# Exploiting structure in humanoid motion planning

Andreas Orthey


# THÈSE

**En vue de l'obtention du**

# DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

**Délivré par :**

Institut National Polytechnique de Toulouse (INP Toulouse)

**Discipline ou spécialité :**

Robotique et Informatique

---

## Présentée et soutenue par :

M. ANDREAS ORTHEY

le jeudi 24 septembre 2015

## Titre :

EXPLOITER LA STRUCTURE POUR LA PLANIFICATION DE MOUVEMENT HUMANOIDE

---

**Ecole doctorale :**

Systèmes (Systèmes)

**Unité de recherche :**

Laboratoire d'Analyse et d'Architecture des Systèmes (L.A.A.S.)

**Directeur(s) de Thèse :**

M. OLIVIER STASSE

**Rapporteurs :**

Mme MAREN BENNEWITZ, FREIBURG UNIVERSITAT

M. PHILIPPE FRAISSE, UNIVERSITE MONTPELLIER 2

**Membre(s) du jury :**

M. PHILIPPE SOUERES, LAAS TOULOUSE, Président

M. OLIVIER STASSE, CNRS VITRY, Membre

M. SETHU VIJAYAKUMAR, UNIVERSITY OF EDINBURGH, Membre

# Acknowledgement

Foremost, I would like to thank my supervisor Olivier Stasse. Thank you so much for helping me to kickstart my life in Toulouse, for removing all those administrative obstacles, for giving me the freedom to pursue my own ideas, and for always grounding my abstract ideas in reality. If I look back, I couldn't have been luckier to have someone like you as a supervisor.

I would like to thank equally Sethu Vijayakumar, who was a great supporter during my time in Edinburgh. And to Vladimir Ivan and Yiming Yang for being amazing lab mates. To Philippe Fraisse and Maren Bennewitz to review my manuscript. To Eiichi Yoshida, for helping me with my research proposals and for being part of my jury.

Special thanks to the whole team GEPETTO. For accompanying me those three years. For all the creative input, especially from Philippe, Jean-Paul, and Florent, who gave me a lot of inspiration. For the friendly atmosphere: Nicolas, Michel, Bruno, Justin, Max, Mehdi, Joseph, Steve, Ganesh, Mathieu, Galo, Paolo, Aiva, Mylene, and Naoko. And of course to our visitors, Mukunda, Coline, and Ixchel. You made me really feel at home. Thanks to all the individuals who inspired me. To Antonio, who build the wall leading us to discover irreducible trajectories. To Olivier Roussel, a constant source of counter-examples, discussions about lie group theory, and for being a good friend. To Andrea del Prete, for many tips and advices for the upcoming postdoc life. And finally, thank you Christian

and Nemo. You made my time in our group really enjoyable. Thank you!

Of course, this thesis would not have existed without those masses of people, of people who contributed to our current stable political situation, of people who inspired me by their science-fiction novels, of people who discovered our current knowledge in physics, computer science, mathematics. Too many to name.

Thanks to my family and friends in Germany. Thanks to everyone who supported me.

# Abstract

If humanoid robots should work along with humans and should be able to solve repetitive tasks, we need to enable them with a skill to autonomously plan motions. Motion planning is a longstanding core problem in robotics, and while its algorithmic foundation has been studied in depth, motion planning is still an NP-hard problem lacking efficient solutions. We want to open up a new perspective on the problem by highlighting its structure: the behavior of the robot, the mechanical system of the robot, and the environment of the robot. We will investigate the hypothesis that each structural component can be exploited to create more efficient motion planning algorithms. We present three algorithms exploiting structure, based on geometrical and topological arguments: first, we exploit the behavior of a walking robot by studying the feasibility of footstep transitions. The resulting algorithm is able to plan footsteps avoiding up to 60 objects on a 6 square meters planar surface. Second, we exploit the mechanical system of a humanoid robot by studying the linear linkage structures of its arms and legs. We introduce the concept of an irreducible motion, which is a completeness-preserving dimensionality reduction technique. The resulting algorithm is able to find motions in narrow environments, where previous sampling-based methods could not be applied. Third, we exploit the environment by reasoning about the topological structure of contact transitions. We show that analyzing the environment is an efficient method to precompute relevant information for efficient motion planning. Based on those

results, we come to the conclusion that exploiting structure is an essential component of efficient motion planning. It follows that any humanoid robot, who wants to act efficiently in the real world, needs to be able to understand and to exploit structure.

# Abstract

Afin que les robots humanoïdes puissent travailler avec les humains et être en mesure de résoudre des tâches répétitives, nous devons leur permettre de planifier leurs mouvements de façon autonome. La planification de mouvement est un problème de longue date en robotique, et tandis que sa fondation algorithmique a été étudiée en profondeur, la planification de mouvement est encore un problème NP-difficile et qui manque de solutions efficaces. Nous souhaitons ouvrir une nouvelle perspective sur le problème en mettant en évidence sa structure: le comportement du robot, le système mécanique du robot et l'environnement du robot. Nous allons nous intéresser à l'hypothèse que chaque composante structurelle peut être exploitée pour créer des algorithmes de planification de mouvement plus efficaces. Nous présentons trois algorithmes exploitant la structure, basés sur des arguments géométriques et topologiques: d'abord, nous exploitons le comportement d'un robot de marche en étudiant la faisabilité des transitions des traces de pas. L'algorithme qui en résulte est capable de planifier des traces de pas tout en évitant jusqu'à 60 objets situés sur une surface plane 6 mètres carrés. Deuxièmement, nous exploitons le système mécanique d'un robot humanoïde en étudiant les structures des liaisons linéaires de ses bras et de ses jambes. Nous introduisons le concept d'une trajectoire irréductible, qui est une technique de réduction de dimension préservant la complétude. L'algorithme résultant est capable de trouver des mouvements dans des environnements étroits, où les méthodes d'échantillonnage

ne pouvaient pas être appliquées. Troisièmement, nous exploitons l'environnement en raisonnant sur la structure topologique des transitions de contact. Nous montrons que l'analyse de l'environnement est une méthode efficace pour pré-calculer les informations pertinentes pour une planification de mouvement efficace. En s'appuyant sur ces résultats, nous arrivons à la conclusion que l'exploitation de la structure est une composante essentielle de la planification de mouvement efficace. Il en résulte que tout robot humanoïde, qui veut agir efficacement dans le monde réel, doit être capable de comprendre et d'exploiter la structure.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Robotics has influenced our society in profound ways. Nowadays, we have vacuum cleaners autonomously operating in our houses, manufacturing robots assembling aircrafts, and drones autonomously mapping and surveying our planet.

*"Motion is one important building block of intelligent Behavior"*

However, many tasks are out of reach for current robot technology. Robots still cannot replace construction workers, cannot enforce law and order, and cannot autonomously mine natural resources. All those tasks require sufficiently agile robots, robots which can act in the same way as human beings. Humanoid robots have therefore become an important research direction in robotics, with many research groups focusing on the design and construction [1][2][3][4][5][6], and on the control and stabilization [7][8][9][10] of humanoid robots. Despite those focused efforts, no humanoid robot is able to match the agility, the speed and the real-world problem-solving skills of a human being. It is therefore natural to ask what is the missing piece to bring humanoid robots into the real-world.

I am arguing here that one missing piece is a lack of true understanding of motion. Motion is one important building block of intelligent behavior: if we want to understand the function of a bicycle, we need to interact with the bicycle, which requires motion. If we want to write a letter, we need to coordinate our fingers, which requires motion. If we want to build a spaceship, we need to move and concatenate small pieces, which requires motion.

It comes to no surprise that planning a motion — the motion planning problem — is therefore one of the core problems in robotics. The field of *motion planning* has become the principled way of formalizing and understanding motions, as described in the comprehensive book "Planning Algorithms" by Steve LaValle [11], which forms the corner-stone of modern motion planning research. One key area of motion planning are humanoid robots. Humanoid robots are of particular interest, because they are interesting to study in their own right, involving high-dimensional problems of stabilization, planning and reasoning, and because they could enable technological breakthroughs, involving automatization of the agricultural, food and the service industry. The book "Motion Planning for Humanoid Robots" by Editors Harada, Yoshida and Yokoi [12] gives a comprehensive overview. The key message is: the problem is theoretically solved. However, the problem lies in one of the hardest computational complexity classes and so a problem which can be solved by a human in one second, can be solved by a humanoid robot only in hours or days. As a particular example, one experiment from this thesis found that with state-of-the-art methods moving a robot through a narrow environment like a wall (see Figure 4.8) took in the worst case up to 44 hours [13].

Clearly, 44 hours is too much if we want to deploy humanoid robots in the real-world. I am arguing here, that we can reduce the planning time if we are able to understand and to exploit the structure of the underlying motion planning problem. This argument will form my main hypothesis: Efficient motion planning

**Figure 1.1:** Motion planning for humanoid robots is computationally hard, but could enable breakthrough technologies and could provide us with a deeper algorithmic understanding of intelligent behavior. Left: Robot at the DARPA robotics challenge manipulating a valve. Right: Robot HRP-2 walking through a wall. *Orthey, Andreas. 2015.*

requires exploitation of structure. To show that this hypothesis is valid, i will use a constructive approach by developing specifically tailored algorithms, which each take advantage of a particular structural component of the problem.

To be able to do that, we have to clarify which components are contributing to motion planning. Informally, we have identified three main components: the desired behavior of the robot, the mechanical system of the robot, and the environment surrounding the robot. We claim that each component can be exploited to yield more efficient motion planning algorithms. I will set out to show the correctness of this claims in the chapters that follow.

*"Hypothesis: Efficient motion planning requires exploitation of structure"*

We start in Chapter 2 to simplify contact motion planning [14][7] by exploiting common walking behavior of a humanoid robot. Our algorithm is able to achieve realtime motion planning by using a large set of simplifications. While being computationally very efficient, the methods developed in this chapter are not complete, and will not work in very narrow environments.

The key contribution of this thesis is the introduction of irreducible trajectories in Chapter 3, where we prove that a dimensionality reduction from the space of all possible trajectories to the space of irreducible trajectories preserves completeness. We show that this concept can be applied to the arms of a humanoid robot, to simplify motion planning in narrow passage.

Since narrow passage in configuration space are clearly visible in workspace, we have exploited the workspace topology in Chapter 4. Instead of planning in configuration space, we formulate the complete motion planning problem in the workspace as the problem of planning for a set of particles. Most importantly, using this novel view on the problem we have been able to estimate the number of local minima in a specific case.

## 1.1 Contributions

To summarize the contributions of this thesis:

1. Chapter 2 investigates methods to simplify motion planning by exploiting a desired behavior. If the robot only uses upright walking behavior, then we can precompute swept volume structures for feasibility analysis. In particular, we precompute if a transition between two contact points is feasible, given behavior and simple environment primitives

2. In Chapter 3 we exploit linear linkage structures, which are a main component of the mechanical system of a humanoid robot. We introduce the key concept of an irreducible trajectory, which is an abstract notion of how we can reduce the dimensionality of motion planning while preserving completeness.

3. In Chapter 4 we exploit the topological structure of a given environment. In particular, we exploit the homotopy classes by conducting continuous optimization for a set of particles in workspace.

## 1.2 Publications

Parts of this thesis have been made public in Journal or Conference articles. The complete list of publications from this thesis is given here as a reference

**JOURNALS**

- [15] **A.Orthey**, O.Roussel, O.Stasse, M.Taix *Irreducible Motion Planning by Exploiting Linear Linkage Structures* , Transactions on Robotics (T-RO), 2015, **(Submitted to)**

**CONFERENCES**

- [16] **A.Orthey**, V.Ivan, M.Naveau, Y.Yang, O.Stasse, S.Viyajakumar, *Homotopic Particle Motion Planning*, International Conference on Humanoid Robots (Humanoids), Seoul, Korea, 2015, **(Submitted to)**

- [13] **A.Orthey**, O.Stasse, F.Lamiraux, *Motion Planning and Irreducible Trajectories*, International Conference on Robotics and Automation (ICRA), Seattle, WA, USA, 2015

- [17] O.Stasse, **A.Orthey**, F. Morsillo, M. Geisert, N. Mansard, M.Naveau, C.Vassallo, *Airbus/Future of Aircraft Factory, HRP-2 as Universal Worker Proof of Concept*, International Conference on Humanoid Robots (Humanoids), Madrid, Spain, 2014

- [18] **A.Orthey**, O.Stasse, *Towards Reactive Whole-Body Motion Planning in Cluttered Environments by Precomputing Feasible Motion Spaces*, International Conference on Humanoid Robots (Humanoids), Atlanta, GA, USA, 2013

## 1.3 Related Work

The study of planning a motion in arbitrary environments for a mechanical system $\mathcal{R}$ is called *motion planning*. Motion planning resolves around the concepts of degrees-of-freedom (DOF) and the configuration space (C-space). The DOFs of a mechanical system $\mathcal{R}$ are defined as the maximum number of independent parameters which fully specify the transformations applied to the system [11]. The space spanned by the parameters of the DOFs is called the C-space, denoted by $\mathcal{C}$. The C-space is divided into the free C-space $\mathcal{C}_f$, which contains all those configurations which are physically feasible, i.e. they respect the following constraints

1. Joint Limits
2. Self-Collision
3. Stability
4. Collision with Environment

Given an initial configuration $q_i \in \mathcal{C}_f$ and a goal configuration $q_G \in \mathcal{C}_f$, the goal of motion planning is to find a continuous trajectory $\tau : [0, 1] \rightarrow \mathcal{C}_f$, which lies entirely in $\mathcal{C}_f$.

Since the seminal works by Lozano-Perez [19], Reif [20] and Canny [21], the main focus of motion planning was on the algorithmic side. In particular, sampling-based methods like probabilistic roadmaps (PRM) [11] and the rapidly-exploring random tree (RRT) [22] have been successfully applied not only in robotic settings [23], [9],[24], but also in protein folding tasks [25],[26], [27]. Research effort in the last years has been concentrated immensively on the algorithmic development of sampling-based methods. Many variants of the RRT have been proposed, among the most prominent ones are: bidirectional RRT (BiRRT) [11] which grows simultaneously two trees in C-space; RRT* [28], the probabilistic optimal variant of

RRT; CBiRRT [29] introduces workspace task regions into RRT; AtlasRRT [30] builds the topological structure of the configuration space to facilitate sampling. A recent survey for sampling-based methods in motion planning has been published [31]. For a basic tutorial and future directions consider the two-part paper by LaValle [32], [33].

While algorithmic developments are an important component for handling an NP-hard problem, algorithmic developments are not the only possibility to approach the problem. I am arguing here, that it is at least equally important to understand and exploit the structure of the underlying problem. Having a good understanding of the structure, we can then use this understanding to guide algorithmic design.

As motivated in Chapter 1, our interest lies in motion planning for humanoid robots [34]. Humanoid robots have been successfully employed for many tasks including climbing walls [35][14], climbing ladders [9], cooking pancakes [36], opening doors [37], manipulating valves [38], kicking footballs [39], and moving through narrow wall passages [13]. In the recent DARPA robotics challenge semi-autonomous robots showed remarkable real-world results. However, to obtain fully autonomous humanoid robots, we argued in Chapter 1 that we need to exploit structure. Structure can be found in the mechanical system of the robot, in the environment, or in intended the behavior. To exploit this structure, research in humanoid robotics has concentrated on finding an efficient decomposition of the problem.

The most basic decomposition is *footstep planning*, whereby first we plan footsteps, and then in a second step, along the footsteps, we plan upper-body motions. Different variants of this concept have been employed on different robotic systems. The most prominent examples include the Asimov robot from HONDA [40], HRP-2 [41],[18] from AIST in Japan, NAO from Aldebaran Robotics, [42],[43] and ATLAS [7] from Boston Dynamics. The goal of footstep planning is to restrict the con-

figuration space of the robot to those subspaces which have valid contacts. While being pratically efficient, however, we sacrifice completeness if we apply footstep planning.

Another line of research has focused on decomposing motion planning by first finding multi-body contacts for the robot, and then using those contacts to search for whole-body configuration. A generalization of footstep planning to multi-contact planning. Succesful systems have been designed by Hauser [14], and Escande [44]. An important simplification for multi-contacts is the quantity of force applicable at a contact, formalized as the force ellipsoid [45]. Tonneau et al. [46] used this concept in motion planning for virtual avatars to achieve near real-time performance. While those heuristics work well in practice, again, we sacrifice completeness by decomposing the problem.

A completeness-preserving decomposition can be obtained by first computing the topological structure of the environment, and afterwards planning whole-body configurations only in the path-connected parts of the environment. This has been pioneered by Brock and Kavraki [47], who developed a sphere expansion algorithm to efficiently find a tunnel, a path-connected component in the environment between initial and goal configuration. In an extension of this work, Yang and Brock [48] investigated how trajectories can be deformed if the tunnel deforms. Deforming a trajectory based on deformation of the environment gives us a clue that there might be a mapping from environment to trajectories, which can be learned with machine learning techniques. Jetchev and Toussaint [49] developed such a machine learning based mapping for a robot manipulator. A recent rigourous treatment of exploiting the topological structure of the environment is given by Bhattacharya et al. [50]. Farber [51] investigated the topological complexity of the configuration space, a notion which could be used to analyse the complexity of the environment. In Chapter 4 we build upon those works to decompose the contact surfaces

of the environment in its homotopy classes and develop a continuous trajectory optimization method inside each homotopy class.

Despite decomposing the problem into appropriate subproblems, we can exploit structure also by dimensionality reduction techniques. The core idea is to ignore dimensions of the planning problem, which might not contribute as much to the solution. For example the configuration of the finger joint of the hand will not contribute as much to the change in the swept volume compared to the hip joint. Simplifications based on dimensionality reduction have been for example employed for directing sampling-based planners [52], in manipulation planning [53][54] and on deformable robots [55]. We will review dimensionality reduction techniques in more detail in Chapter 3, where we also introduce the concept of irreducible trajectories, which provides a principled framework to reduce the dimensionality of linkage structures.

More conceptually, we like to point out that exploiting structure is a general strategy to simplify problems, a strategy to find efficient solutions to NP-hard problems and a strategy to acquire understanding. Hendrickson [56] exploited structure to efficiently solve an NP-hard global optimzation problem related to finding invariances in molecular structures. Poupart [57] investigated general structural components in markov decision processes (MDPs) to generate efficient solving strategies. Ryan [58] exploited graph structures to simplify multi-robot path planning. Burns and Brock [59] explicitly try to exploit problem structure by steering sampling-based methods to high-utility regions. Hutchinson [60] exploited visual features in motion planning. Even highly successful mathematical theories can be seen as mechanisms to exploit structure. Group theory [61] exploits symmetrical structures of objects and transformations, topology [62] exploits deformation invariances in geometrical structure, and convex optimization [63] exploits the structure of specific functional spaces. Motivated by those successes, this thesis is

a first step to investigate the structural components in humanoid motion planning in a principled way. The goal of exploiting structure in motion planning is twofold: first we want to acquire a deeper understanding of motion planning and second, we want to develop highly efficient motion planning algorithms.

# Chapter 2

# Reactive Whole-Body Motion Planning

*"To perceive is [..] to learn how the environment structures one's possibilities for [..] action afforded by the environment."*

— Alva Noë, Action in Perception

## 2.1 Summary

To solve complex whole-body motion planning problems in near real-time, it is essential to precompute as much information as possible, including our intended movements and how they affect the geometrical reasoning process. In this chapter, we focus on the precomputation of the feasibility of contact transitions in the context of discrete contact planning. Our contribution is twofold: First, we introduce the *contact transition*

*"We demonstrate how the prior knowledge about object geometries can achieve near real-time performance in highly-cluttered environments"*

*and object* (CTO) space, a joint space of contact states and geometrical information. Second, we develop an algorithm to precompute the decision boundary between feasible and non-feasible spaces in the CTO space. This boundary is used for online-planning in classical contact-transition spaces to quickly prune the number of possible future states. By using a classical planning setup of A* together with a $l_2$-norm heuristic, we demonstrate how the prior knowledge about object geometries can achieve near real-time performance in highly-cluttered environments, thereby not only outperforming the state-of-the-art algorithm, but also having a significantly lower model sparsity.

## 2.2 Introduction

Consider the problem in Fig. 2.1: A highly-cluttered environment has to be traversed by a humanoid robot, without stepping onto obstacles on the ground. The goal is to find a set of footsteps, which allows us to move the robot towards the goal region. This problem is problematic from different point of views: First, for each footstep we want to take, we have to compute a control law for each degree of freedom of the robot, such that we fulfill certain constraints like joint limits, dynamics and stability. Second, we have to check if the body of the robot is in collision with objects in the environment. Due to the large number of objects and the nearness of the robot to the objects, this is generally not possible in real-time. In this chapter, we provide an algorithm, which generates a footstep path for a humanoid robot which is faster than the state-of-the-art approach, and runs in real-time even for challenging environment like the one in Fig. 2.1, where 30 objects are randomly placed.

The underlying problem is the one of real-time planning of motions for a high-dimensional degrees of freedom robot. We approach this problem by using an

approximation method to precompute if a motion between two contact points will be feasible. Our contribution is twofold: First, we introduce the *contact transition and object* (CTO) space: The union of a reduced set of contact points and the parameters of approximated objects in the environment. Second, we perform an analysis of the decision boundary between feasible and non-feasible subspace within the CTO space. We hereby focus on a sparse and approximate representation of this boundary, which allows us to discriminate very fast between feasible and non-feasible contact points.

This work can be seen as an additional simplification of planning in the contact space of a robot [44]. It further develops the ideas of [41], which used the swept volume – defined as *"the space, occupied by a robot during the execution"* [64] – to precompute if a motion between two contact points will be feasible. We further advance this precomputation idea by including the geometrical information of objects in the environment.

The chapter is organized by first considering related work in Section 2.3. Section 2.4 will focus on background in contact planning, motion generation and swept volume approximation. In Section 2.5 we introduce the CTO space, Section 2.6 discusses the sampling and approximation of the feasibility function, and Section 2.7 demonstrates the applicability of our approach in a highly-cluttered environment.

## 2.3 Related Work

We provide in this section a basic overview about contact planning, with emphasis on footstep planning but also on how to construct a general contact space framework. For each approach, we focus on its relation to our work.

Chestnutt et al. [65, 40] pose the problem of footstep planning as a discrete

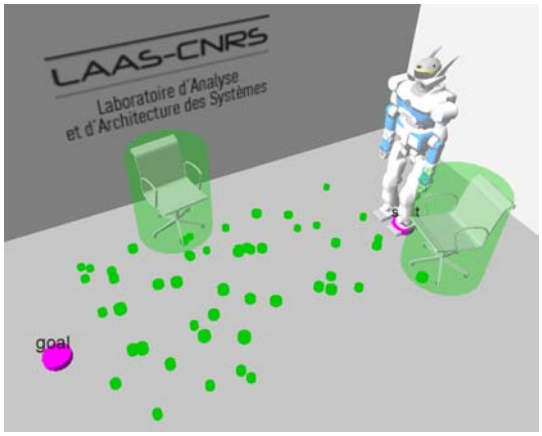



**Figure 2.1**

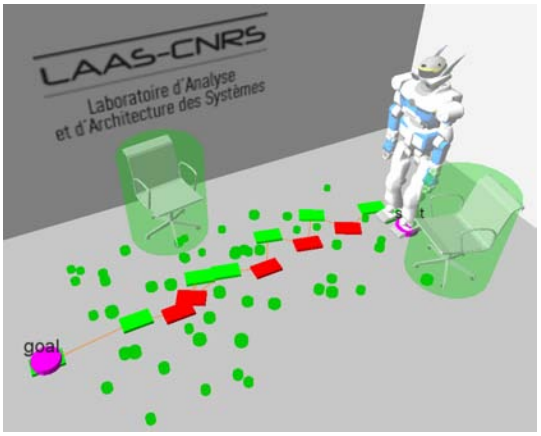



**Figure 2.2**

**Figure 2.3:** Left: A highly-cluttered environment with 30 randomly placed objects, where the robot has to avoid stepping onto objects, while reaching a goal region. Right: Our algorithm finds a feasible footstep path in $0.3s$ by (1) approximating objects with simple geometrical shapes and (2) adding this geometrical information to the precomputation of the feasibility of motions.

search problem, and are approximating its heuristic by a mobile robot planner. Complementary to their work we use a simple heuristic, and instead focus solely on a fast decision about which steps will be feasible in the present of obstacles.

Escande et al. [44] provide a complete framework for multi-contact planning, in which they investigate how to choose contact points, and how to generate paths between them. Our work focuses on the first point, for which we provide an approximate solution.

Hauser et al. [14] are planning general multi-contact points for a humanoid climbing robot. Their approach focuses on using motion primitives of contact points as an initial trajectory for a sampling based algorithm. While their work is concerned with finding a probabilistically complete algorithm, we focus on simplifications for real-time planning.

Hornung et al. [43] are using a anytime variant of the A* algorithm to plan footsteps for the Nao robot. This can be seen complementary to our work, in which we try to approximate the feasibility of footstep transitions.

Perrin et al. [41] are using swept volumes to approximate the contact transition between footsteps. While they require the storage of complete swept volumes for collision checking, we devised an approximate mapping from contact points to feasibility by incorporating the object geometry directly into the precomputation process.

## 2.4 Background

Our approach is based on three core topics, which will be explained in the following sections: First, we introduce the concept of contact-space planning to reduce the dimensionality of a robotic system in Section 2.4.1. Second, we discuss how the whole-body motion of a robot is generated between two contact points in Section 2.4.2, and finally, we introduce approximation via swept volumes in Section 2.4.3.

### 2.4.1 Planning in Contact Space

Planning a movement for a robotics system, with many degrees of freedom (dof), is commonly seen as intractable, because their complexity is exponential in the number of dof [11]. A simplification, which reduces the planning dimensions, is the contact-space planning approach [65, 44, 14]. Planning is posed as a discrete search problem of finding a sequence of contact-points, which can be used to reach a desired goal region. For transitions between contact-points, local optimization methods can be used. In our work, we will make the further simplification, that contact-points are restricted to footsteps. The long-term goal of our research is the inclusion of hand-environment contacts, which is why we formulate our approach in terms of general contact-points, rather than foot-contacts. We also note, that we are interested in fast real-time planning methods, which is contrary to algorithms which try to find a complete trajectory in the general contact-point

space [14, 44]. Earlier research in motion planning made this distinction explicit by dividing algorithms into coarse and fine motion planning [66] — whereby our work can be considered coarse motion planning.

### 2.4.2 Optimal whole-body motion between contact points

For finding a trajectory between two *contact points* $\mathbf{x}_I$ and $\mathbf{x}_G$, we assume that there is an optimization function $p : \mathbb{R}^M \times \mathcal{K} \to \mathbb{R}^d$, which maps a contact point $\mathbf{x}$, of dimension $M$, into a joint configuration $q$, of dimension $d$, which we will call a *contact configuration*. The space $\mathcal{K}$ defines a behaviour of the robot, i.e. how the rest of the body is positioned. Given one behaviour, and assuming zero noise, the mapping $p$ is uniquely defined, so that we can further operate on contact configurations, without loss of generality. Between two contact configurations $q_I$ and $q_G$, we then utilize a local optimization function formalized as a classical optimal control problem

$$\begin{aligned} \underset{u}{\text{minimize}} \quad & \int_{t_0}^{t_f} C(u(t), q(t))dt \\ \text{subject to} \quad & \dot{q}(t) = f(q(t), u(t)) \end{aligned}$$

whereby $q(t)$ is the configuration at time $t$, $u(t)$ is the control input, $f$ is the dynamics of the robot, and $C$ is the cost function, which could depend on the task and the behaviour we want to achieve. We now assume the existence of an algorithm $g$, which solves the whole-body generation problem between two contact configurations:

$$q_{q_I \to q_G} = g(q_I, q_G, C) \tag{2.1}$$

whereby $q_{q_I \to q_G}$ is the final trajectory of the robot, $q_I$ and $q_G$ are the start and the goal configurations, and $C$ is the above mentioned cost function. Besides being a non-chaotic system, we make no restrictions on the optimization algorithm $g$ and the cost function $C$. Therefore, we can make use of potential functions [11], nonlinear attractors like the dynamical motion primitive [67], stochastic optimal control solvers [68], or – as in our case – a hierarchical null space control framework, called the stack of task [69]. In this case, we use costs depending on distance to self collisions, distance to joint limits, and dynamical stability.

In the absence of noise, we assume that the optimization problem is uniquely defined, i.e. for a pair of $q_I$,$q_G$, optimizer $g$, and cost function $C$, $g$ returns one unique trajectory.

### 2.4.3 Swept-Volume Approximations

The unique trajectory from Eq. (2.1) defines directly a swept volume of the robot body [41], which we will denote as $\mathcal{SV}_{q_I,q_G}$. The number of possible contact transitions is infinite and needs to be reduced to make planning computationally tractable. We therefore use a set of $N$ contact points, which are a discretization of all mechanically feasible footsteps of the robot. This implies the computation of $\binom{N}{2}$ swept volumes (one for each transition pair). By adding a waypoint, as reported in [41], one can assume, that each transition will have a common end point, which prunes the number of swept volumes to $N$. Using this setting, Perrin et al. [41] have demonstrated real-time motion planning in a constrained environment with fixed upper body and stepping capabilities. Our goal in the next section is to show, how to speed up this approach by (1) introducing the geometry of objects directly into the precomputation algorithm and (2) approximating the decision boundary between feasible and non-feasible space in the joint space of objects and contact points.

## 2.5 Contact Transition and Object (CTO) Space

To plan a discrete set of contacts for a robot, we want to precompute if the transition between two contact points is *feasible*. The feasibility is a function of the environment and the underlying controller. It is therefore necessary to represent the environment, which we do by using an object-centered approach and by fitting generalized geometrical shapes to those objects.

To decide if a contact transition will be feasible, a common approach [41] is to use precomputed swept volumes for each contact transitions and check each swept volume for collisions with all visible objects in scene. In this work, we go one step back and analyse directly the joint space of contact points $\mathcal{X}$ and objects $O$. Instead of recalling the swept volume and doing collision checking to determine feasibility, our goal is to approximate a feasibility function $f : \mathcal{X} \times O \to \mathbb{R}$ directly by learning a discriminative function of the form $\hat{f} : \mathcal{X} \times O \to \mathbb{R}$, such that we minimize the distance between them.

For making this tractable, we apply two simplifications: First, we use a discrete set of contact points $\tilde{\mathcal{X}}$, which was obtained from all possible contact points $\mathcal{X}$ by (A) utilizing the symmetries of the robot body and a waypoint contact as discussed in section 2.4.3, (B) uniformly discretizing contact points from $\mathcal{X}$, and (C) pruning contact points not satisfying internal constraints — like joint limits and self collisions. This provides us with a set of $N$ contact points, which all have the same common goal contact point $\mathbf{x}_G$. For example to go from an arbitrary contact $\mathbf{x}_0$ to another contact $\mathbf{x}_2$, we concatenate $\mathbf{x}_0$ to $\mathbf{x}_G$, and $\mathbf{x}_G$ to $\mathbf{x}_2$. The contact points are a set with an underlying structure, in this case an geometrical ordering (position of contacts) and a metric (distance between contacts). Set and structure define together a mathematical space, such that we can define:

**Definition 1** (Reduced Contact-Transition Space)**.** *A discrete set of contact points*

$\mathbf{x}_0, \cdots, \mathbf{x}_N$, *which have a common goal contact point* $\mathbf{x}_G$

$$\tilde{\mathcal{X}} = \{\mathbf{x}_0, \cdots, \mathbf{x}_N\} \tag{2.2}$$

In this chapter, one contact point is defined as $\mathbf{x} = \{(x, y, \theta, \bar{q})^T | x, y, \theta \in SE(2), \bar{q} \in \{L, R\}\}$, whereby $x$, $y$ are the middle point of one foot, and $\theta$ is the inclination, and $\bar{q}$ is the support foot.

Second, we observe that the detailed shape of an object is not important for coarse motion planning [66], where one is interested in a first reasonable guess of the trajectory. We therefore build the reduced object space $\tilde{O}$ from the complete object space $O$ by assuming that objects can be approximated by basic geometrical shapes. As an intermediate representation between a set of basic shapes (cylinder, sphere, box) and a complete mesh triangle representation, we utilize a generalization of basic shapes, called the superellipsoid. The superellipsoid allows us to describe different basic shapes by one formula with a sparse set of parameters [70]

$$S(x, y, z; \vec{\theta}, \vec{\lambda}) = \left( \left( \frac{x}{\lambda_1} \right)^{\frac{2}{\theta_2}} + \left( \frac{y}{\lambda_2} \right)^{\frac{2}{\theta_2}} \right)^{\frac{\theta_2}{\theta_1}} + \left( \frac{z}{\lambda_3} \right)^{\frac{2}{\theta_1}} \tag{2.3}$$

whereby $\vec{\theta} > 0$ specifies the shape (e.g. a cylinder), and $\vec{\lambda} > 0$ specifies the elongations along the axes (e.g. the height and radius of a cylinder). Eq. (2.3) is called the inside-outside function, referring to points $x, y, z$ as being outside the object for $S(x, y, z) > 1$ and inside or on the surface for $S(x, y, z) \leq 1$. Examples include the ellipsoid ($\theta_1 = 1, \theta_2 = 1$), cylindroid ($\theta_1 \ll 1, \theta_2 = 1$) and the quader ($\theta_1 \ll 1, \theta_2 \ll 1$). For this work, we restrict objects to the cylindrical space by defining

**Definition 2** (Reduced Object Space)**.** *The set of objects* $\mathbf{o}$*, which can be approx-*

*imated by a superellipsoid in the form*

$$\begin{aligned}\tilde{O} = \{(x, y, \phi, \vec{\theta}, \vec{\lambda})^T | x, y, \phi \in SE(2), \\ \vec{\theta} = (0.01, 1)^T, \\ \vec{\lambda} \in \mathbb{R}_+\}\end{aligned} \tag{2.4}$$

Together with the contact points, we can now define the CTO space:

**Definition 3** (Contact Transition and Object Space)**.** *The union of reduced contact space and reduced object space*

$$\mathcal{C}_{CTO} = \{\tilde{\mathcal{X}} \cup \tilde{O}\} \tag{2.5}$$

Having defined the $\mathcal{C}_{\text{CTO}}$ space, the rest of the chapter is devoted to the computation of the decision boundary between the feasible subspace and the non-feasible subspace. This is formulated as finding a discriminative function $\hat{f}$, which minimizes an optimization problem of the form

$$\begin{aligned}\underset{\hat{f}}{\text{argmin}} \quad & ||f(\mathbf{x}, \mathbf{o}) - \hat{f}(\mathbf{x}, \mathbf{o})||_2 \\ \text{s.t.} \quad & \mathbf{o} \in \tilde{O}, \mathbf{x} \in \tilde{\mathcal{X}}\end{aligned}$$

Whereby $f$ and $\hat{f}$ are computing the feasibility of a contact transition as depicted in Fig. 2.4: $f$ first optimizes a controller to traverse the contact points, then computes the swept volume along its trajectory and finally conducts collision checking with objects in the environment; $\hat{f}$ simplifies this computation by acting as a discriminative function for the $\mathcal{C}_{\text{CTO}}$ space, to directly decide if a contact transition and an object are in the feasible subspace. In the next section, we will focus on the sampling of $f$ and its approximation $\hat{f}$.

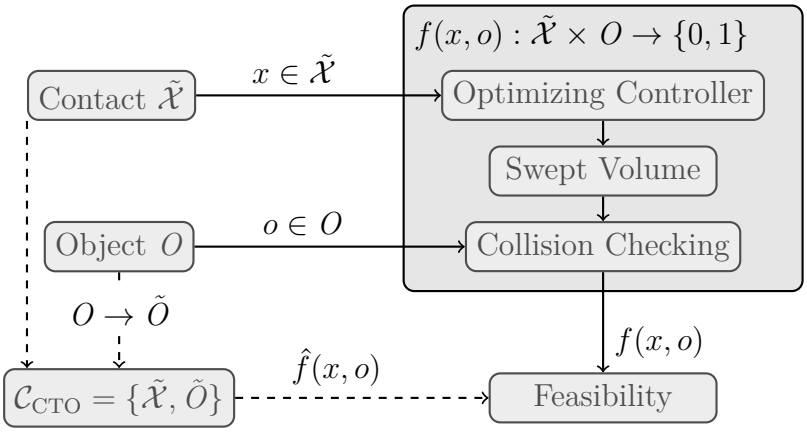



**Figure 2.4:** From Contact Transitions to Feasibility. Dashed lines present the precomputation functions, which form a shortcut for efficient online planning

## 2.6 Precomputation of Decision Boundary in CTO Space

To estimate $\hat{f}$, we first generate samples from the true feasibility function $f$. This requires the definition of a probability distribution, which provides us samples near the decision boundary, such that objects have a distance of $d \approx 0$ to the swept volume. A particularity of this distribution is its elongated shape, which requires the usage of a momentum variable to efficiently sample the distribution.

After acquiring samples, we finally discuss the estimation procedure for $\hat{f}$ by using nonlinear discriminative analysis [71].

### 2.6.1 Sampling of the feasibility function

We divide the sampling stage of $\hat{f}$ into two phases: First, we acquire $N$ contact points by using an uniform discretization. We recall, that every contact point has a unique goal, and together with a controller defines implicitly a unique trajectory. The unique trajectory in turn defines a swept volume by using a function $\mathcal{S} : \tilde{\mathcal{X}} \to \mathcal{T}$, whereby $\mathcal{T}$ will be a triangle mesh. The complete set of swept volumes can

then be defined as

$$\mathcal{SV}_{1:N} = [\mathcal{S}(\mathbf{x}_1), \ldots, \mathcal{S}(\mathbf{x}_N)] \tag{2.6}$$

For each swept volume, we start obtaining samples $\mathbf{o}_i \in \tilde{O}$, by defining a probability distribution, which provides us with the properties we want: High probability around the decision boundary, low probability otherwise. One possible choice is the normal distribution, defined as

$$p(x_j, o_i) = \mathcal{N}(d[\mathcal{S}(\mathbf{x}_j), M(\mathbf{o}_i)]; \mu = 0, \sigma) \tag{2.7}$$

whereby $M$ computes the triangle meshes of the object $i$ at position $\mathbf{o}_i$, $\mathcal{S}(\mathbf{x}_j)$ is the swept volume from contact position $\mathbf{x}_j$, and $d$ is defined as the norm between the nearest points on the object and on the swept volume – or the farthest points inside the swept volume, if the object is in collision. Finally the standard deviation $\sigma$ is a measurement of how much we tolerate samples away from the boundary.

Eq. (2.7) is an elongated probability distribution, which can be very narrow, depending on our choice of $\sigma$. Therefore, standard sampling techniques like the Markov Chain Monte Carlo (MCMC) algorithm will generally be inefficient, i.e. require too many samples before converging to the stationary distribution [71]. One algorithm, which can handle elongated distributions is the *Hamiltonian Monte Carlo* (HMC) algorithm [72], which adds a momentum variable to the sampling process, in order to faster converge to the stationary distribution. The most important feature of HMC is its ability to follow the contour curve of the distribution by simulating the hamiltonian dynamics. In our case, this translates to following the decision boundary of the $\mathcal{C}_{\text{CTO}}$ space. The underlying algorithm to simulate

this contour-curve following behaviour is called the leap-frog algorithm, and progresses by using a number of steps $\tau$ and step width $\epsilon$. The initial momentum in a certain direction is defined by a proposal distribution. In our experiments, we used $\epsilon = 0.3$, $\tau = 13$ and a multivariate normal distribution $\mathcal{N}(\mu, \mathbf{\Sigma}) = \mathcal{N}(\mathbf{0}, 0.09 \cdot \mathbf{I})$ as the proposal. For Eq. (2.7), we have chosen $\sigma = 0.17$.

For simplifications, we consider in our experiments objects approximated by a cylindrical representation, by setting the $\theta$ parameters of Eq. (2.3) to $\theta_1 = 0.01, \theta_2 = 1$. The $\lambda$ parameters are allowed to vary, and are defined for a cylindrical representation as $\lambda_1 = \lambda_2 = r$ and $\lambda_3 = h$, whereby $r$ is the radius of the cylinder and $h$ the height. Sampling is then conducted explicitly in the space of $\tilde{O} = \{(x, y, r, h)^T | x, y \in \mathbb{R}, r, h \in \mathbb{R}_+\}$.

### 2.6.2 Nonlinear Discriminative Analysis

After obtaining the samples from the true function $f$, we have to select a model to approximate $f$ by $\hat{f}$. The choice of this model is mainly determined by its online performance: The more often we can call the function per second, the better will be our planning performance. One widely used choice is the *multilayer perceptron* (MLP), which can lead to compact models and faster evaluation, but is harder to train than common kernel machines, because its objective function is non-convex [71]. Because we need to reduce the time for online performance as much as possible, we have chosen the MLP with one hidden layer and trained the network from the sampled data by utilizing the FANN[1] library.

#### Network Optimization

We applied several standard machine learning tricks to obtain a robust and stable approximation of $\hat{f}$. First, we splitted our training data into a training set (70%)

[1] http://leenissen.dk/fann/

and a validation set (30%) and used an early stopping criterion by observing the model error on the validation set. Second, we used multiple restarts with random initializations. Third, we combined two samplers to avoid spurious non-feasible regions: A uniform coarse sampling technique to avoid spurious non-feasible regions, and the aforementioned HMC algorithm to accentuate the decision boundary.

Finally, we summarized the essentials steps of the precomputation in Algorithm 3.11. For each contact point $\mathbf{x} \in \tilde{\mathcal{X}}$, we first compute the whole-body motion to the waypoint $\mathbf{x}_G$, by using the optimizer $g$ and cost function $C$. The resulting trajectory $q_{q_I \to q_G}$ defines a swept volume $\mathcal{SV}$, which we approximate by using a function $\mathcal{S}$. For the class of objects $\tilde{O}$, we acquire $M$ samples $\mathbf{o}_{1:M}$ by using the HMC sampling algorithms with parameters $\tau$ and $\epsilon$. Afterwards, we split our sampling data and start the nonlinear discriminative algorithm to approximate $\hat{f}$. $\hat{f}$ is finally saved in our complete feasibility structure $\mathcal{F}$.

**Algorithm 1:** Precomputing feasible motion space

**Data**: $\mathcal{C}, \tilde{O}, \tau > 0, \epsilon > 0, H > 0, M > 0$

**Result**: $\mathcal{C}, \tilde{O}, M, H, \tau, \epsilon$

$\mathcal{F} \leftarrow \varnothing$;

**forall the** $\mathbf{x} \in \tilde{\mathcal{X}}$ **do**

- $q_{q_I \to q_G} \leftarrow g(\mathbf{x}, \mathbf{x}_G, \mathcal{C})$ [69];
- $\mathcal{SV} \leftarrow \mathcal{S}(q_{q_I \to q_G})$ [41];
- $\mathbf{o}_{1:M} \leftarrow \text{Sampler}(\mathcal{SV}, \tilde{O}, M, \tau, \epsilon)$ [72];
- $\mathbf{o}_{train}, \mathbf{o}_{validation} \leftarrow \text{split}(\mathbf{o}_{1:M})$;
- $\hat{f} \leftarrow \text{NDA}(H, \mathbf{o}_{train}, \mathbf{o}_{validation})$ [71];
- $push(\mathcal{F}, \hat{f})$

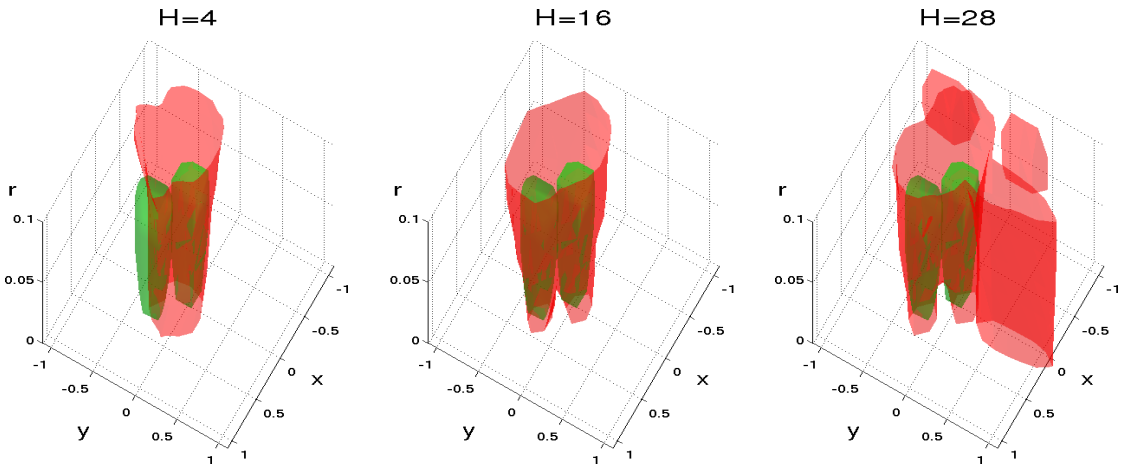



**Figure 2.5:** Influence of the model complexity on the approximated feasibility function: Each graph shows the object space $\tilde{O}$ for the same swept volume with changed complexity parameter $H$. For visualization, we have shown the non-feasible regions for the $(x, y, r)^T$ parameters of a cylinder, whereby we fixed $h = 0.03$. Shown is the isosurface of the feasibility function for the zero value, first the real feasibility function (green), and second the approximated function $\hat{f}$ (red). Depending on the complexity parameter $H$ of the model we can observe different performances: a low complexity like $H = 4$, leads to an underfitting of $\hat{f}$, while a high complexity $H = 28$ leads to overfitting, visible by several spurious non-feasible regions. The goal is to find a parameter, like $H = 16$, which balances model complexity and error rate.

### 2.6.3 Algorithmic analysis

The offline-precomputation of the feasibility function required ~6 hours on a 8 core, 3.0Ghz PC with 8GB working memory. The online performance requires the computation of two matrix multiplications in our MLP, and therefore scales with $\mathcal{O}(H * (N + 1))$, whereby $H$ is the complexity parameter and $N$ the number of dimensions of $\tilde{O}$. At the moment, we have no theoretical guarantee that the algorithm is strictly conservative, i.e. that it declares a footstep as valid, if it is not. We could approach this by proving that the derivative of the feasibility function is bounded, i.e. K-lipschitz continuous, and using this as a hard constraint during the optimization of the approximated model.

## 2.7 Experiments

In our experiments, we use a feasibility function $\hat{f}$ with a reasonable model complexity of $H = 16$, which avoids under- and overfitting, as discussed in Fig. 2.5. We refer to this algorithm as $FP(16)$, whereby $FP$ stands for *feasibility precomputation*. For comparison, we use a reimplementation of the swept volume approximation (SVA) algorithm, proposed by [41], which stores swept volumes for each action, and afterwards used a collision checking algorithm for feasibility checks [73]. Both algorithms are integrated into our planning framework, and tested in a challenging environment, where we randomly place objects.

### 2.7.1 Planning

For planning, we utilize a standard A* algorithm [11] with a classical euclidean $l_2$-norm heuristic to the goal. The heuristic is complementary to our work: We focus not on the heuristic, but on approximating the extension of nodes in the graph search. The choice of the heuristic can further speed up planning [65], but is beyond the scope of this work.

### 2.7.2 Walking in Cluttered Environment

To evaluate and compare the performance of our feasibility precomputation, we consider a highly-cluttered and constrained environment, where $K$ small objects are located randomly over a flat, horizontal floor, as visualized in Fig. 2.6. We generate the objects by using a uniform sampler $\mathcal{U}$ and bounding cylinders in the form of $x = \mathcal{U}(-0.8m, 0.8m), y = (0.2m, 2.8m), r = \mathcal{U}(0.01m, 0.03m), h = \mathcal{U}(0.01m, 0.1m)$.

The robot is allowed to set footsteps, which are constrained to be between $x = [-0.8m, 0.8m]$ and $y = [-0.2m, 3.2m]$.

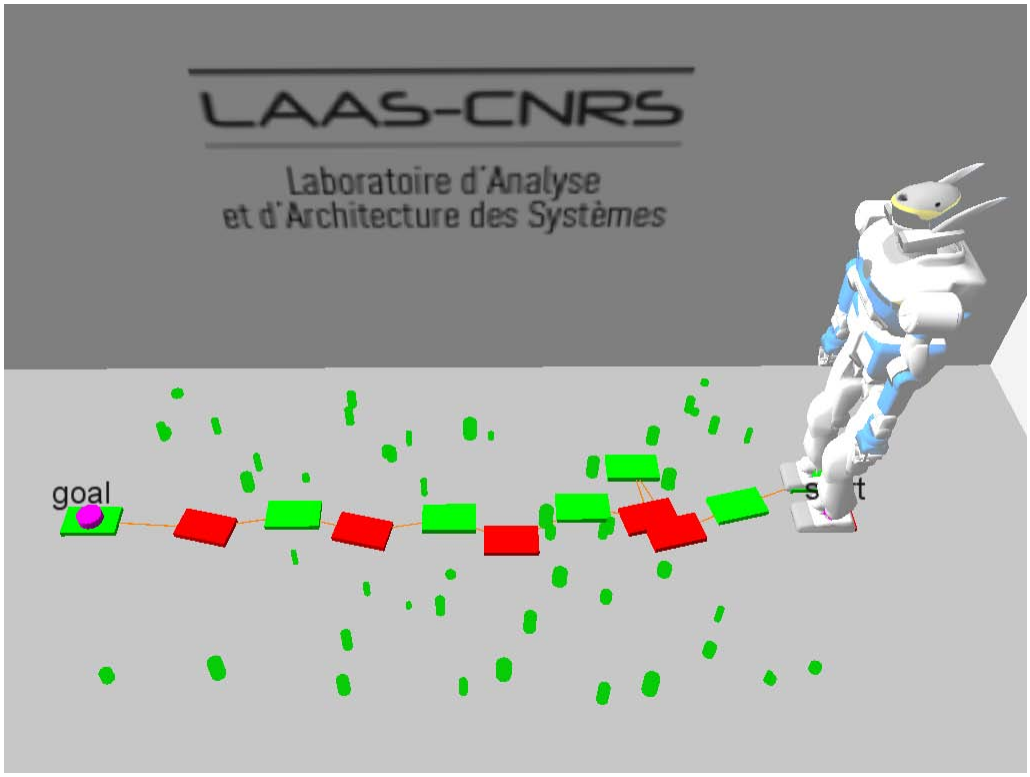



**Figure 2.6:** A cluttered environment, which we consider in our experimental verification. A number of cylinders are used as obstacles, and determines the complexity of the scene. In a real world setting, those cylinders would correspond to approximations of objects, similar to the chair in Fig. 2.3.

The planning tasks is to move the feet, starting with the left foot at coordinates $(x_I, y_I) = (0, 0)$, towards the goal at $(x_G, y_G) = (3, 0)$, i.e. having one foot in the vicinity ($< 0.3m$) of the goal. We compare the two approaches, mentioned above: For the SVA algorithm, we obtain the cylinders as triangle meshes from the simulator and store them offline for efficient collision checks. For $FP(16)$, we use the $x, y, r, h$ values as the input for $\hat{f}$. Before each execution, we apply a homogeneous transformation to move the object into the coordinate system with the support foot as origin, such that they coincide with the precomputation stage. Moreover we prune all objects, which have a certain distance to the robot ($< 1.1m$) before we apply the algorithms.

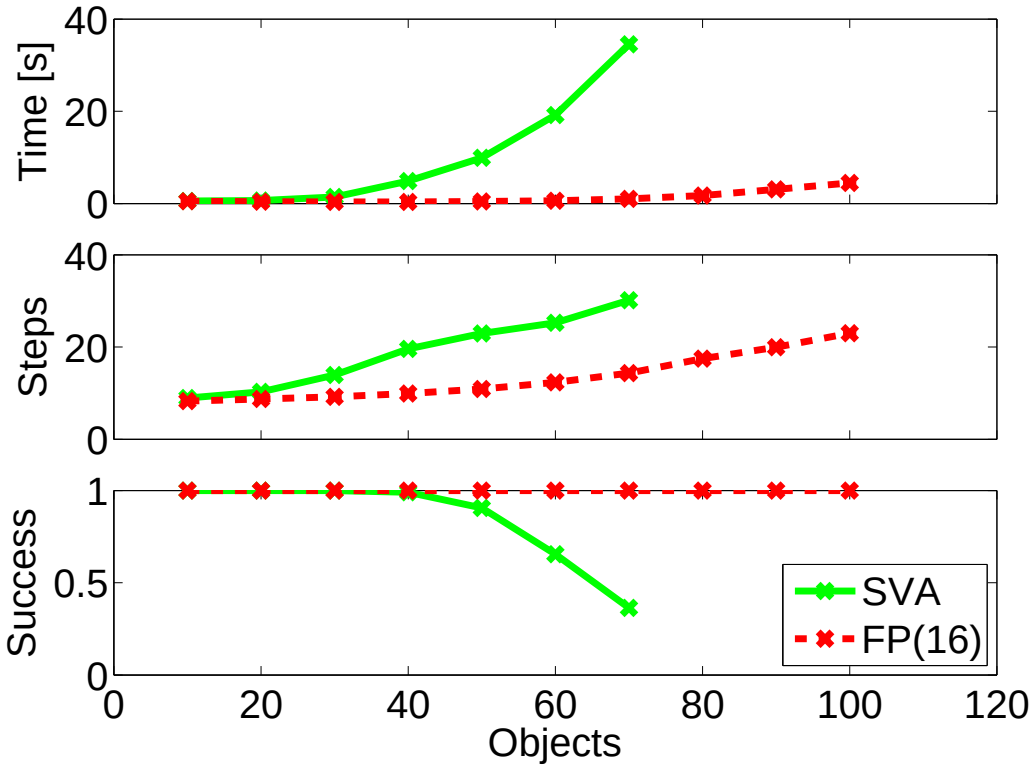



**Figure 2.7:** Comparison between swept volume approximations (SVA) [41] (green) and the precomputation of the feasibility function (red). Each point represents the average over 100 trials in the cluttered environment situation, where the robot had to traverse a distance of $3.0m$, while avoiding $M$ objects, randomly distributed on the floor.

Fig. 2.7 shows the performance of the two algorithms on this task: In the first row, we show the average planning time for successful plans versus the number of objects in the scene. It can be seen, that the time for planning with the SVA algorithm (green) increases rapidly with the number of objects. In comparison, our algorithm (red) increases only marginally and stays bounded by $< 1s$ even for $N = 60$ objects. Also, we obtain a lower number of steps toward the goal as seen in the second row. The last row shows the success rate of the planner, i.e. after a fixed time $T$, we stop the planner and consider the task unsuccessful. Those tasks are cleared from our system and are not considered for the time and step graphs.

## 2.8 Integration into Industrial Project

In collaboration with the aircraft manufacturer AIRBUS, we developed a proof-of-concept, in which we want to demonstrate how a humanoid robot can work inside of an aircraft factory. One aspect of this problem requires that the humanoid robot is able to react reactively to changes in the environment. We have used the methods developed in this chapter to address this problem. In particular we consider a flat floor on which tool-box containers are moving in an unpredictable fashion.

### 2.8.1 Implementation Details

Our algorithm consists of three stages: first, we need to acquire the geometry of the environment and the position of the robot and the obstacles. Second, we plan a feasible motion, and we send the motion towards a ROS module, which executes the motion on the real robot. Third, after every step, we replan a new motion by taking the new position of the robot and the obstacles into account. To avoid the moving obstacles, we continuously replan our motion.

In detail, for the first stage we equip all obstacles and the robot with motion capturing sensors. Using the VICON capturing system, we equipped each obstacle with at least 6 sensors, then used the VICON software to estimate position of each object. We then send the position commands by using ROS to our algorithm, which uses the information together with the URDF version of each object to display the object in rviz. Simultaneously, based on the object geometry, we approximate the object by a cylinder, such that we can apply our developed techniques. In the second stage, we use the information about the cylinder and the foot position to obtain a feasible footstep plan, by using the A* algorithm as explained. We then send the first three steps of the plan towards another ROS module, which handles

the control of the robot by using a hierarchical task-space framework, called the stack-of-task [69].

After executing one step, the task-space framework sends back an acknowledgment. Once received, we start replanning with the updated informations from the VICON software. A video of the replanning procedure can be found here

`https://www.youtube.com/watch?v=iFV-13XlJvI`

Additionally, in Fig. 2.8, you can see three screenshots from the final motion: first, the robot has planned a feasible footstep path towards a goal object. Second, an object moves into its way, and the robot adapts its footstep path. Third, following the adapted path, the robot reaches the goal position.

## 2.9 Conclusion

In this chapter, we presented the *contact transition and object* space, a joint space of contact points and geometrical information of approximated objects. We developed an algorithm to precompute the feasibility of specific objects and contact points, by approximating the decision boundary between feasible and non-feasible subspaces. As a result we obtain a sparse discriminative function, which allowed us to quickly prune non-feasible contact-points – while at the same time preserving the important stepping-over capability of humanoid robots.

In our simulated experiments, we demonstrated that our approach can be used to generate whole-body motions for a humanoid robot in highly-cluttered environments in near real-time, thereby outperforming a state-of-the-art algorithm, which used swept volume approximations. Moreover, our algorithm has a significantly lower memory fingerprint: instead of saving the complete swept volumes, we require only the model parameters of our discriminative function to be saved.

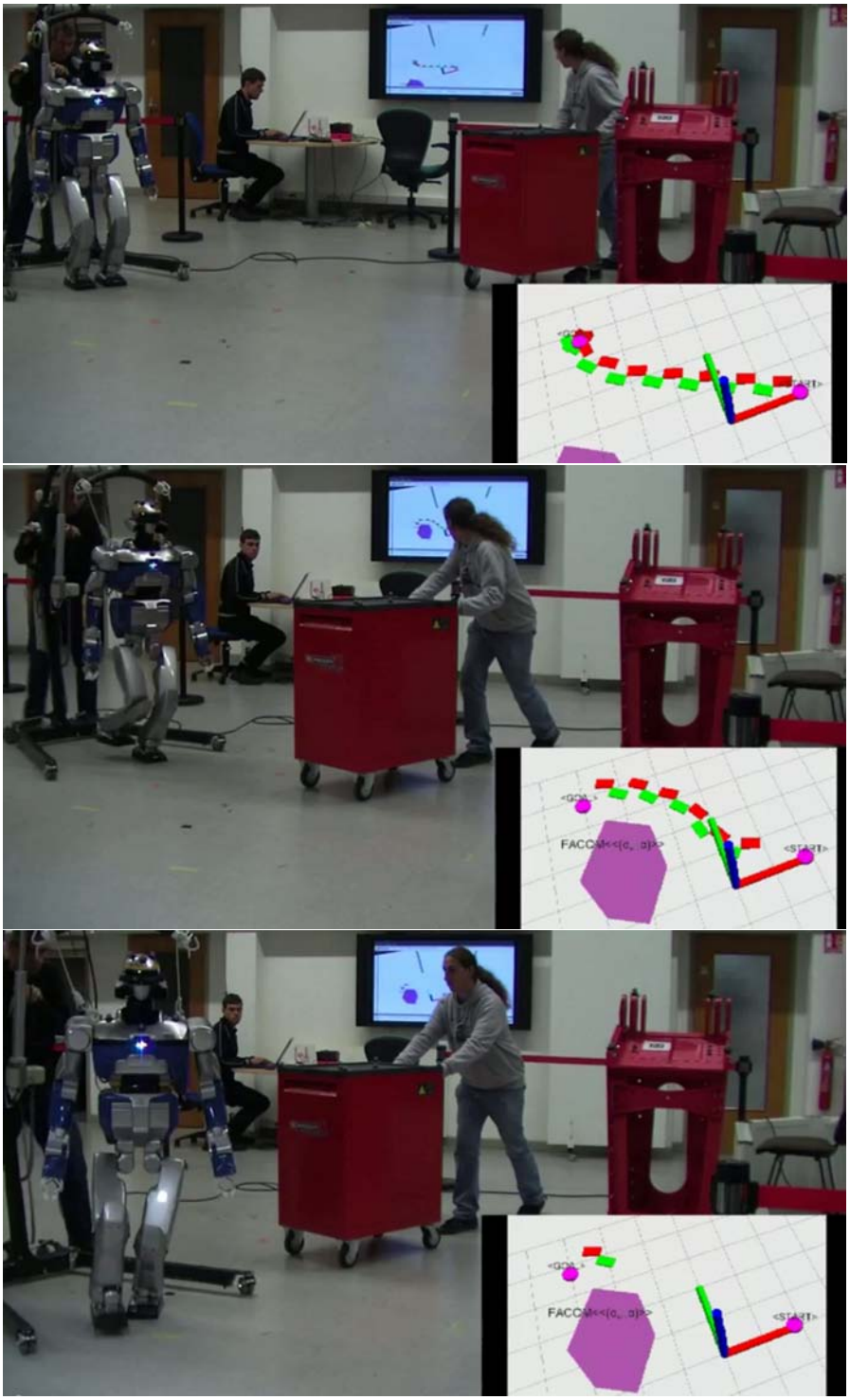


**Figure 2.8:** Online replanning on HRP-2. Top: the robot has planned a feasible footstep path towards a goal object. Middle: an object moves into its way, and the robot adapts its footstep path. Bottom: Following the adapted path, the robot reaches the goal position.

This comes at the price of a lower accuracy at run-time: due to the approximation of the objects by simple geometrical shapes, we lose the ability to move close to objects and conduct for example fine-manipulation planning. However, for certain behaviors like walking, fine-manipulation is per se not required. Also, we see our method as a first reasonable guess of the trajectory, which could be further refined locally.

Possible future research directions for this chapter are the incorporation of object velocities into the precomputation, the estimation of the decision boundary for the general superellipsoid space of objects, the augmentation of the action space and the verification on our robotics platform using vision systems. As a natural extension we like to extend this framework to different behaviors like walking, crouching and holding objects while walking.

# Chapter 3

# Irreducible Motion Planning by Exploiting Linear Linkage Structures

*"If a sign is not necessary then it is meaningless. That is the meaning of Occam's Razor."*

— Ludwig Wittgenstein, Tractatus Logico-Philosophicus

*"Irreducibility is a theoretical framework for completeness-preserving dimensionality reduction in motion planning"*

A humanoid robot can be represented as a mechanical system structure. This mechanical system structure consists of components which are recurrent and which differ in size. Can we take advantage of this recurrency, by exploiting structural components in the mechanical system? We answer this question in the affirmative, and we develop a theoretical concept to study mechanical structures, a concept which we call *irreducibility*. Irreducibility is a theoretical

framework for completeness-preserving dimensionality reduction in motion planning. While classical motion planning searches the full space of continuous trajectories, irreducible motion planning searches the space of minimal swept volume trajectories, called the irreducible trajectory space. We proof that planning in the irreducible trajectory space preserves completeness. We then apply this theoretical result to linear linkage structures, which can be found in several mechanical systems, among them humanoid robots. Our main result establishes that we can reduce the dimensionality of linear linkages in the case where the first link moves on curvature-constrained curves. We further develop a curvature projection method, which can be shown to be curvature-complete, a weaker version of general completeness. As an application, we consider the simplification of humanoid motion planning by considering the arms and legs as linear linkages.

## 3.1 Introduction

In this chapter, we will investigate a specific property of mechanical systems which we call irreducibility. Irreducibility classifies configuration space trajectories into two categories: reducible and irreducible trajectories. An irreducible trajectory is a trajectory with a minimal swept volume in the environment. We will prove here the important fact that planning with only irreducible trajectories preserves completeness. It follows that motion planning can be conducted entirely inside the irreducible trajectory space.

However, it is not obvious how one would analytically define this irreducible trajectory space. In this chapter, we therefore concentrate on a specific mechanical structure, the linear linkage, and investigate how irreducibility can be defined on it. We note that linear linkages are prevalent in a variety of mechanical systems, which are all consequently susceptible to our reduction concept. Four examples

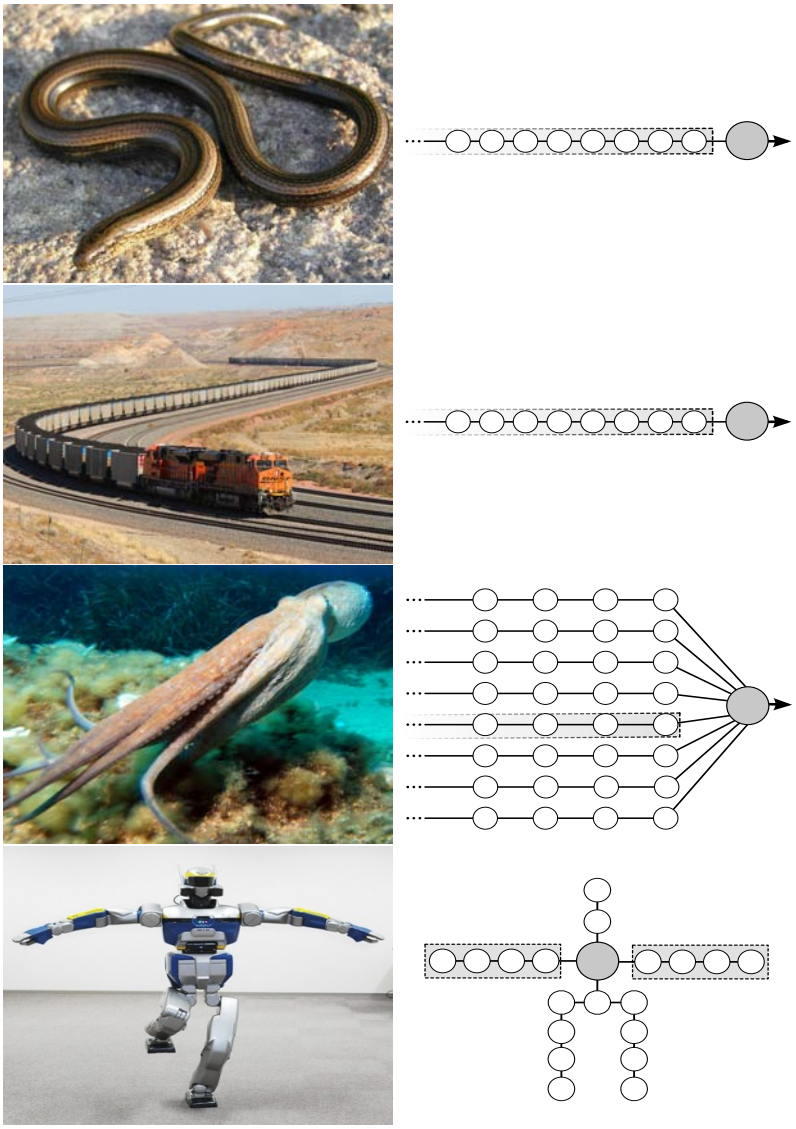


**Figure 3.1:** Examples of free linear linkages in mechanical systems. On the left are different mechanical systems, and on the right is the abstracted idealized linkage structure. **Top**: a snake has one linear linkage with the head as a root link. **Top Middle**: a train has one linkage with the locomotive as the root link. **Bottom Middle**: an octopus has eight arms, each is one linear linkage with the head as a common root link. **Bottom**: humanoid robot HRP-2 has two linear linkages for its arms (with the chest as root link), and two linear linkages for its legs (with the hip as root link).
@Photograph Courtesy:

- Marek Bydg. *Anguis Fragilis Slowworm*. 2004. Poland. `http://en.wikipedia.org/wiki/Anguis_fragilis`. Web. Accessed April 28th, 2015.
- Unknown Author. *BNSF ES44Ac leading a coal train through S-curve in Powder River Basin*. 2013. Colorado, US. `http://www.4rail.net`. Web. Accessed April 28th, 2015.
- Albert Kok. Octopus Vulgaris. 2007. Location Unknown. `http://en.wikipedia.org/wiki/Octopus`. Web. Accessed April 28th, 2015.
- Vincent Fournier. HRP-2 #1 [Kawada], Promet Developed by AIST, 2010. Tochigi, Japan. `http://www.vincentfournier.co.uk`. Web. Accessed April 28th, 2015.

are shown in Fig. 3.1, a snake and a train with each one linear linkage, an octopus with eight linear linkages and a humanoid robot with four linear linkages.

This work is based on previous published results in [13]. In particular, Sec. 3.3 and parts of the experimental results have been already published. Our additional contributions are

- Introduced the irreducibility property for linear linkages
- Introduced the concept of curvature completeness
- Proved that linear linkages are curvature complete for a specific functional space
- Developed a linear-time irreducibility projection algorithm for linear linkages in 3d.
- Conducted simulated experiments for the humanoid robot HRP-2

In terms of prerequisites, we will assume a rudimentary knowledge of differential geometry in our proofs, as can be found for example in [74].

After summarizing related work in Sec. 3.2, we provide definitions and proofs of the main theorems of irreducible motion planning in Sec. 3.3. Our main result is summarized in Corollary 2, which provides a proof of completeness for irreducible trajectories.

After those preliminaries, we concentrate on linear linkage structures in Sec. 3.4. We provide a definition of linear linkages and we proof conditions under which we can ignore certain parts of the linkage. This brings us to the concept of a curvature complete algorithm, for which we design in Sec. 3.5 a linear-time algorithm in the number of links. In Sec. 3.6, we finally conduct a set of experiments for a swimming snake robot in simulation and for the real humanoid robot platform HRP-2.

## 3.2 Related Work

Motion planning for humanoid robots is a well studied field [34], which has demonstrated its potential in the DARPA Robotics Challenge (DRC) in 2015. Humanoid robots are able to solve difficult tasks, like manipulation planning in kitchen environments [75], contact planning in constrained environments [44], or ladder climbing tasks [9]. Techniques for solving those problems range from optimal control planning [23] over motion database approaches [14] to fast contact planning by identifying convex surfaces [7].

The mechanism and locomotion system for snake robots has long been studied [76]. However, path planning for snake robots has been investigated in relatively few papers. Some of them are classical approaches using numerical potential field [77], genetic algorithms [78] or Generalized Voronoi Graph [79]. The idea of a simplified model is studied by [80], who define a frame that is consistent with the overall shape of the robot in all configurations. In [81] the authors plan a trajectory only for a portion of the snake robot.

Ultimately, all those approaches try to exploit structure to reduce the computational complexity of the problem. In this chapter, we concentrate on dimensionality reduction techniques, which have been extensively studied in the motion planning literature. Dalibard et al. [52] have used a principal component analysis (PCA) to bias random sampling. In the context of manipulation motion planning, the powerful eigengrasps [53][54] have been introduced to identify a low-dimensional representation of grasping movements. Reduction techniques have been especially used in cable motion planning. Mahoney et al. [55] perform a PCA for a high-dimensional cable robot by sampling deformations. Kabul et al. [82] plan the motion of a cable by first planning a motion for the head. Those works showed remarkable results, and demonstrate the effectiveness of reduction techniques. Our work is complementary, in that we are undertaking a formal treatment of condi-

tions under which dimensionality reduction can be performed.

In particular, we show in our work a connection between the curvature and the dimensionality of the problem. Curvature constrained curves have been investigated in the framework of computational geometry [74]. For example, Bereg et al. [83] introduce the term reducibility in the context of sweeping of disks along a planar curve. Our work generalizes this concept by giving completeness guarantees of curves which are non-reducible or as we call it irreducible.

Ahn et al. [84] developed algorithms to compute the reachable regions for curvature constraint motions inside convex polygons. Our work builds upon their theoretical contribution to proof when a system is irreducible.

Guha et al. [85] discuss curvature and torsion constraint on space curves in the context of data point approximation. This work hints at a generalization of our ideas in Sec. 3.4.6, which we left as a conjecture.

Finally, we use the result described in [86], who showed that a dynamical humanoid robot is small-space controllable, i.e. we can minimize the oscillations of the upper body — and thereby the swept volume — by minimizing its step-size and its step-period. Taking this towards the extreme, the Center-Of-Mass trajectory can be planned as if the robot was sliding on the floor. This sliding motion can be seen as an irreducible motion and thereby provides a first necessary condition for feasibility.

## 3.3 Irreducible Trajectories

We restate relevant motion planning definitions, following the classical formulation by [11, Chapter 4]

**Definition 4** (Motion Planning Problem)**.** *Let* $A = \{\mathcal{R}, \mathcal{C}, q_I, q_G, \mathbf{E}\}$ *be a motion planning problem, with* $\mathcal{R}$ *the robotic system,* $\mathcal{C}$ *the configuration space,* $q_I$ *the*

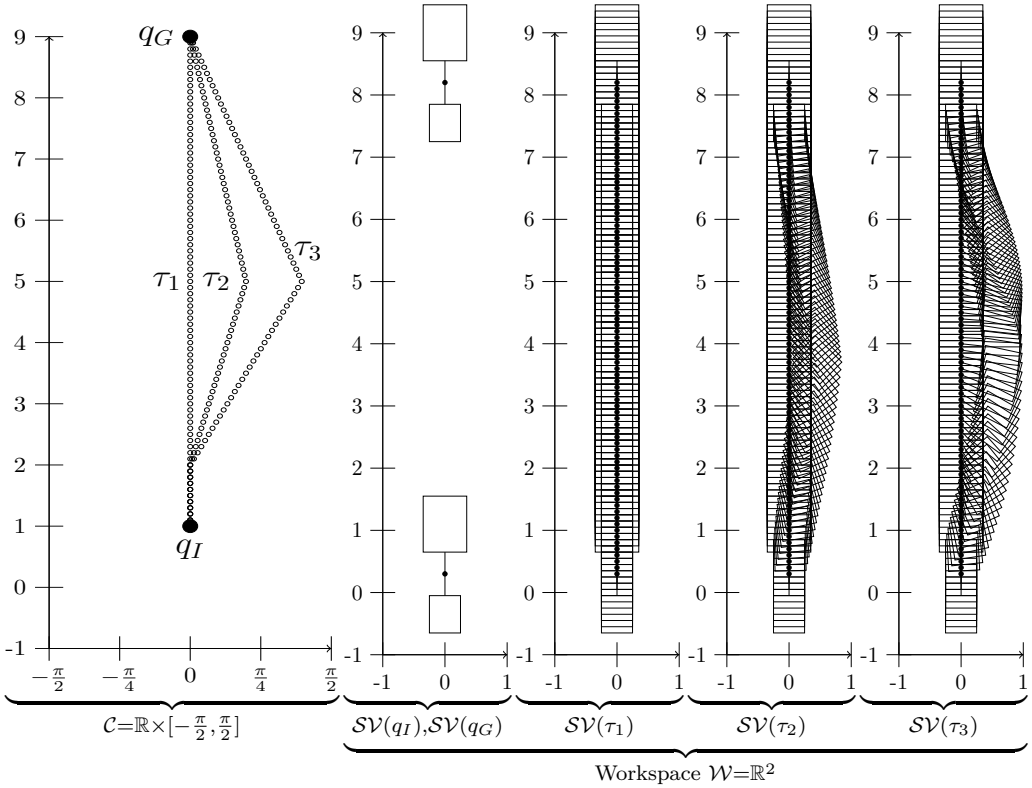



**Figure 3.2:** Explanatory example of irreducible trajectories for a 2-link, 2-dof robot, which can move along the $y$-axis, and which has one rotational joint between its two links, such that its configuration space is $\mathcal{C} = \mathbb{R} \times [-\frac{\pi}{2}, \frac{\pi}{2}]$. **Left.** Three configuration space trajectories $\tau_1, \tau_2, \tau_3$ with $\tau_1(0) = \tau_2(0) = \tau_3(0) = q_I$, $\tau_1(1) = \tau_2(1) = \tau_3(1) = q_G$. **Right.** The workspace volume of the starting configurations $q_I$, $q_G$, and the swept volume of the three trajectories, whereby we have that $\mathcal{SV}(\tau_1) \subset \mathcal{SV}(\tau_2)$ and $\mathcal{SV}(\tau_1) \subset \mathcal{SV}(\tau_3)$, i.e. $\tau_2$ and $\tau_3$ are reducible by $\tau_1$, and $\tau_1$ is in fact irreducible. Adapted from [13].

*initial configuration, $q_G$ the goal configuration, and* $\mathbf{E}$ *the environment.*

**Definition 5** (Configuration Space Trajectory)**.** *Let $A$ be given. Then we denote by $\mathcal{F}(q_I, q_G) = C^1([0,1], \mathcal{C})$ the set of continuously differentiable functions from $[0,1]$ to the configuration space $\mathcal{C}$, with the property that if $\tau \in \mathcal{F}(q_I, q_G) \Rightarrow \tau(0) = q_I, \tau(1) = q_G$.*

**Definition 6** (Swept Volume)**.** *The workspace volume swept by the trajectory $\tau \in \mathcal{F}(q_I, q_G)$ will be denoted by $\mathcal{SV}(\tau)$.*

**Definition 7** (Feasible Trajectory)**.** *A trajectory $\tau \in \mathcal{F}(q_I, q_G)$ is called feasible in an environment $\mathbf{E}$, if $\mathcal{SV}(\tau) \cap \mathbf{E} = \emptyset$.*

**Definition 8** (Feasible Configuration Space Trajectory)**.** *Let $\mathcal{S} \subset \mathcal{F}(q_I, q_G)$ be a set of Configuration space trajectories. Let $A$ be a specific motion planning problem. If there exist $\tau \in \mathcal{S}$ such that $\tau$ solves $A$, then $\mathcal{S}$ is said to be feasible w.r.t. $A$.*

We denote by $\subset$ the proper subset. Let $A = \{\mathcal{R}, \mathcal{C}, q_I, q_G, \mathbf{E}\}$ be given, and let $\mathcal{F} = \mathcal{F}(q_I, q_G)$.

**Definition 9.** *A trajectory $\tau' \in \mathcal{F}$ is called reducible, if there exist $\tau \in \mathcal{F}$ such that $\mathcal{SV}(\tau) \subset \mathcal{SV}(\tau')$. Otherwise $\tau'$ is called* irreducible.

Fig. 3.2 provides a visualization of the irreducible definition for trajectories. We show three configuration space trajectories $\tau_1, \tau_2, \tau_3$, and its swept volumes in workspace. Applying the definition, we have that $\tau_2$ and $\tau_3$ are reducible by $\tau_1$. We will now show why irreducibility is important for motion planning.

**Theorem 1.** *Let $\tau, \tau' \in \mathcal{F}$ be such that $\mathcal{SV}(\tau) \subset \mathcal{SV}(\tau')$, i.e. $\tau'$ is reduced by $\tau$.*

$$\textit{If } \tau \textit{ is infeasible} \Rightarrow \tau' \textit{ is infeasible}$$
$$\textit{If } \tau' \textit{ is feasible} \Rightarrow \tau \textit{ is feasible}$$

*Proof in Appendix.*

**Definition 10** (Irreducible Trajectories)**.** *Let the set of all irreducible configuration space trajectories be defined as*

$$\mathbf{I} = \{\tau \in \mathcal{F} | \tau \text{ is irreducible}\} \tag{3.1}$$

**Lemma 1.** *Let $\tau \in \mathcal{F} \setminus \mathbf{I}$. Then there exist $\tau' \in \mathbf{I}$, with $\mathcal{SV}(\tau') \subset \mathcal{SV}(\tau)$.*

*Proof in Appendix.*

**Theorem 2.** *If $\mathbf{I}$ is infeasible then $\mathcal{F}$ is infeasible*

*Proof in Appendix.*

**Corollary 1.** *Motion planning is complete in $\mathbf{I}$*

*Proof in Appendix.*

Going back to the example in Fig. 3.2, we can now make the statement, that trajectories $\tau_2$ and $\tau_3$ can be ignored for motion planning, while still being complete. This means we now have a formed a geometric argument, which allows us to reduce the dimensionality of a motion planning problem while preserving completeness.

## 3.4 Irreducibility for Linear Linkages

We will now use the theoretical concept of an irreducible trajectory to study linear linkages. A linear linkage is a mechanical system consisting of $N + 1$ links, which are connected in a chain, as depicted in Fig. 3.3. We will call the first link in the chain the *root link*, denoted by $L_0$, and the other $N$ links as *sublinks*. If the root link is moveable we call the linear linkage *free*, otherwise non-free. We will

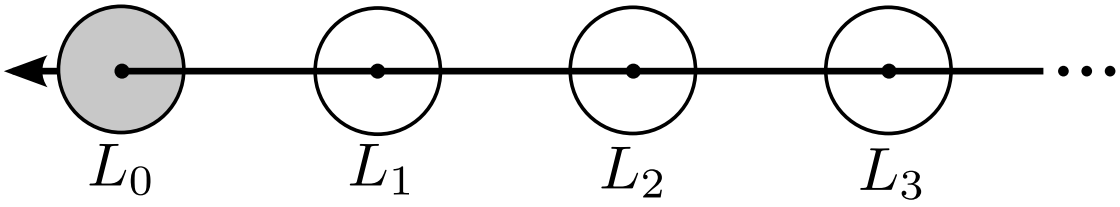



**Figure 3.3:** A free linear linkage with $L_0$ being the root link, and $L_1, L_2, \cdots$ are called the sublinks. The black arrow gives the movement direction of $L_0$. In this chapter, we will give conditions under which only the root link $L_0$ has to be planned for, while the sublinks can be ignored while preserving a curvature completeness property.

exclusively work with free linear linkages with a finite number of sublinks, if not otherwise stated. We will study in this section conditions for movements of $L_0$, such that we can ignore the sublinks.

This whole section is dedicated to the task of finding conditions on the movement of the root link $L_0$, such that all sublinks can be ignored for motion planning. Our main idea is that if the root link moves on curvature-constrained curves in $\mathbb{R}^2$, we can always reduce the sublinks, and thereby preserving a weak form of completeness. We will give an informal treatment of this idea, then proceed to proof the case of curvature-constraint motion planning in $\mathbb{R}^2$ and finally discuss the $\mathbb{R}^3$ case, which we leave as a conjecture.

### 3.4.1 Swept Volume of a Train

Let us observe that a train is a linear linkage with the locomotive as a root link, and its $N$ railroad cars as sublinks. If the train moves between two stations on given railroad tracks, then we can state the following: the swept volume of the train with $N$ railroad cars is equal to the swept volume of the train with zero railroad cars. The reason is that we constrain the movement of the locomotive to be bounded by a minimal curve radius. Given a minimal curve radius, we are allowed to construct arbitrary railroad tracks for a train, to move from one city to the next. More abstractly, we can translate this to: we are allowed to construct

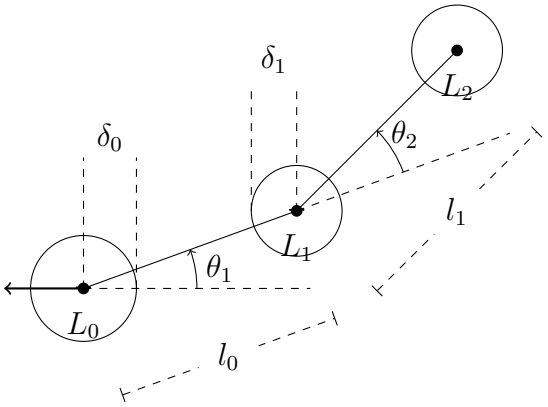



**Figure 3.4:** $N = 2$ linear linkage system

space curves $f$ from a functional space $\mathcal{F}$, under the constraint that every function $f$ has a bounded curvature.

To come back from our train example to arbitrary mechanical systems, let us call the locomotive a root link, and each railroad car a sublink. Intuitively, if the root link is big enough, i.e. the locomotive is bigger than the railroad cars, and if the root link moves on space curves bounded by a certain curvature, then we can state that there exists a configuration of the sublinks, such that the swept volume of the sublinks is a subset of the swept volume of the root link. The big implication here is: if we find a physically feasible railroad track for a locomotive, then we can add a finite number of railroad cars, while still being feasible. A train in this sense is redundant, i.e. there are links which can be ignored for motion planning. Our goal now is to formalize those ideas rigorously. We will start by defining a functional space for the root link $L_0$, then proof that there always exist configurations for the sublinks $L_1, \cdots, L_N$, such that they are inside of the swept volume of $L_0$.

### 3.4.2 Curvature Functional Space

Let us consider a $N = 1$ linear linkage with links $L_0, L_1$ in the plane $\mathbb{R}^2$, connected by a rotational joint at the center of $L_0$, with distance $l_0$ to $L_1$. A $N = 2$ linear linkage is visualized in Fig. 3.4. The rotational joint has an allowed rotation of $\theta \in [-\theta^L, \theta^L]$. Let us denote by $s = (s_0, s_1) \in \mathbb{R}^2$ the position of $L_0$, and by $s'$ its orientation. Let us define a cone $\mathcal{K}_{\theta^L}(s) = \{(x_0, x_1) \in \mathbb{R}^2 | \|x_1 - s_1\|^2 \leq (x_0 - s_0) \tan \theta^L\}$ with apex $s$, orientation $s'$, and aperture $\theta^L$. Then given $L_0$ at $(s, s')$, we can define the set $\partial P_0$ of all possible positions of $L_1$ as a circle intersecting $\mathcal{K}_{\theta^L}(s)$ and the corresponding disk segment $P_0$ as a disk intersecting $\mathcal{K}_{\theta^L}(s)$.

$$\begin{aligned} P_0 &= \{x \in \mathbb{R}^2 | \|x - s\| \leq l_0\} \cap \mathcal{K}_{\theta^L}(s) \\ \partial P_0 &= \{x \in \mathbb{R}^2 | \|x - s\| = l_0\} \cap \mathcal{K}_{\theta^L}(s) \end{aligned} \tag{3.2}$$

whereby $P_0$ and $\partial P_0$ are visualized in Fig. 3.5.

We will now construct a functional space $\mathcal{F}_{\kappa_0}$ by hand, and then prove that all functions from $\mathcal{F}_{\kappa_0}$ starting at $(s, s')$ will *necessarily* have to leave $P_0$ by crossing $\partial P_0$.

Let us define the functional space

$$\mathcal{F}_{\kappa_0} = C^2([0, 1], \mathbb{R}^2) \tag{3.3}$$

with given $\tau(0) = s$, $\tau'(0) = s'$, $\tau(1) \notin P_0$ and for all $\tau \in \mathcal{F}_{\kappa_0}$ we define a maximum curvature by

$$\kappa_0 = \frac{2 \sin(\theta^L)}{l_0} \tag{3.4}$$

The curvature $\kappa_0$ has been constructed in the following way: first, let us observe that for any point $\tau(t)$ on $\tau$ the curvature is defined by $\kappa_0 = \frac{1}{R_0}$ whereby $R_0$ is the radius of the osculating circle at $\tau(t)$[74]. We will now consider trajectories parametrized by arc-length, such that $\tau'(t) \cdot \tau''(t) = 0$. The center of the osculating

circle has to lie therefore in the direction of vector $\tau''(t)$. We are searching for the minimal ball, which ensures that all functions will necessarily leave $P_0$ through $\partial P_0$. This ball touches the most extreme point of $\partial P_0$, which we call $x_M$:

$$x_M = (l_0 \cos(\theta^L), l_0 \sin(\theta^L))^T \tag{3.5}$$

See also Fig. 3.5 for clarification. The ball can be found by solving the equation

$$\|x_M - (0, R_0)^T\|^2 = R_0^2 \tag{3.6}$$

The solution is given by

$$R_0 = \frac{l_0}{2 \sin(\theta^L)} \tag{3.7}$$

Please note that $l_0 \leq 2R_0$, which will be important in the upcoming proof.

### 3.4.3 Reducibility theorems of $\mathcal{F}_{\kappa_0}$

We are now going to proof some elementary properties of this functional space $\mathcal{F}_{\kappa_0}$, which will ultimately show that under certain conditions, we can ignore the sublinks for motion planning. The reader is encouraged to visualize the theorems by thinking about the train example and the maximum curvature under which the swept volume of the cars will be inside the swept volume of the locomotive. Our first theorem builds upon the pocket lemma introduced by [87]. It also uses a slightly modified version of a result by [84, Lemma 6]

**Theorem 3.** *For all $\tau \in \mathcal{F}_{\kappa_0}$ there exists $t_0 \in [0, 1]$ such that $\tau(t_0) \in \partial P_0$ and $\tau(t) \in P_0$ for all $t \leq t_0$.*

Explanation: every trajectory from our constructed functional space $\mathcal{F}_{\kappa_0}$ will leave the region $P_0$ by crossing $\partial P_0$. Visualized in Fig. 3.6.

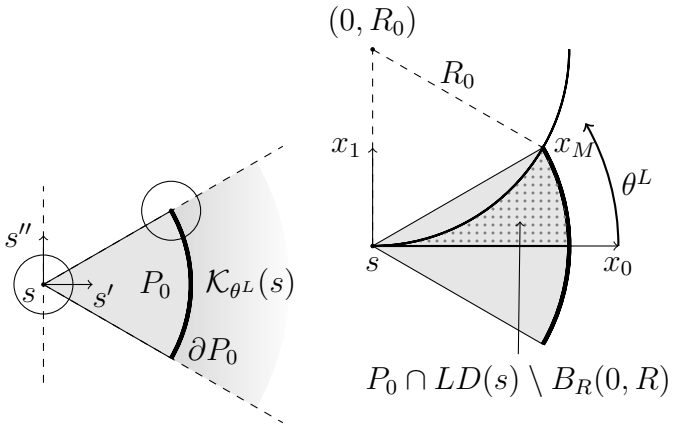



**Figure 3.5:** $\partial P_0$ is the space of all possible positions of link $L_1$, constrained by link $L_0$. We establish in this section that for a specifically constructed functional space $\mathcal{F}_{\kappa_0}$ any function which starts at $s$ and has first derivative equal to $s'$ will leave the area $P$ by crossing $\partial P_0$.

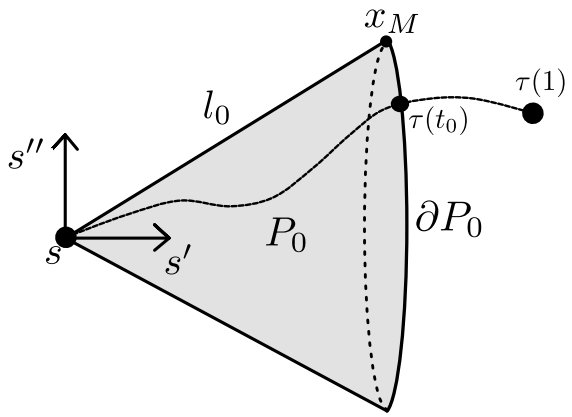



**Figure 3.6:** Cone spanned by the length $l_0$, the limit angle $\theta^L$ and the position of $s$. Every function from $\mathcal{F}_{\kappa_0}$ will necessarily leave $P_0$ by crossing $\partial P_0$ at $\tau(t_0)$ to reach a point $\tau(1)$ outside $P_0$.

*Proof.* Let us decompose the problem into two parts. First, we consider the left side of $s$, which we define as $LD(s) = \{(x_0, x_1) \in \mathbb{R}^2 | x_0 \geq 0, x_1 \geq 0\}$. Our proof will first establish that all circles with center $(0, R)$ and radius $R \geq R_0$ will intersect $\partial P_0$. Second, we use a lemma from [84] to establish that all trajectories from $\mathcal{F}_{\kappa_0}$ will necessarily leave $P_0$ by crossing $\partial P_0$, and that there is no trajectory crossing the ball $B_R(0, R)$ for given curvature $\kappa = \frac{1}{R}$.

- We will proof that every circle with center $(0, R)$ and radius $R \geq R_0$ will intersect $\partial P_0$. By construction we have that the circle $R_0$ intersects $\partial P_0$ at the point specified by angle $\theta^L$. We define the angle depending on $R$ by $\theta(R) = \text{asin}\left(\frac{l_0}{2R}\right)$. We want to establish that indeed $\theta^L \geq \theta(R) \geq 0$, i.e. a ball with radius $R \geq R_0$ will always intersect $\partial P_0$ at a point below $x_M$ and above 0. Since asin is monotone increasing on $[0, 1]$, $l_0, R \geq 0$ and $l_0 \leq 2R$, we have that $\theta(R) \geq 0$. To establish $\theta^L \geq \theta(R)$ we note that given $\text{asin}\left(\frac{l_0}{2R_0}\right) \geq \text{asin}\left(\frac{l_0}{2R}\right)$ we can write $\frac{l_0}{2R_0} \geq \frac{l_0}{2R}$, since asin is monotone increasing. It follows that $R \geq R_0$ as required.

- Let us now construct a polygonal chain for one $R \geq R_0$ in the following way: we start on the boundary of $P_0$ at point $s$ and follow direction $s'$ until we reach $\partial P_0$. At $\partial P_0$ we move upwards on $\partial P_0$ until we meet the ball with radius $R$, which intersects $\partial P_0$. This construct is a polygonal chain and specifically called a forward chain by [84]. This chain follows the boundary of $P_0 \cap LD(s)$. Ergo, we can apply Lemma 6 of [84], which states that if such a forward chain intersects the circle of unit radius, then the reachable region of all trajectories in $\mathcal{F}_{\kappa_0}$ is given by $P_0 \cap LD(s) \setminus B_R(0, R)$ (the unit radius can be obtained by scaling the space). See Fig. 3.5 for visualization. One interpretation of the pocket lemma from [87] let us now state the following: no trajectory can escape the region $P_0 \cap LD(s) \setminus B_R(0, R)$ except through $\partial P_0$ or the lower boundary. Since the same arguments apply for the lower

part, i.e. with $RD(s) = \{x \in \mathbb{R}^2 | x_0 \geq 0, x_1 \leq 0\}$ instead of $LD(s)$, we can reason that any function from $\mathcal{F}_{\kappa_0}$ starting in $s$ can only escape the region $P_0 \setminus (B_R(0,R) \cup B_R(0,-R)) \subset P_0$ through the arc segment $\partial P_0$. Since $\tau(1) \notin P_0$, the result follows.

□

This assures that for a moving particle, it will always cross the arc segment $\partial P_0$. Now we consider the sweeping of disks $D_\delta = \{x \in \mathbb{R}^2 | \|x\| \leq \delta\}$ with radius $\delta$ along a trajectory $\tau \in \mathcal{F}_{\kappa_0}$. Let us define $L_0 = D_{\delta_0}(s_0)$, $L_1 = D_{\delta_1}(s_1)$, and $s_1 = (l_0 \cos(\theta), l_0 \sin(\theta))$.

**Theorem 4.** *Let $L_0 = D_{\delta_0}(s)$. Then there exists $\theta \in [-\theta^L, \theta^L]$ such that for all $\tau \in \mathcal{F}_{\kappa_0}$ there exists $t_0 \in [0,1]$ such that $L_1 \subset (\tau(t_0) \oplus L_0)$ if $\delta_1 \leq \delta_0$.*

*Proof.* Due to Theorem 3 we have that a point starting from $s_0$ following a trajectory from $\tau \in \mathcal{F}_{\kappa_0}$ will necessarily cross $\partial P_0$. Let $\tau(t) \in \partial P_0$ be the crossing point. Let us choose $s_1 = \tau(t)$ as the position of link $L_1$. $\theta$ can be recovered by $\theta = \text{acos}\left(\frac{(s_1 - s_0)^T s'}{\|s'\| l_0}\right)$. Now at $s_1$ we have that the volume of $(\tau(s_1) \oplus L_0)$ is smaller than $(\tau(s_1) \oplus L_1)$ exactly when $\delta_1 \leq \delta_0$.

□

### 3.4.4 Generalization to $N$ sublinks

Let us define a linear linkage in *canonical form* in the following way: Let $L_0, \cdots, L_N \in D^2$ be disk links of radius $\delta_0, \cdots, \delta_N$ connected by lines of equal length $l_0, \cdots, l_{N-1}$ with $l_0 = \cdots = l_{N-1}$, $\delta_i > 0$, $l_i > \delta_i + \delta_{i+1}$, $\delta_i \leq \delta_0$ for all $i \in [0, N]$ and joints limits $\{\{-\theta_0^L, \theta_0^L\}, \cdots, \{-\theta_{N-1}^L, \theta_{N-1}^L\}\}$ with $\theta_0^L = \cdots = \theta_{N-1}^L$. We will refer to this canonical linear linkage structure as $\mathcal{R}_L^N$.

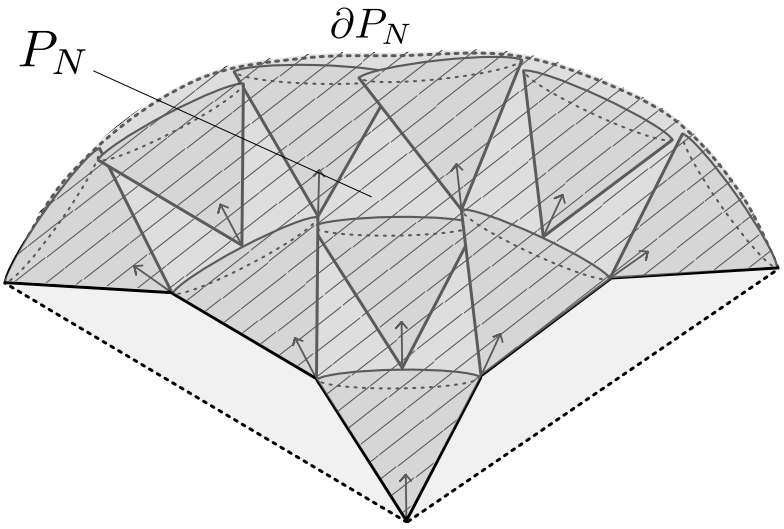



**Figure 3.7:** A succession of cones, spanning the space between $s$ and $\partial C_s$, which necessarily has to be traversed by any function from $\mathcal{F}_{\kappa_N}$.

Let us define by $P_N$ the interior of the space spanned by all possible sublink configurations, as depicted in Fig. 3.7. Let us define analog a functional space $\mathcal{F}_{\kappa_N}$ as

$$\mathcal{F}_{\kappa_N} = C^1([0,1], \mathbb{R}^2) \tag{3.8}$$

with $\tau(0) = s$, $\tau'(0) = s'$, $\tau(1) \notin P_N$, and for all $\tau \in \mathcal{F}_{\kappa_N}$ we have a maximum curvature given by

$$\kappa_N = \frac{2\sin(\theta^L)}{N l_0}, N > 1 \tag{3.9}$$

### 3.4.5 Irreducibility of Linear Linkage

For $N = 1$, we proved that there exist $\theta_1$ such that $L_1 \in \tau$. For $N > 1$, the tangent $t$ of $\tau$ might differ from the normal $n$ of the line $(L_0L_1)$. We denote the angle between $t$ and $n$ as $\theta_{D_i}$. See Fig. 3.8 for clarification. To ensure that we can always find a feasible configuration, such that all links are on $\tau$, we therefore need

to ensure that $\theta_i = \theta_{D_i} + \theta_t \leq \theta^L$ for all $i \in [0, N]$.

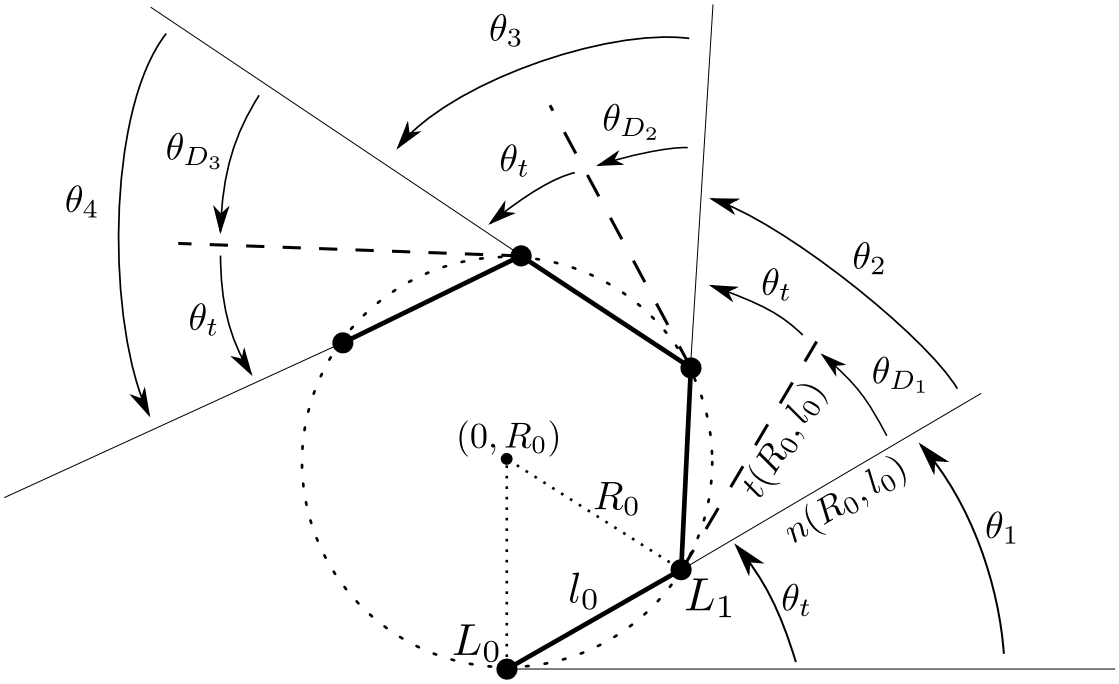



**Figure 3.8:** Linear Linkage along a curve $\tau$. The angle between the tangent to the osculating circle $t$ and the normal $n$ of the line $(L_0L_1)$ is given by $\theta_{D_i}$. The angle $\theta_t$ denotes the maximum angle given a maximal constant curvature $\kappa_N$.

For our proofs, we assume two premises to be true.

**P1** If $l_0 = l_i$ for all $i \in [1, N]$, then $\theta_{D_1} = \theta_{D_i}$ for all $i \in [1, N]$

**P2** Maximum angle between $t$ and $n$ can be found for $\tau = \tau_{\kappa_N}$ with $\tau_{\kappa_N}$ being the constant maximum curvature trajectory with curvature $\kappa_N$ everywhere.

We now want to determine the angles $\theta_t$,$\theta_{D_1}$ depending on the radius $R_0$ of the osculating circle.

By geometrical arguments of circle-circle intersection [1], we can write

$$\begin{aligned} n(R_0, l_0) &= \begin{pmatrix} \frac{l_0}{2R_0}\sqrt{4R_0^2 - l_0^2} \\ \frac{l_0^2}{2R_0} \end{pmatrix} \\ t(R_0, l_0) &= \begin{pmatrix} \frac{-l_0^2}{2R_0} + R_0 \\ \frac{l_0}{2R_0}\sqrt{4R_0^2 - l_0^2} \end{pmatrix} \end{aligned} \tag{3.10}$$

$$\begin{aligned} \theta_t &= \arctan\left(\frac{l_0}{\sqrt{4R_0^2 - l_0^2}}\right) \\ \theta_{D_1} &= \arccos\left(\frac{n \cdot t}{\|n\|\|t\|}\right) = \arccos\left(\frac{4R_0^2 - l_0^2}{4R_0^2}\right) \end{aligned} \tag{3.11}$$

We now choose a certain $R_0$, and prove that $\theta_t(R_0) + \theta_{D_1}(R_0) \leq \theta^L$. Let

$$R_0 = \frac{N l_0}{2 \sin \theta^L} \tag{3.12}$$

such that $\mathcal{F}_{\kappa_N}$ is defined by $\kappa_N = \frac{1}{R_0}$.

**Lemma 2.** *Given premises* **P1**, **P2** *and a trajectory* $\tau \in \mathcal{F}_{\kappa_N}$, *then for* $t \in [0, 1]$ *and* $\delta_0 = 0, \delta_i = 0$, *there exist joint configurations* $\theta_1, \cdots, \theta_N$ *for the linear linkage* $\mathcal{R}_L^N$, *such that every* $L_i$ *is located on* $\tau$. *Furthermore, the maximum distance between* $\tau$ *and the lines* $(L_0L_1)\cdots(L_{N-1}L_N)$ *is given by* $d_{\kappa_N} = R_0 - \sqrt{R_0^2 - \frac{l_0^2}{4}}$

*Proof.* $\theta_t$,$\theta_{D_1}$ evaluates to

$$\begin{aligned} \theta_t &= \arctan\left(\frac{\sin \theta^L}{\sqrt{N^2 - \sin^2 \theta^L}}\right) \\ \theta_{D_1} &= \arccos\left(\frac{N^2 - \sin^2 \theta^L}{N^2}\right) \end{aligned} \tag{3.13}$$

for $N > 1$.

[1] Circle-Circle Intersection – Wolfram Mathworld

Due to premise **P2**, we know that $\theta_t + \theta_{D_1}(R_0) \geq \theta_t + \theta_{D_1}(R)$ for $R \geq R_0$, and so we can concentrate on the maximum curvature case $R_0$. Due to premise **P1**, we now only have to prove that $\theta_t + \theta_{D_1} \leq \theta^L$. By induction on $N$, we get for $N = 2$

$$\begin{aligned}
\theta_t(2) &= \arctan\left(\frac{\sin\theta^L}{\sqrt{4-\sin^2\theta^L}}\right) \leq \arctan\left(\frac{\sin\theta^L}{2}\right) \\
&\leq \frac{\sin\theta^L}{2} \leq \frac{\theta^L}{2} \\
\theta_{D_1}(2) &= \arccos\left(1-\frac{\sin^2\theta^L}{4}\right) = 2\arctan\left(\frac{2\sin\theta^L}{8-\sin^2\theta^L}\right) \\
&\leq \frac{4\sin\theta^L}{8-\sin^2\theta^L} \leq \frac{4\sin\theta^L}{8} = \frac{\sin\theta^L}{2} \leq \frac{\theta^L}{2}
\end{aligned} \tag{3.14}$$

whereby we relied on the fact that for $x > 0$ we have $\arctan(x) \leq x$ since $\arctan'(x) = \frac{1}{1+x^2} \leq 1$, for $x > 0$ we have $\sin(x) \leq x$ since $\sin'(x) = \cos(x) \leq 1$, and that $\arccos(x) = 2\arctan\left(\frac{\sqrt{1-x^2}}{1+x}\right)$.

We now observe that

$$\begin{aligned}
\theta_t(N) &= \arctan\left(\frac{\sin\theta^L}{\sqrt{N^2-\sin^2\theta^L}}\right) \geq \arctan\left(\frac{\sin\theta^L}{N}\right) \\
&\geq \arctan\left(\frac{\sin\theta^L}{\sqrt{(N+1)^2-\sin^2\theta^L}}\right) = \theta_t(N+1) \\
\theta_{D_1}(N) &= \arccos\left(\frac{N^2-\sin^2\theta^L}{N^2}\right) \geq \arccos\left(1-\frac{\sin^2\theta^L}{N^2}\right) \\
&\geq \arccos\left(1-\frac{\sin^2\theta^L}{(N+1)^2}\right) = \theta_{D_1}(N+1)
\end{aligned} \tag{3.15}$$

which shows that $\theta_t(N) + \theta_{D_1}(N) \geq \theta_t(N+1) + \theta_{D_1}(N+1)$. Therefore we have $\theta^L \geq \theta_t(2) + \theta_{D_1}(2) \geq \cdots \geq \theta_t(N) + \theta_{D_1}(N)$ for $N > 1$ as required.

Now given the constant maximum curvature we have that the points $L_0, (0, R_0)$ and $L_1$ are creating an isosceles triangle. See Fig. 3.8 for visualization. The maximum distance of the line $(L_0L_1)$ and the circle can be obtained by the height

of the triangle, such that $d_{\kappa_N} = R_0 - \sqrt{R_0^2 - \frac{l_0^2}{4}}$.

□

**Theorem 5.** *Let $\tau = \tau_I \circ \tau_{\kappa_N} \circ \tau_E$ with $\tau \in \mathcal{F}_{\kappa_N}$ and $\tau_I, \tau_G$ be the linear extensions of $\tau$. If the root link $L_0$ moves along $\tau$, then for $\delta_i \leq \delta_0$ and $\frac{l_0^2}{2R_0} \leq \delta_0$, we have that there exists sublink configurations $\theta_1, \cdots, \theta_N$ such that the volume of the linear linkage $\mathcal{R}_L^N$ is a subset of $\tau \oplus L_0$*

*Proof.* By Lemma 2, the maximum distance of the linear linkage to $\tau$ is given by $d_{\kappa_N}$. If $\delta_0 \geq d_{\kappa_N}$, then any point on the linear linkage curve will be inside $\tau \oplus L_0$. By Theorem 2 we can choose $\theta_1, \cdots, \theta_N$, such that the center of every $L_i$ is located on $\tau$. Then there exists an instance $t$ such that $L_i = \tau(t)$. $L_i$ is a subset of $\tau \oplus L_0$ exactly if $\delta_0 \geq \delta_i$. □

We have showed that if the root link of a linear linkage moves on a $\kappa_N$-curvature constrained trajectory, then there exists a sublink configuration at every instance, such that all sublinks are inside of the swept volume of the root link.

### 3.4.6 3-Dimensional Conjecture

In 3 dimensions, a space curve is defined by its curvature and torsion [74]. We will conjecture that our results apply also to 3 dimensions. Let us define the following functional space

$$\mathcal{F}_{\kappa,T} = C^2([0,1], \mathbb{R}^3) \tag{3.16}$$

with $\tau \in \mathcal{F}_{\kappa,T} \Rightarrow \tau(0) = s, \tau'(0) = s', \tau''(0) = s''$ and that $\tau(1)$ is outside a cone $P_0$ spanned by $s$, and the length of the link $l_0$, as depicted in Fig. 3.6. The

curvature $\kappa$ and torsion $T$ of $\tau$ is constrained to be

$$\kappa = \frac{2\sin(\theta^L)}{l_0}, \quad T \in \mathbb{R} \tag{3.17}$$

**Conjecture 1.** *Theorem 5 holds for $\mathcal{F}_{\kappa,T}$ in 3-dimensions.*

We will use this conjecture in our planning algorithm, to verify it experimentally and let the proof for future work.

Finally, we want to point out that completeness is not maintained for $\mathcal{F}_{\kappa,T}$

**Theorem 6.** *The motion planning problem A for $\mathcal{R}^N_L$ is not complete in $\mathcal{F}_{\kappa,T}$*

*Proof.* Since we constraint the functional space to not allow functions with curvatures $> \kappa$, we can trivially construct a counterexample in the following way: let us consider a disk $D^2 = \{x \in \mathbb{R}^2 | \|x\| \leq \delta\}$ with radius $\delta$, starting at a point $s$ and having direction $s'$. We construct an environment $E$ by sweeping the disk along a constant $\kappa'$ curvature curve $\phi$, connecting $(s, s')$ to $(s, s')$, whereby $\kappa' > \kappa$. Let us now look at the motion planning problem of planning for $D^2$ from $(s, s')$ to a point $(p, p')$, with $(p, p') \in E$. Visualized in Fig. 3.9. Since the environment is not intersecting the boundary of the cone $P_0$, which is constructed by $s, s', \kappa'$, it follows from Theorem 4 that no function can reach $(p, p')$. □

We established so far that if we can find a feasible trajectory for link $L_0$ under a curvature constraint, then we can find a trajectory for the whole linear linkage, which is feasible. We showed that this is not complete, however we can define a weaker version of completeness, which we call $\kappa$-curvature completeness

**Definition 11.** *A motion planning algorithm is $\kappa$-curvature complete if it finds a trajectory in the functional space $\mathcal{F}_{\kappa_0} \subset \mathcal{F}$, if one exists, or correctly reports that no such exist.*

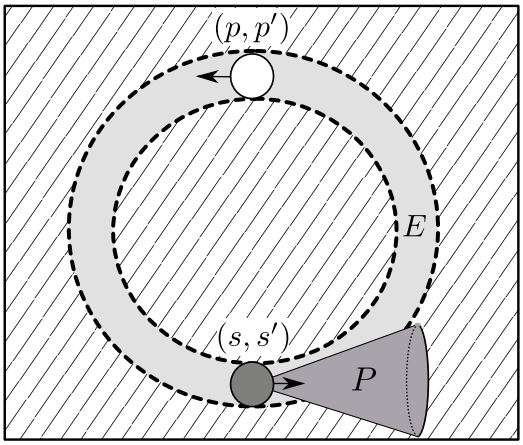



**Figure 3.9:** Visualization of a simple completeness counterexample, in which an environment $E$ has to be solved, which follows a $\kappa' > \kappa$ curvature curve.

We observe that this is a weaker version, such that completeness would imply $\kappa$-curvature completeness, but not the other way round. This is depicted schematically in Fig. 3.10.

The next section will be devoted to develop a $\kappa$-curvature complete algorithm.

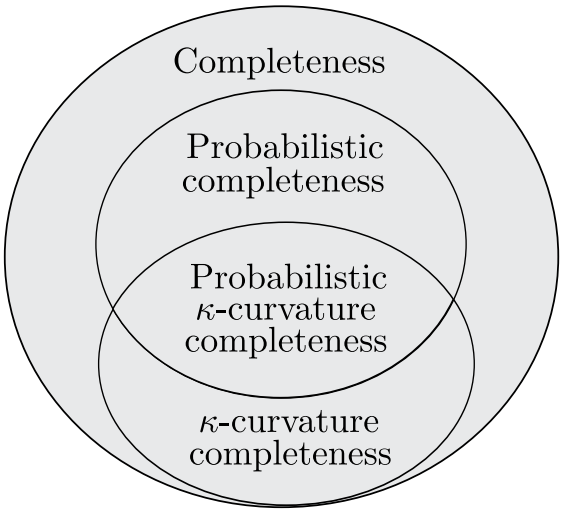



**Figure 3.10:** The $\kappa$-curvature completeness property and its relation to probabilistic completeness and completeness.

## 3.5 Irreducible Curvature Complete Algorithm

In Theorem 5, we established that a linear linkage $\mathcal{R}_L^N$ with links $L_0 \to \cdots \to L_N$ has a feasible solution if we can find a feasible solution for $L_0$ which respects a certain curvature $\kappa$. Here, we describe an algorithm to compute this solution. We will use spherical joints for the sublinks, such that we have joint configurations $\theta_1, \cdots, \theta_N, \gamma_1, \cdots, \gamma_N$.

Now, given a trajectory $\tau \in \mathcal{F}_{\kappa_N}$ for $L_0$, we compute feasible joint configurations for the sublinks $L_1, \cdots, L_N$. A rotational joint can be seen as a special case with $\gamma_1 \cdots, \gamma_N = 0$.

Let $A = \{\mathcal{R}_L^N, \mathcal{C}, q_I, q_G, \mathbf{E}\}$ be a motion planning problem for $\mathcal{R}_L^N$. Let $\tau \in \mathcal{F}_{\kappa_N}$ be the trajectory of the root link $L_0$. If $\tau \oplus L_0$ is a feasible solution, then by Theorem 5 we are guaranteed to find a feasible configuration such that $\tau \oplus (L_0 \cup \cdots \cup L_N)$ is a feasible solution. We will describe now how to find the configurations given a trajectory $\tau \in \mathcal{F}_{\kappa_N}$.

For all $t_0 \in [0, 1]$ we compute $\theta_1$ by the following procedure: start at $\tau(t_0)$ and move along $\tau$ in backward direction. See Fig. 3.12. Since we are guaranteed by Theorem 3 that we will meet $\partial P_0$, we can denote the intersection point as $t_n < t_0$ with $\|\tau(t_n) - \tau(t_0)\| = l_0$. Then we have

$$\theta = \operatorname{acos}\left(\frac{-\tau'(t_0)^T(\tau(t_n) - \tau(t_0))}{\|\tau'(t_0)\|\|\tau(t_n) - \tau(t_0)\|}\right) \tag{3.18}$$

from $t_q$ we recursively compute all $\theta$ values.

As a technical detail, we note that this requires that even at $q_I$, we can follow the trajectory backwards. Therefore, we need to extend the trajectory by moving along the sublinks at $q_I$ to obtain an extended trajectory $\tau = \tau_I \circ \tau$.

The resulting algorithm is described in Fig. 3.11. It takes the input trajectory $\tau$ and produces a resulting configuration vector at each instance $t \in [0, 1]$ along $\tau$,

such that the resulting swept volume of all links is inside the swept volume of the root link, i.e. $(\tau \oplus L_0 \cup \cdots \cup L_N) \subseteq (\tau \oplus L_0)$. The complexity scales with $\mathcal{O}(N)$. The algorithm has been implemented in python and is available as a standalone module

`https://github.com/orthez/irreducible-curvature-projection/`

### 3.5.1 Irreducibility Assurance Controller

The analytical computation of the irreducible configuration at instance $t$ enables us to design a control algorithm, which pushes the robot body towards an irreducible trajectory.

Let us denote by $\phi : \mathcal{F} \times [0,1] \to \mathbb{R}^N \times \mathbb{R}^N$ the computation of joint angles for our spherical joint from the current trajectory $\tau \in \mathcal{F}$ of body $L_0$ at instance $t_0 \in [0,1]$. The output are joint angles $\theta, \gamma$ specifying the position of the spherical joints at instance $t_0$. Let us denote by $\varphi : [0,1] \to \mathbb{R}^N \times \mathbb{R}^N$ the *measured joint angles* at instance $t_0 \in [0,1]$.

A proportional gain controller can be constructed as $u(t) = K_p e(t)$ with $e(t) = \|\phi(t) - \varphi(t)\|$. This gives a hint at the possibilities of this geometrical inspired approach. In general, using the controller will minimize the swept volume, which could be useful in different areas. We note that minimal swept volume loosely relates to minimal air resistance. For example, an octopus robot could use this to let the arms trail behind its body while moving, such that water resistance is minimized. A road train — a tractor unit pulling two or more trailers — could minimize its air resistance to minimize gas consumption.

**Algorithm 2:** Irreducible Curvature Projection

**Data**: $t_0, \tau, \tau', \tau'', \delta_{0:N}, l_{1:N}, \Delta t$
**Result**: $\theta_{1:N}, \gamma_{1:N}$
$\mathbf{e}_1 \leftarrow \tau'(t_0)$;
$\mathbf{e}_2 \leftarrow \tau''(t_0)$;
$\mathbf{e}_3 \leftarrow \tau'(t_0) \times \tau''(t_0)$;
$t_{cur} \leftarrow t_0$;
$\mathbf{R} \leftarrow \begin{pmatrix} \mathbf{e}_1 \cdot \mathbf{e}_x & \mathbf{e}_2 \cdot \mathbf{e}_x & \mathbf{e}_3 \cdot \mathbf{e}_x \\ \mathbf{e}_1 \cdot \mathbf{e}_y & \mathbf{e}_2 \cdot \mathbf{e}_y & \mathbf{e}_3 \cdot \mathbf{e}_y \\ \mathbf{e}_1 \cdot \mathbf{e}_z & \mathbf{e}_2 \cdot \mathbf{e}_z & \mathbf{e}_3 \cdot \mathbf{e}_z \end{pmatrix}$;
**for** $i \leftarrow 1$ **to** $N$ **do**
  $t_n \leftarrow t_0$;
  **while** $\|\tau(t_n) - \tau(t_{cur})\| \leq l_i$ **do**
    $t_n \leftarrow t_n - \Delta t$
  $\tau_n \leftarrow \tau(t_n)$;
  $p_I \leftarrow \tau(t_n) - \tau(t_{cur})$;
  $p_W \leftarrow \mathbf{R}^T p_I$;
  $x_L \leftarrow (-1, 0, 0)^T$;
  $p_{xy} \leftarrow p_W - (p_W^T \mathbf{e}_z)\mathbf{e}_z$;
  $p_{zx} \leftarrow p_W - (p_W^T \mathbf{e}_y)\mathbf{e}_y$;
  $\theta_i \leftarrow \text{acos}(\frac{p_{xy} x_L}{\|p_{xy}\| \|x_L\|})$;
  $\gamma_i \leftarrow \text{acos}(\frac{p_{zx} x_L}{\|p_{zx}\| \|x_L\|})$;
  **if** $p_W^T \mathbf{e}_z < 0$ **then**
    $\gamma_i \leftarrow -\gamma_i$;
  **if** $p_W^T \mathbf{e}_y > 0$ **then**
    $\theta_i \leftarrow -\theta_i$;
  $\mathbf{R} \leftarrow \mathbf{R} \cdot \mathbf{R}_Y(\gamma_i) \cdot \mathbf{R}_Z(\theta_i)$;
  $\mathbf{e}_1 \leftarrow \mathbf{R}\mathbf{e}_x$;
  $\mathbf{e}_2 \leftarrow \mathbf{R}\mathbf{e}_y$;
  $\mathbf{e}_3 \leftarrow \mathbf{R}\mathbf{e}_z$;
  $t_{cur} \leftarrow t_n$;

**Figure 3.11:** Irreducible Curvature Projection Algorithm. $\mathbf{e}_x, \mathbf{e}_y, \mathbf{e}_z$ represent the $x, y, z$ basis vectors, respectively.

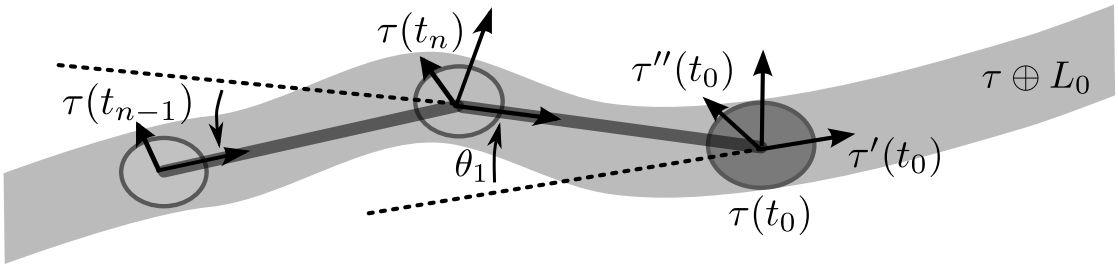



**Figure 3.12:** Given a trajectory $\tau \in \mathcal{F}_{\kappa_N}$, we can analytically compute the joint configurations, such that sublinks of the linear linkage are reduced, i.e. they are inside of the swept volume of $\tau \oplus L_0$..

## 3.6 Experiments

We performed two experiments to verify our theoretical results. First, a swimming snake in a 2d and a 3d environment. Planning is conducted for the head of the snake under a curvature constraint. After finding a feasible head trajectory we can use the Irreducible Curvature Projection Algorithm to project the remaining sublinks into the swept volume of the head. Second, we planned a constrained motion for the humanoid robot HRP-2, where we plan a motion for a reduced mechanical model with 7 dimensions. After planning a motion, we then use our projection algorithm to find the position of the remaining links.

### 3.6.1 Swimming Snake

For the snake simulation, we have choosen a bounded curvature, and estimated the number of links, such that we obtain the longest possible irreducible snake. Our values were $\kappa = 1\text{m}^{-1}$, $\delta_0 = 0.23\text{m}$, $\delta_i = 0.138\text{m}$, $l_0 = 0.33\text{m}$ and $\theta^L = \frac{\pi}{2}$ giving rise to

$$N = \left\lfloor \frac{2\sin(\theta^L)}{\kappa l_0} \right\rfloor = 6 \tag{3.19}$$

Planning with our curvature-constrained functional space is equivalent to plan-

ning a path for the non-holonomic snake's head subject to differential constraints describing forward non-slipping motions and for which we will assume constant speed. Note that this is equivalent to the model of Dubin's car. This can be solved in both 2d and 3d using kinodynamic planning [11].

In 2d, the configuration space of the snake's head is $SE(2)$ with $q = (x, y, \theta)^T$ and the differential model is given by

$$\begin{aligned} \dot{x} &= \cos\theta \\ \dot{y} &= \sin\theta \\ \dot{\theta} &= u \end{aligned} \tag{3.20}$$

where the control space is defined by the steering angle $u$. In 3d, the configuration space is $SE(3)$ and the differential model is similar to a driftless airplane given by

$$\dot{q} = q\left(\sum_{i=1}^{3} u_i X_i + X_4\right) \tag{3.21}$$

where

$$X_1 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad X_2 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad X_3 = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$
$$X_4 = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad X_5 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad X_6 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

is a basis for $\mathfrak{se}(3)$, the Lie algebra of $SE(3)$.

The controls $u_1, u_2$ and $u_3$ are then the roll, pitch and yaw steering angles, respectively. We have performed one experiment in 2d in a rocky environment, and averaged the results for the classical and the irreducible case over 100 experiments, as reported in Tab. 3.3. While having the same success rate, the planning time is reduced by one order of magnitude. We further planned a single motion in 3d, where the snake has to swim through holes in a formation of rocks. Fig. 3.13 shows the results of our projection algorithm with the swept volume of the head in magenta.

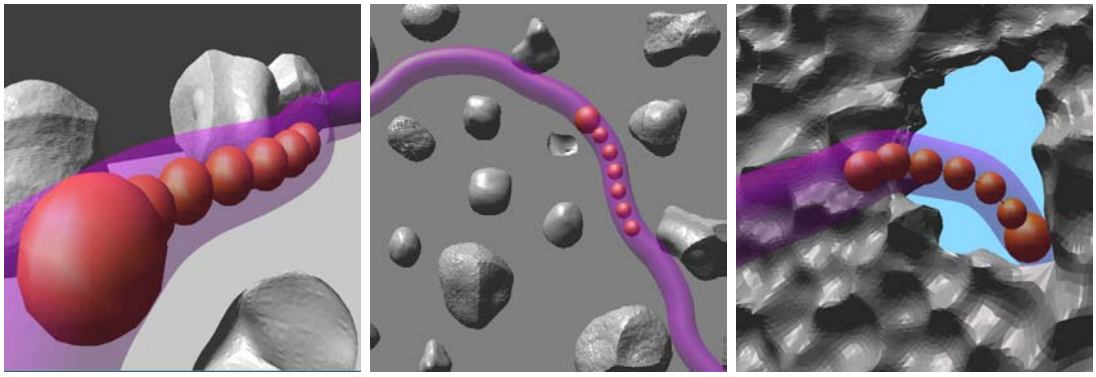


**Figure 3.13:** Planning for the head of a swimming snake in 2D (left, middle), and in 3d (right). The swept volume of the head is shown in magenta. The position of the sublinks is an output of the curvature projection algorithm.

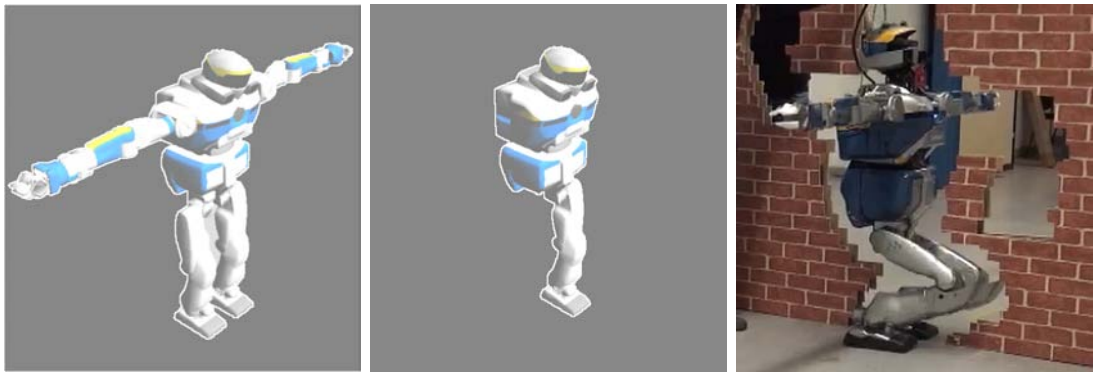


**Figure 3.14:** We use a reduced mechanical model for motion planning, which preserves curvature completeness for the linear arm linkages with respect to the chest (left,middle). After planning for the reduced model, we can project the remaining links into the swept volume, and thereby solving very narrow environments (right, adapted from [16]).

### 3.6.2 Humanoid Robot

Next, we conduct motion planning for the humanoid robot HRP-2, by abstracting away the two arms as linear linkages. Also, we consider the right leg as a linear linkage connected to the left leg. We additionally approximate the head by a sphere, so that yaw rotations leave the head invariant. This leaves us with an effective configuration space of $\mathbb{R}^8$, which is shown in Table 3.1. Motion planning can now be conducted with a *reduced* mechanical model, as shown in Fig. 3.14.

### Curvature constraint for chest HRP-2

Each arm of HRP-2 is a linear linkage, which we will approximate by four spheres as depicted in Fig. 3.15. We positioned the spheres at the moveable joints of the robot. The resulting linear linkage has $N = 4$ links with length $L_0 = 0.25\text{m}$ and sphere radius of $\delta = 0.08\text{m}$. We choose a common joint interval $[-\frac{\pi}{4}, \frac{\pi}{4}]$ for the free joints. We can compute the resulting maximum curvature as

$$\kappa = \frac{2\sin(\frac{\pi}{4})}{3L_0} = 1.8856\text{m}^{-1} \tag{3.22}$$

Meaning, if we can find a trajectory of the chest (without considering the arms), which has a bounded $\kappa$ curvature, then we are guaranteed to find joint angles for the arm, such that the swept volume of the arms and the chest is a subset of the swept volume of the chest. The resulting joint limits for the arms of HRP-2 are shown in Table 3.2.

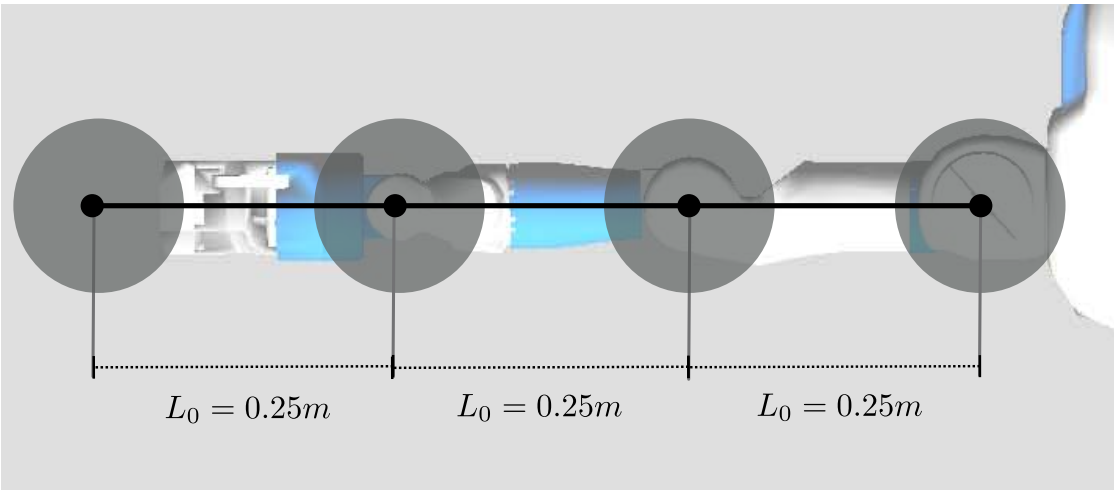



**Figure 3.15:** Approximation of the arm as a linear linkage in canonical form

**Table 3.1:** Variable Joints of Humanoid Robot HRP-2, and the corresponding range. If the value is set to $\phi$, then the joints are ignored for motion planning, and are determined by the irreducible projection algorithm in a post-processing stage.

| Joint | Fixed Value | Anatomical Name | Range |
|---|---|---|---|
| HEAD0 | 0.0 | | |
| HEAD1 | - | Neck | $[-0.52, 0.79]$ |
| CHEST0 | 0.0 | | |
| CHEST1 | - | Waist | $[-0.09, 1.05]$ |
| RARM | $\phi$ | Right Arm | |
| LARM | $\phi$ | Left Arm | |
| LLEG0 | 0.0 | | |
| LLEG1 | 0.0 | | |
| LLEG2 | - | Hip | $[-2.18, 0.73]$ |
| LLEG3 | - | Knee | $[-0.03, 2.62]$ |
| LLEG4 | - | Ankle | $[-1.31, 0.73]$ |
| LLEG5 | 0.0 | | |
| RLEG | $\phi$ | Right Leg | |
| LSOLE_X | - | Left Foot | $[-0.5, 0.5]$ |
| LSOLE_Y | - | Left Foot | $[-3.0, 3.0]$ |
| LSOLE_$\theta$ | - | Left Foot | $[0, 2\pi]$ |

**Table 3.2:** Values for the approximated linear linkage structure of the arms of HRP-2. Our curvature algorithm determines the exact values based on the movement of the chest.

| Joint | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|
| Left Arm | $\frac{-\pi}{2}$ | $[\frac{\pi}{4}, \frac{3\pi}{4}]$ | $\frac{-\pi}{2}$ | $[\frac{-\pi}{4}, \frac{\pi}{4}]$ | 0.0 | $[\frac{-\pi}{4}, \frac{\pi}{4}]$ | 0.1 |
| Right Arm | $\frac{-\pi}{2}$ | $[\frac{-3\pi}{4}, \frac{-\pi}{4}]$ | $\frac{-\pi}{2}$ | $[\frac{-\pi}{4}, \frac{\pi}{4}]$ | 0.0 | $[\frac{-\pi}{4}, \frac{\pi}{4}]$ | 0.1 |

### Implementation Details

For our simulations, we use the humanoid path planner (HPP) framework [88]. It is a general motion planning framework based on random sampling techniques [11], tailored for planning on humanoid robots like HRP-2. We will make use of a planning algorithm based on sliding motions. A sliding motion is dynamically stable, as we discussed in Sec. 3.2, and is particulary suitable for constrained environment as it locally minimizes the swept volume by minimizing oscillations. From a motion planning point of view, a sliding motion is easier to deal with computationally: while planning discrete contact steps gives rise to a combinatorial explosion, a continuous sliding motion can be optimized by taking derivative informations into account.

To plan a single motion, we use the rapidly-exploring tree (RRT) [22] algorithm. We replace the basic configuration shooter, which samples a random configuration from the configuration space by an irreducible configuration shooter, to only sample inside the subspace generated by ignoring the arms and the right leg. After planning, we compute the reduced configurations by using the irreducible curvature projection algorithm.

The irreducible configuration shooter has been released as an open-source submodule for the HPP framework, which can be found here

`https://github.com/orthez/hpp-motion-prior/`

### Experimental Results

To test our theoretical results, we have chosen a motion planning problem, where the robot HRP-2 has to move through a wall, as shown in Fig. 3.16. Those results have been taken from [13]. Due to the wall constraint, a solver has to find a narrow passage in the configuration space to solve the problem. In the classical

35-dof setting, this problems has not been solved, since in practice the probability to find a feasible configuration vanishes towards zero. We consider here the 8-dof setting without waypoints, by using the irreducible subspace.

The results of 10 runs are reported in Table 3.3. Since the passage is narrow, RRT can take a long time to converge, for our experiment, it took between 44 minutes up to 43 hours. This shows that sampling-based methods are becoming inefficient in narrow environments, which is closely related to the $\epsilon$-goodness criteria [89], which states that the convergence rate of sampling-based methods is inversly proportional to the volume of the free configuration space.

We have successfully applied the irreducibility concept on the HRP-2 humanoid walk through the wall. This experiment, however, uses a different planning algorithm which exploits environmental structure, and follows the resulting trajectory by using a hierarchical task-space controller. We submitted those results in [16].

Since this chapter is concerned with a feasibility study, the resulting motion will be non-optimal, assumes infinitesimal small footsteps and might appear unnatural to a human observer. However, having a first feasible trajectory is a prerequisite for fast convergence of local planning algorithms like CHOMP [90] or AICO [68].

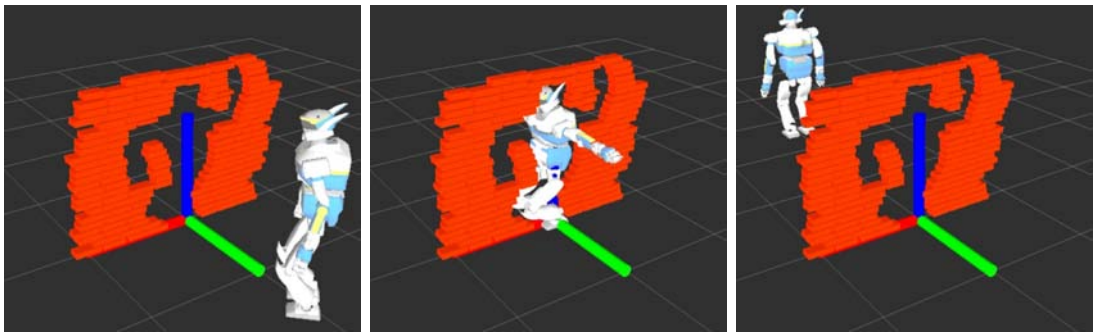


**Figure 3.16:** Wall Motion Planning Problem. **Left** initial configuration **Middle** one irreducible configuration on the final trajectory found by an RRT on the irreducible subspace **Right** goal configuration. Adapted from [13].

**Table 3.3:** Simulation results for the snake and for the humanoid robot. The "snake 2d Rocks" and "Snake 3d Rock Formation" refers to the environment shown in Fig. 3.13. *HRP-2 Wall* refers to the experiment in Fig. 3.16. Results taken from [13].

| Planning Problem | $\mathcal{C}$ Dimension | #Success/ #Experiments | $\sigma$(s) | $\mu$(s) |
|---|---|---|---|---|
| Snake 2d Rocks (Classical) | $\mathbb{R}^{3+N}$ | 100/100 | 54.15s | 94.36s |
| Snake 2d Rocks (Irreducible) | $\mathbb{R}^3$ | 100/100 | 1.34s | 1.04s |
| HRP-2 Wall (Classical) | $\mathbb{R}^{35}$ | Not solveable (> 3days) | | |
| HRP-2 Wall (Irreducible)[13] | $\mathbb{R}^7$ | 10/10 | 12h14m | 9h34m |

## 3.7 Discussion

The theoretical framework presented is able to simplify motion planning problems by exploiting the linear linkage structure, which can be found in a diverse number of mechanical systems, including snakes, octopuses and humanoid robots.

Our conceptual idea is a completeness-preserving dimensionality reduction technique. To apply this concept in practice, we introduced a new concept called $\kappa$-curvature completeness. This $\kappa$-curvature completeness is in general a proper subset of completeness, and therefore we can always find certain situations in which we cannot find a solution, even if one exists. We believe, however, that for some mechanical systems $\kappa$-curvature completeness and completeness are equivalent, for example for systems which resemble Dubin's car with trailers and positive velocity.

Motion planning can now be simplified by first planning under a certain curvature constraint in the reduced dimensionality space. If a motion plan has been

found, we can execute it. If no plan has be found, we can increase the dimensionality.

In the larger scheme, we think about irreducibility as one component of *motion prior* information: developing efficient motion planning algorithms requires us to make use of the underlying structure of the problem. Here, we showed that certain mechanical systems allow us to exploit their linear linkage structure.

Finally, it seems that linear linkages are quite common in nature. Irreducibility could be a way to motivate why the octopus aligns its limbs behind its head during swimming. Besides minimizing water resistance, it could also thereby simplify motion planning. We think there is a variety of interesting phenomena which could be studied by exploiting our concept of an irreducible trajectory in motion research.

## 3.8 Conclusion

We described the concept of irreducibility, which allows us to conduct completeness-preserving dimensionality reduction for motion planning. The main result in Theorem 2 states that finding no feasible trajectory in the space of irreducible trajectories implies that there is no feasible trajectory in the space of all configuration space trajectories, i.e. that motion planning is complete w.r.t. irreducible trajectories.

We have described how irreducibility can be applied to linear linkages by using the concept of $\kappa$-curvature completeness. Based on those results, we developed a linear-time algorithm to project configurations into the swept volume of the root links of a linear linkage. Finally, we conducted a set of experiments for the humanoid robot HRP-2, by considering the arms as linear linkages.

Future research will focus on the automatic discovery of the irreducible trajec-

tory space, on the correctness of the conjectures in Sec. 3.4.6, and on applying our principle to more general linkage structures.

# Chapter 4

# Homotopic particle motion planning for humanoid robotics

*"Clearly a complete understanding of walking requires a theory of spatial memory"*

— David Rosenbaum, Human Motor Control

*"A good model should account for the environment"*

— de Groot, A. & Gobet, F., Perception and Memory in Chess

## 4.1 Summary

We showed in Chapter 3, that exploiting the inherent mechanical structure of a robot is essential to an understanding of motion planning. In this chapter, we want to show that the environment equally consists of a rich structure which we can exploit. In particular, we exploit the topology of the environment to discover connected components. Inside a connected component, instead of planning one trajectory in configuration space, motion planning can be seen as optimizing a set

of homotopically equivalent particle trajectories. Our contributions are: i) finding the homotopy classes of a single footstep trajectory in an environment, ii) finding a single footstep trajectory in a single homotopy class formulated as a convex optimization problem, and iii) finding a feasible upper body trajectory given a footstep trajectory, formulated as a set of convex optimization problems. This view provides us with important insights into the difficulty of motion planning, and – under some assumptions – allows us to provide the number of local minima of a given motion planning problem. We demonstrate our approach on a real humanoid platform with 36-dof in a highly restricted environment.

## 4.2 Introduction

We recall that we define the motion planning problem as $A = \{\mathcal{R}, \mathcal{C}, q_I, q_G, \mathbf{E}\}$ [11] with $\mathcal{R}$ be a robotic system, $\mathcal{C}$ the configuration space of $\mathcal{R}$, $q_I \in \mathcal{C}$ the initial configuration, $q_G \in \mathcal{C}$ the goal configuration and $\mathbf{E}$ the environment.

The motion planning problem was shown to be NP-hard [20], and for humanoid robots, computational time can become several hours in a narrow environment [13, 23, 44]. We argue that the main problem is the reliance on random sampling techniques [22]: if the subset of feasible configuration gets arbitrarily small, the convergence rate of random sampling gets arbitrarily high [89]. While random sampling is excellent for solving the problem in general, we argue here that to design truly efficient algorithms, we need to study, understand and exploit the underlying structure of the motion planning problem.

Here, we concentrate on investigating and exploiting the environment structure by extracting homotopy classes. A homotopy class is a set of functions, which can be continuously deformed into each other, as depicted in Fig. 4.1. For each homotopy class, we consider the trajectories of a set of particles $\{\tau_k\}_{k=0}^{\eta}$ on the

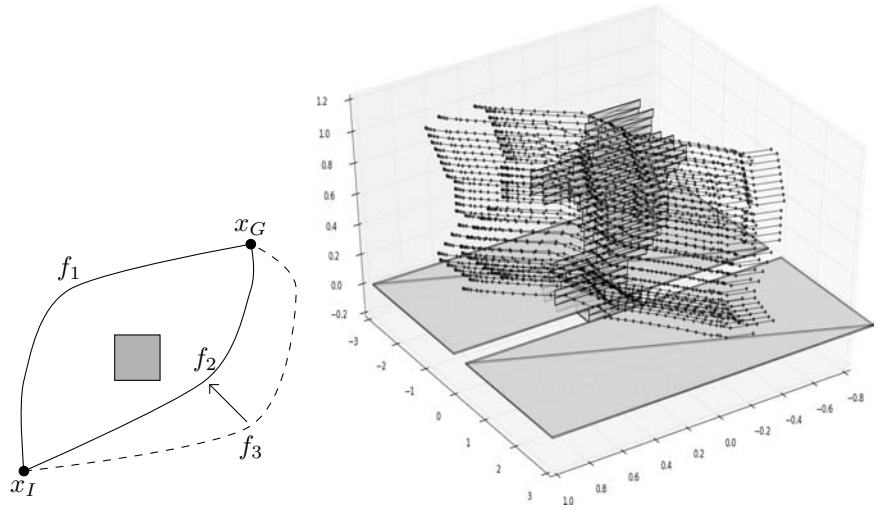



**Figure 4.1: Left**: Two functions are in the same homotopy class, if they can be continuously deformed into each other, while fixing their end points. $f_1$ would be not homotopically equivalent to $f_2$, while $f_3$ would be. **Right**: In 3d, we conduct homotopic motion planning for a set of particles, particles with homotopically equivalent space curves.

robot body moving through $\mathbb{R}^3$ on space curves of the form $\tau_k : [0, 1] \to \mathbb{R}^3$. We assume here that all particle trajectories are homotopically equivalent. Motion planning can then be conducted in the environment by first finding a single particle trajectory, and then finding all particles on the robot body by restricting them to belong to the same homotopy class as the single particle trajectory.

Towards this goal, we decompose the open space of the environment into smaller volumes and analyze their covering to compute homotopy classes of robot particles moving through open space. We argue here that performing motion planning locally in one homotopy class ensures continuity, which is a requirement for optimization based planners. This decomposition of motion planning is called homotopic motion planning [50].

Our contributions are

- Identification of the homotopy classes in a given environment for a sliding footstep and the approximation of the free space

- Formulation of the problem of finding a sliding contact trajectory in one homotopy class as a convex optimization problem
- Formulation of the problem of finding a set of particle trajectories on the robot body, which are constrainted by the contact trajectory, as a set of convex optimization problems

Sec. 4.4 describes how we decompose an environment into walkable surfaces, homotopy classes and the free space inside of one homotopy class. Sec. 4.5 formulates optimizing a single footstep trajectory as a convex optimization problem, and Section 4.6 formulates the upper body optimization as a set of convex optimization problems under convex inequality constraints from the environment. The reader is refered to consult Fig. 4.2 for a technical overview.

## 4.3 Related Work

Bhattacharya et al. [50] compute homotopy classes in the environment, and use them as a constraint for graph-based search. Our work is complementary in the sense that we investigate how to formulate planning in one homotopy class as a set of convex optimal problems, while their work investigates how to compute the homotopy classes in the first place.

The technique presented in [47] estimates a single homotopy class by growing random spheres. Our approach tries to be more systematic in that we reason about contact surfaces, and restrict the free space by the robots geometry. Also, we consider planning inside a homotopy class not as a potential field controller, but as a global optimization procedure.

The work by [91] consider sweeping a spherical object to find weakly collision free footstep positions. Our work is similar for footsteps but precomputes homotopy classes to identify high-level minima.

[92] identifies narrow passage in the environment, and computes important waypoint configurations inside those narrow passages. This idea inspired our computation of connector elements, elements which connect two contact surfaces.

Ivan et al. [93] introduce different topological representations which makes it easier to solve certain subproblems of motion planning. Our work is complementary, in that we would be able to analyze which representation to use given a certain problem.

The authors of [7] discover convex regions of footsteps in an environment, and employ mixed-integer programming to find a solution. Our work explores how adding more structure in form of connectivity can help to discover the homotopy classes, and formulate the resulting problem as a set of convex optimization problems.

Farber [51] introduced the topological complexity of a configuration space. Our work can be seen as a practical means of identifying the covering of the workspace volume and thereby its topological complexity. Our optimization algorithms are then one proposal to find paths inside of a given covering.

This work is fundamentally based on the work by [13], who introduced irreducible configuration for humanoid robots. Our work is complementary in that we are restricting our motions to the space of irreducible configuration while exploiting environment structure.

## 4.4 Environment Homotopy Decomposition

In this section, we describe how we compute the free space of a given environment, and its connectivity. We start by reasoning about surfaces on which a foot contact is possible, which we call *walkable surfaces*. For each walkable surface, we compute its free space stack, a set of boxes on top of the surface in which the *swept volume* of

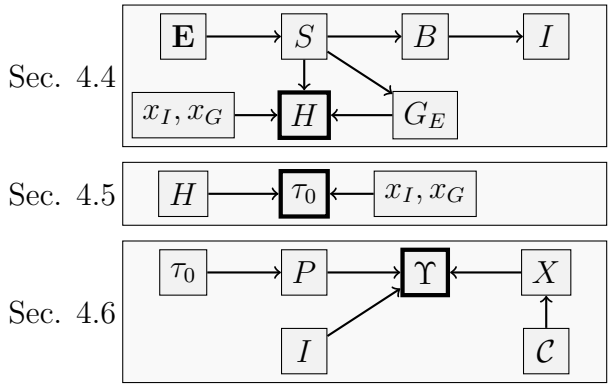



**Figure 4.2:** The conceptual overview about this chapter. **Top**: in section 4.4 we decompose the environment $\mathbf{E}$ into walkable surfaces $S$, intersections $I$ and intersections $I$. From the start contact $x_I$, the goal contact $x_G$ and the connectivity graph $G_E$ we compute the homotopy classes $H$ on $S$. **Middle**: Given the homotopy classes $H$, the start contact $x_I$ and the goal contact $x_G$, we compute a sliding footstep trajectory $\tau_0$, supported on $H$. **Bottom**: We compute upper-body particle trajectories $\Upsilon$ from a given footstep trajectory $\tau_0$, planes $P$, cuboids $B$, and from cross-sections $X$ generated by sampling robot configurations $\mathcal{C}$.

the robot $\mathcal{R}$ necessarily has to lie. We further represent the connectivity of surfaces by a graph structure. This graph structure then enables us identify homotopy classes.

We will consider a decomposition of the environment into a set of objects as

$$\mathbf{E} = O_1 \sqcup \cdots \sqcup O_\alpha \tag{4.1}$$

with $O_i$ being a bounded convex polytope

$$O_i = \{x \in \mathbb{R}^3 | a_j^{(i)T} x \leq b_j^{(i)}, \|a_j^{(i)}\|_2 = 1, j \in [1, \alpha_i]\} \tag{4.2}$$

We make here the assumption that every object $O_i$ is a convex polytope. If an object is not a convex polytope, we decompose it into convex subobjects [94], such that we can operate without loss of generality on convex polytopes.

For every object $O_i$, we define the $p$-th surface element as

$$\begin{aligned} S_i^p = \{x \in \mathbb{R}^3 | a_p^{(i)T} x = b_p^{(i)}, a_j^{(i)T} x \leq b_j^{(i)}, \\ j = 1, \cdots, p-1, p+1, \cdots, \alpha_i\} \end{aligned} \tag{4.3}$$

with $a_p^{(i)}$ the surface normal, and $b_p^{(i)}$ the distance to the origin.

**Definition 12** (Walkable Surface)**.** *A surface element $S_i^p$ is called* walkable*, if*

1. *the slope of $S_i^p$ is smaller than the maximum slope $\mathcal{R}_\theta$ the robot can stand on*

$$\|a_p^{(i)} - v_g\| \leq \sqrt{(2 - 2\cos(\mathcal{R}_\theta))} \tag{4.4}$$

   *with $v_g = (0, 0, 1)^T$*

2. *the foot of radius $\mathcal{R}_{FR}$ is fully contained inside $S_i^p$, meaning the following convex problem is feasible (based on the maximum inscribed circle problem [63])*

$$\begin{aligned} \underset{x \in \mathbb{R}^n, r \in \mathbb{R}}{\textit{maximize}} \quad & R \\ \textit{subject to} \quad & a_i^T x + R a_i^T a_i' \leq b_i, \\ & i = \{1, \cdots, p-1, p+1, \cdots, \alpha_i\} \\ & a_p^T x = b_p \\ & R \geq \mathcal{R}_{FR} \end{aligned} \tag{4.5}$$

   *whereby $r$ is the radius of the circle, $x$ the center, $a_i'$ is the orthogonal projection onto the hyperplane of $a_p$, i.e. $a_i' = a_i - (a_i^T a_p) a_p$. Visualized in Fig. 4.3.*

We now add a notion of connectivity:

**Definition 13** (Connectivity)**.** *Two walkable surfaces $S_i, S_j$ are called* connected*,*

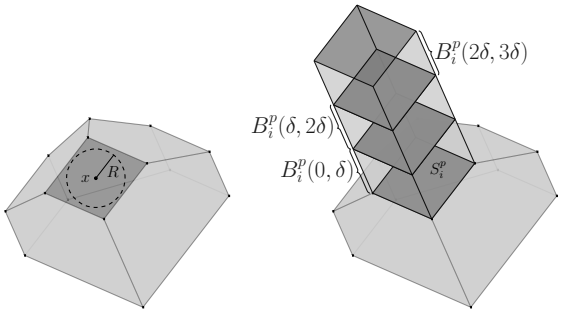



**Figure 4.3: Left**: a polytope (light gray), a surface element (dark gray) and an inscribed circle with radius $R$ and center $x$. **Right**: a set of cuboids $B_i^p$ on top of one surface element $S_i^p$ (dark gray)

*iff* $d(S_i, S_j) < \mathcal{R}_{StepWidth}$*, with* $\mathcal{R}_{StepWidth}$ *being the maximum step size of the robot* $\mathcal{R}$

This connectivity gives rise to a graph structure $G_E$, which contains walkable surface nodes, and an edge between two connected surfaces.

To approximate the free space, we stack cuboids on top of each walkable surface. A cuboid of height $\delta$ and with distance $\Delta_L$ to $S_i^p$ is defined as $B_i^p(\Delta_L, \Delta_L + \delta)$ See Fig. 4.3. The stack of cuboids on $S_i^p$ will be denoted by

$$\begin{aligned} B_i^p &= \{B_{i,k}^p\}_{k=1}^{\beta} \\ B_{i,k}^p &= B_i^p(k\delta, (k+1)\delta)\} \end{aligned} \tag{4.6}$$

$\beta$ is choosen such that $\beta > \frac{\mathcal{R}_{\mathrm{H_U}}}{\delta}$ with $\mathcal{R}_{\mathrm{H_U}}$ the maximum height of the robot. For each $B_{i,k}^p$, we apply a clipping algorithm [95] to decompose it into smaller convex cuboids.

Additionally we define the intersection element between two stacks of cuboids $B_i$, $B_j$ as $I_{ij} = B_i \cap B_j$.

Now, given two configurations $q_I, q_G \in \mathcal{C}$ of the robot, we compute the right foot position as $x_I = T(q_I), x_G = T(q_G)$ by using a forward kinematics operator $T$. Given $x_I, x_G$, we define $S_I = \underset{S_k \in S}{\operatorname{argmin}}\ d(x_I, S_k)$ to be the initial surface, and

$S_G = \underset{S_k \in S}{\text{argmin}}\ d(x_G, S_k)$ to be the goal surface.

Given $S_I, S_G$, we compute $\mathcal{H} = \{H_1, \cdots, H_R\}$, the set of $R$ simple connected paths on the environment graph $G_E$. We call $H \in \mathcal{H}$ a homotopy, and we will write the connection of walkable surfaces as $H : S_0 \to \cdots \to S_{R_H}$. As a note, the complexity of finding all connected paths in a graph with $V$ vertices is $\mathcal{O}(|V|!)$ [96].

To summarize, in this section we preprocessed the environment $\mathbf{E}$, to decompose it into

- A set of $N_w$ walkable surfaces $S_1, \cdots, S_{N_w}$
- A set of $N_w$ stack of cuboids $B_1, \cdots, B_{N_w}$
- A set of $N_i$ connector elements $I_1, \cdots, I_{N_i}$
- The environment graph $G_E$, describing the connectivity between walkable surfaces
- A set of homotopies $\mathcal{H}$ for given $x_I, x_G$

## 4.5 Convex optimization of Footpath Homotopies

Given $H : S_0 \to \cdots \to S_{R_H}$, our goal is to find a sliding footstep trajectory supported on the surfaces $S_0, \cdots, S_{R_H}$. More formally, we will consider the functional space of space curves as

$$\Omega(x_I, x_G, H) = C^1([0,1], \mathbb{R}^3) \tag{4.7}$$

under the constraints that for all $\tau \in \Omega(x_I, x_G, H)$ we have $\tau(0) = x_I$, $\tau(1) = x_G$, and that a segment $\tau(t)$ for $t \in [t_i, t_{i+1}]$ has support on a walkable surface $S_i$, as depicted in Fig. 4.4. In between support, we assume that the function is not

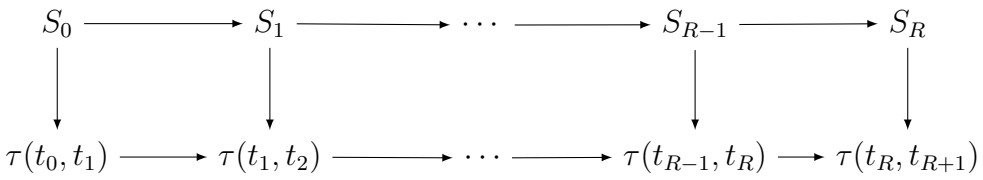



**Figure 4.4:** A function $\tau$ has support on a walkable surface $S_i$ at the time $[t_i, t_{i+1}]$.

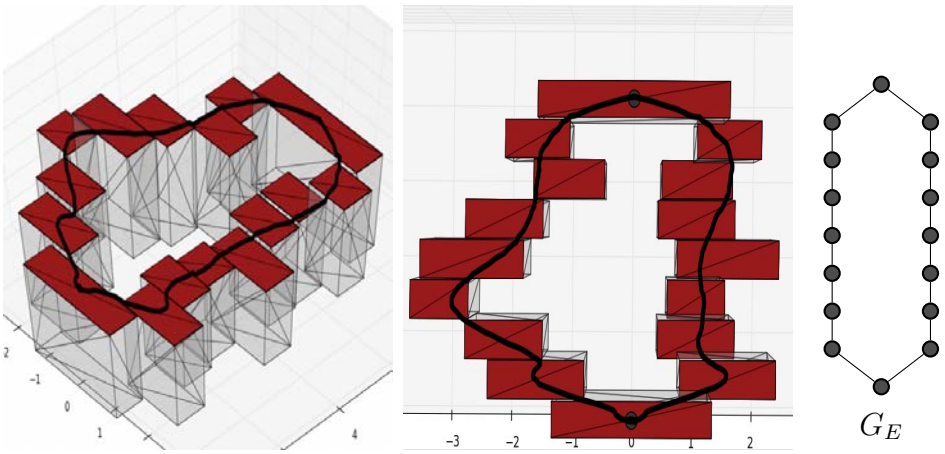



**Figure 4.5:** Two homotopy classes of footsteps, and two solutions, obtained by solving one convex optimization program in each homotopy class. Also we show the environment graph $G_E$ for this particular example, which represents connectivity between walkable surfaces.

supported, i.e. the foot can freely move through space, under the restriction that the non-support movement is smaller than the maximum stepsize.

One way to represent a function from the functional space $\Omega(x_I, x_G, H)$ is by a linear combination of basis functions [97]. We assume here that all functions are of polynomial form, i.e. a function $\tau \in \Omega(x_I, x_G, H)$ is represented at instance $t$ by a polynomial $\tau(t) = \sum_{i=0}^{K} w_i t^i$. We will make the assumption that higher-order terms are negligible such that we choose a finite $K \gg 0$, and use $p(t) = \sum_{i=0}^{K-1} w_i t^i$. We will denote $F = \{x^0, \cdots, x^{K-1}\} \in \mathbb{R}^{K \times D}$, with $K$ basis functions, and $D$ the discretization of $[0, 1]$. For all $t \in [0, 1]$ we denote the approximation by $\tau = W^T F(t)$, with $W \in \mathbb{R}^{K \times 3}$. The resulting optimization problem is convex in the parameters $W$ [63], and we can apply the constraints that every $\tau$ lies

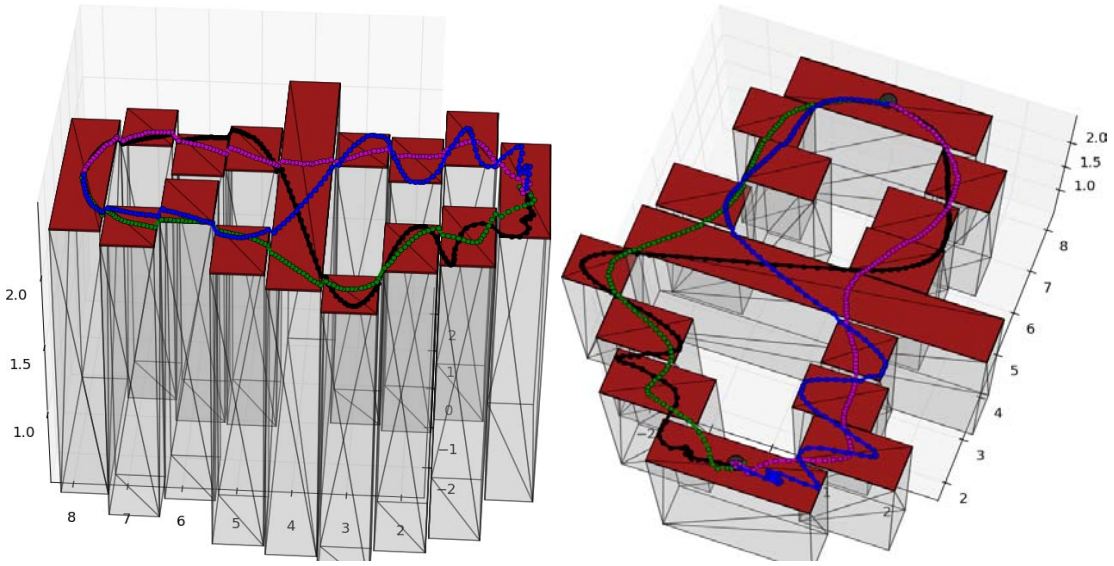



**Figure 4.6:** Four homotopy classes in the environment: Our algorithm finds the homotopy classes via graph search, and then solves one convex optimization problem in each class to find the global optimal solution trajectory.

in the homotopy class $H$ as convex inequality constraints. The resulting convex optimization problem in homotopy class $H$ becomes

$$\underset{\tau \in \Omega(x_I, x_G, H)}{\text{minimize}} \quad c(\tau) \tag{4.8}$$

$$\text{subject to} \quad \tau(0) = x_I, \tau(1) = x_G \tag{4.9}$$

$$\forall S_i \in \{S_0, \cdots, S_R\}, t \in [t_i, t_{i+1}] : \quad A_i \tau(t) \leq b_i \tag{4.10}$$

$$\forall t \in [0,1] : \quad \|\tau(t) - \tau(t + \Delta t)\| \leq \mathcal{R}_{\text{StepWidth}} \tag{4.11}$$

$$\forall t \in [0,1] : \quad \|\tau(t) - W^T F(t)\| \leq \epsilon \tag{4.12}$$

whereby we have the following parameters

**Table 4.1:** Results for planning paths in the two scenarios shown in Fig. 4.5 and Fig. 4.6. $R$ is the number of homotopies, $T_W$ is the time to extract walkable surfaces from the environment, $T_G$ the time to compute the connectivity between surfaces, $T_P$ the planning time of solving $R$ convex optimization problems, and $T$ is the accumulated time of all stages together (averaged over 10 runs, rounded).

| Environment | $R$ Homotopies | $T_W$(s) | $T_G$(s) | $T_P$(s) | $T$(s) |
|---|---|---|---|---|---|
| Stepping 1 (Fig. 4.5) | 2 | 0.27 | 1.92 | 6.53 | 8.73 |
| Stepping 2 (Fig. 4.6) | 4 | 0.26 | 1.68 | 20.34 | 22.28 |

- $\mathcal{R}_{\text{StepWidth}}$ the maximum step size of the robot
- $\epsilon > 0$ approximation constant to circumvent numerical instabilities
- $K$ number of basis functions
- $c(\tau)$ is a convex objective function on $\tau$, for example the shortest path

The given convex problem describes the set of all trajectories restricted to one homotopy class. We can easily add other convex inequality constraints, for example a valid footstep at $t$ can be modeled as another convex inequality constraint:

$$\forall t \in [0,1] : A_i \tau(t) \leq b_i - \mathcal{R}_F \, \mathbf{diag}(A_i^T A_i') \tag{4.13}$$

whereby $A_i = \{a_0, \cdots, a_{M_i}\}$ contains the normals of the polytope associated to the walkable surface $S_i$, $A_i' = \{a_0', \cdots, a_{M_i}'\}$ with $a_j' = a_j - (a_j^T a_p)a_p$, and $\mathcal{R}_F$ being the radius of the foot. Compare to (4.5).

Fig. 4.5 shows the result of our convex optimization problem for an environment with 2 homotopy classes. A more complex version with 4 homotopy classes is shown in Fig. 4.6. The final planning results are depicted in Table 4.1, all generated by using the splitting conic solver (SCS) [98] inside cvxpy [99].

## 4.6 Upper Body Optimization

We have showed so far how to optimize a single footstep trajectory $\tau_0 \in \Omega(x_I, x_G, H)$ in one homotopy class of the environment via a convex optimization problem. Now we assume that $\tau_0$ is fixed. Our goal is to find a set of particle trajectories in the same homotopy class as $\tau_0$, belonging to the swept volume of $\mathcal{R}$, such that those particles are feasible in $\mathbf{E}$. To put if differently, instead of searching for a single configuration space trajectory, we are searching for a set of mutually constrained particle trajectories in the environment. This section describes one possible way to constrain those particle trajectories to lie in the same homotopy class as $\tau_0$. Please consult also Fig. 4.2 for an overview.

Each particle of the swept volume moves along a space curve in $\mathbb{R}^3$. Let $\Upsilon = \{\{\tau_k^l, \tau_k^r\}\}_{k=0}^{\eta}$ be the set of space curves of $2(\eta+1)$ particles, with $\tau_k^{\{l,r\}} \in C^1([0,1], \mathbb{R}^3)$, and $\tau_k^l$ represents the left outer hull of the swept volume of the robot at height $k\delta$, and $\tau_k^r$ the right hull. If we take a cross-section of the swept volume, then $\tau_k$ is represented by a point at height $k\delta$, as depicted by the red dots in Fig. 4.7. To achieve this, we apply three constraints on the functional space $\Upsilon$

1. $\tau_k(t) \in P(t)$, the plane orthogonal to the foot trajectory $\tau_0$ at instance $t$ (cmp. Fig. 4.8)

$$P(t) = \{x \in \mathbb{R}^3 | a_P^T(t)x = b_P(t)\} \tag{4.14}$$

with $a_P(t) = \frac{\tau_0'(t)}{\|\tau_0(t)\|}$ and $b_P(t) = a^T(t)\tau_0(t)$.

2. $\tau_k(t)$ has to be feasible in $\mathbf{E}$, i.e. if $\tau_0$ is supported on $S_i^p$ at $t$, then

$$\tau_k(t) \in B_i^p(k\delta, (k+1)\delta) \tag{4.15}$$

3. At instance $t$, all particles resemble a cross-section $X_k$ of the robot

$$\tau_k \in X_k \tag{4.16}$$

The first two constraints are a linear equality and a convex inequality, respectively. The third constraint however is non-convex. To obtain $X_k$, we sample the configuration space $\mathcal{C}$ and compute cross-sections. A cross-section of a configuration is defined as its swept volume on the plane in movement direction, i.e. at $t$, the volume of $q \in \mathcal{C}$ is projected onto $P(t)$, as depicted in Fig. 4.7. As a simplification, we use only irreducible configurations of the robot [13]. An irreducible configuration is a configuration which has a minimal swept volume. Basically, we sample $\{q_1, \cdots, q_\sigma\} \in \mathcal{C}$, apply a cross-section operator $\phi : \mathcal{C} \to X \times X$ and compute the cross-sections $X = \{\{x_{l,1}, x_{r,1}\}, \cdots, \{x_{l,N}, x_{r,N}\}\}$. $x_l$ stands for the left points of the swept volume, and $x_r$ for the right points and we note that there is a linear transformation $A_L$ such that $x_r = A_L x_l$.

We now have to find a feasible cross-section for each plane. As a simplification, we consider the cross-sections only at intersections $I_r$ of the environment, since those intersections represent the narrow passages. We note that we have only convex boxes in-between intersections, and so we assume that we can linearly interpolate two intersection points.

Our algorithm proceeds in the following manner: we compute the feasibility of $N$ cross-sections by solving $N$ convex optimization problems $\Theta_1^i, \cdots, \Theta_N^i$ for all intersections $i \in [1, V]$ in $I_1, \cdots, I_V$. A feasible path is then a sequence $\lambda_1, \cdots, \lambda_V$ of feasible cross-sections $\Theta_{\lambda_1}^1, \cdots, \Theta_{\lambda_V}^V$ with $\Theta_{\lambda_j}^i < \infty$. We can represent this as a solution matrix

$$\Lambda = \begin{bmatrix} \Theta_1^1 & \cdots & \Theta_1^V \\ \vdots & \ddots & \vdots \\ \Theta_N^1 & \cdots & \Theta_N^V \end{bmatrix} \tag{4.17}$$

whereby we have that $\Theta_j^i$ solves the problem of feasibility of a cross-section $X_j$ on an intersection element $I_i$. Let $t_i$ be such that $\tau(t_i) \in I_i$. Then $\Theta_j^i$ becomes

$$\Theta_j^i = \underset{\{\tau_0,\cdots,\tau_\eta\}\in\Upsilon}{\text{minimize}} \quad c(\tau_0, \cdots, \tau_\eta) \tag{4.18}$$

$$\text{subject to} \quad \forall k \in [0, \eta] :$$

$$\tau_k(t_i), A_L\tau_k(t_i) \in P(t_r) \tag{4.19}$$

$$\tau_k(t_i), A_L\tau_k(t_i) \in I_i(k\delta, (k+1)\delta) \tag{4.20}$$

$$\tau_k(t_i), A_L\tau_k(t_i) = X_j \tag{4.21}$$

whereby $c(\tau_0, \cdots, \tau_\eta)$ is an arbitrary convex cost function.

Fig. 4.8 represents the complete algorithmic output: from an environment, we compute walkable surfaces, we compute a footstep trajectory, we compute planes orthogonal to the footstep, we solve a set of convex optimization problems at each intersection, and we compute a final set of particle trajectories in the environment.

Finally, our main point here is that we have investigated the structure of the planning problem. Given our assumptions, we can compute the number of local minima of our planning problem as

$$L = \sum_{i=1}^{R} N^{V_i} \tag{4.22}$$

with $R$ the number of homotopy classes, $N$ the number of cross-sections, and $V_i$ the number of intersections inside the $i$-th homotopy class. For the wall envi-

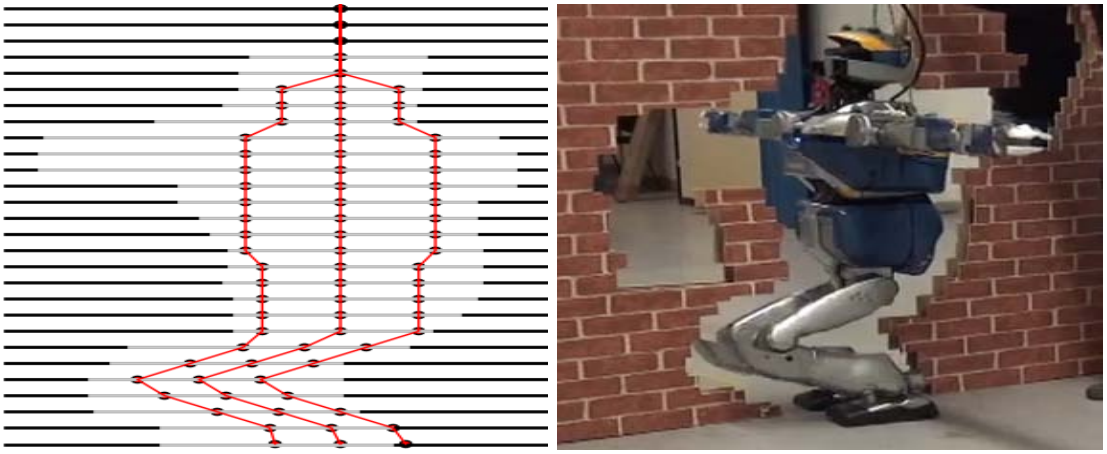


**Figure 4.7: Left**: The cross-section space for a humanoid robot. Each line represents a certain height above a walkable surface. The cross-section of the robot intersects each height at two points, which we call $x_L$ and $x_R$ for left and right, respectively. We assume that every cross-section gives rise to only two points, i.e. we ignore configurations, where this is not the case. Overlaid (white line segments) are the constraints by the wall environment, which impose a convex box inequality on the cross-sections.**Right**: the final experiment, with the humanoid robot HRP-2 walking through a narrow environment.

ronment in Fig. 4.8 we have $N = 144$,$V = 2$,$R = 1$ and so $L = 20736$.

While our work is preliminary and non-complete, we want to stress the fact that knowing the number of local minima is important for understanding the inherent complexity of motion planning.

## 4.7 Experiments

We implemented the algorithms in python, and used cvxpy [99] to compute solutions to the local minima. The source code to reproduced the experiments is available at

`https://github.com/orthez/mpp-path-planner`

For experimental verification, we have chosen the wall environment depicted in Fig. 4.8, which contains 145 objects.

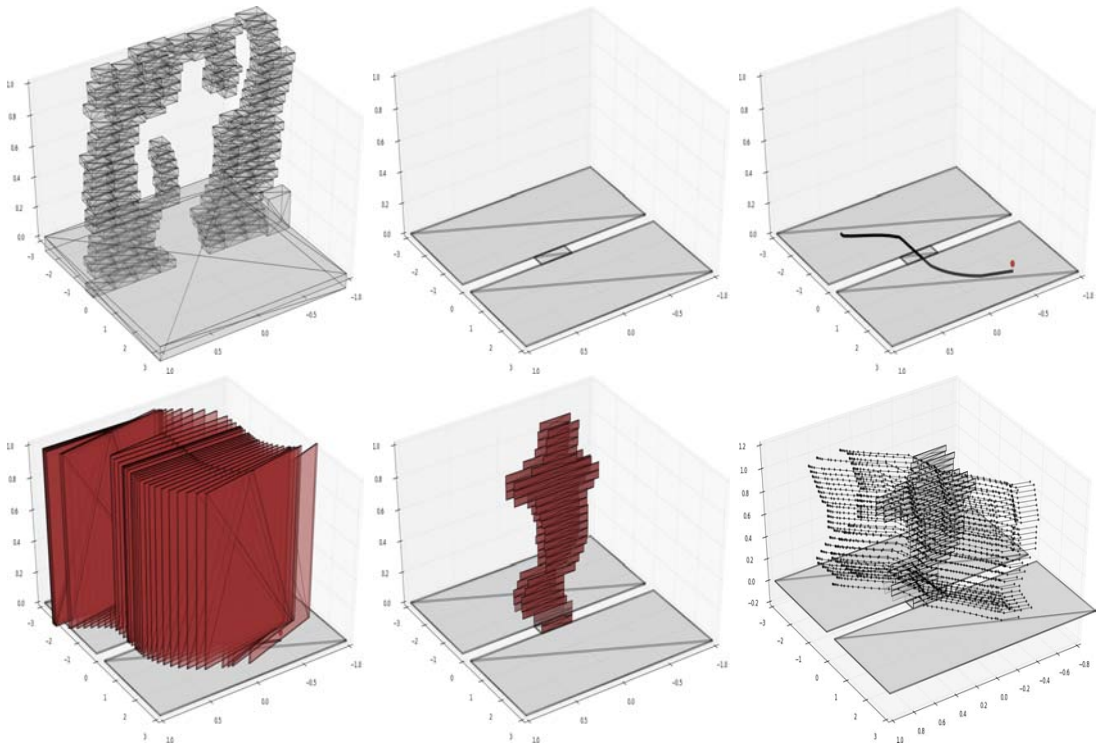
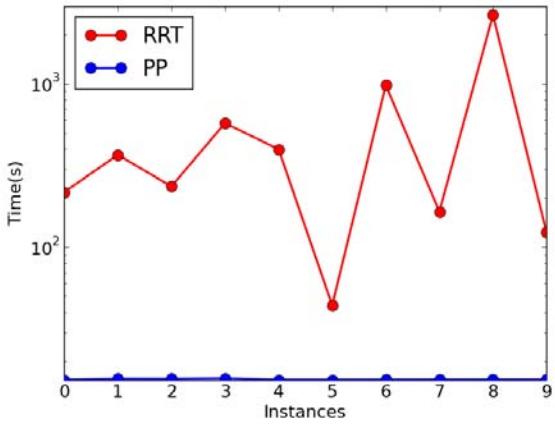


**Figure 4.8:** From Left to Right, Top to Bottom: 1) all polytopes in the wall environment 2) the extracted walkable surfaces from our algorithm, 3) the footstep trajectory in the single homotopy class, computed by solving the convex optimization problem (4.8), 4) the vertical planes along the footstep trajectory, 5) the intersection of boxes on each walkable surface to create the connection elements 6) the final plan of workspace trajectories, all homotopically equivalent and representing a configuration space trajectory.




| Planning Instance | RRT [13] | PP |
|---|---|---|
| 0 | 3h37m | 15m41s |
| 1 | 6h07m | 16m03s |
| 2 | 3h55m | 16m01s |
| 3 | 9h34m | 16m12s |
| 4 | 6h37m | 15m41s |
| 5 | 0h44m | 15m41s |
| 6 | 16h19m | 15m42s |
| 7 | 2h45m | 15m43s |
| 8 | 43h57m | 15m45s |
| 9 | 2h03m | 15m16s |

**Figure 4.9:** Comparison of running time on 10 instances for RRT (red) (adapted from [13]) and for our particle planning (PP) algorithm (blue). For PP, the time depends on the discretization

The following parameters were used: $\beta = 40$, $\delta = 0.05$, such that $\beta\delta = 2.0 > \mathcal{R}_H$ with $\mathcal{R}_H = 1.539m$ the maximum height of HRP-2 [1]. For Problem (4.8), we used $\epsilon = 0.02$, $D = 1000$, $K = 2000$, and we used a minimum number of $D_i = 15$ samples for each walkable surface $S_i$.

The environment decomposition took $3m20s$, and our algorithm computed $N = 144$ cross-section configurations. We employ a greedy version of our algorithm, which computes all local minima for the first intersection, and then checks if the next intersection can be solved by the solution to the first intersection. For each intersection, we computed all minima and found out that 11/144 have been feasible (7.64%). The computation took $12m15s$ (averaged over 10 runs). All together we have a total computation time of $15m35s$. We compared the results of all runs with the results of a rapidly exploring random tree [22], operating on the irreducible configuration space [13]. The results in Fig. 4.9 show that we have a lower variance while performing better at the given sampling resolution.

To move the robot in the real world, we add small footsteps along the path, one every $0.1m$. Given footsteps and trajectory of the upper body, we use a dynamical solver to compute zero-moment point trajectories for the robot. We have used

those results to verify the motion in the dynamical simulator OpenHRP [100], and executed it on the humanoid robot HRP-2 [1]. The video can be found here

`http://homepages.laas.fr/aorthey/videos/wall-homotopy.mp4`

## 4.8 Conclusion

We decomposed the general motion planning problem into a set of homotopic motion planning problems, with the goal of developing more efficient algorithms for the homotopic motion planning problem.

We presented three results: I) how to identify homotopy classes in an environment, based on walkable surfaces, surfaces on which a robot can make a foot contact. II) how to find a single contact trajectory inside a given homotopy class, formulated as a single convex optimization problem, and III) how to find a feasible upper body trajectory by solving a set of convex optimization problems.

Regarding future work, we currently work on incorporating our particle planning into a local motion planning algorithm to produce a dynamical feasible motion. We also would like to investigate when a surface is walkable, depending on physical properties like density, geometry, maximum pressure, and slippage. Finally, we would like to investigate the complexity properties of homotopic particle motion planning.

## 4.9 $H$ space to $Q$ space

Fig. 4.10 depicts the geometry of computing the joint angles $q_3, q_4$, given $h_2$ and the foot direction. Elementary geometry provides us with the following values

($\mathbf{R}_\mathrm{Y}$ depicts a rotation around the $y$ axis in counter-clock wise manner):

$$
\begin{aligned}
l_1 &= h_1 - d_k \\
l_2 &= h_3 - d_2 - h_1 \\
d_5 &= \sqrt{(h_2^2 + l_2^2)} \\
l_3 &= \frac{d_5^2 - d_4^2 + d_3^2}{2d_5} \\
a &= \frac{1}{2l_1}\sqrt{4l_1^2 d_0^2 - (l_1^2 - d_1^2 + d_0^2)^2} \\
b &= \frac{1}{2d_5}\sqrt{4d_5^2 d_3^2 - (d_5^2 - d_4^2 + d_3^2)^2} \\
l_0 &= \sqrt{d_0^2 - a^2} \\
p_1 &= d_k \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^T \\
p_3 &= h_1 \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^T \\
p_4 &= (h_1 + d_2) \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^T \\
p_6 &= h_3 \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^T + h_2 \begin{pmatrix} 1 & 0 & 0 \end{pmatrix}^T \\
\mathbf{v}_k &= d_k \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^T \\
\mathbf{v}_0 &= l_0 \frac{p_3 - p_1}{\|p_3 - p_1\|} + a \frac{\mathbf{R}_{\mathrm{Y}}(\frac{\pi}{2})(p_3 - p_1)}{\|p_3 - p_1\|} \\
\mathbf{v}_0' &= l_0 \frac{p_3 - p_1}{\|p_3 - p_1\|} + a \frac{\mathbf{R}_{\mathrm{Y}}(-\frac{\pi}{2})(p_3 - p_1)}{\|p_3 - p_1\|} \\
\mathbf{v}_1 &= p_3 - p_1 - \mathbf{v}_0 \\
\mathbf{v}_1' &= p_3 - p_1 - \mathbf{v}_0' \\
\mathbf{v}_2 &= d_2 \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^T \\
\mathbf{v}_3 &= l_3 \frac{p_6 - p_4}{\|p_6 - p_4\|} + b \frac{\mathbf{R}_{\mathrm{Y}}(\frac{\pi}{2})(p_6 - p_4)}{\|p_6 - p_4\|} \\
\mathbf{v}_3' &= l_3 \frac{p_6 - p_4}{\|p_6 - p_4\|} + b \frac{\mathbf{R}_{\mathrm{Y}}(-\frac{\pi}{2})(p_6 - p_4)}{\|p_6 - p_4\|} \\
\mathbf{v}_4 &= p_6 - p_4 - \mathbf{v}_3 \\
\mathbf{v}_4' &= p_6 - p_4 - \mathbf{v}_3'
\end{aligned}
\tag{4.23}
$$

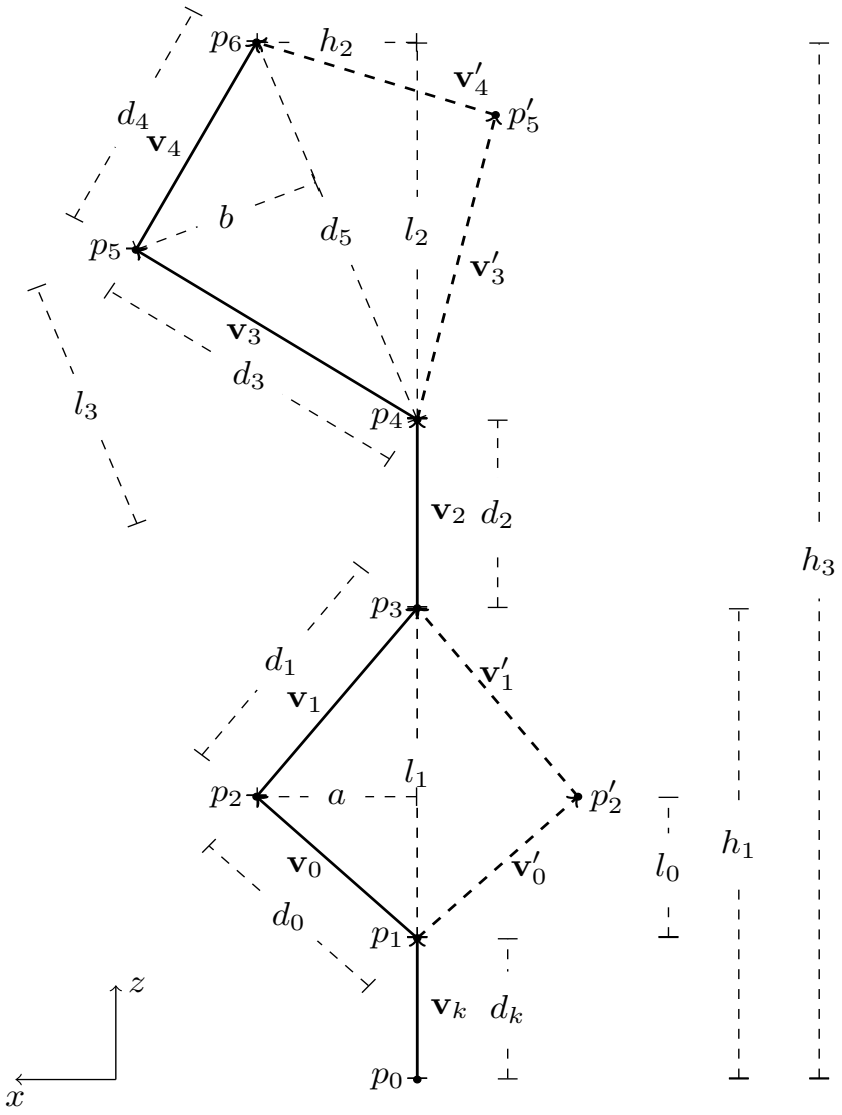



**Figure 4.10:** Sideview of the cross section skeleton of a humanoid robot. $d_k$ is the ankle length, $d_0$ the lower limb length, $d_1$ the thigh length, $d_2$ the hip length, $d_3$ the chest length, and $d_4$ the head length. Also, we have $p_0$ the sole, $p_1$ the ankle, $p_2$ the knee, $p_3$ the hip, $p_4$ the waist, $p_5$ the neck and $p_6$ the head. Under the assumption that the hip $p_3$ and the waist $p_4$ are always on one line with $p_0$ and $p_1$, we can parametrize the configurations by four parameters $K, h_1, h_2, h_3$. $K$ is the arrangement, meaning for $K = 0$ we have an active link path $p_0p_1p_2p_3p_4p_5p_6$, for $K = 1$ we have $p_0p_1p_2p_3p_4p'_5p_6$, for $K = 2$ we have $p_0p_1p'_2p_3p_4p_5p_6$, and for $K = 3$ we have $p_0p_1p'_2p_3p_4p'_5p_6$.

# Chapter 5

# Conclusion

*"Go Adam, go Eve. The world is yours."*

—Karel Čapek, R.U.R.

*"Exploitation of Structure is an essential component for Motion Planning"*

We have investigated the hypothesis that motion planning can be simplified by exploiting structure. To show that this hypothesis is correct, we constructed three motion planning algorithms, which exploit the desired behavior of the robot, the mechanical system and the environment, respectively. As a result, we found that yes, motion planning can be simplified by exploiting structure. In particular, we found that knowing the behavior is useful to precompute the expected swept volume, knowing the mechanical system is useful to find irreducible swept volume movements, and knowing the environment is useful to precompute high-level minima.

Since we established that exploiting structure is viable for motion planning, we like to point the interested reader into some future directions. In particular, we have identified four main directions, which we think fruitful towards the long-term

goal of developing efficient motion planning algorithms for humanoid robots.

First, we discussed in Ch. 4, that a complete understanding of motion planning would require us to know the number of local minima. Knowing the number of local minima can be used to reason about the problem on an abstract level and it can be used to come up with efficient locally convex optimization programs. We demonstrated that in principle this is possible in a simplified setting, and future research should address the problem of finding the number of local minima in more general settings. Especially interesting in this regard is the connection between the workspace topology and the configuration space topology in the sense of Farber [51].

Second, we observe that in many real-life situations the topology structure of an environment is not stable, but it is changing in a continuous manner. As an example, consider a robot trying to cross a street with cars, as depicted in Fig. 5.1. The homotopy classes of movements in the environment are not static but are evolving dynamically, some homotopy classes are created, some homotopy classes are closed over time. One extension of topological motion planning is to investigate *persistency* of homotopy classes, i.e. for safe and stable motion planning, we would like the robot to choose a maximal persistent homotopy class in the environment [101]. In particular, we would like to choose a persistent homotopy class, by leveraging the mathematical area of persistent homology [102]. Persistent homology has already been successfully applied to topological data analysis [103][104], and would fit perfectly for choosing the maximum persistent homotopy class in a principled way.

Third, we would like to extend our concepts to multiple contacts. Multiple contacts are important for stabilization of the robot, especially in environments where the robot can hardly move without hand contacts, for example in rock climbing, balancing over a beam, or unstable, rocky environments. Pioneer work

has been conducted in this area by Hauser et al. [14], and Escande [44]. We would like to extend our particle planning framework from Ch. 4 to include also hand contacts, and analyze the environment based on possible hand contact surface structures. Also, the connection between contact planning and irreducible motion planning is still an interesting open research question to be investigated.

Fourth, we believe that truly efficient motion planning algorithm need to be able to abstract information and reason on higher abstract levels. Higher abstraction levels are studied in the literature under the names of hierarchical motion planning [105],[106],[107] and task or symbolic motion planning [108],[109], [110]. However, there is not yet a framework which integrates both symbolic reasoning and motion planning. First works in this direction have been undertaken in optimizing grasp planning [111]. We believe that this integration of different abstraction levels is an essential step to exploit structural components. We like to point out that any abstraction has to provide us with a guarantee of success, meaning any abstraction has to preserve completeness of motions in some way.

Let us conclude this thesis by reminding you that motion planning is an NP-hard problem. While this implies that general-purpose algorithms are not efficient, we have shown that humanoid motion planning exhibits nevertheless a rich exploitable structure. We showed that exploiting structural components can be beneficial towards algorithmic development. In the future, we hope that more structural components are discovered and investigated, to allow us to efficiently solve the motion planning problem for humanoid robots. A general solution would not only enable autonomous capabilities like space exploration, food harvesting, or construction work, but could also bring solutions to seemingly disconnected fields like molecular modeling, protein folding, or architectural design. I hope that by now I have been able to convince you that exploiting structure is viable, and that this thesis has made a small but purposeful contribution towards the general goal

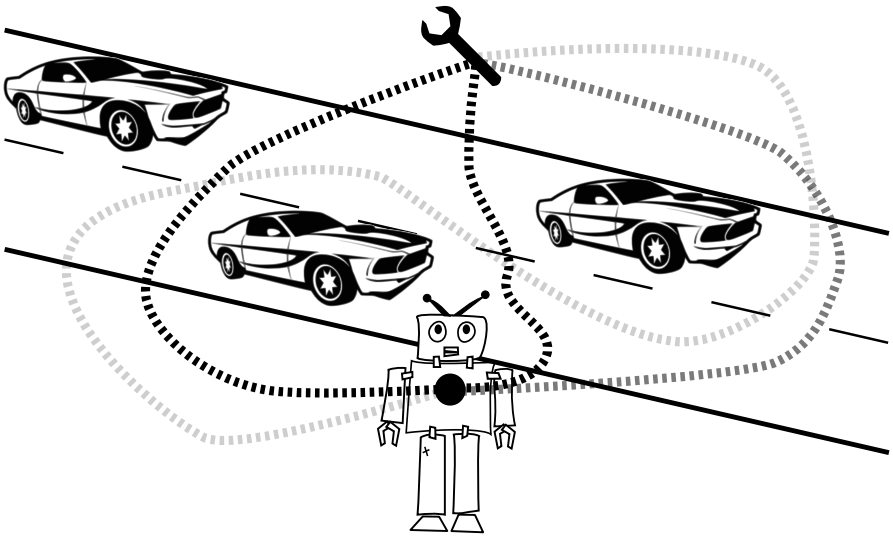


**Figure 5.1:** A complicated real-life scenario: the robot needs to cross a street, where the movement of cars opens and closes homotopy classes in relative short times. The robot needs to be able to predict the movement of the cars, such that it can understand the topological changes necessary to decompose the motion planning problem.

of autonomously acting machines.

# Appendix A

# Proofs

*Proof of Theorem 1.* Let $s = \mathcal{SV}(\tau)$ and $s' = \mathcal{SV}(\tau')$. $s$ is feasible if $s \cap \mathbf{E} = \emptyset$. We proceed by direct proof:
(1) Let $s$ be infeasible, then $\exists v \in s$, such that $v \cap \mathbf{E} = v$. Since $s \subset s'$, we have that $v \in s'$. Since $v$ exists, we can conclude that at least $s' \cap \mathcal{E} > v$, which makes $s'$ infeasible.
(2) $\tau'$ being feasible means $s' \cap \mathbf{E} = \emptyset$. Since $s \subset s'$, it follows from elementary set theory that $s \cap \mathbf{E} = \emptyset$, which proofs that $\tau$ is feasible. □

*Proof of Lemma 1.* Let $\mu$ be the lebesque measure on the workspace $\mathcal{W}$. First, let us see that if $\mathcal{SV}(\tau_1) \subset \mathcal{SV}(\tau_0)$, then $\mu(\mathcal{SV}(\tau_1)) < \mu(\mathcal{SV}(\tau_0))$.

Now, by definition, if $\tau_0 \notin \mathbf{I}$, then $\exists \tau_1 \in \mathcal{F}$, such that $\mathcal{SV}(\tau_1) \subset \mathcal{SV}(\tau_0)$. Then either $\tau_1 \in \mathbf{I}$, and we are done. Or $\tau_1 \notin \mathbf{I}$, and by definition, $\exists \tau_2 \in \mathcal{F}$, such that $\mathcal{SV}(\tau_2) \subset \mathcal{SV}(\tau_1)$. Let us assume that there is no trajectory $\tau_i \in \mathbf{I}$, such that we obtain an infinite sequence $\Pi = \{\tau_0, \tau_1, \tau_2, \cdots\}$ of reducible trajectories $\tau_i \in \mathcal{F}$, such that $\forall \tau_i \in \Pi : \mathcal{SV}(\tau_{i+1}) \subset \mathcal{SV}(\tau_i)$. Since we have $\forall \tau_i \in \Pi : \mu(\mathcal{SV}(\tau_{i+1})) < \mu(\mathcal{SV}(\tau_i))$ and $\mu(\mathcal{SV}(\tau)) > 0$, the sequence is strictly monotonically decreasing and bounded, and will therefore converge to its maximum lower bound, which we call $C$, i.e. $\lim_{n\to\infty} \mu(\mathcal{SV}(\tau_i)) = C$. Consequently, since the maximum lower bound is

obtained, there cannot exists another trajectory $\tau'$, such that $\mu(\mathcal{SV}(\tau')) < C$. By definition, the sequence is converged in $\mathbf{I}$, and therefore we conclude that every element $\tau \in \mathcal{F} \setminus \mathbf{I}$ is reducible by $\tau' \in \mathbf{I}$.

□

*Proof of Theorem 2.* Let us assume that $\exists \tau \in \mathcal{F}$, with $\tau$ being feasible, and that $\forall \tau' \in \mathbf{I} : \tau'$ is not feasible. Since $\tau$ is feasible, it follows that $\tau \notin \mathbf{I}$. Then by definition there has to be a $\tau'' \in \mathcal{F}$ such that $\mathcal{SV}(\tau'') \subset \mathcal{SV}(\tau)$. Then $\tau''$ is feasible by Theorem 1. Further, either we have that $\tau'' \in \mathbf{I}$. Then we have a contradiction. Or we have $\tau'' \notin \mathbf{I}$, which means that we can still find another $\tau''' \in \mathcal{F}$ reducing $\tau''$. By Lemma 1, we know that such a sequence can be reduced by a $\tilde{\tau} \in \mathbf{I}$ . So we reach a contradiction, too.

□

*Proof of Corollary 1.* By definition, motion planning is complete, if we can find a solution (a trajectory), if one exist. By Theorem 2, we know that if we cannot find a solution in $\mathbf{I}$, then there is no solution in $\mathcal{F}$. Conversely, if there is a solution in $\mathcal{F}$, then by Theorem 1, there exists a solution in $\mathbf{I}$. □

# Bibliography

[1] K. Kaneko, F. Kanehiro, S. Kajita, H. Hirukawa, T. Kawasaki, M. Hirata, K. Akachi, and T. Isozumi, "Humanoid robot HRP-2," in *International Conference on Robotics and Automation*, 2004, pp. 1083–1090.

[2] R. Tellez, F. Ferro, S. Garcia, E. Gomez, E. Jorge, D. Mora, D. Pinyol, J. Oliver, O. Torres, J. Velazquez *et al.*, "Reem-b: An autonomous lightweight human-size humanoid robot," in *International Conference on Humanoid Robots*, 2008.

[3] K. Kaneko, F. Kanehiro, M. Morisawa, K. Miura, S. Nakaoka, and S. Kajita, "Cybernetic human hrp-4c," in *International Conference on Humanoid Robots*, 2009.

[4] A. Parmiggiani, M. Maggiali, L. Natale, F. Nori, A. Schmitz, N. Tsagarakis, J. S. Victor, F. Becchi, G. Sandini, and G. Metta, "The design of the icub humanoid robot," 2012.

[5] T. Asfour, J. Schill, H. Peters, C. Klas, J. Bucker, C. Sander, S. Schulz, A. Kargov, T. Werner, and V. Bartenbach, "Armar-4: A 63 dof torque controlled humanoid robot," in *International Conference on Humanoid Robots*, 2013.

[6] Y. Sakagami, R. Watanabe, C. Aoyama, S. Matsunaga, N. Higaki, and K. Fujimura, "The intelligent asimo: System overview and integration," in *International Conference on Intelligent Robots and Systems*, vol. 3, 2002.

[7] R. Deits and R. Tedrake, "Footstep Planning on Uneven Terrain with Mixed-Integer Convex Optimization," in *International Conference on Humanoid Robots*, 2014.

[8] T. Koolen, T. De Boer, J. Rebula, A. Goswami, and J. Pratt, "Capturability-based analysis and control of legged locomotion, part 1: Theory and application to three simple gait models," *International Journal of Robotics Research*, vol. 31, no. 9, 2012.

[9] Y. Zhang, J. Luo, K. Hauser, H. A. Park, M. Paldhe, C. Lee, R. Ellenberg, B. Killen, P. Oh, J. H. Oh *et al.*, "Motion planning and control of ladder climbing on DRC-Hubo for DARPA Robotics Challenge," in *International Conference on Robotics and Automation*, 2014.

[10] S. Feng, X. Xinjilefu, W. Huang, and C. G. Atkeson, "3d walking based on online optimization," in *Humanoid Robots (Humanoids), 2013 13th IEEE-RAS International Conference on*. IEEE, 2013, pp. 21–27.

[11] S. M. LaValle, *Planning Algorithms*. Cambridge University Press, 2006.

[12] K. Harada, S. Hattori, H. Kurokawa, M. Morisawa, S. Kajita, and E. Yoshida, "Two-stage time-parametrized gait planning for humanoid robots," *IEEE/ASME Trans. on Mechatronics*, vol. 15, no. 5, pp. 694–703, 2010.

[13] A. Orthey, F. Lamiraux, and O. Stasse, "Motion Planning and Irreducible Trajectories," in *International Conference on Robotics and Automation*, 2015.

[14] K. Hauser, T. Bretl, K. Harada, and J.-C. Latombe, "Using motion primitives in probabilistic sample-based planning for humanoid robots," in *Algorithmic foundation of robotics VII*. Springer, 2008, pp. 507–522.

[15] A. Orthey, O. Roussel, O. Stasse, and M. Taix, "Irreducible Motion Planning by Exploiting Linear Linkage Structures," in *Transactions on Robotics*, 2015 (Submitted to).

[16] A. Orthey, V. Ivan, M. Naveau, Y. Yang, O. Stasse, and S. Vijayakumar, "Homotopic Particle Motion Planning," in *International Conference on Humanoid Robots*, 2015 (Submitted to).

[17] O. Stasse, A. Orthey, F. Morsillo, M. Geisert, N. Mansard, M. Naveau, and C. Vassallo, "Airbus/Future of Aircraft Factory, HRP-2 as Universal Worker Proof of Concept," in *International Conference on Humanoid Robots*, 2014, p. Video Session. [Online]. Available: https://www.youtube.com/watch?v=iFV-13XlJvI

[18] A. Orthey and O. Stasse, "Towards Reactive Whole-Body Motion Planning in Cluttered Environments by Precomputing Feasible Motion Spaces ," in *International Conference on Humanoid Robots (Humanoids)*, Atlanta, GA, USA, 2013.

[19] T. Lozano-Pérez, "Spatial planning: A configuration space approach," *IEEE Trans. Computers*, vol. 32, no. 2, pp. 108–120, 1983.

[20] J. H. Reif, "Complexity of the mover's problem and generalizations," in *Conference on Foundations of Computer Science*, 1979, pp. 421–427.

[21] J. F. Canny, *The complexity of robot motion planning*. MIT press, 1988.

[22] S. M. Lavalle and J. J. Kuffner Jr, "Rapidly-Exploring Random Trees: Progress and Prospects," in *Algorithmic and Computational Robotics: New Directions*, 2000.

[23] A. El Khoury, F. Lamiraux, and M. Taix, "Optimal Motion Planning for Humanoid Robots," in *International Conference on Robotics and Automation*, 2013.

[24] J. Kuffner, K. Nishiwaki, S. Kagami, M. Inaba, and H. Inoue, "Motion planning for humanoid robots," in *International Symposium of Robotics Research*, 2005.

[25] I. Al-Bluwi, T. Siméon, and J. Cortés, "Motion planning algorithms for molecular simulations: A survey," *Computer Science Review*, vol. 6, no. 4, pp. 125–143, 2012.

[26] K. A. Dill and J. L. MacCallum, "The protein-folding problem, 50 years on," *Science*, vol. 338, no. 6110, pp. 1042–1046, 2012.

[27] D. Devaurs, T. Siméon, and J. Cortés, "Efficient sampling-based approaches to optimal path planning in complex cost spaces," in *Algorithmic Foundations of Robotics XI*. Springer International Publishing, 2015, pp. 143–159.

[28] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *International Journal of Robotics Research*, vol. 30, no. 7, 2011.

[29] D. Berenson, S. S. Srinivasa, D. Ferguson, and J. J. Kuffner, "Manipulation planning on constraint manifolds," in *International Conference on Robotics and Automation*. IEEE, 2009.

[30] L. Jaillet and J. M. Porta, "Path planning under kinematic constraints by rapidly exploring manifolds," *Transactions on Robotics*, 2013.

[31] M. Elbanhawi and M. Simic, "Sampling-based robot motion planning: A review," *Access, IEEE*, vol. 2, pp. 56–77, 2014.

[32] S. M. LaValle, "Motion planning: The essentials," *IEEE Robotics and Automation Society Magazine*, vol. 18, no. 1, pp. 79–89, 2011.

[33] ——, "Motion planning: Wild frontiers," *IEEE Robotics and Automation Society Magazine*, vol. 18, no. 2, pp. 108–118, 2011.

[34] K. Y. Kensuke Harada, Eiichi Yoshida, Ed., *Motion Planning for Humanoid Robots*. Springer, 2010.

[35] T. Bretl, S. Lall, J.-C. Latombe, and S. Rock, "Multi-step motion planning for free-climbing robots," in *Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2004.

[36] M. Beetz, U. Klank, I. Kresse, A. Maldonado, L. Mosenlechner, D. Pangercic, T. Ruhr, and M. Tenorth, "Robotic roommates making pancakes," in *International Conference on Humanoid Robots*, 2011.

[37] S. Dalibard, A. Nakhaei, F. Lamiraux, and J. Laumond, "Manipulation of documented objects by a walking humanoid robot ," in *International Conference on Humanoid Robots*, 2010.

[38] M. Johnson, B. Shrewsbury, S. Bertrand, T. Wu, D. Duran, M. Floyd, P. Abeles, D. Stephen, N. Mertins, A. Lesman *et al.*, "Team ihmc's lessons learned from the darpa robotics challenge trials," *Journal of Field Robotics*, 2015.

[39] S. Lengagne, N. Ramdani, and P. Fraisse, "Planning and fast replanning safe motions for humanoid robots," *Transactions on Robotics*, vol. 27, no. 6, pp. 1095–1106, 2011.

[40] J. Chestnutt, M. Lau, K. M. Cheung, J. Kuffner, J. K. Hodgins, and T. Kanade, "Footstep planning for the honda asimo humanoid," in *International Conference on Robotics and Automation*, April 2005.

[41] N. Perrin, O. Stasse, L. Baudouin, F. Lamiraux, and E. Yoshida, "Fast humanoid robot collision-free footstep planning using swept volume approximations," *Transactions on Robotics*, vol. 28, no. 2, 2012.

[42] A. Hornung and M. Bennewitz, "Adaptive level-of-detail planning for efficient humanoid navigation," in *International Conference on Robotics and Automation*. IEEE, 2012, pp. 997–1002.

[43] A. Hornung, D. Maier, and M. Bennewitz, "Search-based footstep planning," in *ICRA Workshop on Progress and Open Problems in Motion Planning and Navigation for Humanoids*, Karlsruhe, Germany, 2013.

[44] A. Escande, A. Kheddar, and S. Miossec, "Planning contact points for humanoid robots," *Robotics and Autonomous Systems*, vol. 61, no. 5, pp. 428 – 442, 2013.

[45] T. Yoshikawa, "Analysis and control of robot manipulators with redundancy," in *International Symposium of Robotics Research*. Mit Press Cambridge, MA, 1984.

[46] S. Tonneau, J. Pettré, and F. Multon, "Using task efficient contact configurations to animate creatures in arbitrary environments," *Computers and Graphics*, 2014.

[47] O. Brock and L. E. Kavraki, "Decomposition-based Motion Planning: A Framework for Real-time Motion Planning in High-dimensional Configuration Spaces," in *International Conference on Robotics and Automation*, 2001.

[48] Y. Yang and O. Brock, "Elastic roadmaps - motion generation for autonomous mobile manipulation," *Autonomous Robots*, vol. 28, no. 1, pp. 113–130, 2010.

[49] N. Jetchev and M. Toussaint, "Fast motion planning from experience: trajectory prediction for speeding up movement generation," *Autonomous Robots*, 2013.

[50] S. Bhattacharya, M. Likhachev, and V. Kumar, "Topological constraints in search-based robot path planning," *Autonomous Robots*, vol. 33, no. 3, pp. 273–290, 2012.

[51] M. Farber, "Topological complexity of motion planning," *Discrete and Computational Geometry*, vol. 29, no. 2, pp. 211–221, 2003.

[52] S. Dalibard and J.-P. Laumond, "Linear Dimensionality Reduction in Random Motion Planning," *International Journal of Robotics Research*, 2011. [Online]. Available: https://hal.archives-ouvertes.fr/hal-00486793

[53] M. Ciocarlie, C. Goldfeder, and P. Allen, "Dimensionality reduction for hand-independent dexterous robotic grasping," in *International Conference on Intelligent Robots and Systems*, 2007.

[54] P. Allen, M. Ciocarlie, and C. Goldfeder, "Grasp planning using low dimensional subspaces," in *The Human Hand as an Inspiration for Robot Hand Development*, ser. Springer Tracts in Advanced Robotics, R. Balasubramanian and V. J. Santos, Eds. Springer International Publishing, 2014. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-03017-3_24

[55] A. Mahoney, J. Bross, and D. Johnson, "Deformable robot motion planning in a reduced-dimension configuration space," in *International Conference on Robotics and Automation*, 2010.

[56] B. Hendrickson, "The molecule problem: Exploiting structure in global optimization," *SIAM Journal on Optimization*, vol. 5, no. 4, 1995.

[57] P. Poupart, "Exploiting structure to efficiently solve large scale partially observable markov decision processes," Ph.D. dissertation, Department of Computer Science, University of Toronto, 2005.

[58] M. R. K. Ryan, "Exploiting subgraph structure in multi-robot path planning," 2008.

[59] B. Burns and O. Brock, "Sampling-based motion planning using predictive models," in *International Conference on Robotics and Automation*, 2005.

[60] S. Hutchinson, "Exploiting visual constraints in robot motion planning," in *International Conference on Robotics and Automation*, 1991.

[61] T. Judson, *Abstract algebra: Theory and Applications*. Orthogonal Publishing, 2014.

[62] J. Munkres, *Topology*. Pearson, 2000.

[63] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge university press, 2004.

[64] P. Jiménez, F. Thomas, and C. Torras, "Collision detection algorithms for motion planning," in *Robot Motion Planning and Control*, J.-P. Laumond, Ed. Berlin: Springer-Verlag, 1998, pp. 1–53.

[65] J. Chestnutt, "Navigation and Gait Planning," in *Motion Planning for Humanoid Robots*, K. Harada, E. Yoshida, and K. Yokoi, Eds. Springer London, 2010, pp. 1–28.

[66] Y. K. Hwang and N. Ahuja, "Gross motion planning—a survey," *ACM Computing Surveys (CSUR)*, vol. 24, no. 3, pp. 219–291, 1992.

[67] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal, "Dynamical Movement Primitives: Learning Attractor Models for Motor Behaviors," *Neural computation*, 2013.

[68] M. Toussaint, "Robot trajectory optimization using approximate inference," in *International Conference on Machine Learning*, 2009.

[69] N. Mansard, O. Khatib, and A. Khedar, "A unified approach to integrate unilateral constraints in the stack of tasks," *Transactions on Robotics*, vol. 25, no. 3, June 2009.

[70] A. H. Barr, "Superquadrics and angle-preserving transformations," *IEEE Computer graphics and Applications*, 1981.

[71] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.

[72] R. M. Neal, "Mcmc using hamiltonian dynamics," Handbook of Markov Chain Monte Carlo (editors S. Brooks, A. Gelman, G. Jones, XL Meng). Chapman and Hall/CRC Press, 2010.

[73] E. Larsen, S. Gottschalk, M. C. Lin, and D. Manocha, "Fast proximity queries with swept sphere volumes," Technical Report TR99-018, Department of Computer Science, University of North Carolina, Tech. Rep., 1999.

[74] M. Spivak, *A comprehensive introduction to differential geometry. Vol. I.* Publish or Perish Inc., 1979.

[75] N. Vahrenkamp, D. Berenson, T. Asfour, J. Kuffner, and R. Dillmann, "Humanoid motion planning for dual-arm manipulation and re-grasping tasks," in *International Conference on Intelligent Robots and Systems*, 2009.

[76] S. Hirose and H. Yamada, "Snake-like robots [Tutorial]," *Robotics and Automation Magazine*, 2009.

[77] E. S. Conkur and R. Gurbuz, "Path Planning Algorithm for Snake–Like Robots," *Information Technology And Control*, vol. 37, no. 2, pp. 159–162, 2008.

[78] J. Liu, Y. Wang, B. Ii, and S. Ma, "Path planning of a snake-like robot based on serpenoid curve and genetic algorithms," in *Intelligent Control and Automation*, 2004.

[79] W. Henning, F. Hickman, and H. Choset, "Motion Planning for Serpentine Robots," in *Proceedings of ASCE Space and Robotics*, 1998.

[80] D. Rollinson and H. Choset, "Virtual Chassis for Snake Robots," in *International Conference on Intelligent Robots and Systems*, 2011.

[81] E. Cappo and H. Choset, "Planning end effector trajectories for a serially linked, floating-base robot with changing support polygon," in *American Control Conference*, 2014.

[82] I. Kabul, R. Gayle, and M. C. Lin, "Cable route planning in complex environments using constrained sampling," in *ACM Symposium on Solid and Physical Modeling*. ACM, 2007.

[83] S. Bereg and D. Kirkpatrick, "Curvature-bounded traversals of narrow corridors," in *Symposium on Computational geometry*. ACM, 2005.

[84] H.-K. Ahn, O. Cheong, J. Matoušek, and A. Vigneron, "Reachability by paths of bounded curvature in a convex polygon," *Computational Geometry*, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0925772111000617

[85] S. Guha and S. D. Tran, "Reconstructing curves without delaunay computation," *Algorithmica*, 2005.

[86] S. Dalibard, A. Khoury, F. Lamiraux, M. Taix, and J. Laumond, "Small-Space Controllability of a Walking Humanoid Robot," in *International Conference on Humanoid Robots*, 2011.

[87] P. K. Agarwal, T. Biedl, S. Lazard, S. Robbins, S. Suri, and S. Whitesides, "Curvature-constrained shortest paths in a convex polygon," *SIAM Journal on Computing*, 2002.

[88] F. Lamiraux and J. Mirabel, "HPP: a new software framework for manipulation planning," 2015, submitted to International Conference on Intelligent Robots and Systems. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01138118

[89] L. E. Kavraki, J.-C. Latombe, R. Motwani, and P. Raghavan, "Randomized query processing in robot path planning," in *Symposium on Theory of Computing*. ACM, 1995, pp. 353–362.

[90] M. Zucker, N. Ratliff, A. Dragan, M. Pivtoraiko, M. Klingensmith, C. Dellin, J. A. D. Bagnell, and S. Srinivasa, "CHOMP: Covariant Hamiltonian Optimization for Motion Planning," *International Journal of Robotics Research*, May 2013.

[91] N. Perrin, O. Stasse, F. Lamiraux, and E. Yoshida, "Weakly collision-free paths for continuous humanoid footstep planning," in *International Conference on Intelligent Robots and Systems*, 2011.

[92] Y. Yang and O. Brock, "Efficient Motion Planning Based on Disassembly," in *Robotics: Science and Systems*, Cambridge, USA, June 2005.

[93] V. Ivan, D. Zarubin, M. Toussaint, T. Komura, and S. Vijayakumar, "Topology-based representations for motion planning and generalization in dynamic environments with interactions," *The International Journal of Robotics Research*, vol. 32, no. 9-10, pp. 1151–1163, 2013.

[94] J.-M. Lien and N. M. Amato, "Approximate convex decomposition of polyhedra and its applications," *Computer Aided Geometric Design*, vol. 25, no. 7, pp. 503–522, 2008.

[95] B. R. Vatti, "A Generic Solution to Polygon Clipping," *Communications of ACM*, vol. 35, no. 7, pp. 56–63, Jul. 1992. [Online]. Available: http://doi.acm.org/10.1145/129902.129906

[96] R. Sedgewick, "Part 5: Graph algorithms," in *Algorithms in C*. Addison Wesley Professional, 2001.

[97] W. Rudin, *Functional analysis*, 2nd ed., ser. International Series in Pure and Applied Mathematics. New York: McGraw-Hill Inc., 1991.

[98] B. O'Donoghue, E. Chu, N. Parikh, and S. Boyd, "Operator splitting for conic optimization via homogeneous self-dual embedding," *arXiv preprint arXiv:1312.3039*, 2013.

[99] S. Diamond, E. Chu, and S. Boyd, "CVXPY: A Python-embedded modeling language for convex optimization," http://cvxpy.org/, May 2014.

[100] F. Kanehiro, H. Hirukawa, and S. Kajita, "OpenHRP: Open architecture humanoid robotics platform," *The International Journal of Robotics Research*, vol. 23, no. 2, pp. 155–165, 2004.

[101] P. Lehner, A. Sieverling, and O. Brock, "Incremental, Sensor-Based Motion Generation for Mobile Manipulators in Unknown, Dynamic Environments," in *International Conference on Robotics and Automation*, 2015.

[102] R. Ghrist, *Elementary Applied Topology*. CreateSpace Independent Publishing Platform, 2014.

[103] ——, "Barcodes: the persistent topology of data," *Bulletin of the American Mathematical Society*, vol. 45, no. 1, pp. 61–75, 2008.

[104] H. Edelsbrunner and J. Harer, *Computational topology: an introduction*. American Mathematical Soc., 2010.

[105] D. Zarubin, V. Ivan, M. Toussaint, and S. Vijayakumar, "Hierarchical motion planning in topological representations," in *Robotics: Science and Systems*, 2012.

[106] F. Stulp and S. Schaal, "Hierarchical reinforcement learning with motion primitives," in *International Conference on Humanoid Robots*, 2011.

[107] L. P. Kaelbling and T. Lozano-Pérez, "Hierarchical task and motion planning in the now," in *International Conference on Robotics and Automation*, 2011, pp. 1470–1477.

[108] E. Plaku and G. D. Hager, "Sampling-based motion and symbolic action planning with geometric and differential constraints," in *International Conference on Robotics and Automation*, 2010, pp. 5002–5008.

[109] M. Toussaint, "Logic-geometric programming: An optimization-based approach to combined task and motion planning," in *International Joint Conference on Artificial Intelligence*, 2015.

[110] T. Lozano-Pérez and L. P. Kaelbling, "A constraint-based method for solving sequential manipulation planning problems," in *International Conference on Intelligent Robots and Systems*, 2014.

[111] A. Orthey, M. Toussaint, and N. Jetchev, "Optimizing Motion Primitives to Make Symbolic Models More Predictive," in *International Conference on Robotics and Automation*, Karlsruhe, Germany, 2013.