end{aligned} \tag{A. 10}$$

where we have defined $\bar{\mathcal{S}} = \mathcal{S} \setminus \{\bar{s}\}$ for convenience. Hence, we can regroup the three terms and rewrite them as

$$\begin{aligned} M_{\boldsymbol{\phi}}(\bar{s}) &= \langle n_{\bar{s}s}\rangle_{s\in\mathcal{B}_{\bar{s}}} - \varphi(\bar{s})\langle u_{\bar{s}s}\rangle_{s\in\mathcal{B}_{\bar{s}}} \\ &\quad - \frac{d_{\bar{s}}}{S_N - 1} \langle n_{\bar{s}s}\tau_\infty(s,\bar{s})\rangle_{s\in\mathcal{B}_{\bar{s}}} + \langle d_s \langle n_{ss'}\, (\tau_\infty(s,\bar{s}) - \tau_\infty(s',\bar{s}))\rangle_{s'\in\mathcal{B}_s}\rangle_{s\in\bar{\mathcal{S}}} \\ &\quad + \frac{d_{\bar{s}}}{S_N - 1} \varphi(\bar{s}) \langle u_{\bar{s}s}\tau_\infty(s,\bar{s})\rangle_{s\in\mathcal{B}_{\bar{s}}} - \langle \varphi(s) d_s \langle u_{ss'}\, (\tau_\infty(s,\bar{s}) - \tau_\infty(s',\bar{s}))\rangle_{s'\in\mathcal{B}_s}\rangle_{s\in\bar{\mathcal{S}}}\,, \end{aligned} \tag{A. 11}$$

where we have used the boundary condition $\tau_\infty(\bar{s},\bar{s}) = 0$.

Since

$$\frac{S_N - 1}{d_{\bar{s}}} = \tau^{\mathrm{r}}_\infty(\bar{s}) = \sum_{s\in\mathcal{B}_{\bar{s}}} p_\infty(\bar{s},s)\tau_\infty(s,\bar{s}) = \frac{1}{d_{\bar{s}}} \sum_{s\in\mathcal{B}_{\bar{s}}} \tau_\infty(s,\bar{s}) = \langle \tau_\infty(s,\bar{s})\rangle_{s\in\mathcal{B}_{\bar{s}}}\,, \tag{A. 12}$$

where $p_\infty(\bar{s},s)=\gamma_\infty(\bar{s},s)t_\infty(\bar{s})=1/d_s$ is the infinite-temperature limit of $p_{\boldsymbol{\phi}}(\bar{s},s)$, we can rewrite the first and third term of Eq. (A. 11) as

$$\begin{aligned}
&\langle n_{\bar{s}s}\rangle_{s\in\mathcal{B}_{\bar{s}}}-\frac{d_{\bar{s}}}{S_N-1}\langle n_{\bar{s}s}\tau_\infty(s,\bar{s})\rangle_{s\in\mathcal{B}_{\bar{s}}}=\\
&\qquad=\frac{1}{\tau^{\mathrm{r}}_\infty(\bar{s})}\langle n_{\bar{s}s}\left(\tau^{\mathrm{r}}_\infty(\bar{s})-\tau_\infty(s,\bar{s})\right)\rangle_{s\in\mathcal{B}_{\bar{s}}}\\
&\qquad=\frac{1}{\tau^{\mathrm{r}}_\infty(\bar{s})}\langle n_{\bar{s}s}\left(\langle\tau_\infty(s',\bar{s})\rangle_{s'\in\mathcal{B}_{\bar{s}}}-\tau_\infty(s,\bar{s})\right)\rangle_{s'\in\mathcal{B}_{\bar{s}}}\\
&\qquad=\frac{1}{\tau^{\mathrm{r}}_\infty(\bar{s})}\langle\left(n_{\bar{s}s}-\langle n_{\bar{s}s'}\rangle_{s'\in\mathcal{B}_{\bar{s}}}\right)\left(\langle\tau_\infty(s',\bar{s})\rangle_{s'\in\mathcal{B}_{\bar{s}}}-\tau_\infty(s,\bar{s})\right)\rangle_{s'\in\mathcal{B}_{\bar{s}}}\\
&\qquad=-\frac{1}{\tau^{\mathrm{r}}_\infty(\bar{s})}g_n(\bar{s},\bar{s})\,,
\end{aligned}\tag{A. 13}$$

where in the last line we have used the definition of $g_n(s,\bar{s})$ of Eq. (2.31). The same manipulation can be used for the second and fifth term of Eq. (A. 11), to obtain

$$\begin{aligned}
&-\varphi(\bar{s})\langle u_{\bar{s}s}\rangle_{s\in\mathcal{B}_{\bar{s}}}+\frac{d_{\bar{s}}}{S_N-1}\varphi(\bar{s})\langle u_{\bar{s}s}\tau_\infty(s,\bar{s})\rangle_{s\in\mathcal{B}_{\bar{s}}}=\\
&\qquad=-\frac{\varphi(\bar{s})}{\tau^{\mathrm{r}}(\bar{s})}\langle\left(u_{\bar{s}s}-\langle u_{\bar{s}s'}\rangle_{s'\in\mathcal{B}_{\bar{s}}}\right)\left(\langle\tau_\infty(s',\bar{s})\rangle_{s'\in\mathcal{B}_{\bar{s}}}-\tau_\infty(s,\bar{s})\right)\rangle_{s'\in\mathcal{B}_{\bar{s}}}\\
&\qquad=\frac{\varphi(\bar{s})}{\tau^{\mathrm{r}}_\infty(\bar{s})}g_u(\bar{s},\bar{s})\,,
\end{aligned}\tag{A. 14}$$

where in the last line we have used the definition of $g_u(s,\bar{s})$ of Eq. (2.32).

Using the high-temperature recursion relation Eq. (2.22), we can obtain an alternative writing for $\tau_\infty(s,\bar{s})$:

$$\begin{aligned}
\langle\tau_\infty(s',\bar{s})\rangle_{s'\in\mathcal{B}_s}&=\frac{1}{d_s}\sum_{s'\in\mathcal{B}_s}\tau_\infty(s',\bar{s})-\frac{1}{d_s}\sum_{s'\in\mathcal{B}_s}\tau_\infty(s,\bar{s})+\tau_\infty(s,\bar{s})\\
&=\tau_\infty(s,\bar{s})-\frac{1}{d_s}\sum_{s'\in\mathcal{B}_s}\left(\tau_\infty(s,\bar{s})-\tau_\infty(s',\bar{s})\right)\\
&=\tau_\infty(s,\bar{s})-\frac{1}{d_s}\quad\Rightarrow\quad\tau_\infty(s,\bar{s})=\frac{1}{d_s}+\langle\tau_\infty(s',\bar{s})\rangle_{s'\in\mathcal{B}_s}\,,
\end{aligned}\tag{A. 15}$$

and use this expression to manipulate the remaining fourth and sixth term of Eq. (A. 11). Let us start from the fourth one

$$\begin{aligned}
&\langle d_s\langle n_{ss'}\left(\tau_\infty(s,\bar{s})-\tau_\infty(s',\bar{s})\right)\rangle_{s'\in\mathcal{B}_s}\rangle_{s\in\bar{\mathcal{S}}}=\\
&=\left\langle d_s\left\langle n_{ss'}\left(\frac{1}{d_s}+\langle\tau_\infty(s'',\bar{s})\rangle_{s''\in\mathcal{B}_s}-\tau_\infty(s',\bar{s})\right)\right\rangle_{s'\in\mathcal{B}_s}\right\rangle_{s\in\bar{\mathcal{S}}}\\
&=\langle\langle n_{ss'}\rangle_{s'\in\mathcal{B}_s}\rangle_{s\in\bar{\mathcal{S}}}+\langle d_s\langle n_{ss'}\left(\langle\tau_\infty(s'',\bar{s})\rangle_{s''\in\mathcal{B}_s}-\tau_\infty(s',\bar{s})\right)\rangle_{s'\in\mathcal{B}_s}\rangle_{s\in\bar{\mathcal{S}}}\\
&=\langle\langle n_{ss'}\rangle_{s'\in\mathcal{B}_s}\rangle_{s\in\bar{\mathcal{S}}}+\langle d_s\langle\left(n_{ss'}-\langle n_{ss''}\rangle_{s''\in\mathcal{B}_s}\right)\left(\langle\tau_\infty(s'',\bar{s})\rangle_{s''\in\mathcal{B}_s}-\tau_\infty(s',\bar{s})\right)\rangle_{s'\in\mathcal{B}_s}\rangle_{s\in\bar{\mathcal{S}}}\\
&=\langle\langle n_{ss'}\rangle_{s'\in\mathcal{B}_s}\rangle_{s\in\bar{\mathcal{S}}}-\langle d_s g_n(s,\bar{s})\rangle_{s\in\bar{\mathcal{S}}}\,,
\end{aligned}\tag{A. 16}$$

where in the last line we have used the definition of $g_n(s,\bar{s})$ of Eq. (2.31). Then, we can apply

the same manipulation to the sixth term, to obtain

$$
\begin{aligned}
&- \langle \varphi(s) d_s \langle u_{ss'} \left(\tau_\infty(s,\bar{s}) - \tau_\infty(s',\bar{s})\right)\rangle_{s'\in\mathcal{B}_s}\rangle_{s\in\bar{\mathcal{S}}} = \\
&= -\langle \varphi(s) \langle u_{ss'}\rangle_{s'\in\mathcal{B}_s}\rangle_{s\in\bar{\mathcal{S}}} - \langle \varphi(s) d_s \langle (u_{ss'} - \langle u_{ss''}\rangle_{s''\in\mathcal{B}_s}) \left(\langle \tau_\infty(s'',\bar{s})\rangle_{s''\in\mathcal{B}_s} - \tau_\infty(s',\bar{s})\right)\rangle_{s'\in\mathcal{B}_s}\rangle_{s\in\bar{\mathcal{S}}} \\
&= -\langle \varphi(s) \langle u_{ss'}\rangle_{s'\in\mathcal{B}_s}\rangle_{s\in\bar{\mathcal{S}}} + \langle \varphi(s) d_s g_u(s,\bar{s})\rangle_{s\in\bar{\mathcal{S}}} \,,
\end{aligned}
\tag{A. 17}
$$

where in the last line we have used the definition of $g_u(s,\bar{s})$ of Eq. (2.32).

Regrouping all six terms, we finally obtain

$$
\begin{aligned}
M_{\boldsymbol{\phi}}(\bar{s}) =& - \frac{1}{\tau^{\mathrm{r}}_\infty(\bar{s})} g_n(\bar{s},\bar{s}) + \frac{\varphi(\bar{s})}{\tau^{\mathrm{r}}_\infty(\bar{s})} g_u(\bar{s},\bar{s}) \\
&+ \langle \langle n_{ss'}\rangle_{s'\in\mathcal{B}_s}\rangle_{s\in\bar{\mathcal{S}}} - \langle d_s g_n(s,\bar{s})\rangle_{s\in\bar{\mathcal{S}}} \\
&- \langle \varphi(s) \langle u_{ss'}\rangle_{s'\in\mathcal{B}_s}\rangle_{s\in\bar{\mathcal{S}}} + \langle \varphi(s) d_s g_u(s,\bar{s})\rangle_{s\in\bar{\mathcal{S}}} \\
=& - \frac{d_{\bar{s}}}{S_N - 1} g_n(\bar{s},\bar{s}) + \langle \langle n_{ss'}\rangle_{s'\in\mathcal{B}_s} - d_s g_n(s,\bar{s})\rangle_{s\in\bar{\mathcal{S}}} \\
&+ \frac{d_{\bar{s}}}{S_N - 1} \varphi(\bar{s}) g_u(\bar{s},\bar{s}) - \langle \varphi(s) \left(\langle u_{ss'}\rangle_{s'\in\mathcal{B}_s} - d_s g_u(s,\bar{s})\right)\rangle_{s\in\bar{\mathcal{S}}} \\
=& \frac{S_N}{S_N - 1} \langle (1-\delta_{s\bar{s}}) \langle n_{ss'}\rangle_{s'\in\mathcal{B}_s} - d_s g_n(s,\bar{s})\rangle_{s\in\mathcal{S}} \\
&- \frac{S_N}{S_N - 1} \langle \varphi(s) \left((1-\delta_{s\bar{s}}) \langle u_{ss'}\rangle_{s'\in\mathcal{B}_s} - d_s g_u(s,\bar{s})\right)\rangle_{s\in\mathcal{S}} \,,
\end{aligned}
\tag{A. 18}
$$

which is identical to Eq. (2.29).

A.5 Convergence analysis of value iteration

We have performed a study of the accuracy of the value iteration algorithm (reported in pseudocode in Fig. A.2) as a function of the convergence threshold θ. We have used a small threshold $\theta_{\mathrm{ref}} = 10^{-5}$ as a reference. We then ran the value iteration algorithm to obtain the optimal return time to target for various targets with $N \leq 9$, at the temperatures $T = 0.1$, $T = 0.3$, $T = 0.6$, and $T = 1$, for increasing values of the convergence threshold θ. We were not able to perform this analysis on larger clusters because the algorithm required too much computational time to converge with the reference threshold $\theta_{\mathrm{ref}} = 10^{-5}$. We then calculated, for each θ, the error relative to the reference threshold as

$$
\text{Relative error} = \frac{|\tau^{\mathrm{r}}_\theta(\bar{s}) - \tau^{\mathrm{r}}_{\mathrm{ref}}(\bar{s})|}{\tau^{\mathrm{r}}_{\mathrm{ref}}(\bar{s})} \,,
\tag{A. 19}
$$

where here $\tau^{\mathrm{r}}_\theta(\bar{s})$ indicates the optimal return time to target obtained with threshold θ, and $\tau^{\mathrm{r}}_{\mathrm{ref}}(\bar{s})$ the optimal return time obtained with θ_{ref}.

The results of this analysis are reported in Fig. A.6. First, we can see that the relative error grows linearly with the convergence threshold with a slope ~ 1, and that the error is higher when the temperature is higher. Next, we note that there is little variation with respect to cluster size, while target compactness has a greater influence on the error, with higher compactness corresponding to lower error. However, at higher temperatures, the effect of the compactness is greatly reduced. Overall, we can see that the trend of the relative error with the threshold θ behaves very similarly for the highest temperature considered $T = 1$, for all the cluster sizes and shapes.

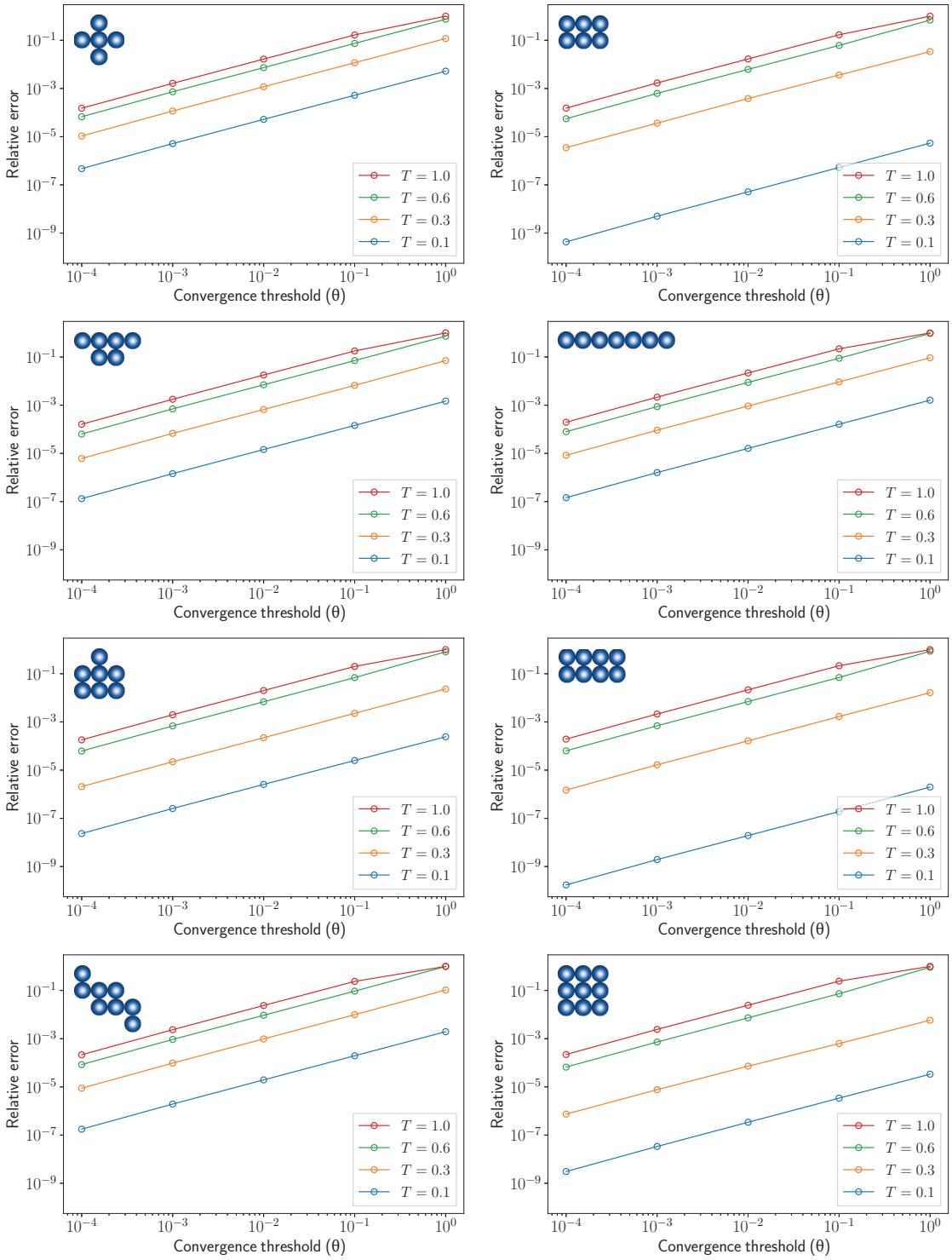


Figure A.6: Relative error of the accuracy of the value iteration algorithm as a function of the convergence threshold θ, for some targets with $N \leq 9$.

Based on these results, we decided to set the convergence threshold as a function of the cluster size N, in order to obtain a relative error at worse equal to $\sim 10^{-2}$ for $T = 1$. With an empirical study, we have set the following thresholds, that are a compromise between computational time and accuracy:

- $\theta = \theta_{\text{ref}} = 10^{-5}$ for $N \leq 5$,
- $\theta = 10^{-4}$ for $N = 6$,
- $\theta = 10^{-3}$ for $7 \leq N \leq 11$,
- $\theta = 10^{-2}$ for $N = 12$.

These thresholds should ensure a relative error $\lesssim 10^{-2}$ for the majority of targets and temperature ranges that we considered in this work.

A.6 State encoding

To encode a state into a numerical identifier, we use a simple method based on binary numbers. This approach was inspired by the "light reflection" method proposed in Ref. [203].

The first step is to determine the bounding box of the cluster. Then, lattice sites within the box are filled with a 1 if they contain an atom, or with a 0 if they are empty, as shown in Fig. A.7. The 1's and 0's are then read as single binary number, from top to bottom. In the example in the figure, this number is 110111110. Finally, the identifier of the state is composed of two numbers: the horizontal length of the bounding box and the base-10 conversion of the binary number. In the example, the identifier is $(3, 446)$.

Note that it is necessary to include the length of the bounding box, otherwise the identifier would not be unique. A simple case to show this is to consider the two states of a dimer, both of which correspond to the binary number 11, or 3 in base 10, and which would be indistinguishable without specifying their horizontal (or, alternatively, their vertical) length.

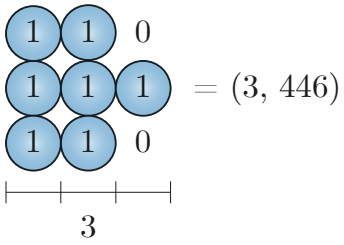


Figure A.7: Schematic representation of the method we have used to encode polyominoes into numerical identifiers.

A.7 Parabolic fit to obtain M_0 and M_* from simulations

In Fig. A.8 we show the procedure that we used to extract the values of the high-temperature slopes $M_0^{\text{sim}}(\bar{s})$ and $M_*^{\text{sim}}(\bar{s})$ from simulations. In this figure, we show an example for the zero-force case. We have followed the same procedure for the optimal-policy case. The square markers in blue correspond to the expected return time to target $\tau_0^{\text{r}}(\bar{s})$ obtained from iterative numerical methods. The three points at $T = 5, 10, 20$ have been fitted with a parabola. Then, the slope $M_0^{\text{sim}}(\bar{s})$ is obtained from the tangent of this parabola calculated at $1/T = 0$.

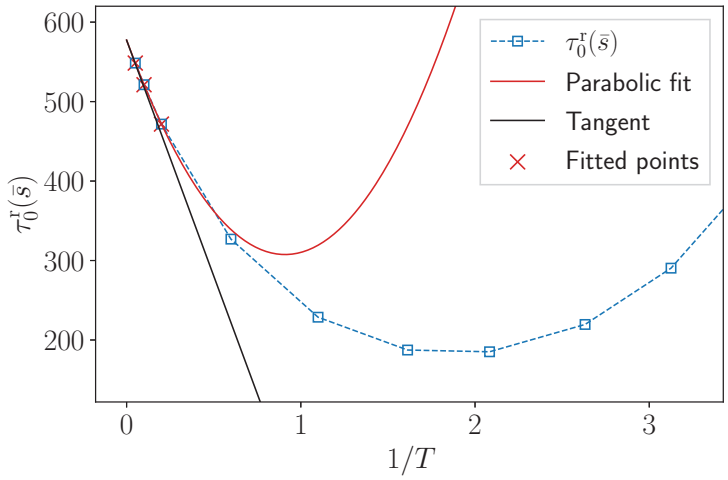


Figure A.8: Example of the procedure to obtain $M_0^{\mathrm{sim}}(\bar{s})$ from simulation data. The high-temperature points ($T = 5, 10, 20$) of $\tau_0^{\mathrm{r}}(\bar{s})$ are fitted with a parabola (in red). The black line corresponds to the tangent of the parabola at $1/T = 0$. The slope of this line corresponds to $M_0^{\mathrm{sim}}(\bar{s})$. In this example, the target is the 9-atom, 3×3 square target.

A.8 Procedure to check degeneracy of the actions

In the pseudocode of the value iteration algorithm reported in Fig. A.2 we can see that, to obtain an optimal policy from the estimated optimal value function, we need to evaluate the expression $\mathrm{argmax}_a \sum_{s',r} p(s', r \mid s, a)\,[r + \beta V(s')]$. The argmax_a operator outputs the set of optimal actions in state s. If this set contains more than one action, then these actions are degenerate.

Computationally, to evaluate the argmax_a operator, one needs to compare the values of the three possible actions in $\mathcal{A}$ and check if there are values that are equal. However, due to numerical error, this check needs to be done within a certain "tolerance" threshold, in the sense that, if the absolute difference of the values of two actions is smaller than this threshold, than we consider these actions as both optimal and therefore degenerate. If the tolerance threshold is too close to zero, we risk to undercount degenerate states, while if it is too high we risk to overcount.

In Fig. A.9 we can see the effect of the tolerance threshold on the fraction of degenerate states $D_N(\bar{s})/S_N$, for three different targets with $N = 5$, 7, 9. We have checked several combinations of the tolerance threshold and the convergence threshold θ of the value iteration algorithm, for different values of the temperature: $T = 0.1$, 0.3, 0.6, 1. We can see that there are several factors that are affecting the value of $D_N(\bar{s})/S_N$. First, we notice that, for values of the convergence threshold $\theta \leq 10^{-2}$, the results are essentially unchanged for all the three targets considered. This further supports our choice of $\theta = 10^{-2}$ as the maximum threshold value for $N = 12$ reported in Appendix A.5.

Second, we can see that, for higher temperatures, the variation of $D_N(\bar{s})/S_N$ as a function of the tolerance threshold is larger. In all the three cases, however, when the value of the tolerance threshold is smaller than $10^{-4} \sim 10^{-5}$, then the value of $D_N(\bar{s})/S_N$ reaches a constant value for all the temperatures considered.

Based on this analysis, and on the analysis of other targets with $N \leq 9$ whose results are analogous to those reported in Fig. A.9, we have decided to set the tolerance threshold for the check of degenerate actions to 10^{-5}.

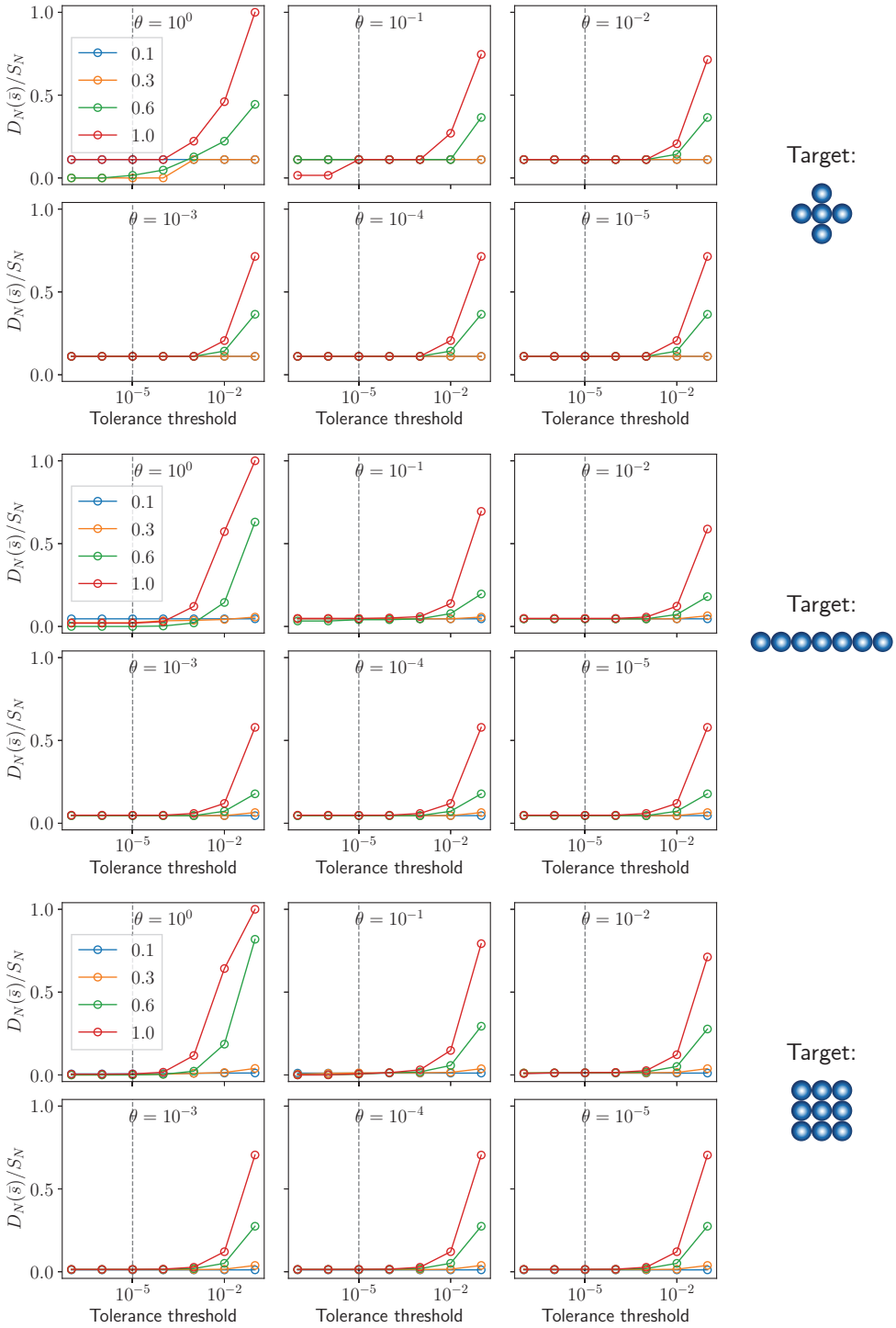


Figure A.9: Variation of the fraction of degenerate states $D_N(\bar{s})/S_N$ as a function of the tolerance threshold, at $T = 0.1$, 0.3, 0.6, 1 and for increasing values of the convergence threshold θ, for three different targets with $N = 5$, 7, 9.

A.9 Variation of the fraction of degenerate states with temperature

In Fig. A.10 we plot the fraction of degenerate states $D_N(\bar{s})/S_N$ as a function of inverse temperature, for several targets with $4 \leq N \leq 12$. The color of the lines and the markers corresponds to the symmetry group of the target, while the thickness of the lines is proportional to the size of the target. We can see that the the fraction $D_N(\bar{s})/S_N$ varies weakly with temperature.

Note that we have checked all possible targets with $4 \leq N \leq 7$ (as in Fig. 2.27 of the main text) only for $T = 0.24$, while for all the other temperatures we have checked only some selected targets. We did not investigate the reason of the appearance of degenerate states at high temperature for some targets belonging to the $\mathcal{H}$ and $\mathcal{V}$ symmetry classes, but one possible reason is the tolerance threshold being too high for these particular targets in this temperature range, as discussed in Appendix A.9.

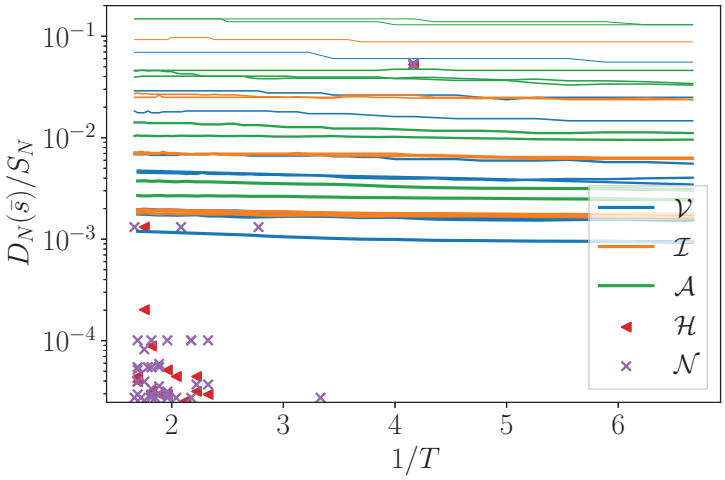


Figure A.10: Fraction of degenerate states $D_N(\bar{s})/S_N$ as a function of $1/T$, for several targets with $4 \leq N \leq 12$. The thickness of the lines is proportional to the size of the cluster. For the $\mathcal{H}$ and $\mathcal{V}$ symmetry classes, we show scatter points because in this two cases $D_N(\bar{s})/S_N = 0$ for most of the temperatures.

A.10 Temperature transitions for the trimer

In Fig. A.11 we plot the first derivative of the optimal time to reach the target shown in Fig. 2.28 $\tau_*(\bar{s})$ with respect to inverse temperature, starting from the target itself (in orange), and from all the other states of the system (in blue). We can see that the derivative has a discontinuity at the transition temperature T_c (vertical dashed line) for all the states of the system.

A.11 Effect of an additional bond on the mean return time to target

In Fig. A.12 we show the effect on the mean return time to target of including one extra bond with energy J to all the hopping barriers in the system. We can see that this procedure, which

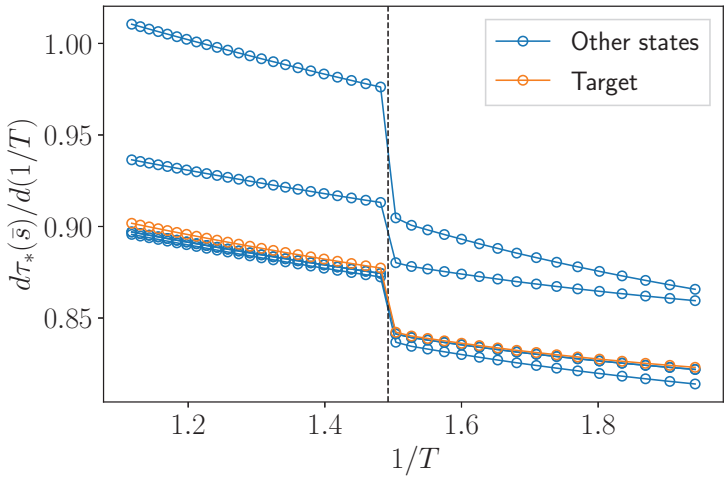


Figure A.11: Derivative of the expected optimal time to reach the target starting from the target and from the other 5 states in the system for the trimer target of Fig. 2.28.

simply corresponds to a rescaling of all the timescales by $\exp(J/k_BT)$, leads to results that are qualitatively identical to those discussed in Chapter 2, but with a decrease of the depth of the minimum in the mean return time to target.

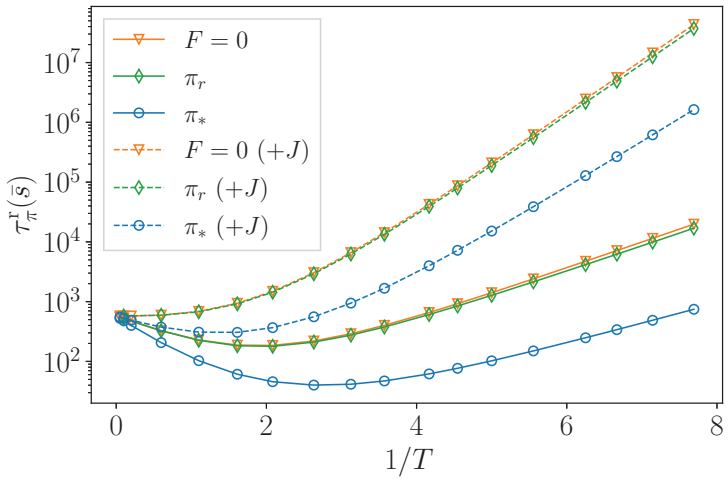


Figure A.12: Comparison of the mean return time to target as a function of inverse temperature with (dashed lines) and without (solid lines) the inclusion of an extra particle-substrate bond of energy J in the hopping energy barriers.

Bibliography

[1] R. P. Feynman. "There's Plenty of Room at the Bottom". *Engineering and Science magazine* 23.5 (1960), pp. 22–36.

[2] F. J. Rubio-Sierra, W. M. Heckl, and R. W. Stark. "Nanomanipulation by Atomic Force Microscopy". *Adv. Eng. Mater.* 7.4 (2005), pp. 193–196.

[3] Q. Cao et al. "Recent advances in manipulation of micro- and nano-objects with magnetic fields at small scales". *Materials Horizons* 7.3 (2020), pp. 638–666.

[4] B. W. Smith and K. Suzuki, eds. *Microlithography: Science and Technology, Second Edition*. CRC Press, 2018.

[5] F. M. Raymo. "Nanomaterials Synthesis and Applications: Molecule-Based Devices". *Springer Handbook of Nanotechnology*. Springer Berlin Heidelberg, 2010, pp. 17–45.

[6] S. Kasani, K. Curtin, and N. Wu. "A review of 2D and 3D plasmonic nanostructure array patterns: fabrication, light management and sensing applications". 8.12 (2019), pp. 2065–2089.

[7] D. M. Eigler and E. K. Schweizer. "Positioning single atoms with a scanning tunnelling microscope". *Nature* 344.6266 (1990), pp. 524–526.

[8] G. Binnig et al. "Surface Studies by Scanning Tunneling Microscopy". *Phys. Rev. Lett.* 49.1 (1982), pp. 57–61.

[9] G. Binnig, C. F. Quate, and C. Gerber. "Atomic Force Microscope". *Phys. Rev. Lett.* 56.9 (1986), pp. 930–933.

[10] J. E. Curtis, B. A. Koss, and D. G. Grier. "Dynamic holographic optical tweezers". *Opt. Commun.* 207.1-6 (2002), pp. 169–175.

[11] Y. Jiang et al. "Trapping and manipulation of a single micro-object in solution with femtosecond laser-induced mechanical force". *Appl. Phys. Lett.* 90.6 (2007), p. 061107.

[12] D. G. Grier. "A revolution in optical manipulation". *Nature* 424.6950 (2003), pp. 810–816.

[13] H. Lee, A. M. Purdon, and R. M. Westervelt. "Manipulation of biological cells using a microelectromagnet matrix". *Appl. Phys. Lett.* 85.6 (2004), pp. 1063–1065.

[14] J. Yan, D. Skoko, and J. F. Marko. "Near-field-magnetic-tweezer manipulation of single DNA molecules". *Phys. Rev. E* 70.1 (2004).

[15] IBM. *A Boy And His Atom: The World's Smallest Movie*. URL: www.youtube.com/watch?v=oSCX78-8-q0.

[16] A. Martínez-Galera et al. "Towards scalable nano-engineering of graphene". *Sci. Rep.* 4.1 (2014), pp. 1–6.

[17] K. Xu, A. Kalantari, and X. Qian. "Efficient AFM-Based Nanoparticle Manipulation Via Sequential Parallel Pushing". *IEEE Trans. on Nanotechnol.* 11.4 (2012), pp. 666–675.

[18] D. Barredo et al. "An atom-by-atom assembler of defect-free arbitrary two-dimensional atomic arrays". *Science* 354.6315 (2016), pp. 1021–1023.

[19] A. Requicha et al. "Algorithms and Software for Nanomanipulation with Atomic Force Microscopes". *Int. J. Rob. Res.* 28.4 (2009), pp. 512–522.

[20] C. Onal, O. Ozcan, and M. Sitti. "Automated 2-D nanoparticle manipulation with an atomic force microscope". *2009 IEEE Int. Conf. Robot. Autom.* IEEE, 2009.

[21] P. McCormack, F. Han, and Z. Yan. "Self-Organization of Metal Nanoparticles in Light: Electrodynamics–Molecular Dynamics Simulations and Optical Binding Experiments". *J. Phys. Chem. Lett.* 9.3 (2018), pp. 545–549.

[22] P. Kuhn et al. "Complex Shape Evolution of Electromigration-Driven Single-Layer Islands". *Phys. Rev. Lett.* 94 (16 2005), p. 166105.

[23] M. Mahadevan and R. M. Bradley. "Simulations and theory of electromigration-induced slit formation in unpassivated single-crystal metal lines". *Phys. Rev. B* 59 (16 1999), pp. 11037–11046.

[24] O. Pierre-Louis and T. L. Einstein. "Electromigration of single-layer clusters". *Phys. Rev. B* 62.20 (2000), pp. 13697–13706.

[25] S. V. Khare, N. C. Bartelt, and T. L. Einstein. "Diffusion of Monolayer Adatom and Vacancy Clusters: Langevin Analysis and Monte Carlo Simulations of their Brownian Motion". *Phys. Rev. Lett.* 75 (11 1995), pp. 2148–2151.

[26] K. C. Lai, D.-J. Liu, and J. W. Evans. "Diffusion of two-dimensional epitaxial clusters on metal (100) surfaces: Facile versus nucleation-mediated behavior and their merging for larger sizes". *Phys. Rev. B* 96 (23 2017), p. 235406.

[27] M. J. Solomon. "Tools and Functions of Reconfigurable Colloidal Assembly". *Langmuir* 34.38 (2018), pp. 11205–11219.

[28] T. D. Edwards and M. A. Bevan. "Controlling Colloidal Particles with Electric Fields". *Langmuir* 30.36 (2014), pp. 10793–10803.

[29] J. J. Juárez and M. A. Bevan. "Feedback Controlled Colloidal Self-Assembly". *Adv. Funct. Mater.* 22.18 (2012), pp. 3833–3839.

[30] J. Xavier et al. "Controlled formation and manipulation of colloidal lattices by dynamically reconfigurable three dimensional interferometric optical traps". *Appl. Phys. Lett.* 101.20 (2012), p. 201101.

[31] S. Bleil, D. W. M. Marr, and C. Bechinger. "Field-mediated self-assembly and actuation of highly parallel microfluidic devices". *Appl. Phys. Lett.* 88.26 (2006), p. 263515.

[32] Y. Xue et al. "Optimal Design of a Colloidal Self-Assembly Process". *IEEE Trans. Control Syst. Technol.* 22.5 (2014), pp. 1956–1963.

[33] J. J. Juárez et al. "Multiple electrokinetic actuators for feedback control of colloidal crystal size". *Lab Chip* 12.20 (2012), p. 4063.

[34] T. Michely and J. Krug. *Islands, Mounds and Atoms*. Springer Berlin Heidelberg, 2004.

[35] C. Tao, W. G. Cullen, and E. D. Williams. "Visualizing the Electron Scattering Force in Nanostructures". *Science* 328.5979 (2010), pp. 736–740.

[36] C. P. Joshi et al. "Critical island size, scaling, and ordering in colloidal nanoparticle self-assembly". *Physical Review E* 90.3 (2014).

[37] J. Nozawa et al. "Kink Distance and Binding Energy of Colloidal Crystals". *Cryst. Growth Des.* 18.10 (2018), pp. 6078–6083.

[38] L. D. Landau and E. M. Lifshitz. *Statistical Physics*. 3rd ed. Elsevier, 2013.

[39] S. Curiotto et al. "Shape changes of two-dimensional atomic islands and vacancy clusters diffusing on epitaxial (111) interfaces under the impact of an external force". *J. Cryst. Growth* 520 (2019), pp. 42–45.

[40] M. Rusanen, P. Kuhn, and J. Krug. "Kinetic Monte Carlo simulations of oscillatory shape evolution for electromigration-driven islands". *Physical Review B* 74.24 (2006).

[41] U. Nakaya. *Snow Crystals*. Harvard University Press, 1954.

[42] K. G. Libbrecht. "Physical Dynamics of Ice Crystal Growth". *Annual Review of Materials Research* 47.1 (2017), pp. 271–295.

[43] K. G. Libbrecht. *Snow crystals*. 2019. URL: `https://arxiv.org/abs/1910.06389`.

[44] J. Hoffman. "Q&A: The snowflake designer". *Nature* 480.7378 (2011), pp. 453–454.

[45] Veritasium. *The Snowflake Mystery*. URL: `www.youtube.com/watch?v=ao2Jfm35XeE`.

[46] T. Gibaud and P. Schurtenberger. "A closer look at arrested spinodal decomposition in protein solutions". *Journal of Physics: Condensed Matter* 21.32 (2009), p. 322201.

[47] N. Vladimirova, A. Malagoli, and R. Mauri. "Diffusion-driven phase separation of deeply quenched mixtures". *Physical Review E* 58.6 (1998), pp. 7691–7699.

[48] J. Vollmer, G. K. Auernhammer, and D. Vollmer. "Minimal Model for Phase Separation under Slow Cooling". *Physical Review Letters* 98.11 (2007).

[49] D. Vollmer, J. Vollmer, and A. J. Wagner. "Oscillatory kinetics of phase separation in a binary mixture under constant heating". *Physical Chemistry Chemical Physics* 4.8 (2002), pp. 1380–1385.

[50] P. Colli and J. Sprekels. "Optimal Control of an Allen–Cahn Equation with Singular Potentials and Dynamic Boundary Condition". *SIAM Journal on Control and Optimization* 53.1 (2015), pp. 213–234.

[51] L. Blank et al. "Optimal Control of Allen-Cahn Systems". *International Series of Numerical Mathematics*. Springer International Publishing, 2014, pp. 11–26.

[52] M. H. Farshbaf-Shaker. "A Penalty Approach to Optimal Control of Allen-Cahn Variational Inequalities: MPEC-View". *Numerical Functional Analysis and Optimization* 33.11 (2012), pp. 1321–1349.

[53] P. Benner and M. Stoll. "Optimal Control for Allen-Cahn Equations Enhanced by Model Predictive Control". *IFAC Proceedings Volumes* 46.26 (2013), pp. 139–143.

[54] R. Kurita. "Control of pattern formation during phase separation initiated by a propagated trigger". *Scientific Reports* 7.1 (2017).

[55] T. Tsukada and R. Kurita. "Pattern Formation during Phase Separation by Radial Quenching at the Base of a Three-Dimensional Box". *Journal of the Physical Society of Japan* 88.4 (2019), p. 044603.

[56] T. Tsukada and R. Kurita. "Mechanism behind columnar pattern formation during directional quenching-induced phase separation". *Physical Review Research* 2.1 (2020).

[57] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. Second. The MIT Press, 2018.

[58] D. Luenberger. *Introduction to dynamic systems: theory, models, and applications*. New York, NY: Wiley, 1979.

[59] R. Bellman. *Eye of the Hurricane*. World Scientific, 1984.

[60] S. Dreyfus. "Richard Bellman on the Birth of Dynamic Programming". *Operations Research* 50.1 (2002), pp. 48–51.

[61] C. J. Watkins and P. Dayan. "Q-learning". *Machine learning* 8.3-4 (1992), pp. 279–292.

[62] V. Mnih et al. *Playing Atari with Deep Reinforcement Learning*. 2013. URL: `http://arxiv.org/abs/1312.5602`.

[63] H. van Hasselt and M. A. Wiering. "Reinforcement Learning in Continuous Action Spaces". *2007 IEEE International Symposium on Approximate Dynamic Programming and Reinforcement Learning*. Honolulu, HI, USA: IEEE, 2007, pp. 272–279.

[64] M. Riedmiller. "Neural Fitted Q Iteration – First Experiences with a Data Efficient Neural Reinforcement Learning Method". *Machine Learning: ECML 2005*. Springer Berlin Heidelberg, 2005, pp. 317–328.

[65] S. Lange and M. Riedmiller. "Deep auto-encoder neural networks in reinforcement learning". *The 2010 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2010.

[66] V. Mnih et al. "Human-level control through deep reinforcement learning". *Nature* 518.7540 (2015), pp. 529–533.

[67] L.-J. Lin. "Self-improving reactive agents based on reinforcement learning, planning and teaching". *Machine Learning* 8.3-4 (1992), pp. 293–321.

[68] H. van Hasselt. "Double Q-learning". *Advances in Neural Information Processing Systems*. Ed. by J. Lafferty et al. Vol. 23. Curran Associates, Inc., 2010.

[69] H. van Hasselt, A. Guez, and D. Silver. "Deep Reinforcement Learning with Double Q-Learning". *Proceedings of the AAAI Conference on Artificial Intelligence* 30.1 (2016).

[70] D. Gruznev et al. "Stepwise self-assembly of C60 mediated by atomic scale moiré magnifiers". *Nature Communications* 4.1 (2013).

[71] R. Ganapathy et al. "Direct Measurements of Island Growth and Step-Edge Barriers in Colloidal Epitaxy". *Science* 327.5964 (2010), pp. 445–448.

[72] M. Giesen-Seibert et al. "Time fluctuations of steps on Cu(11n) surfaces investigated by temperature variable tunneling microscopy". *Surface Science* 329.1-2 (1995), pp. 47–60.

[73] M. Giesen and G. S. Icking-Konert. "Equilibrium fluctuations and decay of step bumps on vicinal Cu (111) surfaces". *Surface Science* 412-413 (1998), pp. 645–656.

[74] M. Bott, T. Michely, and G. Comsa. "The homoepitaxial growth of Pt on Pt(111) studied with STM". *Surface Science* 272.1-3 (1992), pp. 161–166.

[75] M. Giesen. "Step and island dynamics at solid/vacuum and solid/liquid interfaces". *Prog. Surf. Sci.* 68.1-3 (2001), pp. 1–154.

[76] B. C. Hubartt and J. G. Amar. "Critical island-size, stability, and morphology of 2D colloidal Au nanoparticle islands". *J. Chem. Phys.* 142.2 (2015), p. 024709.

[77] M. Mahadevan and R. M. Bradley. “Stability of a circular void in a passivated, current-carrying metal film”. *Journal of Applied Physics* 79.9 (1996), pp. 6840–6847.

[78] Y.-H. Mao et al. “Epitaxial growth of highly strained antimonene on Ag(111)”. *Front. Phys.* 13.3 (2018).

[79] L. Helden, R. Eichhorn, and C. Bechinger. “Direct measurement of thermophoretic forces”. *Soft Matter* 11.12 (2015), pp. 2379–2386.

[80] L. Becu et al. “Resonant reshaping of colloidal clusters on a current carrying wire”. *The European Physical Journal E* 40.12 (2017).

[81] N. Wei, H.-Q. Wang, and J.-C. Zheng. “Nanoparticle manipulation by thermal gradient”. *Nanoscale Res. Lett.* 7.1 (2012), p. 154.

[82] A. Dong et al. “A Generalized Ligand-Exchange Strategy Enabling Sequential Surface Functionalization of Colloidal Nanocrystals”. *J. Am. Chem. Soc.* 133.4 (2011), pp. 998–1006.

[83] M. Braibanti, D. Vigolo, and R. Piazza. “Does Thermophoretic Mobility Depend on Particle Size?” *Phys. Rev. Lett.* 100.10 (2008).

[84] A. Würger. “Thermal non-equilibrium transport in colloids”. *Rep. Prog. Phys.* 73.12 (2010), p. 126601.

[85] D.-J. Liu and J. D. Weeks. “Quantitative theory of current-induced step bunching on Si(111)”. *Phys. Rev. B* 57 (23 1998), pp. 14891–14900.

[86] N. G. Van Kampen. *Stochastic processes in physics and chemistry*. Elsevier, 1992.

[87] S. Glasstone, K. J. Laidler, and H. Eyring. *The Theory of Rate Processes: The Kinetics of Chemical Reactions, Viscosity, Diffusion and Electrochemical Phenomena*. International chemical series. McGraw-Hill Book Company, Inc., 1941.

[88] H. Ibach. *Physics of Surfaces and Interfaces*. Springer, 2006.

[89] K. C. Lai et al. “Reshaping, Intermixing, and Coarsening for Metallic Nanocrystals: Nonequilibrium Statistical Mechanical and Coarse-Grained Modeling”. *Chemical Reviews* 119.11 (2019), pp. 6670–6768.

[90] P. J. Rous and D. N. Bly. “Wind force for adatom electromigration on heterogeneous surfaces”. *Physical Review B* 62.12 (2000), pp. 8478–8486.

[91] D. Kandel and E. Kaxiras. “Microscopic Theory of Electromigration on Semiconductor Surfaces”. *Physical Review Letters* 76.7 (1996), pp. 1114–1117.

[92] O. Pierre-Louis. “Local Electromigration Model for Crystal Surfaces”. *Physical Review Letters* 96.13 (2006).

[93] N. Combe and H. Larralde. “Low-temperature shape relaxation of two-dimensional islands by edge diffusion”. *Phys. Rev. B* 62 (23 2000), pp. 16074–16084.

[94] J. R. Sanchez and J. W. Evans. “Diffusion of small clusters on metal (100) surfaces: Exact master-equation analysis for lattice-gas models”. *Phys. Rev. B* 59.4 (1999), pp. 3224–3233.

[95] L. Lovász. “Random Walks on Graphs: A Survey”. *Combinatorics, Paul Erdős is eighty* 2 (1993), pp. 1–46.

[96] J. D. Noh and H. Rieger. “Random Walks on Complex Networks”. *Phys. Rev. Lett.* 92 (11 2004), p. 118701.

[97] A. Baronchelli and V. Loreto. “Ring structures and mean first passage time in networks”. *Phys. Rev. E* 73 (2 2006), p. 026103.

[98] Y. Lin, A. Julaiti, and Z. Zhang. "Mean first-passage time for random walks in general graphs with a deep trap". *J. Chem. Phys.* 137.12 (2012), p. 124104.

[99] V. Tejedor, O. Bénichou, and R. Voituriez. "Global mean first-passage times of random walks on complex networks". *Phys. Rev. E* 80 (6 2009), p. 065104.

[100] P. Jensen et al. "Kinetics of shape equilibration for two dimensional islands". *The European Physical Journal B* 11.3 (1999), pp. 497–504.

[101] D.-J. Liu and J. W. Evans. "Sintering of two-dimensional nanoclusters in metal(100) homoepitaxial systems: Deviations from predictions of Mullins continuum theory". *Physical Review B* 66.16 (2002).

[102] C. Misbah, O. Pierre-Louis, and Y. Saito. "Crystal surfaces in and out of equilibrium: A modern view". *Reviews of Modern Physics* 82.1 (2010), pp. 981–1040.

[103] D. B. Dougherty et al. "Experimental Persistence Probability for Fluctuating Steps". *Phys. Rev. Lett.* 89 (13 2002), p. 136102.

[104] M. Constantin et al. "Infinite Family of Persistence Exponents for Interface Fluctuations". *Phys. Rev. Lett.* 91 (8 2003), p. 086103.

[105] S. W. Golomb. *Polyominoes: puzzles, patterns, problems, and packings*. Second. Princeton University Press, 1994.

[106] A. J. Guttmann. *Polygons, Polyominoes and Polycubes*. Lecture Notes in Physics 775. Springer, 2009.

[107] I. Jensen and A. J. Guttmann. "Statistics of lattice animals (polyominoes) and polygons". *J. Phys. A Math. Theor.* 33.29 (2000), pp. L257–L263.

[108] Y. Saito. *Statistical physics of crystal growth*. World Scientific, 1996.

[109] R. Livi and P. Politi. *Nonequilibrium Statistical Physics: A Modern Perspective*. Cambridge University Press, 2017.

[110] R. Bellman. *Dynamic Programming*. Dover Publications, Inc., 2003.

[111] R. E. K. Mosteller F.; Rourke and G. B. Thomas. *Probability and Statistics*. Addison-Wesley, 1961.

[112] E. W. Weisstein. *Random Walk 1-Dimensional*. URL: `https://mathworld.wolfram.com/RandomWalk1-Dimensional.html`.

[113] D. H. Redelmeier. "Counting polyominoes: yet another attack". *Discrete Math.* 36.2 (1981), pp. 191–203.

[114] T. Oliveira e Silva. *Animal enumerations on the* $\{4, 4\}$ *Euclidean tiling*. URL: `http://sweet.ua.pt/tos/animals/a44.html`.

[115] R. Ferrando and G. Tréglia. "Anisotropy of diffusion along steps on the (111) faces of gold and silver". *Phys. Rev. B* 50.16 (1994), pp. 12104–12117.

[116] B. D. Yu and M. Scheffler. "Ab initio study of step formation and self-diffusion on Ag(100)". *Phys. Rev. B* 55.20 (1997), pp. 13916–13924.

[117] H. Mehl et al. "Models for adatom diffusion on fcc (001) metal surfaces". *Phys. Rev. B* 60.3 (1999), pp. 2106–2116.

[118] R. Nelson et al. "Energies of steps, kinks, and defects on Ag{100} and Ag{111} using the embedded atom method, and some consequences". *Surf. Sci.* 295.3 (1993), pp. 462–484.

[119] Y. Han et al. "Kinetics of Facile Bilayer Island Formation at Low Temperature:Ag/NiAl(110)". *Phys. Rev. Lett.* 100.11 (2008).

[120] Y. Han et al. “Formation and coarsening of Ag(110) bilayer islands on NiAl(110): STM analysis and atomistic lattice-gas modeling”. *Phys. Rev. B* 81.11 (2010).

[121] P. García et al. “Photonic Glass: A Novel Random Material for Light”. *Adv. Mater.* 19.18 (2007), pp. 2597–2602.

[122] J. A. Pariente et al. “Vacancies in Self-Assembled Crystals: An Archetype for Clusters Statistics at the Nanoscale”. *Small* 16.42 (2020), p. 2002735.

[123] A. Bortz, M. Kalos, and J. Lebowitz. “A new algorithm for Monte Carlo simulation of Ising spin systems”. *Journal of Computational Physics* 17.1 (1975), pp. 10–18.

[124] A. Prados, J. J. Brey, and B. Sánchez-Rey. “A dynamical monte carlo algorithm for master equations with time-dependent transition rates”. *Journal of Statistical Physics* 89.3-4 (1997), pp. 709–734.

[125] P. J. Van Laarhoven and E. H. Aarts. “Simulated annealing”. *Simulated annealing: Theory and applications*. Springer, 1987, pp. 7–15.

[126] Y. Chen, L. Schomaker, and M. Wiering. “An Investigation Into the Effect of the Learning Rate on Overestimation Bias of Connectionist Q-learning”. *Proceedings of the 13th International Conference on Agents and Artificial Intelligence*. SCITEPRESS - Science and Technology Publications, 2021.

[127] E. Even-Dar, Y. Mansour, and P. Bartlett. “Learning Rates for Q-learning.” *Journal of machine learning Research* 5.1 (2003).

[128] R. Plass et al. “Self-assembled domain patterns”. *Nature* 412.6850 (2001), pp. 875–875.

[129] J. Heinonen et al. “Island Diffusion on Metal fcc (100) Surfaces”. *Phys. Rev. Lett.* 82 (13 1999), pp. 2733–2736.

[130] F. Leroy et al. “Electric forces on a confined advacancy island”. *Phys. Rev. B* 102 (23 2020), p. 235412.

[131] J.-J. Métois, J.-C. Heyraud, and A. Pimpinelli. “Steady-state motion of silicon islands driven by a DC current”. *Surface Science* 420.2-3 (1999), pp. 250–258.

[132] B. VanSaders and S. C. Glotzer. “Sculpting crystals one Burgers vector at a time: Toward colloidal lattice robot swarms”. *Proc. Natl. Acad. Sci. U.S.A* 118.3 (2021), e2017377118.

[133] R. Huang et al. “Direct observations of shape fluctuation in long-time atomistic simulations of metallic nanoclusters”. *Phys. Rev. Materials* 2 (12 2018), p. 126002.

[134] O. Trushin et al. “Atomic mechanisms of cluster diffusion on metal fcc(100) surfaces”. *Surf. Sci.* 482-485 (2001), pp. 365–369.

[135] S. A. Sukhishvili et al. “Diffusion of a polymer ‘pancake’”. *Nature* 406.6792 (2000), pp. 146–146.

[136] A. Weddemann et al. “Magnetic Field Induced Assembly of Highly Ordered Two-Dimensional Particle Arrays”. *Langmuir* 26.24 (2010), pp. 19225–19229.

[137] D. Rabaud et al. “Acoustically Bound Microfluidic Bubble Crystals”. *Phys. Rev. Lett.* 106.13 (2011).

[138] D. Rabaud et al. “Manipulation of confined bubbles in a thin microchannel: Drag and acoustic Bjerknes forces”. *Phys. Fluids* 23.4 (2011), p. 042003.

[139] M. Lanoy et al. “Manipulating bubbles with secondary Bjerknes forces”. *Applied Physics Letters* 107.21 (2015), p. 214101.

[140] J. Palacci et al. “Living Crystals of Light-Activated Colloidal Surfers”. *Science* 339.6122 (2013), pp. 936–940.

[141] F. Ginot et al. “Aggregation-fragmentation and individual dynamics of active clusters”. *Nature Communications* 9.1 (2018).

[142] I. Theurkauff et al. “Dynamic Clustering in Active Colloidal Suspensions with Chemical Signaling”. *Physical Review Letters* 108.26 (2012).

[143] J. Schneiderheinze et al. “High efficiency separation of microbial aggregates using capillary electrophoresis”. *FEMS Microbiology Letters* 189.1 (2000), pp. 39–44.

[144] S. Protière et al. “Sinking a Granular Raft”. *Physical Review Letters* 118.10 (2017).

[145] C. Misbah. *Complex Dynamics and Morphogenesis: An Introduction to Nonlinear Science*. 1st ed. Dordrecht: Springer Netherlands, 2017.

[146] M. Cross and H. Greenside. *Pattern Formation and Dynamics in Nonequilibrium Systems*. Cambridge University Press, 2009.

[147] F. Charru. *Hydrodynamic instabilities*. Cambridge texts in applied mathematics 37. Cambridge University Press, 2011.

[148] E. Ben-Jacob and P. Garik. “The formation of patterns in non-equilibrium growth”. *Nature* 343.6258 (1990), pp. 523–530.

[149] H. Tanaka. “Viscoelastic phase separation”. *Journal of Physics: Condensed Matter* 12.15 (2000).

[150] J. S. Gutmann, P. Müller-Buschbaum, and M. Stamm. “Complex pattern formation by phase separation of polymer blends in thin films”. *Faraday Discussions* 112 (1999), pp. 285–297.

[151] E. Ben-Jacob and H. Levine. “Self-engineering capabilities of bacteria”. *Journal of The Royal Society Interface* 3.6 (2005), pp. 197–214.

[152] Z. Niroobakhsh, M. Litman, and A. Belmonte. “Flow instabilities due to the interfacial formation of surfactant–fatty acid material in a Hele-Shaw cell”. *Physical Review E* 96.5 (2017).

[153] R. A. L. Almeida and K. A. Takeuchi. “Phase-ordering kinetics in the Allen-Cahn (Model A) class: Universal aspects elucidated by electrically induced transition in liquid crystals”. *Phys. Rev. E* 104 (5 2021), p. 054103.

[154] L. McNally et al. “Killing by Type VI secretion drives genetic phase separation and correlates with increased cooperation”. *Nat. Comm.* 8.1 (2017).

[155] D. E. Kirk. *Optimal control theory: an introduction*. Courier Corporation, 2004.

[156] I. Rafols. “Formation of concentric rings in bacterial colonies”. MSc thesis. Chuo University, Japan, 1998.

[157] L. Vassallo, D. Hansmann, and L. A. Braunstein. “On the growth of non-motile bacteria colonies: an agent-based model for pattern formation”. *The European Physical Journal B* 92.9 (2019).

[158] G. Demange et al. “Growth kinetics and morphology of snowflakes in supersaturated atmosphere using a three-dimensional phase-field model”. *Phys. Rev. E* 96 (2 2017), p. 022803.

[159] A. Karma and W.-J. Rappel. “Quantitative phase-field modeling of dendritic growth in two and three dimensions”. *Physical Review E* 57.4 (1998), pp. 4323–4349.

[160] O. Pierre-Louis. “Phase field models for step flow”. *Physical Review E* 68.2 (2003).

[161] L. Cueto-Felgueroso and R. Juanes. “A phase-field model of two-phase Hele-Shaw flow”. *Journal of Fluid Mechanics* 758 (2014), pp. 522–552.

[162] L. M. Pismen. *Patterns and interfaces in dissipative dynamics*. Springer Science & Business Media, 2006.

[163] A. M. Lacasta et al. “Modeling of spatiotemporal patterns in bacterial colonies”. *Physical Review E* 59.6 (1999), pp. 7036–7041.

[164] H. Soner. “Motion of a Set by the Curvature of Its Boundary”. *Journal of Differential Equations* 101.2 (1993), pp. 313–372.

[165] G. J. Fix. “Phase field methods for free boundary problems” (1982).

[166] G. J. Fix. “Numerical simulation of free boundary problems using phase field methods”. *The Mathematics of Finite Elements and Applications* (1982), pp. 265–279.

[167] J. S. Langer. “Models of Pattern Formation in First-Order Phase Transitions”. *Series on Directions in Condensed Matter Physics*. World Scientific, 1986, pp. 165–186.

[168] K. R. Elder et al. “Sharp interface limits of phase-field models”. *Physical Review E* 64.2 (2001).

[169] K. Huang. *Statistical Mechanics*. Wiley, 1987.

[170] N. Provatas and K. Elder. *Phase-Field Methods in Materials Science and Engineering*. John Wiley & Sons, 2010.

[171] J. D. Van der Waals. “The thermodynamic theory of capillarity under the hypothesis of a continuous variation of density”. *Journal of Statistical Physics* 20.2 (1979), pp. 200–244.

[172] S. M. Allen and J. W. Cahn. “A microscopic theory for antiphase boundary motion and its application to antiphase domain coarsening”. *Acta Metall.* 27.6 (1979), pp. 1085–1095.

[173] L. F. Cugliandolo. “Coarsening phenomena”. *Comptes Rendus Physique* 16.3 (2015), pp. 257–266.

[174] H. Abels and Y. Liu. “Sharp Interface Limit for a Stokes/Allen–Cahn System”. *Archive for Rational Mechanics and Analysis* 229.1 (2018), pp. 417–502.

[175] A. J. Bray. “Coarsening dynamics of phase-separating systems”. *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences* 361.1805 (2003). Ed. by T. C. B. McLeish et al., pp. 781–792.

[176] T. Benincasa, L. D. D. Escobar, and C. Moroşanu. “Distributed and boundary optimal control of the Allen–Cahn equation with regular potential and dynamic boundary conditions”. *International Journal of Control* 89.8 (2016), pp. 1523–1532.

[177] P. M. Pardalos and V. A. Yatsenko. *Optimization and Control of Bilinear Systems: Theory, Algorithms, and Applications*. Vol. 11. Springer Science & Business Media, 2010.

[178] P. K. Jaiswal, S. Puri, and K. Binder. “Phase separation in thin films: Effect of temperature gradients”. *EPL (Europhysics Letters)* 103.6 (2013), p. 66003.

[179] L. G. Stanton and A. A. Golovin. “Global feedback control for pattern-forming systems”. *Physical Review E* 76.3 (2007).

[180] M. C. Cross and P. C. Hohenberg. “Pattern formation outside of equilibrium”. *Reviews of Modern Physics* 65.3 (1993), pp. 851–1112.

[181] S. Cox and P. Matthews. “Exponential Time Differencing for Stiff Systems”. *Journal of Computational Physics* 176.2 (2002), pp. 430–455.

[182] K. Kawasaki and T. Ohta. “Kink dynamics in one-dimensional nonlinear systems”. *Physica A: Statistical Mechanics and its Applications* 116.3 (1982), pp. 573–593.

[183] D. Patel et al. “Improved robustness of reinforcement learning policies upon conversion to spiking neuronal network platforms applied to Atari Breakout game”. *Neural Networks* 120 (2019), pp. 108–115.

[184] I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning*. OCLC: 1183962587. 2016.

[185] L. Matignon, G. J. Laurent, and N. Le Fort-Piat. “Reward Function and Initial Values: Better Choices for Accelerated Goal-Directed Reinforcement Learning”. *Artificial Neural Networks - ICANN 2006*. Ed. by S. D. Kollias et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 840–849.

[186] R. Folch et al. “Pattern formation and interface pinch-off in rotating Hele-Shaw flows: A phase-field approach”. *Physical Review E* 80.5 (2009).

[187] G. Caginalp. “Stefan and Hele-Shaw type models as asymptotic limits of the phase-field equations”. *Phys. Rev. A* 39 (11 1989), pp. 5887–5896.

[188] S. Kitsunezaki. “Interface Dynamics for Bacterial Colony Formation”. *Journal of the Physical Society of Japan* 66.5 (1997), pp. 1544–1550.

[189] E. Ben-Jacob et al. “Adaptive self-organization during growth of bacterial colonies”. *Physica A: Statistical Mechanics and its Applications* 187.3-4 (1992), pp. 378–424.

[190] W. W. Mullins and R. F. Sekerka. “Morphological Stability of a Particle Growing by Diffusion or Heat Flow”. *Journal of Applied Physics* 34.2 (1963), pp. 323–329.

[191] J. S. Langer. “Instabilities and pattern formation in crystal growth”. *Reviews of Modern Physics* 52.1 (1980), pp. 1–28.

[192] P. Ball. *Branches. Nature’s patterns: a tapestry in three parts*. Oxford University Press, USA, 2009.

[193] L. Paterson. “Radial fingering in a Hele Shaw cell”. *Journal of Fluid Mechanics* 113.-1 (1981), p. 513.

[194] P. G. Saffman and G. I. Taylor. “The penetration of a fluid into a porous medium or Hele-Shaw cell containing a more viscous liquid”. *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences* 245.1242 (1958), pp. 312–329.

[195] J. C. Butcher. *Numerical methods for ordinary differential equations*. John Wiley & Sons, 2016.

[196] R. Kupferman, O. Shochet, and E. Ben-Jacob. “Numerical study of a morphology diagram in the large undercooling limit using a phase-field model”. *Phys. Rev. E* 50 (2 1994), pp. 1005–1008.

[197] Y. Mingqiang, K. Kidiyo, and R. Joseph. “A Survey of Shape Feature Extraction Techniques”. *Pattern Recognition Techniques, Technology and Applications*. InTech, 2008.

[198] F. Mokhtarian and A. Mackworth. “Scale-based description and recognition of planar curves and two-dimensional shapes”. *IEEE transactions on pattern analysis and machine intelligence* 1 (1986), pp. 34–43.

[199] F. Mokhtarian and M. Bober. *Curvature scale space representation: theory, applications, and MPEG-7 standardization*. Vol. 25. Springer Science & Business Media, 2013.

[200] F. Mokhtarian. “Silhouette-based isolated object recognition through curvature scale space”. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 17.5 (1995), pp. 539–544.

[201] F. Mokhtarian, S. Abbasi, and J. Kittler. “Robust and Efficient Shape Indexing through Curvature Scale Space”. *Procedings of the British Machine Vision Conference 1996*. British Machine Vision Association, 1996.

[202] S. Abbasi, F. Mokhtarian, and J. Kittler. “Curvature scale space image in shape similarity retrieval”. *Multimedia Systems* 7.6 (1999), pp. 467–476.

[203] S. V. Jablan and S. Radmila. *LinKnot: knot theory by computer*. Vol. 21. World Scientific, 2007.