The Raymond and Beverly Sackler
Faculty of Exact Sciences
The Blavatnik School of Computer Science

Multi-Robot Motion Planning: Theory and Practice

Dissertation submitted in partial fulfillment of the requirements for the degree of "doctor in philosophy" in the School of Computer Science, Tel-Aviv University

by

Kiril Solovey

This work has been carried out under the supervision of
Prof. Dan Halperin

March 2018

Acknowledgments

I wish to express my endless gratitude to my advisor Prof. Dan Halperin. I had a great privilege to work with this kind, smart, professional, generous, and inspiring person. I thank him for instilling in me the passion for science, learning, and exploring the unknown.

I would like to thank my dear friends and lab mates Michal Kleinbort, Oren Salzman, and Omri Perez, with whom I've had memorable moments, and who made my time at the lab extremely enjoyable. I also wish to thank all the members of the applied computational geometry group over the years and the algorithmic game theory group for their companionship. I also thank the administrative staff of the School of Computer Science, Tel Aviv University, and especially Gilit Zohar-Oren, Pnina Neria Barzilay, Larisa Slepyan-Eshed, Anat Amirav, and Mika Shahar.

I wish to thank my parents for being remarkably supportive throughout my life and especially so for my years in academia. The last person I thank is my partner Nadia, without whom this life would have been mostly a dull journey through space.

I would also like to thank the Clore Israel Foundation for supporting me during my Ph.D. studies. My work has also been supported in part by the Israel Science Foundation (grants nos. 1102/11 and 825/15), by the Blavatnik Computer Science Research Fund, and by the Blavatnik Interdisciplinary Cyber Research Center at Tel Aviv University.

Thank you all,
Kiril

Preface

This thesis presents a collection of seven published papers that were prepared during my Ph.D. studies. It consists of three journal publications and five peer-reviewed conference publications (see below).

Journal Publications

- **Kiril Solovey** and Dan Halperin, "On the Hardness of Unlabeled Multi-Robot Motion Planning." Special issue **(invited)**, *International Journal of Robotics Research*, 35(14): 1750–1759, 2016.
  Also in *Robotics: Science and Systems*, **finalist for best paper, and winner of best student paper**, 2015.

- **Kiril Solovey**, Oren Salzman and Dan Halperin, "Finding a Needle in an Exponential Haystack: Discrete RRT for Exploration of Implicit Roadmaps in Multi-Robot Motion Planning." Special issue **(invited)**, *International Journal of Robotics Research*, 35(5): 501–513, 2016.
  Also in *Workshop on Algorithmic Foundations of Robotics*, 2014.

- Aviv Adler, Mark de Berg, Dan Halperin and **Kiril Solovey** (alphabetical order), "Efficient Multi-Robot Motion Planning for Unlabeled Discs in Simple Polygons." Special issue **(invited)**, *Transactions on Automation Science and Engineering*, 12(4): 1309–1317, 2015.
  Also in *Workshop on Algorithmic Foundations of Robotics*, 2014.

Conference Publications

- **Kiril Solovey** and Michal Kleinbort, "The Critical Radius in Sampling-based Motion Planning" In *Robotics: Science and Systems*, Carnegie Mellon University, Pittsburgh, PA, USA, 2018.

- **Kiril Solovey** and Dan Halperin, "Efficient Sampling-based Bottleneck Pathfinding over Cost Maps." In *International Conference on Intelligent Robots and Systems*, Vancouver, BC, Canada, 2017.

- **Kiril Solovey** and Dan Halperin, "Asymptotically-Optimal Bottleneck Pathfinding with Applications to Fréchet-Type Optimization." In *European Symposium on Algorithms*, 76:1–76:16, Aarhus, Denmark, 2016.

- **Kiril Solovey**, Oren Salzman and Dan Halperin, "New Perspective on Sampling-based Motion Planning via Random Geometric Graphs." In *Robotics: Science and Systems*, University of Michigan, MI, USA, 2016.

- **Kiril Solovey**, Jingjin Yu, Or Zamir and Dan Halperin, "Motion Planning for Unlabeled Discs with Optimality Guarantees." In *Robotics: Science and Systems*, Sapienza University of Rome, Italy, 2015.

Abstract

Motion planning is a fundamental problem in robotics concerned with allowing autonomous robots to efficiently navigate in environments cluttered with obstacles. Although motion planning was initially studied as a theoretical problem in computer science, it is applied nowadays in various fields such as computational biology, computer graphics, and most naturally robotics. Unfortunately, motion planning is notoriously challenging—both theoretically and practically—and many aspects of the problem are not sufficiently understood, even after 30 years of extensive multidisciplinary research.

Perhaps the most challenging aspect of motion planning is the complex and the high-dimensional search space that it induces, which accounts for the geometric structure of the robot, the physical constraints that it needs to satisfy, and the environment in which it operates.

This thesis is mainly devoted to the study of theoretical and practical issues of high-dimensional systems consisting of multiple robots. In particular, we consider *multi-robot* motion planning (MRMP), which is a natural extension of motion planning concerned with the collision-free coordination of multiple robots operating in a shared environment.

We present a theoretical hardness result concerning MRMP, which applies to several variations of the problem, including the *unlabeled* setting which consists of identical and interchangeable robots. We show that it also applies in the standard labeled case. This is the strongest MRMP hardness result in that it pertains to the simplest possible robots over previous proofs: unit squares. We then show that by making relatively-mild simplifying assumptions on the problem one can drastically reduce its complexity. In particular, we consider two settings with such assumptions and devise algorithms that run in time that is (low-)polynomial both in the number of robots and the complexity of the environment. Notably, one of the approaches even comes up with solutions whose cost, i.e., the total length the robots traverse, is (theoretically) close to the optimum. We also consider a sampling-based approach for MRMP and present a new algorithm, which applies to more general settings of the problem, and outperforms the best previously known solutions in practice.

The second part of the thesis is devoted to the study of sampling-based planners, with emphasis on their theoretical guarantees. In particular, we present a general approach to analyze such algorithms by establishing a novel connection to the a well-studied mathematical subject of random geometric graphs. We then present two papers that are devoted to sampling-based motion planning under bottleneck cost, which had only been lightly treated before them in the literature, despite its importance.

Contents

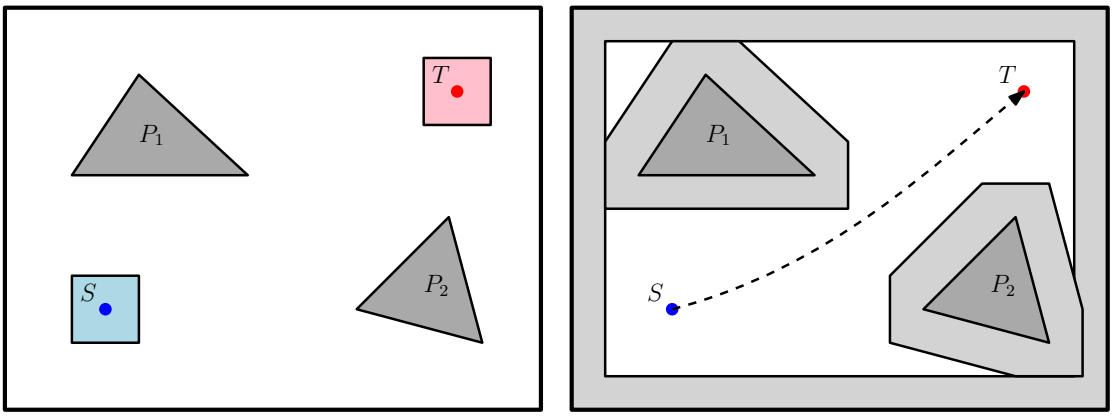
1 Introduction

Remark: This notation, which appears throughout the introduction, indicates that the corresponding text describes a contribution of the author that is part of this thesis.

Motion planning is a fundamental problem in robotics concerned with allowing autonomous robots to efficiently navigate in environments cluttered with obstacles. Although motion planning was initially studied as a theoretical problem in computer science [29], nowadays it is applied in various fields. Notably, it arises in coordination of multiple autonomous vehicles [23], steering surgical needles [9], and planning trajectories of spacecrafts in orbit [94], to name just a few examples. Unfortunately, motion planning is notoriously challenging—both theoretically and practically—and many aspects of the problem are not sufficiently understood, even after 30 years of extensive multidisciplinary research.

The complexity of the problem can be attributed to its continuous and high-dimensional search space, which accounts for (i) the geometric structure of the robot, which may consist of several rigid bodies connected through joints, (ii) the physical constraints that it needs to satisfy, e.g., velocity constraints and limited steering angle as a car has, (iii) and the environment in which it operates, which may change with time, contain unexpected obstacles (as pedestrians jumping into the road), or be a priori unknown altogether.

In its most basic form, motion planning consists of finding collision-free paths for a robot in a (two or three-dimensional) *workspace* cluttered with static obstacles. The spatial pose of the robot, or its *configuration*, is uniquely defined by its degrees of freedom (DoFs). The set of all configurations $\mathcal{C}$ is termed the *configuration space* of the robot, and decomposes into the disjoint sets of free and forbidden configurations, namely $\mathcal{F} \subseteq \mathcal{C}$ and $\mathcal{C} \setminus \mathcal{F}$, respectively. Thus, given start and target configurations, the



(a) (b)

Figure 1.1: Motion planning for a square robot translating in the plane. In (a) the start and target configurations S, T, are denoted by the blue and pink squares, respectively. The robot is confined to a room (black border) which contains two triangular obstacles (gray). In (b), the forbidden regions of the configurations space are drawn in gray, whereas the white region is the free space. Observe that the depicted path is entirely collision free.

problem can be restated as the task of finding a continuous curve in $\mathcal{F}$ connecting the two configurations. See a simple example in Figure 1.1.

It is important to note that the input to the problem consists of only a representation of the robot $\mathcal{R}$ and the workspace in which it operates, whereas the structure of the free space $\mathcal{F}$ is implicitly induced by those parameters. Moreover, although the workspace is typically a planar or a three-dimensional space, the dimension of $\mathcal{C}$ can be much higher, as it corresponds to the number of DoFs of the robot. For instance, if we allow the robot from Figure 1.1 to translate and rotate, it will have three DoFs: x, y coordinates of a specific reference point, and a degree of rotation θ. More interestingly, a rigid body in 3D that is allowed to translate and rotate induces a six-dimensional configuration space, and a planar arm with 8 rotational joints has 8 DoFs (see more examples in Lynch and Park [61]). The humanoid robot that appears in Figure 1.2 has 38 DoFs.

Initial efforts in motion planning can be primarily attributed to computer scientists working in the field of *computational geometry* (see the survey of Halperin et al. [29]). Their focus was on the design of *complete* and *exact* algorithms for motion planning, which are guaranteed to return a solution path if one exists, or to report that none exists otherwise. Such techniques typically construct an explicit representation of the free space of the problem by examining *critical* robot configurations in which the robot exactly touches obstacles without penetrating them. Those form the boundary of $\mathcal{F}$. Such techniques yield efficient (polynomial-time algorithms) for some low-dimensional instances of the problem [29]. However, in general, motion planning is computationally intractable when the number of DoFs is no longer fixed (see, e.g., [13, 69]). Moreover, there are some low-dimensional instances of the problem that are computationally intractable. For instance, finding the *shortest* path for a point robot amid polyhedral obstacles in three dimensions is NP-hard [14], whereas *curvature constrained* motion

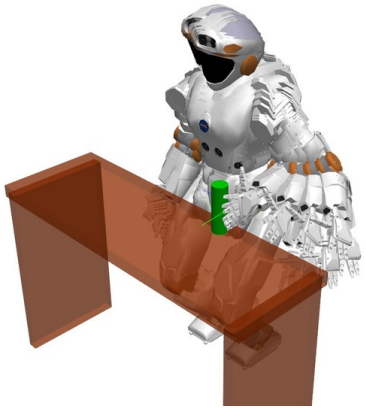
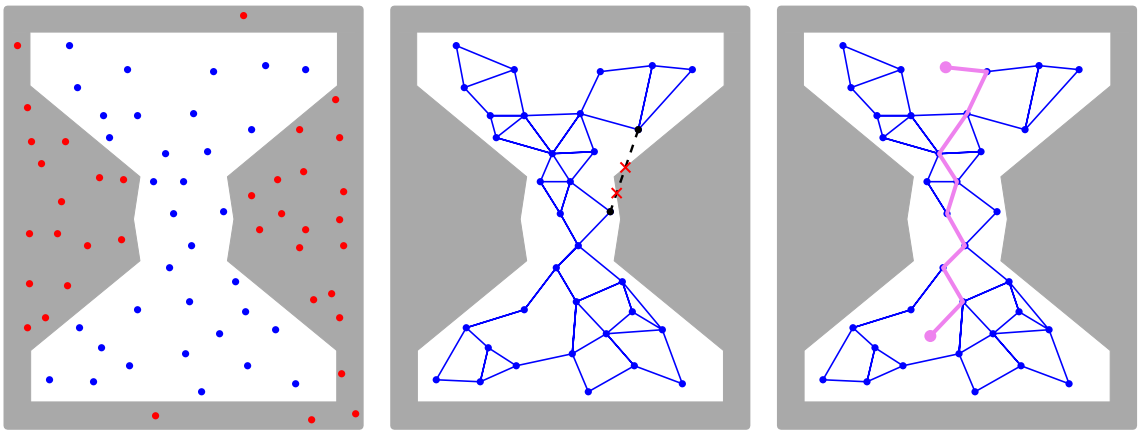


Figure 1.2: The NASA Valkyrie humanoid robot. See Yiming et al. [107] where motion planning for this robot is considered.

planning is NP-hard already in a planar setting [42].

The high computational complexity of exact solutions to motion planning have led to the development of *sampling-based planners* (SBPs). These algorithms, which trade completeness with applicability in practical settings, aim to capture the connectivity of $\mathcal{F}$ in a graph data structure, called a roadmap, by randomly sampling $\mathcal{C}$ and connecting nearby samples with local collision-free paths.

Consider for instance the popular *probabilistic roadmap method* (`PRM`) [41], which is illustrated in Figure 1.3. The input for `PRM` consists of a representation of the robot $\mathcal{R}$ and a workspace, which is usually inferred from the description of the obstacles within. `PRM` also accepts as parameters the number of samples $n \in \mathbb{N}_+$ and a connection radius $r > 0$. In the *preprocessing stage* the algorithm generates n sample configurations (typically in a randomized fashion) in $\mathcal{C}$. Note that the sampling is not conditioned on the free space $\mathcal{F}$, which is unknown a priori, and so some samples may represent colliding configurations. Thus, in the next step `PRM` identifies the collision-free configurations and gets rid of the rest. In particular, it employs a black-box component called *collision detector*, which can perform the detection quickly (see, e.g., [58]). The remaining *collision-free* configurations form the vertex set V of a graph G. In the next step, the algorithm introduces edges to G in the following manner. It identifies pairs of configurations $v, v' \in V$ such that $\|v - v'\| \leqslant r$, i.e., the Euclidean distance between v and v' is at most r, using *nearest-neighbor search* methods (see, e.g., [44]). Then, an edge between such vertices v, v' is added to G only if the straight-line path between the two vertices is collision free. This verification is performed using another dedicated component called the *local planner*, which typically validates the motion by densely sampling the path and checking the resulting configurations using the collision detector. By construction, every path along G between two vertices induces a collision-free path between the corresponding configurations. Given a *query*, which consists of the start and target configurations s, t it remains to connect the two in a similar manner to G and perform a graph search from s to t. If a graph path is found it is easily translated into a collision-free path for the robot in the workspace.



(a) (b) (c)

Figure 1.3: Illustration of PRM. The gray and white regions represent the forbidden and free spaces, respectively. In (a) a set of n random configurations is generated. The forbidden configurations (red) are identified and discarded. In (b) the edges of the graph are generated. Observe that the black dashed edge is not added to the graph, even if its length is below r, since it represents a straight-path motion in which the robot collides with obstacles. In (c) a query is given (depicted as the two magenta bullets), the query configurations are connected to the graph, and G is searched for a path that connects the two configurations.

SBPs have several benefits over their exact counterparts, which made them the weapon of choice in modern motion-planning applications. In particular, such planners are easy to implement and several basic building blocks that are readily available in software exist (we mention the OMPL software library [96], which provides implementation for a variety of SBPs). Furthermore, SBPs are defined independently of the robotic system to which they are applied, which makes them general and applicable to a variety of problems. But perhaps most importantly, SBPs solve motion-planning problems quickly in realistic settings, including high-dimensional instances.

However, all those benefits come at the price of weaker theoretical guarantees, when compared to complete methods. In particular, most of the theoretical properties of SBP are stated in terms of their *asymptotic* behavior, i.e., assuming that the number of samples is sufficiently large, which makes it difficult to assess the outcome of the planner for a specific sample number. Moreover, an inherent drawback of SBPs, which follows from their stochastic nature, is that they are unable to determine situations in which solutions do not exist.

The remainder of this chapter will be devoted to two subjects. The first part, and the main subject of this thesis, is *multi-robot* motion planning, which involves multiple moving robots that operate in a shared environment. We will learn about different theoretical aspects of the problem, consider complete algorithms that can solve it efficiently, even when the number of robots involved is large. The second part is devoted to a careful examination of SBP, mostly for the single-robot case, with emphasis on their theoretical guarantees.

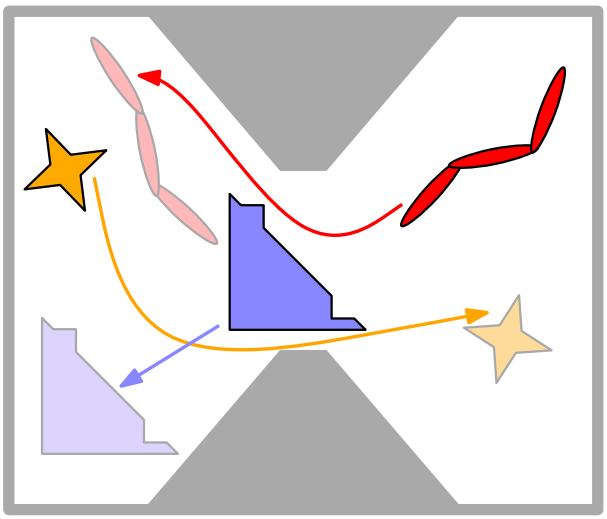


Figure 1.4: Multi-robot motion planing for three robots.

1.1 Multi-robot motion planning

Multi-robot motion planning (MRMP) is a natural extension of motion planning, which involves the collision-free coordination of multiple robots operating a shared environment. See illustration in Figure 1.4. Here the robots should avoid collisions with obstacles, as well as collisions with other robots. In the past few years, we have witnessed the rapid development and adaptation of autonomous multi-robot and multi-vehicle systems in a wide variety of application domains. The most prominent example is arguably the success of Kiva Systems, now part of Amazon, which developed a warehousing system employing hundreds of mobile robots to facilitate order assembly tasks [106]. More recently, Amazon, DHL, and Google have demonstrated working prototypes of aerial vehicles capable of automated package delivery. Since the vehicles are intended to operate in an autonomous, swarm-like setting, we can foresee in the near future the demand for efficient path-planning algorithms designed for such systems.

MRMP is notoriously difficult as it often involves many degrees of freedom, and consequently a vast search space [34, 92]. In general, each additional robot introduces several more degrees of freedom to the problem. Nevertheless, there is a rich body of work dedicated to this problem. The earliest research efforts can be traced back to the 1980s [79].

MRMP has been considered in two main flavors. In the *labeled* and standard setting of the problem each robot needs to move from a specific start configuration to another specific target position. This implies that each robot has its own individual task that it has to perform. In some practical settings it happens that the robots are identical in form and functionality and thus are interchangeable. In such cases it is more natural to allow flexible assignment of robots to targets. In the *unlabeled* variant of MRMP (see, e.g., [45]), the robots are given a set of target positions and the goal is to move the robots in a collision-free manner so that each robot ends up at *some* target, without specifying exactly which. We mention that we also studied a generalization consisting of several unlabeled groups of robots, termed

k-colored MRMP [83].

Practical approaches for MRMP can be typically subdivided into two categories. Decoupled techniques (see, e.g., [19, 27, 54, 64, 100, 101]) reduce the size of the search space by partitioning the problem into several subproblems, which are solved separately, and then the different components are combined. In contrast to that, centralized approaches (see, e.g., [5, 45, 63, 72, 88, 103, 104]) usually work in the combined high-dimensional configuration space, and thus tend to be slower than decoupled techniques. However, centralized algorithms often come with stronger theoretical guarantees, such as completeness. Through the rest of this section we will consider such centralized methods.

MRMP has also been considered as a discrete problem on a graph [47]: robots can be viewed as pebbles placed on the vertices of the graph and are bound to move from one set of vertices to another along edges. Many aspects of the discrete case are well understood. In particular, for the labeled setting of the problem there exist efficient feasibility-test algorithms [8, 28, 110], as well as complete planners [46, 48, 59]. For the unlabeled case there even exist complete and efficient planners that generate an optimal solution [39, 111, 113] for different cost functions. While there exists a fundamental difference between the discrete and the continuous setting of the problem, the continuous case being exceedingly more difficult, several recent techniques in the continuous domain [1, 83, 90, 98] have employed concepts that were initially introduced in the discrete domain.

1.1.1 Complete approaches

This subsection is devoted to complete approaches for MRMP. One of the first occurrences of MRMP in the computational-geometry literature can be found in the seminal series of papers on the *Piano Movers' Problem* by Schwartz and Sharir. They first considered the problem in a general setting [78] and then narrowed it down to the case of disc robots moving amidst polygonal obstacles [79]. In the latter work an algorithm was presented for the case of two and three robots, with running time of $O(n^3)$ and $O(n^{13})$, respectively, where n is the complexity of the workspace. Later Yap [108] used the *retraction method* to develop more efficient algorithms, which run in $O(n^2)$ and $O(n^3)$ time for the case of two and three robots, respectively. Several years afterwards, Sharir and Sifrony [80] presented a general approach based on *cell decomposition*, which is capable of dealing with various types of robot pairs and which has a running time of $O(n^2)$. Several techniques exist that reduce the effective number of DoFs [5, 101].

Hardness results

When the number of robots is no longer a fixed constant the problem can become computationally intractable. Specifically, Hopcroft et al. [34] showed that the problem is PSPACE-hard for the setting of multiple rectangular robots bound to translate in a rectangular workspace. Their proof required the rectangular robots to be of varying dimensions. Spirakis and Yap [92] showed that the problem is NP-hard for disc robots in a simple-polygon workspace; here the proof relies strongly on the fact that the discs are of varying radii.

More recently, Hearn and Demaine [31, 32] improved the result of Hopcroft et al. by showing that the robots can be restricted to only two types—2×1 and 1×2 rectangles. Their work is more general:

They introduced in this work the *nondeterministic constraint logic* (NCL) model of computation, for which they describe several PSPACE-hard problems, and from which they derive the PSPACE-hardness of a variety of puzzle-like problems that consist of sliding game pieces. In particular, they applied their technique to the *Sokoban* puzzle where multiple "crates" need to be pushed to target locations.

Another interesting puzzle, where NCL came in handy, is *Rush Hour*: a parking attendant needs to evacuate a car from a parking lot, by clearing a route blocked by other cars. The cars are restricted to move forward or backward, with respect to the direction of the car's front. The construction of Hearn and Demaine involves $2 \times 1, 1 \times 2, 3 \times 1$ and 1×3 cars in a rectangular room, where the front of the car is at the short edge. Flake and Baum [20] used the same building blocks, but their hardness proof does not rely on a reduction from NCL. Another NCL-based hardness proof was developed by Tromp and Cilibrasi [97]. Their construction requires only 2×1 and 1×2 cars in a rectangular room.

Our paper [84] (see also Chapter 2) presents the first hardness result for the unlabeled case of MRMP, where we consider the setting of unit-square robots moving amidst polygonal obstacles. Moreover, we consider three additional variants of the unlabeled problem and show that they are all PSPACE-hard. For instance, we show that the seemingly easier version of the problem where only one of the robots is required to reach a certain target position while the other robots function as movable obstacles, is also computationally intractable. We mention that our proofs can be used to show that the labeled variant, again, for unit-square robots, is PSPACE-hard as well, which sets another precedent, as previous hardness results require the robots to be of different shapes (or at least in different orientations). Additionally, we settle an open problem introduced by Tromp and Cilibrasi [97]: we prove that Rush Hour is PSPACE-hard for unit-square cars moving amidst polygonal obstacles. All our proofs follow from a reduction using the aforementioned NCL model.

Separation assumptions

Although MRMP is computationally intractable in general, several recent works have demonstrated that the problem can be solved efficiently, if one makes some simplifying *separation assumptions*.

Turpin et al. [98] describe an efficient and complete algorithm for unlabeled planning for disc robots, which also guarantees finding the *optimal* solution in terms of the length of the longest path traversed by a robot. Their algorithm makes the assumption that a certain portion of the free space, surrounding each start or target position, is *star-shaped*.

More recently, we studied the same problem [1] (see Chapter 3), although under a milder assumption requiring every pair of start positions, every pair of target positions, and every pair of start-target positions, each have a distance of at least 4 units between the centers of the unit-disc robots, and requiring the workspace being a simple polygon, i.e., a polygon that contains no holes. We mention that a simple-polygon workspace can still induce (for a single robot) a free space that consists of several connected components (see [1]). Notice that we only assume this extra separation between the robots in their static initial and goal placements; we do not assume any extra separation (beyond non-collision) between a robot and the obstacles, nor do we enforce any extra separation between the robots during

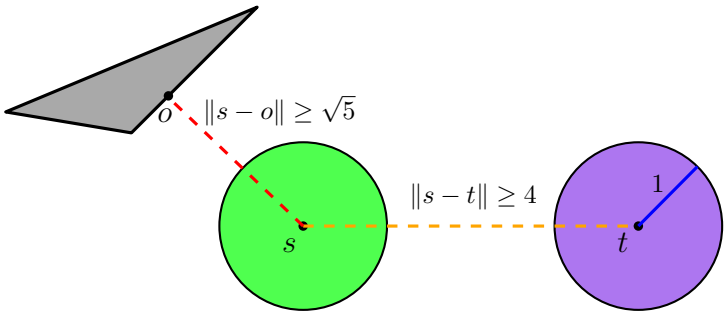


Figure 1.5: Illustration of the separation conditions assumed in our work [90]. The green and purple discs represent two unit-disc robots placed in $s \in S, t \in T$, respectively. The blue line represents the unit radius of the robot (for scale). The distance between s and t is at least 4 units (see dashed orange line). The gray triangle represents an obstacle, and the point o represents the closest obstacle point to s. Notice that the distance between o and s is at least $\sqrt{5}$ units (see dashed red line).

the motion. Even this basic version of the problem turns out to have a rich structure and poses several difficulties and interesting questions. By carefully examining the various properties of the problem we show how to transform it into a discrete pebble-motion problem on graphs. A solution to the pebble problem, which can be generated with rather straightforward techniques, can then be transformed back into a solution to our continuous motion-planning problem. Using this transformation we are able to devise an efficient algorithm whose running time is $O(m^2 + mn)$, where m is the number of robots and n is the complexity of the workspace.

In a more recent paper [90] (see Chapter 4) we consider a similar setting, with the additional requirement that each start or goal position has a (Euclidean) distance of at least $\sqrt{5}$ to any obstacle in the environment (see illustration in Figure 1.5). However, we do lift the assumption of the previous paper that the workspace consists of a single polygon. We follow the methodology of performing target assignments and path planning concurrently [12, 98, 112] and adopting a graph-based vertex ordering argument from Yu and LaValle [111], we develop a complete combinatorial algorithm for the unlabeled problem described above. Given that the separation conditions are fulfilled, our algorithm is guaranteed to generate a solution if one exists, or report that none exists otherwise. It has a running time of $\tilde{O}\left(m^4 + m^2 n^2\right)$, where m is the number of robots, and n is the description complexity of the workspace environment, i.e., the number of edges of the polygons. Furthermore, the total length of the returned solution, i.e., the sum of lengths of the individual paths, is at most $\mathrm{OPT} + 4m$, where OPT is the optimal solution cost. To the best of our knowledge, we are the first to describe a complete algorithm that fully addresses the problem of planning minimum total-distance paths for unlabeled discs.

1.1.2 Sampling-based multi-robot approaches

Sampling-based algorithms designed for a single robot can easily be extended to the multi-robot case by considering the fleet of robots as one composite robot (see, e.g., [75]). Such a naive approach suffers from inefficiency as it overlooks aspects that are unique to the multi-robot problem, and hence can handle

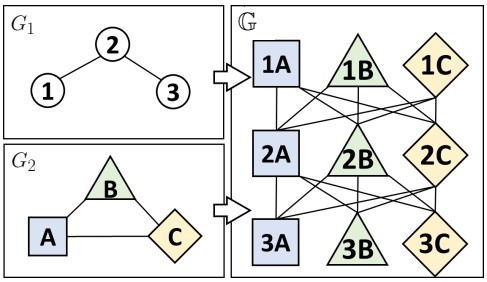


Figure 1.6: An illustration of a tensor product (right) between two graphs (left).

only a very small number of robots. Several techniques tailored for instances of MRMP involving a small number of robots have been described [33, 72].

In a previous work of ours [83] we introduce an SBP for the k-color variant of MRMP. At the heart of the algorithm is a novel technique where the k-color problem is reduced to several discrete pebble-motion problems (see above). These reductions amplify basic samples into massive collections of free placements and paths for the robots. This algorithm manages to cope with some relatively complex scenarios in reasonable running time. An improved version of this algorithm was presented in [49], where it was employed to rearrange multiple objects using a robotic manipulator.

Švestka and Overmars [103] introduced a different approach for labeled MRMP, which leverages the following fundamental observation: the structure of the overall high-dimensional multi-robot free space can be inferred by first considering independently the free space of every robot, and combining these subspaces in a meaningful manner to account for robot-robot collisions. They suggested the following data structure: given a problem consisting of m robots, construct for each robot $1 \leqslant i \leqslant m$ a `PRM` graph G_i, which ignores the existence of the other robots. Then, construct a *composite roadmap* $\mathbb{G} = G_1 \times \ldots \times G_m$, which is a tensor product (see Figure 1.6) of the individual-robot roadmaps: for every m-tuple $(v_1, \ldots, v_m)$ of single-robot vertices, where $v_i \in G_i$, $\mathbb{G}$ has a corresponding vertex; two vertices $(v_1, \ldots, v_m), (v'_1, \ldots, v'_m)$ of $\mathbb{G}$ are connected by an edge if for every $1 \leqslant i \leqslant m$ it holds that $(v_i, v'_i) \in G_i$. That is, a vertex or an edge of $\mathbb{G}$ describe a *simultaneous* placement or motion, respectively, of the m robots. Notice that by definition any path along $\mathbb{G}$ induces a multi-robot motion in which the robots are guaranteed not to collide with obstacles. However, it is still possible that robot-robot collision will occur. Thus, it remains to remove such colliding vertices or edges during the construction of $\mathbb{G}$, or in an online fashion when $\mathbb{G}$ is searched for a path between two specific vertices.

Due to the exponential nature of the resulting roadmap, this technique is only applicable to problems that involve a modest number of robots. A recent work by Wagner and Choset [104] suggests that $\mathbb{G}$ does not necessarily have to be explicitly represented. Instead, they implicitly represent $\mathbb{G}$ by an explicit construction of only $G_1, \ldots, G_m$, which are much more manageable in terms of storage space and running time. They apply their `M*` algorithm to efficiently retrieve paths over $\mathbb{G}$, while minimizing the explored portion of the roadmap. The resulting technique is able to cope with a large number of robots, for certain types of scenarios. However, when the degree of simultaneous coordination between the robots increases,

there is sharp increase in the running time of this algorithm, as it has to consider many neighbors of a visited vertex of $\mathbb{G}$. This makes M* less effective when the motion of multiple robots needs to be tightly coordinated.

Recently we introduced a different sampling-based framework for MRMP, which combines an implicit representation of $\mathbb{G}$ with a novel approach for pathfinding in geometrically-embedded graphs tailored for MRMP [88] (see Chapter 5). Our pathfinding algorithm, *discrete-RRT* (dRRT), is an adaptation of the celebrated RRT algorithm for the discrete case of a graph, and it enables a rapid exploration of the high-dimensional configuration space by carefully walking through an implicit representation of the tensor product of roadmaps for the individual robots. We demonstrated our approach experimentally on scenarios that involve as many as 60 DoFs and on scenarios that require tight coordination between robots. On most of these scenarios our algorithm is faster by a factor of at least ten when compared to existing algorithms that we are aware of.

Later on, we applied dRRT for motion planning of a free-flying multi-link robot [74]. In that case, dRRT allowed to efficiently decouple between costly self-collision checks, which were done offline, and robot-obstacle collision checks, by traversing an implicitly-defined roadmap, whose structure resembles to that of $\mathbb{G}$. dRRT has also been used in the study of the effectiveness of metrics for MRMP, which are an essential ingredient in SBP [7]. Lastly, we developed dRRT* [18], which is an asymptotically-optimal variant of dRRT, i.e., guaranteed to converge to the shortest plan (in different cost functions).

1.2 Theory of sampling-based planning

Sampling-based planners (SBPs) were first described in the mid '90s and several prominent instances of that time, which are still used extensively today, include PRM [41], RRT [55], and EST [35]. In contrast with complete planners, which generate a precise representation of $\mathcal{F}$, SBPs approximate its structure by random sampling and computing a graph data structure. SBPs typically accept as a parameter the number of samples n to be drawn (or a time budget), and an immediate question that follows is how large this number should be (or for how long the planner should be run). Unfortunately, it is not yet clear how to answer this in a precise manner, as it depends heavily on the structure of the free space $\mathcal{F}$, which is unknown a priori. We can however prove for some planners that they will *eventually* find a solution if one exists. That is, a planner is *probabilistically complete* (PC) if the probability of finding a solution tends to 1 as the number of samples n tends to infinity. Indeed, it was shown that all the three algorithms mentioned above have this property [40, 55, 35] for the geometric setting of the motion planning problem. More recently, Ladd and Kavraki [51] have developed a more general proof for PC of PRM, which applies to a variety of robotic systems. Notably, their work exploits measure theory to obtain this broad generality.

Another important parameter of SBP, which dictates both the running time and the quality of the returned solution, is the number of neighbors considered for connection for each added sample. In many techniques, including PRM, this number is directly affected by a connection radius r: Decreasing r reduces the number of neighbors. This in turn reduces the running time of the planner for a given number of

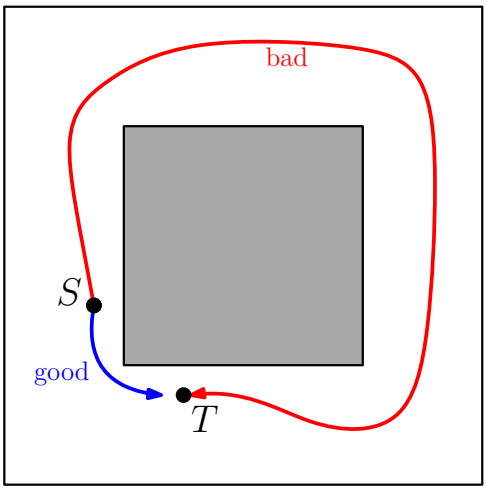


Figure 1.7: Illustration of the scenario used in [62] to prove that RRT is not (asymptotically) optimal.

samples n, but may also reduce the quality of the solution or its availability altogether. Thus, it is desirable to come up with a radius r that is small, but not to the extent that the planner loses its favorable properties.

1.2.1 Quality guarantees

In recent years we have seen an increasing interest in *high-quality* motion planning. Quality can be measured in terms of length, clearance, smoothness, energy, to mention a few criteria. The literature contains many examples of planners, which are shown empirically to produce high-quality paths (for a partial list see [4, 26, 57, 60, 68, 82, 99]). Unfortunately, they are not backed by rigorous proofs pertaining to the quality of the solution produced by the algorithm.

Nechushtan et al. [62] were among the first to address this matter in a mathematically-rigorous manner. This work proved that RRT can produce arbitrarily-bad (long) paths with non-negligible probability. In particular, the authors consider a specific instance of motion planning depicted in Figure 1.7, and prove that with some probability independent of n, RRT would prefer to take the longer route. This route can be made arbitrarily longer than the best solution available by putting S and T closer to each other, but still remain on two separate sides of the corner.

The influential work of Karaman and Frazzoli [38] laid the theoretical foundations for analyzing quality of SBPs. The authors introduced the notion of *asymptotic optimality* (AO), which indicates the solution returned by the planner converges to the optimum as $n \to \infty$. They showed that the following algorithms are AO: PRM*, which is a special case[1] of PRM with a specific value of the connection radius r_n; an AO variant of RRT [55] termed RRT*; RRG, which can be viewed as a combination of RRT and PRM (see Figure 1.8). The analysis in [38] establishes that $r_n = \Theta\left((\log n/n)^{1/d}\right)$ guarantees AO, where the

[1] Throughout this introduction we will refer to the more general algorithm PRM, rather than PRM*, which is a special case of PRM.

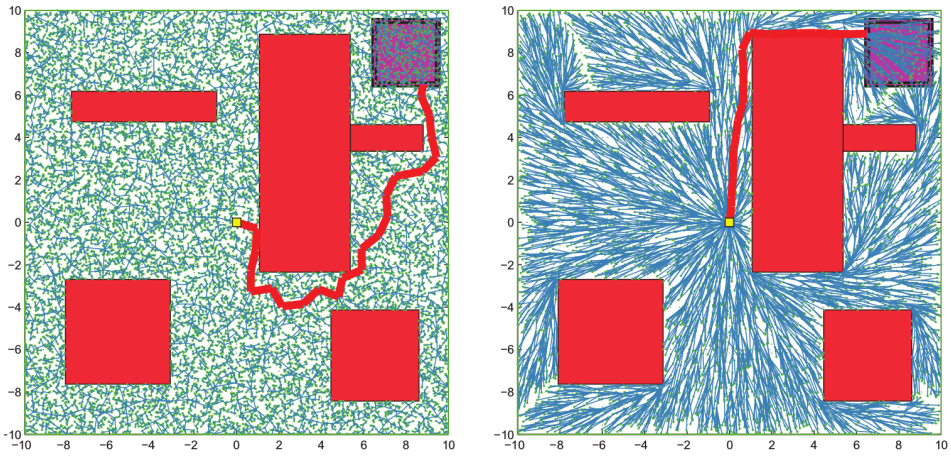


Figure 1.8: Reproduced from [38]. A comparison between RRT (left) and RRT* (right), which illustrates the graphs (trees) obtained by the two planners for the same number of samples. Observe how the RRT* solution converges to the optimal one.

configuration space of the robot is assumed to be $[0,1]^d$. This indicates that the expected number of neighbors used per vertex should be $O(\log n)$. The authors also proved that for sufficiently-small radii of order $\Theta(n^{-1/d})$ the planner is guaranteed to fail (asymptotically) in finding any solution.

Following this breakthrough, other AO planners have emerged (see e.g., [3, 6, 25]). Janson et al. [37] introduced FMT*, which is a single-query planner that traverses an implicitly-represented PRM graph, and is comparable in performance to RRT*. The authors refined the proof technique presented in [38], which allowed them to slightly reduce the connection radius r_n necessary to FMT* and PRM to achieve AO. We do mention that here again $r_n = \Theta\left((\log n/n)^{1/d}\right)$. BFMT*, which is a bidirectional version of FMT*, was introduced in Starek et al. [93]. In their paper the authors also proved that the success rate of PRM, FMT*, BFMT* can be lower bounded by $1 - O\left(n^{-\eta/d}\log^{-1/d} n\right)$, where $\eta > 0$ is a tuning parameter. In this context, we also mention the work of Dobson et al. [17], which bounds the success rate with an expression that depends on the amount of deviation from the optimum.

A number of methods have been developed to reduce the running time or space requirements of existing planners by relaxing AO constraints to asymptotic near optimality[2] (AnO). For instance, LBT-RRT [71] interpolates between the quick RRT and the AO yet slower RRG, according to a predefined parameter. MPLB [70] can be viewed as a relaxation of FMT*. SPARS2 [16] and RSEC [73] perform a sparsification of PRM in an online and offline fashion, respectively, to reduce the space footprint of the produced roadmap.

1.2.2 Analysis of SBP via Random geometric graphs

Interestingly, roadmaps constructed by many sampling-based planners coincide, in the absence of obstacles, with standard models of *random geometric graphs* (RGGs). RGGs have been studied for several

[2] AnO means that the cost of the solution tends to at most a constant factor times the optimum, compared with AO in which this constant is equal to one.

decades and by now a rich body of literature exists analyzing various properties and types of RGGs, see, e.g., Penrose [65]. Indeed, in their seminal work on optimal motion planning, Karaman and Frazzoli observed this relation. They conjectured that a sampling-based planner possesses a certain property if the underlying RGG has this property as well (see [38, Section 6]).

In our recent work [89] (Chapter 6) we settle this conjecture and leverage it for the development of a general framework for the analysis of sampling-based planners. Our framework, which we call *localization-tessellation*, allows for easy transfer of arguments on RGGs from the free unit-hypercube to spaces punctured by obstacles, which are geometrically and topologically much more complex. We demonstrate its power by providing alternative and (arguably) simple proofs for probabilistic completeness and asymptotic (near-)optimality of `PRM`. Our work shows that one can slightly reduce the `PRM` and `FMT`* radius r_n obtained in Janson et al. [37].

Furthermore, we introduce three variants of `PRM` called Soft-`PRM`, Bluetooth-`PRM` and Embedded-`PRM`, which perform connections in a randomized fashion. For instance, in Soft-`PRM` two given vertices are connected with probability that depends on their distance—the smaller the distance the higher the probability. We provide the first mathematically-rigorous study of these types of planners. Additionally, using Soft-`PRM` and Bluetooth-`PRM` we show that the standard `PRM` still maintains its favorable properties even when implemented using *approximate* nearest-neighbor search queries.

1.2.3 Bottleneck cost

The aforementioned papers mainly deal with the cost function of *path length*. Our two recent works [85, 86] (Chapter 7, Chapter 8) considered the *bottleneck-pathfinding* problem in a sampling-based setting. The bottleneck cost is defined as follows: Every robot configuration x is paired with a value $\mathcal{M}(x)$, and the cost of a path is the maximum value of $\mathcal{M}$ along any configuration on the path. As a concrete example, consider $\mathcal{M}$ which assign to x the value $1/\text{dist}(x)$, where $\text{dist}(x)$ is the distance from x to the closest obstacle in the environment. The bottleneck path in this case is the one which maximizes the robot's clearance [105].

Bottleneck pathfinding can also arise in settings involving multiple robots. Suppose that we have m robots, where each robot is assigned with a specific path that it needs to follow, as is often the case in factory assembly lines [91], aviation routes [11] and traffic intersections [15]. We wish to find the *safest* coordination—to tune the robots' velocities while moving along their paths such that maximal distance between the robots is maintained. This problem again, can be reformulated as a bottleneck-pathfinding problem, where m is the dimension of the effective search space.

Another example of bottleneck pathfinding is Fréchet matching, which is a popular similarity measure between curves that has been extensively studied by the computational-geometry community (see, e.g., [2, 30]) and recently was used in robotics [67, 102]. This matching takes into consideration not only the "shape" of curves but also the order in which the points are arranged along them.

In [85] we introduced an SBP termed `BTT` (bottleneck tree) for bottleneck planning and studied its performance on various settings of bottleneck pathfinding. We proved that `BTT` is AO for bottleneck

cost using the connection radius that we obtained in [89]. In a subsequent paper [86] we studied the theoretical implications that follow from the requirement to produce paths that are monotone in all the coordinates. Such a requirement arises for instance in the safest-coordination problem mentioned above, in which the robots are not allowed to backtrack.

1.2.4 Analysis of SBP via Continuum percolation

In a recent paper [87] (see Chapter 9) we develop a new analysis of PRM, which relies on a novel connection between sampling-based planners and *continuum percolation*, which studies the phenomenon of emergence of infinite connected components in various random structures (see, e.g., [22]). Our analysis is tight and proves the existence of a *critical connection* radius $r_n^* = \gamma^* n^{-1/d}$, where $\gamma^* > 0$ is a constant, and $d \geqslant 2$ is the dimension: If $r_n < r_n^*$ then PRM is guaranteed to fail, where d is the dimension. Above the threshold, i.e., when $r_n > r_n^*$, PRM is AO for the bottleneck cost, and AnO with respect to the path-length cost. Furthermore, our analysis yields concrete bounds on the probability of success, which is lower-bounded by $1-O(n^{-1})$. By success, we refer to the event that the algorithm returns a solution that is at most a given multiplicative factor away from the optimum. Notice that this bound is comparable to the one obtained in Starek et al. [93], although we show this for a much smaller radius.

Our analysis is not restricted to PRM and applies to a variety of planners that maintain PRM-like roadmaps, explicitly or implicitly. For instance, when r_n is above the threshold, FMT* is AnO with respect to the path-length cost, while BTT is AO with respect to the bottleneck cost. RRG behaves similarly for the two cost functions. Our results are also applicable to *multi-robot* motion planners such as the recently introduced dRRT* [18], and M* [104] when applied to a continuous domain. See Figure 1.9 for additional PRM-based planners to which our analysis is applicable.

A practical implication of our work is that AO (or AnO), under the regime of uniform random sampling, can be achieved even when every sample is connected to $\Theta(1)$ neighbors. This is in stark contrast with previous work, e.g., [37, 38, 85, 89], which provided a rough estimate of this number, that is proportional to $O(\log n)$. Interestingly, our $\Theta(1)$ bound for uniform samples is comparable to the best known bound for deterministic samples, which was obtained by Janson et al. [36].

Lastly, in this work we also study the asymptotic properties of the aforementioned planners when constructed using a deterministic point process, rather than uniform random sampling which we have mentioned so far. We introduce an analysis which shows that the graphs constructed using such deterministic samples can be sparsified in a randomized fashion while maintaining AO or AnO of the planners. We term this regime as *semi-deterministic sampling*.

1.2.5 Planning under differential constraints

So far our discussion on the theoretical properties of SBPs was restricted to simple holonomic robotic systems whose configuration space is assumed to be the Euclidean space. Two recent papers by Schmerling et al. [76, 77] establish the theoretical foundations of PRM and FMT* planners when applied to robots having certain differential constraints, by examining the geometry induced by the robot's constraints,

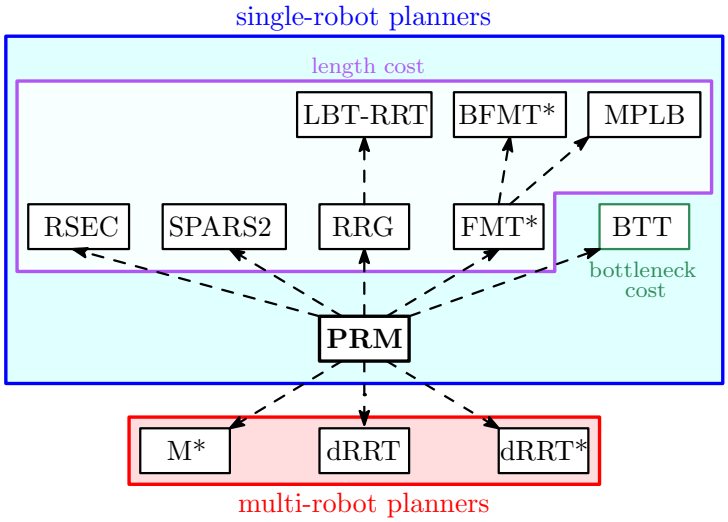


Figure 1.9: PRM-based planners to which the analysis from [87] applies. Single robot and multi-robot planners, are bounded into the blue and red frames, respectively. Planners inside the magenta frame aim to minimize the path-length cost, whereas BTT is designed for bottleneck cost, and RRG works for both costs. Roughly speaking, arrow from "A" to "B" indicates that theoretical properties of "A" extend to "B", or that the B maintains "A" as a substructure.

which is a differential manifold. Li et al. [56] take a different approach and develop a brand-new algorithm for robotic systems with differential constraints, which is shown to be AO. Notably, unlike PRM-flavored planners, the new planner does not assume the existence of a *steering function*, which computes the optimal motion of the robot between any two given configurations in the absence of obstacles. It is often the case for complex robotic systems that precise steering cannot be performed, at least not efficiently (see discussion in [56, Section 1.3]).

Finally, we mention a number of PC techniques, which are known to be highly effective for motion planning with differential constraints and for systems having a large number of degrees of freedom. See [50, 52, 95, 66].

1.3 Organization of articles included in this dissertation

We provide outline for the articles presented in the following chapters, and which were mentioned above.

In the first part we consider different aspects of multi-robot motion planning. Chapter 2 (reference [84]) provides a hardness proof for unlabeled and labeled MRMP, and a new hardness result for Rush Hour. Chapter 3 (reference [1]) describes a complete and efficient algorithm for unlabeled MRMP of disc robots in a simple-polygon environment. The following Chapter 4 (reference [90]) considers a similar setting and describes an efficient algorithm that also guarantees to minimize the total cost traversed by the robots. Chapter 5 (reference [88]) considers the labeled case for far more general robots and introduces the SBP termed dRRT.

The second part is devoted to sampling-based planning, with an emphasis on theoretical proper-

ties. Chapter 6 (reference [89]) describes a new analysis technique of SBPs using random geometric graphs. Chapter 7 (reference [85]) initiates the study of bottleneck cost in a sampling-based setting, and introduces the `BTT` planner. Chapter 8 (reference [86]) considers the implications of monotonicity requirement on `BTT`. Chapter 9 includes our most recent result, which provides a tight analysis of the connection radius required for asymptotic optimality. There we present a tight analysis of SBPs, which applies to length and bottleneck costs functions, and is applicable to a wide range of planners [87].

2

On the Hardness of Unlabeled Multi-Robot Motion Planning

Article

The International Journal of
Robotics Research
2016, Vol. 35(14) 1750–1759

DOI: 10.1177/0278364916672311
ijr.sagepub.com

On the hardness of unlabeled multi-robot motion planning

Kiril Solovey and Dan Halperin

Abstract

In unlabeled *multi-robot motion planning, several interchangeable robots operate in a common workspace. The goal is to move the robots to a set of target positions such that each position will be occupied by* some *robot. In this paper, we study this problem for the specific case of unit-square robots moving amidst polygonal obstacles and show that it is* PSPACE*-hard. We also consider three additional variants of this problem and show that they are all* PSPACE*-hard as well. To the best of our knowledge, this is the first hardness proof for the unlabeled case. Furthermore, our proofs can be used to show that the* labeled *variant (where each robot is assigned a specific target position), again, for unit-square robots, is* PSPACE*-hard as well, which sets another precedent, as previous hardness results require the robots to be of different shapes (or at least in different orientations). Lastly, we settle an open problem regarding the complexity of the well-known* Rush-Hour *puzzle for unit-square cars in environments with polygonal obstacles.*

Keywords

Hardness of motion planning, multi-robot motion planning, robot motion planning, rush hour

1. Introduction

In practical settings where multiple robots operate in a common environment, it is often the case that the robots are identical in form and functionality and thus are interchangeable. Specifically, in *unlabeled* multi-robot motion planning (unlabeled planning, in short) a group of identical robots need to reach a set of target positions. As the robots are identical we only require that in the end of the process each target position will be occupied by *some* robot. This is in contrast to the standard *labeled* (also known as *fully-colored*) multi-robot motion problem, where each robot is required to reach a *specific* target position, and the robots may differ in shape. While labeled planning has been of interest to many researchers for the past four decades, unlabeled planning has only been recently introduced and investigated.

1.1. Related work

We start with the much more intensively studied *labeled* case of multi-robot motion planning. Schwartz and Sharir (1983) were the first to consider the labeled problem from the geometric point-of-view, and in particular studied the case of two discs moving amidst polygonal obstacles and developed an algorithm with a running time of $O(n^3)$, where n is the complexity of the workspace. Yap (1984) also considered this setting and described an algorithm of complexity $O(n^2)$. Later on, Sharir and Sifrony (1991) proposed an $O(n^2)$ algorithm as well, although their algorithm deals with several additional types of robots, besides discs.

When the number of robots is no longer a fixed constant the problem can become computationally intractable. Specifically, Hopcroft et al. (1984) showed that the problem is PSPACE-hard for the setting of rectangular robots bound to translate in a rectangular workspace. Their proof required the rectangular robots to be of varying dimensions. Spirakis and Yap (1984) showed that the problem is NP-hard for disc robots in a simple-polygon workspace; here the proof strongly relies on the fact that the discs are of varying radii.

More recently, Hearn and Demaine (2005, 2009) improved the result of Hopcroft et al. by showing that the robots can be restricted to only two types—2×1 and 1×2 rectangles. Their work is more general: they introduced in this work the *nondeterministic constraint logic* (NCL) model of computation, for which they describe several PSPACE-hard problems, and from which they derive the PSPACE-hardness of a variety of puzzle-like problems that consist of sliding game pieces. (We describe the NCL model

Blavatnik School of Computer Science, Tel-Aviv University, Israel

Corresponding author:
Kiril Solovey Blavatnik School of Computer Science, Tel-Aviv University, Tel Aviv 69978, Israel.
Email: kirilsol@post.tau.ac.il

in detail later on.) In particular, they applied their technique to the *SOKOBAN* puzzle where multiple "crates" need to be pushed to target locations.

Another interesting puzzle, where the NCL model came in handy, is *rush hour*: a parking attendant needs to evacuate a car from a parking lot, by clearing a route blocked by other cars. The cars are restricted to moving forward or backward, with respect to the direction of the car's front. The construction of Hearn and Demaine involves 2×1, 1×2, 3×1 and 1×3 cars in a rectangular room, where the front of the car is at the short edge. Flake and Baum (2002) used the same building blocks, but their hardness proof does not rely on a reduction from NCL. Another NCL-based hardness proof was developed by Tromp and Cilibrasi (2005). Their construction requires only 2×1 and 1×2 cars in a rectangular room.

More hardness results using the NCL model have followed for various problems and puzzles (see, e.g., Buchin and Buchin, 2012; Buchin and Gerrits, 2013; Holzer and Jakobi, 2012; Ito et al., 2012).

Possibly due to the various hardness results related to multi-robot motion-planning problem, the interest of the computational-geometry community in the multi-robot motion planning problem has diminished over the years and has gradually shifted to the robotics community, who started to develop sampling-based tools for the problem. Sampling-based algorithms[1] try to capture the structure of the configuration space of the problem (or, more accurately, its division into "free" and "forbidden" regions) by random sampling of the space and connecting nearby configurations by "simple" paths, to form a *roadmap*. While such methods are incapable of determining whether a solution does not exist, they often provide asymptotic guarantees of completeness and optimality. Even though such techniques can be applied as-is to the labeled planning problem (Sánchez-Ante and Latombe, 2002), many approaches that were specifically designed for the problem have emerged (see, e.g., Hirsch and Halperin, 2002; Salzman et al., 2015; Solovey et al., 2016; Švestka and Overmars, 1998; Wagner and Choset, 2011).

Unlabeled multi-robot motion planning was introduced by Kloder and Hutchinson (2005), who described a sampling-based algorithm for the problem. More recently, Solovey and Halperin (2014) have developed a sampling-based algorithm for the unlabeled problem, as well as for a generalization termed *k-color* planning that consists of k groups of interchangeable robots. Krontiris et al. (2014) describe an adaptation of this work for the problem of rearranging several objects using a robotic manipulator.

Turpin et al. (2013) describe an efficient and complete algorithm for unlabeled planning for disc robots, which also guarantees finding the optimal solution in terms of the length of the longest path traversed by a robot. However, their algorithm makes the assumption that a certain portion of the free space, surrounding each start or target position, is *star-shaped*. More recently, Adler et al. (2015) studied the same problem, although under a milder assumption requiring each pair of start or target positions to be separated by a distance of at least 4, where the radius of the robots is 1. They describe an algorithm with a running time of $O(m^2 + mn)$, where n is the complexity of the polygon and m is the number of robots. In a similar setting, Solovey et al. (2015) show that if two simplifying assumptions are made regarding the distances between pairs of start and target positions and between such positions and the obstacles, an efficient algorithm can be developed. In particular, their algorithm has a running time[2] $\tilde{O}(m^4 + m^2n^2)$. Furthermore, the algorithm returns a solution whose total length (namely the total length traveled by all the robots) is OPT $+ 4m$, where OPT is the optimal solution cost.

A crucial question that follows from these three works is whether the efficient solution of the problem is possible due to the separation constraints or the fact that the robots are unlabeled.

1.2. Contribution

In this paper we study the problem of unlabeled multi-robot motion planning for unit-square robots moving amidst polygonal obstacles. We show that the decision problem, namely, to decide whether a solution exists, is PSPACE-hard. To the best of our knowledge, this is the first hardness proof for the unlabeled case. In fact we consider four variants of the unlabeled problem (see Section 2 for a precise definition) and show that they are all PSPACE-hard. For instance, we show that the seemingly easier version of the problem where only one of the robots is required to reach a certain target position while the other robots function as movable obstacles, is also computationally intractable. We mention that our proofs can be used to show that the labeled variant, again, for unit-square robots, is PSPACE-hard as well, which sets another precedent, as previous hardness results require the robots to be of different shapes (or at least in different orientations); see Hearn and Demaine (2005); Hopcroft et al. (1984); Spirakis and Yap (1984). The various hardness results for multi-robot motion planning are summarized in Table 1. Although our result in itself is negative, we hope that it will motivate research of other variants of the unlabeled problem which may turn out to be polynomially solvable.

Additionally, we settle an open problem introduced by Tromp and Cilibrasi (2005): we prove that *rush hour* is PSPACE-hard for unit-square cars moving amidst polygonal obstacles.

The organization of this paper is as follows. In Section 2 we describe the four variants of the unlabeled problem that will be considered in this paper. In Section 3 the NCL model of computation, which is a key ingredient in our hardness proof, is described. In Section 4 we provide the proofs. In Section 5 we provide PSPACE-hardness proofs for the labeled variant of the problem and for rush hour.

Table 1. Hardness results related to the multi-robot problem.

Contributor	Problem	Complexity	Robots	Workspace
Hopcroft et al. (1984)	Labeled	PSPACE-hard	Rectangles	Rectangle
Spirakis and Yap (1984)	Labeled	Strongly NP-hard	Discs	Simple polygon
Hearn and Demaine (2005)	Labeled	PSPACE-complete	1×2 rectangles	Rectangle
This paper	Unlabeled, labeled	PSPACE-hard	Unit squares	Polygonal obstacles

2. Preliminaries

Let r be a robot operating in a planar workspace $\mathcal{W}$. We denote by $\mathcal{C}(r)$ the *configuration space* of r, and by $\mathcal{F}(r) \subset \mathcal{C}(r)$ the *free space* of r—the collection of all configurations for which the robot does not collide with obstacles. Given $s, t \in \mathcal{F}(r)$, a *path* for r from s to t is a continuous function $\pi : [0,1] \to \mathcal{F}(r)$, such that $\pi(0) = s, \pi(1) = t$.

We say that two robots r, r' are *geometrically identical* if $\mathcal{F}(r) = \mathcal{F}(r')$ for the same workspace $\mathcal{W}$. Let $R = \{r_1, \ldots, r_m\}$ be a set of m geometrically identical robots, operating in a workspace $\mathcal{W}$. We use $\mathcal{F}$ to denote $\mathcal{F}(r_i)$ for any $1 \leqslant i \leqslant m$.

Definition 1. *A collection $C = \{c_1, \ldots, c_m\}$ of m configurations is termed a* multi-configuration*. Such a multi-configuration is* free *if $C \subset \mathcal{F}$, and for every two distinct configurations $c, c' \in C$ it holds that $r(c) \cap r'(c') = \emptyset$, for $r, r' \in R$, where $r(x)$, for $x \in \mathcal{C}$, denotes the area covered by some robot $r \in R$ when placed in x, excluding its boundary, i.e. the robots' boundaries may touch, but they are not allowed to overlap.*

Definition 2. *Given two free multi-configurations, $C = \{c_1, \ldots, c_m\}, C' = \{c'_1, \ldots, c'_m\}$, we say that they are equivalent ($C \equiv C'$) if there exist m paths $\Pi = \{\pi_1, \ldots, \pi_m\}$ that move the m robots from C to C'. More formally, we demand that for every $c \in C$, there exists $1 \leqslant i \leqslant m$ for which $\pi_i(0) = c$; for every $c' \in C'$ there exists $1 \leqslant j \leqslant m$ for which $\pi_j(1) = c'$; for every $\tau \in [0,1]$ the multi-configuration $\Pi(\tau) = \{\pi_1(\tau), \ldots, \pi_m(\tau)\}$ is free.*

Given two equivalent multi-configurations S, T we denote by $\Pi(S,T) = \{\pi_1, \ldots, \pi_m\}$ a collection of m paths that move the robots from S to T by following the restrictions above. Note that Definition 2 requires that every target position will be occupied by *some* robot, in contrast with the classic definition which indicates which robot should reach where.

We define four decision problems that will each be shown to be PSPACE-hard below.

1. Given two free multi-configurations S, T, is it true that $S \equiv T$?
2. Given a free multi-configuration S, and a configuration $t \in \mathcal{F}$, is there a multi-configuration T such that $t \in T$ and $S \equiv T$?
3. Given a free multi-configuration S, a configuration $s \in S$, and another configuration $t \in \mathcal{F}$, is there a multi-configuration T such that $t \in T$, $S \equiv T$ for which there exists $\pi \in \Pi(S,T)$ such that $\pi(0) = s, \pi(1) = t$?
4. Given two configurations $s, t \in \mathcal{F}$, are there two multi-configurations S, T such that $s \in S, t \in T$ and $S \equiv T$?

We will refer to these problems from now on as the *multi-to-multi*, *multi-to-single*, *multi-to-single-restricted*, and *single-to-single* problems, respectively.

For the intuition behind these mathematical definitions we briefly explain two of the problems. The *multi-to-multi* problem corresponds to the standard definition of the unlabeled problem: given two multi-configurations S, T, which represent the start and target positions of the m robots, can the robots move from S to T while avoiding collisions? The *multi-to-single* problem starts with a similar input, only that T is replaced with a single target configuration t: can one of the robots, from those initially positioned in S, reach t, while avoiding collisions. Note that it might be the case that the robots would be required to move collectively in order to ensure that one of them will be able to reach t. This motivates imposing the existence of T, such that $t \in T$ and $S \equiv T$. We also refer the reader to the proof of Theorem 12 for a fuller geometric interpretation of the four definitions.

Note that *multi-to-single* differs from *multi-to-single-restricted* by allowing any robot to reach t. Although the four problems seem to be closely related, it is not clear whether it is possible to reduce one problem to another.

3. Nondeterministic constraint logic

In this section we review the NCL model of computation, and state three decision problems that are based on this model and are shown to be PSPACE-complete (Hearn and Demaine, 2005). An NCL machine is defined by a *constraint* graph $G = (V, E)$, a *weight* function $w : E \to \mathbb{N}$, and a *minimum-flow* constraint $c : V \to \mathbb{N}$.

Definition 3. *A machine state is an orientation $o : E \to \{0,1\}$ such that for every edge $(v, v') \in E$ it holds that $o(v, v') = 1, o(v', v) = 0$, or $o(v, v') = 0, o(v', v) = 1$. An orientation o is valid if for every $v \in V$ the sum of weights of the edges that are directed into v is at least the minimum-flow constraint of the vertex. More formally,*

$$\forall v \in V : \sum_{v' \in N(v)} o(v', v) \cdot w(v', v) \geqslant c(v)$$

where $N(v)$ denotes the set of neighbors of v in G.

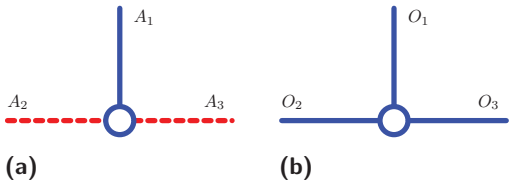


Fig. 1. (a) AND and (b) OR vertices. Red (dashed) edges have a weight of 1 and blue (solid) edges have a weight of 2. The blue (solid) vertex (circle) represents a minimum flow constraint of 2. In (a) A_1 can be directed outward if and only if A_2 and A_3 are both directed inward. In (b) O_1 can be directed outward if and only if any of the other two is directed inward.

A *move* from one orientation to another consists of a reversal of the orientation of a single edge, while maintaining the minimum-flow constraints. Given two orientations o, o' we say that they are *equivalent*, and denote this relation by $o \equiv o'$, if o can be transformed into o' by a series of moves. Given these definitions, the following decision problems are defined by Hearn and Demaine (2005).

1. Given two orientations o_S, o_T, is it true that $o_S \equiv o_T$?
2. Given an orientation o_S and an edge $(v, v') \in E$ is there another orientation o_T such that $o_S \equiv o_T$ and $o_S(\{v, v'\}) \neq o_T(\{v, v'\})$, i.e. the orientation of (v, v') is reversed between o_S and o_T?
3. Let $(v, v'), (u, u') \in E$. Additionally, let $o_{(v,v')}, o_{(u,u')}$ be two orientations for these specific edges. Are there two orientations o_S, o_T such that $o_S \equiv o_T$ and $o_S(v, v') = o_{(v,v')}, o_T(u, u') = o_{(u,u')}$?

These problems are termed *full-to-full*, *full-to-edge* and *edge-to-edge*, respectively. We are interested in a particular setting of the problem where the constraint graph is planar and consists of only two types of vertices, depicted in Figure 1:

- An AND vertex has a minimum-flow constraint of two, and has exactly three incident edges, where one of the edges has a weight of two, and each of the other two edges has a weight of one.
- An OR vertex has a minimum-flow constraint of two and has exactly three incident edges, each with a weight of two.

The following theorem will play a central role in our analysis. Its proof is found in Hearn and Demaine (2005, Theorem 12).

Theorem 4. Full-to-edge, full-to-full, *and* edge-to-edge, *are* PSPACE*-complete, even when the constraints graph is simple, planar and consists of only* AND *and* OR *vertices.*

3.1. Grid-embedded constraint graph

In order to simplify the reduction process in the following sections, we show that given a constraint graph G,

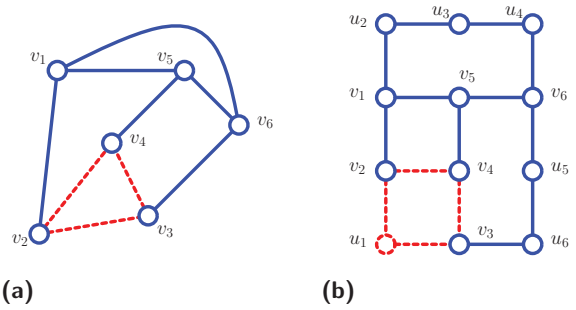


Fig. 2. A grid embedding of a simple planar graph with AND and OR vertices. In (a) we have the original graph $G = (V, E)$, while in (b) we have a grid embedding of it. Recall that blue (solid) edges represent edges with weights of 2, while red (dashed) edges represent weights of 1. Similarly, blue circles represent vertices with min-flow constraints of 2, while red circles represent vertices with min-flow constaints of 1. Note that the embedding preserves the type of the vertices from G, e.g. v_4 is an AND vertex in both cases. In order to make the embedding possible the *connector* vertices $U = \{u_1, \ldots, u_6\}$ were added to H. Note that the minimum-flow constraint of u_1 is 1. The edges of G are represented as paths in H. For instance, $(v_1, v_6) \in E$ is represented by the path $H(v_1, v_6) = \{v_1, u_2, u_3, u_4, v_6\}$ in H.

as described above, it can be transformed into a two-dimensional constraint grid graph H, such that the three NCL decision problems remain PSPACE-hard on H as well (Figure 2). We mention that Hearn and Demaine (2005) use a similar grid embedding, but omit the relevant details. Thus we choose to provide a full description of this process here.

We generate a new constraint graph H whose vertices are grid vertices and edges are grid edges. Each edge of G is transformed into a noncrossing path in H. Such an embedding is possible due to the fact that G is simple and planar. For the purpose of the reduction it suffices to know that such an embedding can be carried out in polynomial time, but we mention that a linear-time algorithm by Liu et al. (1998) exists.

As G is planar and has a degree of three (it is exclusively made of AND and OR vertices), it can be embedded on a planar grid $H = (V_H, E_H)$ which is defined as follows. The set of vertices of H is defined to be $V_H := V \cup U$, where U is an additional set of vertices called *connectors*, and where each $v \in V_H$ corresponds to a point on the grid. Every edge $(v, v') \in E_H$ corresponds to an axis-parallel segment that connects the two points v, v' on the grid. Given two vertices $v, v' \in V$ for which $(v, v') \in E$, we denote by $H(v, v') = (v, u_1, \ldots, u_\ell, v')$ the path in H that corresponds to (v, v').

We also define the weight and capacity functions w_H, c_H, respectively. Each vertex $v \in V$ maintains its original capacity of two from G, that is, $c_H(v) := c(v)$. Let $(v, v') \in E$ and let $u \in U$ such that $u \in H(v, v')$. Then $c_H(u) := w(v, v')$. Additionally, suppose that (u, u') represents an edge on the path $H(v, v')$; then we let $w_H(u, u') := w(v, v')$.

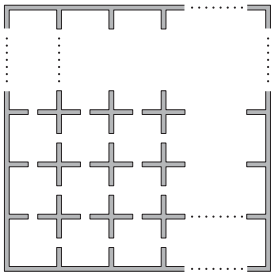


Fig. 3. Placeholders for the gadgets.

Lemma 5. Full-to-edge*,* full-to-full*, and* edge-to-edge*, are* PSPACE*-complete, even for the grid-embedded constraint graph H that consists of only* AND*,* OR *and connector vertices.*

Proof. Every orientation of G can be transformed into an orientation for H, and vice versa. Using this fact the hardness of the these three problems immediately follows from Theorem 4. □

4. From NCL to multi-robot motion planning

In this section we present the reduction from the three NCL problems, which were described in the previous section, to our four unlabeled multi-robot motion planning problems; we will call them unlabeled problems for short. Specifically, we consider the case where the input consists of a grid-embedded constraint graph H, as described in Section 3. Given such a graph H we construct an unlabeled scenario which corresponds to the graph and consists of unit-square robots and polygonal obstacles. We use a grid layout, as depicted in Figure 3, where the cells of this grid are of dimension 5×5 and the walls separating the cells are of thickness $1/2$. Each such cell functions as a placeholder for a gadget which represents and emulates a specific vertex of H. The gadgets are placed according to the positions of their counterpart vertices in H. Note that between every two adjacent cells there is a doorway of width 1 so that an interaction between adjacent gadgets can take place. We may rotate the gadgets depicted below by $90, 180$ or 270 degrees so that gadgets that correspond to two vertices of H that share an edge will share a passage. When two vertices of H share an edge, the corresponding gadgets will share a robot. A similar scheme was employed in Hearn and Demaine (2005), although they used different gadgets as they were interested in showing the hardness of slightly different motion planning problems.

4.1. AND, OR *and connector gadgets*

For each vertex of H we create a gadget that emulates the functionality of this vertex in the NCL machine. For the vertices of U we create a *connector* gadget, while for the vertices of V we create AND and OR gadgets. All the gadgets fit into 5×5 squares (see Figures 4–6) and have either two or three exits through which they connect to other gadgets. Every gadget accommodates several robots and contains polygonal obstacles; the robots are illustrated in purple or green and the obstacles are illustrated in gray. The white regions represent portions of the free workspace. All the robots are placed such that they neither overlap with the obstacles nor with each other, although the boundaries may touch. The AND gadgets also have a point obstacle, illustrated in red (its purpose will be explained below). For an illustration of a connection between two gadgets, see Figure 7.

Every gadget accommodates several unit-square robots which fall into two categories: those that never leave the gadget and those that may penetrate[3] the gadget or leave it to a neighbor gadget. The former are called *vertex* robots (drawn in purple), while the latter are *edge* robots (drawn in green). Edge robots are located at the exits of the gadgets, one robot per exit. As the name suggests, edge robots correspond to the edges of H incident to the vertex. The direction of the edge corresponds to the position of the robot, with respect to the gadget. Specifically, an edge that is directed *inward* corresponds to an edge robot that is located at the exit but does not penetrate the gadget (for example, robot A_1 in Figure 5(a)); an *outward* directed edge corresponds to an edge robot located at the exit such that exactly half of it is located inside the 5×5 square of the gadget (see e.g. robots A_2, A_3 in Figure 5(a)). We will refer to these two positions of the edge robot as *inside* a gadget, and *outside* a gadget, respectively. We note that when an edge robot of one gadget is located outside, it is also inside the adjacent gadget, and vice versa. The inverse relation between the position of the edge robots and the orientation of the edges stems from the fact that we wish to avoid situations where too few edges are directed into a vertex (and thus the minimum-flow constraint is not satisfied), and situations where too many edge robots are inside a gadget (and thus a collision occurs). For example, in the OR gadget in Figure 6 it is not possible for all the three edge robots to be simultaneously inside the gadget, and this ensures that the corresponding OR vertex in the constraint graph will receive a sufficient amount of in-flow.

For simplicity, we only consider configurations of the robots where the center of each robot is located at the vertices of a $1/2 \times 1/2$ grid. We refer to such configurations as *terminal*. For instance, all the robots in Figures 4–6 are placed at terminal configurations. Additionally, the actual terminal configurations are illustrated in Figure 4(b). We also allow a robot to move between two adjacent terminal configurations. The following three lemmas illustrate the relation between the gadgets and the vertices of H. Their proofs are straightforward and therefore omitted.

Lemma 6. *Connector gadgets correspond to connector vertices in H, i.e. one of the two edge robots can be inside, only if the other edge robot is outside.*

Proof. See Figures 4(a) and (b). □

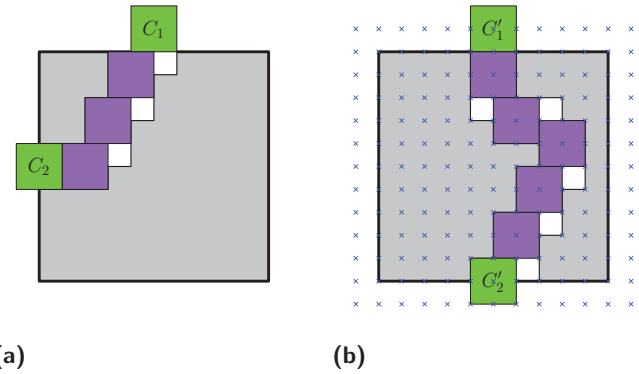


Fig. 4. Connector gadgets: C_1 (similarly for C_1') can be inside only if C_2 (similarly for C_2') is outside, and vice versa. The blue crosses in (b) represent terminal configurations, which are defined in Section 4.1.

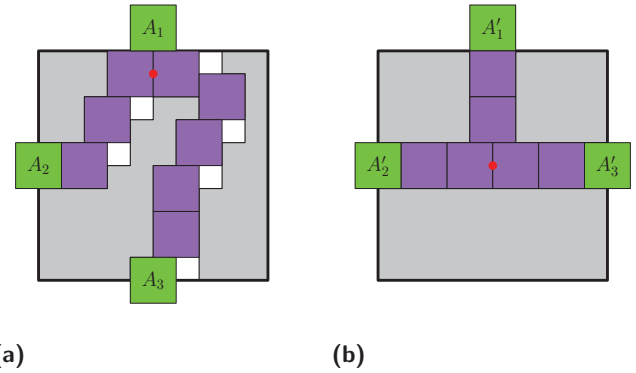


Fig. 5. AND gadgets. A_1 (similarly for A_1') can move inside if A_2, A_3 (similarly for A_2', A_3') are both outside, and vice versa. The purpose of the point obstacle, depicted in red, is to restrict the movement of the edge robot A_1 so that it will only be able to alternate between at most two terminal configurations.

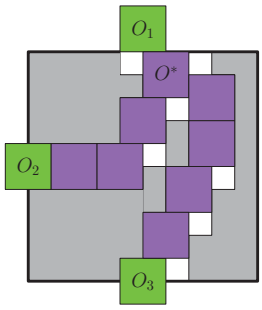


Fig. 6. OR gadget. The three edge robots O_1, O_2, O_3 cannot be inside simultaneously.

Lemma 7. AND *gadgets correspond to* AND *vertices in H, with* A_2, A_3, A_2', A_3' *corresponding to* 1*-weight edges, and* A_1, A_1' *corresponding to* 2*-weight edges; e.g.* A_1 *can move inside the gadget only if* A_2, A_3 *are both outside.*

Proof. See Figures 5(a) and (b). □

Lemma 8. *The* OR *gadgets correspond to* OR *vertices in H, i.e. one of the edge robots can move inside only if at least one of the other edge robots is outside.*

Proof. See Figure 6. □

4.2. Unlabeled motion planning with gadgets

We finalize the details of our reduction and prove its correctness. We first show that the structure of the gadgets that we have produced is very restrictive and allows a limited set of movements for the robots, so that robots cannot wander between different gadgets.

Lemma 9. *Each edge robot can be in at most two distinct terminal configurations.*

Proof. We show that for every possible connection of two gadgets the edge robot can be in at most two terminal configurations. We only consider here the combination of the second AND gadget (Figure 5(b)) and the OR gadget (Figure 6). The other combinations can be treated analogously. Specifically, we consider the case where the connection is made through the edge robot $A_3' = O_2$ (colored in orange), as illustrated in Figure 7. Notice that robot $A_3' = O_2$ is stuck between the robots D and E: D is blocked to the left by the red point obstacle (Figure 7(a)); E is blocked to the right by an obstacle (Figure 7(b)). From this observation we conclude that every edge robot can either be inside or outside, and not in any other terminal configuration. □

Lemma 10. *Each vertex robot can be in at most two distinct terminal configurations, save robot* O^* *in the* OR *gadget, which can be in at most three terminal configurations.*

Proof. First, note that every edge robot can move either horizontally or vertically. Let us consider for example robot $A_3' = O_2$ in Figure 7, which can only move horizontally. Since this robot can only move between two terminal configurations (Lemma 9), the two vertex robots that are directly to its right, cannot move further left than where they appear in Figure 7(a). Additionally, robot E is bounded from the right by an obstacle which does not allow it to move 2 further to the right than it appears in Figure 7(b). A similar reasoning can be applied to all the other vertex robots. As for robot O^*, consider its position in Figure 7(a). It cannot go up as there is an obstacle there. If O_3 leaves the gadget it can move a half step right, and if O_2 leaves the gadget it can move a half step down. Notice however, that it cannot move simultaneously down and right since it will collide with the robot that is immediately to its right. □

The following lemma implies that our motion planning scenario is so tight that each valid multi-configuration can be interpreted in a single way as an assignment of terminal configurations to the robots.

Lemma 11. *Given a specific terminal configuration, there is at most one robot that can be in it.*

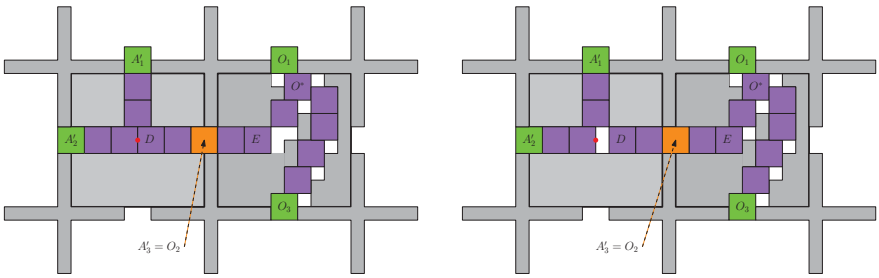


Fig. 7. Illustration for Lemma 9 and a depiction of two gadgets that are connected through an edge robot. In both figures the same AND gadget and OR gadgets, which share an edge robot $A'_3 = O_2$, are illustrated. These two gadgets simulate an AND vertex and an OR vertex which share an edge. For each edge robot A'_i in the AND gadget, denote by a'_i the corresponding edge in the AND vertex. Similarly, denote by o_i the edge corresponding to the edge robot O_i in the vertex. Note that a'_3 and o_2 represent the same edge. In (a) the gadget represents an orientation in which a'_1 is directed inward, a'_2 outward, a'_3 outward with respect to the AND vertex. As a consequence, o_2 is directed inward with respect to the OR vertex. Additionally, o_1 is directed inward, while o_3 is directed outward. In (b) a similar orientation is depicted, with the exception that a'_3 is now directed inward while o_2 is directed outward.

Proof. Follows from Lemmas 9, 10. □

We mention that a similar lemma can be proved for the gadgets used by Hearn and Demaine (2005) in their hardness proof of the labeled variant of the *multi-to-multi* problem. Since their proof uses two different types of robots, their result can also be interpreted as a PSPACE-hardness proof for the 2-color multi-robot motion planning problem (see Solovey and Halperin, 2014), which consists of two groups where the robots within each group are identical and interchangeable.

We return to the PSPACE-hardness proof of the four unlabeled problems discussed in our paper.

Theorem 12. *The problems* multi-to-multi, multi-to-single, multi-to-single-restricted *and* single-to-single, *for unit-square robots translating amidst polygonal obstacles in the plane are all* PSPACE*-hard.*

Proof. We describe reductions from the three NCL-model decision problems to our four unlabeled problems. Specifically we show the following reductions. □

- The *full-to-full* problem, which is concerned with checking whether one NCL machine orientation can be transformed to another, is reduced to the *multi-to-multi* unlabeled problem, which is concerned with deciding whether a collection of robots can be moved between two multi-configurations.
- The *full-to-edge* problem, which is concerned with the existence of an orientation that can be reached from a given orientation where a direction of a specific edge is flipped, is reduced to the unlabeled *multi-to-single* and *multi-to-single-restricted* problems that are concerned with moving an arbitrary or a specific robot (respectively) to a given destination configuration, when a starting multi-configuration is specified for all the robots.
- The *edge-to-edge* problem, which is concerned with the existence of two orientations, where one can be transformed to another, such that the two orientations have an opposite direction for a given edge, is reduced to the unlabeled *single-to-single* problem, which is concerned with the existence of two multi-configurations that are equivalent, where each of the two multi-configurations contains a specific configuration that is given as input (where the configuration is not assigned to a specific robot).

We use the same reduction for all three cases. Only the analysis slightly differs. Given a grid-embedded constraint graph H we generate a scenario for the unlabeled problem as we described above, by placing gadgets corresponding to vertices and connecting gadgets according to the connections in H.

We first note that given an orientation for H it can be transformed into a valid multi-configuration consisting only of terminal configurations, where the directions of edges in H induce configurations for the respective edge robots, and these in turn induce configurations for the vertex robots. In the other direction, given a valid multi-configuration for the robots that consists of only terminal configurations, a valid orientation for H can be easily generated by considering only the positions of edge robots. Note that by Lemma 9 every edge robot can be in at most two terminal configurations that represent two opposite directions of the same edge of H.

We first prove the hardness of the *multi-to-multi* problem by a reduction from the *full-to-full* problem. Let o_S, o_T be two orientations of H, and denote by S, T the two free multi-configurations induced by them. It is clear that if $o_S \equiv o_T$ then also $S \equiv T$. Now, suppose that $S \equiv T$. Notice that in order to show that this implies that $o_S \equiv o_T$ we need to prove the existence of a solution $\Pi(S, T)$ where no two edge robots move at a given time. More accurately, we need

to show that there exists a solution in which every edge robot is in transit between two terminal configurations, only when all the other edge robots are stationary in terminal configurations. We consider for example the AND gadget (Figure 5(a)) and show that each simultaneous movement of edge robots, where several edge robots are located simultaneously at non-terminal configurations, can be carried out in a sequential manner as well. We treat the various combinations of robots moving simultaneously in and out of the gadget. If both A_2 and A_3 move inside then A_1 must be out, and therefore the former two robots do not depend on each other in order to make their move. Now suppose that A_2 and A_3 simultaneously move outside. This means that each of the two gadgets, to which A_2 and A_3 are entering, already moved other edge robots, and vertex robots, so that the entrance of A_2, A_3 will be possible. Thus the fact that A_2 can leave the gadget does not depend on the fact that A_3 leaves the gadget, and vice versa. Therefore, we can move A_2, A_3 in a sequential manner. Therefore, a solution $\Pi(S, T)$ as required always exists, and in the case that $S \equiv T$ it also follows that $o_S \equiv o_T$.

We now proceed to prove the hardness of the *multi-to-single* problem by a reduction from the *full-to-edge* problem. Recall that *full-to-edge* consists of an orientation o_S and edge $e \in E_H$. The question is whether there exists another orientation o_T such that $o_S \equiv o_T$ and $o_S(e) \neq o_T(e)$, that is, the direction of e in the two configurations is reversed. Denote by S the multi-configuration induced by o_S, and by $s \in S$ the terminal configuration that corresponds to the edge e in the direction $o_S(e)$. We now ask whether there exists a multi-configuration T such that $S \equiv T$ such that there exists $t \in T$ that corresponds to e in the opposite direction. Now that we have defined the components of our *multi-to-single* problem it is clear that if there exists an orientation o_T as required, then its induced multi-configuration satisfies the conditions of the *multi-to-single* problem. The opposite direction follows similarly to the *multi-to-multi* proof above.

The difference between *multi-to-single* and *multi-to-single-restricted* is that in the latter we require that a specific robot will move to a specific target configuration. Note that our reduction for *multi-to-single* holds here as well, since a selection of a specific edge of H induces the selection of a specific robot that has to move between two configurations (see Lemma 11). The hardness of the *single-to-single* problem by a reduction from the *edge-to-edge* problem can be proved in a manner similar to the previous three cases.

5. Extensions to other problems

In this section we mention two more problems that can be shown to be PSPACE-hard, using the machinery that we introduced in the previous sections.

5.1. Labeled multi-robot motion planning

Our construction for the hardness proof of the unlabeled problem can also be applied *as-is* to the labeled variant of the *multi-to-multi* problem. In this case, each robot r_i is assigned with specific start and target configurations s_i and t_i, respectively, and the goal is to move each r_i from s_i to t_i while avoiding collisions with robots and obstacles. We note that the construction of the proof uses exactly the same gadgets as in the unlabeled case.

Theorem 13. *Labeled multi-robot motion planning for unit-square robots moving amidst polygonal obstacles is* PSPACE*-hard.*

Proof. We follow the same reasoning as for the *full-to-full* to *multi-to-multi* reduction (Theorem 12) and add that due to Lemma 11 given a placement of the robots induced by the start multi-configuration S, the target multi-configuration can be viewed as an assignment of a specific target for every robot. Specifically, given an initial assignment s_i for robot r_i, there is at most one configuration in T to which it can move (see Lemma 11). □

5.2. Rush hour

Recall that the problem of *Rush Hour* consists of evacuating a car from a parking lot, by clearing a route blocked by other cars. The cars are restricted to moving forward or backward, with respect to the direction of the car's front. It can be viewed as a special case of the multi-to-single-restricted problem: each robot r_i of the m robots is given a specific start position, and one of the robots, r^*, is also given a target position, and the goal is to move r^* to its destination, while avoiding collisions with the other robots and obstacles. What distinguishes this problem from those that we have dealt with so far is the following restriction: each robot has a specific orientation in which it is allowed to move (i.e. horizontally or vertically, but never both). For instance, a horizontal robot that starts in the position $(x, y) \in \mathbb{R}^2$ can only move along the line $\{(x', y) \,|\, x' \in \mathbb{R}\}$.

Note that each of the connector (Figure 4) and the AND (Figure 5) gadgets imposes a specific orientation on each of the robots that reside therein. For example, the second AND gadget (Figure 5(a)) forces the robots A'_2, A'_3, and all the robots in between the two, to move horizontally, whereas the remaining three robots can move only vertically. Thus, we will use those gadgets in the construction of the hardness proof of rush hour, unchanged.

Unfortunately, the original OR gadget (Figure 6) is not suitable for this problem: robot O^* should move horizontally *and* vertically in order that the gadget will properly emulate an NCL-OR vertex. As a workaround, we devise an alternative OR gadget, called OR_{rush} (see Figure 8, in which the arrows indicate the orientation of the robots).

Lemma 14. *The* OR_{rush} *gadgets correspond to NCL-*OR *vertices, i.e. one of the edge robots can move inside if and only if at least one of the other edge robots is outside.*

Proof. In order to let O_1 enter the gadget, O^* must touch the wall directly to its left (or right). To make this motion,

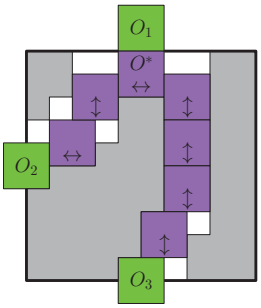


Fig. 8. OR_{rush} gadget for the rush-hour problem.

O_2 must leave the gadget so that the two edge robots to its right clear the way to O^*. □

We now proceed to the concluding theorem.

Theorem 15. *Rush hour for unit-square robots moving amidst polygonal obstacles is* PSPACE*-hard.*

Proof. Unlike previous gadgets, in which every edge robot can only be half-way inside or entirely outside the gadget, in the OR_{rush} gadget O_1 can also reside entirely inside the gadget. Fortunately, this additional "slack" cannot be exploited as any other edge robot can still only be half-way inside or entirely outside a gadget. This can be validated for each combination of an OR_{rush} gadget and another gadget (similarly to Lemma 9). □

6. Concluding remarks

In this paper we studied the problem of motion planning of multiple unlabeled unit-square robots in an environment cluttered with polygonal obstacles. We proved that four variants of this problem are PSPACE-hard. While our result in itself is negative, we hope that it will motivate research of other variants of the unlabeled problem which may turn out to be polynomially solvable.

Acknowledgements

A preliminary version of this paper was presented at the *Robotics: Science and Systems* (RSS) 2015 conference, where it also won the Best Student Paper Award.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: The work was supported by the Israel Science Foundation (grant number 825/15), by the German-Israeli Foundation (grant number 1150-82.6/2011), and by the Hermann Minkowski–Minerva Center for Geometry at Tel Aviv University. Kiril Solovey is also supported by the Clore Israel Foundation. His travel to RSS 2015 was supported by the International Travel Scholarship, Ministry of Science, Technology and Space, Israel (grant number 3-1154).

Notes

1. We only briefly mention this area of research, as it is beyond the scope of this paper.
2. For simplicity of presentation, log factors in the complexity of the algorithm are omitted, and hence the $\tilde{O}$ notation is used.
3. Here "penetrate" signifies that the robot intersects the 5×5 square of the gadget.

References

Adler A, de Berg M, Halperin D, et al. (2015) Efficient multi-robot motion planning for unlabeled discs in simple polygons. *IEEE Transactions on Automation Science and Engineering* 12(4): 1309–1317.

Buchin K and Buchin M (2012) Rolling block mazes are PSPACE-complete. *Journal of Information Processing* 20(3): 719–722.

Buchin K and Gerrits DHP (2013) Dynamic point labeling is strongly PSPACE-complete. In: *International symposium on algorithms and computation*. pp.262–272.

Flake GW and Baum EB (2002) Rush hour is PSPACE-complete, or "Why you should generously tip parking lot attendants". *Theoretical Computer Science* 270(1-2): 895–911.

Hearn RA and Demaine ED (2005) PSPACE-completeness of sliding-block puzzles and other problems through the nondeterministic constraint logic model of computation. *Theoretical Computer Science* 343(1-2): 72–96.

Hearn RA and Demaine ED (2009) *Games, puzzles and computation*. AK Peters.

Hirsch S and Halperin D (2002) Hybrid motion planning: Coordinating two discs moving among polygonal obstacles in the plane. In: *Workshop on the algorithmic foundations of robotics (WAFR)*, 15–17, pp.239–255. Springer, 2004.

Holzer M and Jakobi S (2012) On the complexity of rolling block and Alice mazes. In: Kranakis E, Krizanc D and Luccio F (eds) *Fun With Algorithms*. pp.210–222 Berlin, Springer: Heidelberg.

Hopcroft JE, Schwartz JT and Sharir M (1984) On the complexity of motion planning for multiple independent objects; PSPACE-hardness of the "Warehouseman's problem". *International Journal of Robotics Research* 3(4): 76–88.

Ito T, Kamiski M and Demaine ED (2012) Reconfiguration of list edge-colorings in a graph. *Discrete Applied Mathematics* 160(15): 2199–2207.

Kloder S and Hutchinson S (2006) Path planning for permutation-invariant multi-robot formations. *International conference on robotics and automation*, 22(4): 650–665.

Krontiris A, Shome R, Dobson A, et al. (2014) Rearranging similar objects with a manipulator using pebble graphs. In: *14th IEEE–RAS international conference on humanoid robots*, Madrid, Spain, 18–20 Nov, pp.1081–1087. IEEE.

Liu Y, Morgana A and Simeone B (1998) A linear algorithm for 2-bend embeddings of planar graphs in the two-dimensional grid. *Discrete Applied Mathematics* 81(1–3): 69–91.

Salzman O, Hemmer M and Halperin D (2015) On the power of manifold samples in exploring configuration spaces and the dimensionality of narrow passages. *IEEE Transactions on Automation Science and Engineering* 12(2): 529–538.

Sánchez-Ante G and Latombe J (2002) Using a PRM planner to compare centralized and decoupled planning for multi-robot

systems. In: *International conference on robotics and automation (ICRA)*, Washington, DC, USA, 11–15 May pp.2112–2119. IEEE.

Schwartz JT and Sharir M (1983) On the piano movers problem: III. Coordinating the motion of several independent bodies. *International Journal of Robotics Research* 2(3): 46–75.

Sharir M and Sifrony S (1991) Coordinated motion planning for two independent robots. *Annals of Mathematics and Artificial Intelligence* 3(1): 107–130.

Solovey K and Halperin D (2014) *k*-Color multi-robot motion planning. *International Journal of Robotic Research* 33(1): 82–97.

Solovey K, Salzman O and Halperin D (2016) Finding a needle in an exponential haystack: Discrete RRT for exploration of implicit roadmaps in multi-robot motion planning. *International Journal of Robotic Research* 35(5): 501–513.

Solovey K, Yu J, Zamir O, et al. (2015) Motion planning for unlabeled discs with optimality guarantees. In: *Robotics: Science and systems XI*. Rome: Sapienza University of Rome, Italy, 13–15 July.

Spirakis PG and Yap CK (1984) Strong NP-hardness of moving many discs. *Information Processing Letters* 19(1): 55–59.

Švestka P and Overmars MH (1998) Coordinated path planning for multiple robots. *Robotics and Autonomous Systems* 23: 125–152.

Tromp J and Cilibrasi R (2005) Limits of rush hour logic complexity. *CoRR* abs/cs/0502068.

Turpin M, Michael N and Kumar V (2014) Concurrent assignment and planning of trajectories for multiple robots. *International Journal of Robotics Research*, 33(1): 98–112.

Wagner G and Choset H (2011) M*: A complete multirobot path planning algorithm with performance bounds. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 25–30 September, pp. 3260–3267. Available at http://dx.doi.org/10.1109/IROS.2011.6095022.

Yap CK (1984) Coordinating the motion of several discs. Technical report, Courant Institute of Mathematical Sciences, New York.

3

Efficient Multi-Robot Motion Planning for Unlabeled Discs in Simple Polygons

Efficient Multi-Robot Motion Planning for Unlabeled Discs in Simple Polygons

Aviv Adler, Mark de Berg, Dan Halperin, *Fellow, IEEE*, and Kiril Solovey

***Abstract*—We consider the following motion-planning problem: we are given m unit discs in a simple polygon with n vertices, each at their own start position, and we want to move the discs to a given set of m target positions. Contrary to the standard (labeled) version of the problem, each disc is allowed to be moved to any target position, as long as in the end every target position is occupied. We show that this unlabeled version of the problem can be solved in $O(m^2 + mn)$ time, assuming that the start and target positions are at least some minimal distance from each other. This is in sharp contrast to the standard (labeled) and more general multi-robot motion planning problem for discs moving in a simple polygon, which is known to be strongly NP-hard.**

***Note to Practitioners*—Operating low-cost robots in large groups is becoming a widespread practice. Planning the motion of a group of robots among obstacles is known to be computationally hard. In particular, the worst-case running time of such algorithms grows exponentially in the number of robots. We present a very efficient algorithm for planning collision-free paths for a special case: a set of interchangeable disc (Roomba-like) robots moving in a simple-polygon environment. By interchangeable we mean that we do not care which robot reaches a specific target position, as long as all target positions are occupied at the end of the motion. Our technique is complete and as such it is guaranteed to find a solution if one exists, or report that none exist otherwise. To obtain the efficient solution, we assume a certain minimum separation between the robots in their initial and target positions—we believe that this is the crucial assumption needed to obtain an efficient solution.**

***Index Terms*—Computational geometry, motion planning, multi-robot motion planning.**

Manuscript received December 02, 2014; revised July 14, 2015; accepted August 12, 2015. Date of publication August 26, 2015; date of current version October 02, 2015. This paper was recommended for publication by Associate Editor T. Bretl and Editor F. A. van der Stappen upon evaluation of the reviewers' comments. The work of A. Adler was done while at Tel Aviv University, through a Melvin M. Goldberg Fellowship for Research, Israel. The work of M. de Berg supported was by the Netherlands Organization for Scientific Research (NWO) under Grant 024.002.003. The work of D. Halperin and K. Solovey was supported in part by the Seventh Framework Program for Research of the European Commission under FET-Open Grant 255827 (CGL—Computational Geometry Learning), by the Israel Science Foundation under Grant 1102/11, by the German-Israeli Foundation under Grant 1150-82.6/2011, and by the Hermann Minkowski–Minerva Center for Geometry, Tel Aviv University .

A. Adler is with the Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: aadler1561@gmail.com).

M. de Berg is with the Department of Mathematics and Computing Science, TU Eindhoven, 5612 AZ, Eindhoven, The Netherlands (e-mail: m.t.d.berg@tue.nl).

D. Halperin and K. Solovey are with the Blavatnik School of Computer Science, Tel-Aviv University, Tel-Aviv 69978, Israel (e-mail: danha@post.tau.ac.il; kirilsol@post.tau.ac.il).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TASE.2015.2470096

I. Introduction

THE MULTI-ROBOT motion-planning problem is to plan the motions of several robots operating in a common workspace. In its most basic form, the goal is to move each robot from its start position to some designated target position, while avoiding collision with obstacles in the environment and with other robots. Besides its obvious relevance to robotics, the problem has various other applications, for example, in the design of computer games or crowd simulation. Multi-robot motion planning is a natural extension of the single-robot motion planning problem, but it is much more complex due to the high number of *degrees of freedom* that it entails, even when the individual robots are as simple as discs.

A. Related Work

One of the first occurrences of the multi-robot motion-planning problem in the computational-geometry literature can be found in the series of papers on the *Piano Movers' Problem* by Schwartz and Sharir. They first considered the problem in a general setting [1] and then narrowed it down to the case of disc robots moving amidst polygonal obstacles [2]. In the latter work, an algorithm was presented for the case of two and three robots, with running time of $O(n^3)$ and $O(n^{13})$, respectively, where n is the complexity of the workspace. Later, Yap [3] used the *retraction method* to develop a more efficient algorithm, which runs in $O(n^2)$ and $O(n^3)$ time for the case of two and three robots, respectively. Several years afterwards, Sharir and Sifrony [4] presented a general approach based on *cell decomposition*, which is capable of dealing with various types of robot pairs and which has a running time of $O(n^2)$. Moreover, several techniques that reduce the effective number of degrees of freedom of the problem have been proposed [5], [6].

When the number of robots is no longer a fixed constant, the multi-robot motion-planning problem becomes hard. Hopcroft *et al.* [7] showed that even in the relatively simple setting of n rectangular robots moving in a rectangular workspace, the problem is already PSPACE-hard. Moreover, Spirakis and Yap [8] showed that the problem is strongly NP-hard for disc robots in a simple polygon.

In recent years, multi-robot planning has attracted a great deal of attention from the robotics community. This can be mainly attributed to two reasons. First, it is a problem of practical importance. Second, the emergence of the sampling-based techniques, which are relatively easy to implement, yet are highly effective, have made the problem relevant. These techniques attempt to capture the connectivity of the configuration space through random sampling [9], [10]. Although sampling-based algorithms are usually incomplete—they are not guaranteed to

find a solution—they tend to be very efficient in practice. Hence, they are considered the method-of-choice for motion-planning problems that involve many degrees of freedom. While sampling-based tools for a single robot can be applied directly to the multi-robot problem by considering the group of robots as one large *composite robot* [11], there is a large body of work that attempts to exploit the unique properties of the multi-robot problem [12]–[16].

The aforementioned results deal exclusively with the classical formulation of the multi-robot problem, where the robots are distinct and every robot is assigned a specific target position. The *unlabeled* variant of the problem, where all the robots are assumed to be identical and thus interchangeable, was first considered by Kloder and Hutchinson [17], who devised a sampling-based algorithm for the problem. Recently, a generalization of the unlabeled problem—the k-color motion-planning problem—has been proposed, in which there are several groups of interchangeable robots [18]. Turpin *et al.* [19] considered a special setting of the unlabeled problem with disc robots, namely, where the collection of free configurations surrounding every start or target position is star-shaped. This condition allows them to devise an efficient algorithm that computes a solution in which the maximum path length is minimized. Unfortunately the star-shapedness condition is quite restrictive and, in general, it will not be satisfied. More recently, Solovey *et al.* [20] studied a slightly different setting of the problem in which the start and target positions are assumed to be separated by a distance of at least four (where the radius of every robot is 1), and the area surrounding each such position is obstacle free. They introduced an algorithm which returns a solution whose total length, i.e., the sum of lengths of the individual paths, is near optimal. The running time[1] of the algorithm is $\tilde{O}(m^4 + m^2n^2)$, where m is the number of robots and n represents the complexity of the workspace. Additionally, the cost of the returned solution is at most $\mathrm{OPT} + 4m$, where OPT is the cost of the optimal solution.

Other related work includes papers that study the number of moves required to move a set of discs between two sets of positions in an unbounded workspace, when a move consists of sliding a single disc—see, for example, the paper by Bereg *et al.* [21] which provides upper and lower bounds for the unlabeled case, or the paper by Dumitrescu and Jiang [22] who show that deciding whether a collection of labeled or unlabeled discs can be moved between two sets of positions within k steps is NP-hard. We mention the problem of pebble motions on graph, which can be considered as a discrete variant of the multi-robot motion planning problem. In this problem, pebbles need to be moved from one set of vertices of a graph to another, while following a certain set of rules—see, for example [23]–[28]. Finally, we mention the field of *swarm robotics*, which consists of performing various motion tasks with a large group of primitive robots—see, e.g., [29] and [30].

B. Our Contribution

Surprisingly, the unlabeled version of the multi-robot motion-planning problem has hardly received any attention in the computational-geometry literature. Indeed, we do not know of any papers that solve the problem in an exact and complete manner, except in two settings studied by Turpin *et al.* [19] and Solovey *et al.* [20] that we mentioned above. We therefore study the following basic variant of the problem: given m unit-radius discs (or unit discs, in short) in a simple polygon with n vertices, each at their own start position, and m target positions, find collision-free motions for the discs such that at the end of the motions each disc occupies a target position. We make the additional assumption that the given start and target positions are well-separated. More precisely, any two of the given start and target positions should be at distance at least 4 from each other. Notice that we only assume this extra separation between the robots in their static initial and goal placements; we do not assume any extra separation (beyond noncollision) between a robot and the obstacles, nor do we enforce any extra separation between the robots during the motion. Even this basic version of the problem turns out to have a rich structure and poses several difficulties and interesting questions.

By carefully examining the various properties of the problem we show how to transform it into a discrete pebble-motion problem on graphs.[2] A solution to the pebble problem, which can be generated with rather straightforward techniques, can then be transformed back into a solution to our continuous motion-planning problem. Using this transformation, we are able to devise an efficient algorithm whose running time is $O(m^2 + mn)$, where m is the number of robots and n is the complexity of the workspace. As already mentioned, this is in sharp contrast to the standard (labeled) and more general multi-robot motion planning problem for discs moving in a simple polygon, which is known to be strongly NP-hard [8]. It should also be juxtaposed with a recent result which shows that the unlabeled problem for square robots amid polygonal obstacles is PSPACE-hard, when no separation constraints are imposed [31].

II. Preliminaries

We consider the problem of m indistinguishable unit-disc robots moving in a simple polygonal workspace $\mathcal{W} \subset \mathbb{R}^2$ with n edges. We define $\mathcal{O} \triangleq \mathbb{R}^2 \setminus \mathcal{W}$ to be the complement of the workspace, and we call $\mathcal{O}$ the *obstacle space*. Since our robots are discs, a placement of a robot is uniquely specified by the location of its center. Hence, we will sometimes refer to points $x \in \mathcal{W}$ as *configurations*, and we will say that a robot is *at configuration* x when its center is placed at the point $x \in W$. For given $x \in \mathbb{R}^2$ and $r \in \mathbb{R}_+$, we define $\mathcal{D}_r(x)$ to be the open disc of radius r centered at x.

We consider the unit-disc robots to be open sets. Thus a robot avoids collision with the obstacle space if and only if its center is at distance at least 1 from $\mathcal{O}$, that is, when it is at a configuration

[1]The notation $\tilde{O}$ indicates that log factors are omitted.

[2]We mention that a similar approach was taken in [18] in the context of a sampling-based method, which generates a sequence of pebble graphs by sampling the joint (high-dimensional) configuration space. However, it should be emphasized that our current algorithm is *complete*, i.e., guaranteed to find a solution if one exists, or report that none exists, otherwise. Additionally, it has a polynomial running time. In contrast, the technique described in [18] is unable to detect situations in which a solution does not exist. Moreover, its running time can be exponential in the number of robots or the complexity of the environment. On the positive side, it can deal with various types of robots and does not require that the start and target positions will be well-separated, as we do in the current paper.

located in the *free space* $\mathcal{F} \triangleq \{x \in \mathbb{R}^2 : \mathcal{D}_1(x) \cap \mathcal{O} = \emptyset\}$. We require the robots to avoid collisions with each other, so if a robot is at configuration x then no other robot can be at a configuration $y \in \mathrm{Int}(\mathcal{D}_2(x))$; here $\mathrm{Int}(X)$ denotes the interior of the set X. Furthermore, the notation $\partial(X)$ will be used to refer the boundary of X. We call $\mathcal{D}_2(x)$ the *collision disc* of the configuration x.

Besides the simple polygon $\mathcal{W}$ forming the workspace, we are also given sets $S = \{s_1, s_2, \ldots, s_m\}$ and $T = \{t_1, t_2, \ldots, t_m\}$, such that $S, T \subset \mathcal{F}$. These are, respectively, the sets of *start* and *target* configurations of our m identical disc robots. We assume that the configurations in S and T are *well-separated*:

For any two distinct configurations $x, y \in S \cup T$, we have $\|x - y\| \geqslant 4$.

The problem is now to plan a collision-free motion for our m unit-disc robots such that each of them starts at a configuration in S and ends at a configuration in T. Since the robots are indistinguishable (or: *unlabeled*), it does not matter which robot ends up at which target configuration. Formally, we wish to find paths $\pi_i : [0,1] \to \mathcal{F}$, for $1 \leqslant i \leqslant m$, such that $\pi_i(0) = s_i$ and $\bigcup_{i=1}^{m} \pi_i(1) = T$. Additionally, we require that the robots do not collide with each other: for every $1 \leqslant i \neq j \leqslant m$ and every $\xi \in (0,1)$, we require $\mathcal{D}_1(\pi_i(\xi)) \cap \mathcal{D}_1(\pi_j(\xi)) = \emptyset$. Note that the requirement that the robots do not collide with the obstacle space $\mathcal{O}$ is implied by the paths π_i being inside the free space $\mathcal{F}$.

III. Basic Properties of the Free Space

Recall that the free space $\mathcal{F} \subset \mathcal{W}$ is the set of configurations at which a robot does not collide with the obstacle space. The free space may consist of multiple connected components. We denote these components by $F_1, \ldots, F_q$, where q is the total number of components. For any $i \in \{1, 2, \ldots, q\}$, we let $S_i \triangleq S \cap F_i$ and $T_i \triangleq T \cap F_i$. We assume from now on that $|S_i| = |T_i|$ for all $1 \leqslant i \leqslant q$—if this is not the case, then the problem instance obviously has no solution—and we define $m_i \triangleq |S_i| = |T_i|$ to be the number of robots in F_i.

Before we proceed, we need one more piece of notation. For any $x \in \mathcal{W}$, we define $\mathrm{obs}(x)$, the *obstacle set* of x, as $\mathrm{obs}(x) \triangleq \{y \in \mathcal{O} : \|x - y\| < 1\}$. In other words, $\mathrm{obs}(x)$ contains the points in the obstacle space overlapping with $\mathcal{D}_1(x)$. Note that $\mathrm{obs}(x) = \emptyset$ for $x \in \mathcal{F}$.

In the remainder of this section, we prove several crucial properties of the free space, which will allow us to transform our problem to a discrete pebble problem. We start with some properties of individual components F_i, and then consider the interaction between robots in different components.

A. Properties of a Single Connected Component of $\mathcal{F}$

We start with a simple observation, for which we provide a proof for completeness.

Lemma 1: Each component F_i is simply connected.

Proof: Suppose for a contradiction that F_i contains a hole. Then, $\mathrm{Compl}(\mathcal{F}) \triangleq \mathbb{R}^2 \setminus \mathcal{F}$, the complement of the free space, has multiple connected components. One of these is $C_{\mathcal{O}}$, the unbounded component containing $\mathcal{O}$. Let C be another component of $\mathrm{Compl}(\mathcal{F})$, and let $x \in C$. Since $x \notin \mathcal{F}$, there is a point $y \in \mathcal{O}$ with $\|x - y\| < 1$. But then $\|x' - y\| < 1$ for any point x' on the segment $\overline{xy}$, which implies $\overline{xy} \subset \mathrm{Compl}(\mathcal{F})$ and thus contradicts that C and $C_{\mathcal{O}}$ are different components. □

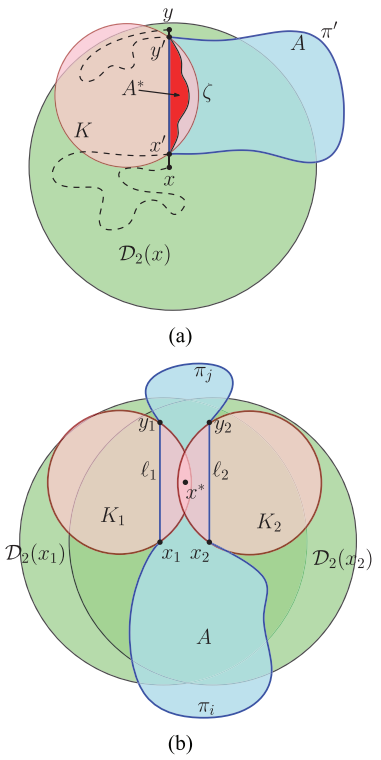


Fig. 1. (a) An illustration of Lemma 2. The disc $\mathcal{D}_2(x)$ is drawn in green. The closed curve λ, which consists of the curve π' and the straight-line $\overline{x'y'}$, is drawn in blue, and A represents the area that is bounded by λ. The disc K of radius 1 that touches x', y' is drawn in pink. Note that the area A^*, which is drawn in red, is contained in A. The dashed black lines represent $\pi \setminus \pi'$. (b) An illustration of Lemma 3, and in particular, the case where $A_1^* \cap A_2^* \neq \emptyset$. For simplicity of presentation, we assume that $\ell_1 = \overline{x_1 y_1}$ and $\ell_2 = \overline{x_2 y_2}$.

Now, consider any x in $\mathbb{R}^2$. Recall that $\mathcal{D}_2(x)$ denotes the collision disc of x, that is, $\mathcal{D}_2(x)$ is the set of all configurations y for which another robot placed at y collides with a robot at x. We now define $D^*(x)$ to be the part of $\mathcal{D}_2(x)$ that is in the same free-space component as x, that is, $D^*(x) \triangleq \mathcal{D}_2(x) \cap F_i$, where F_i is the free-space component such that $x \in F_i$.

The following three lemmas constitute the theoretical basis on which the correctness and efficiency of our algorithm relies.

Lemma 2: For any $x \in \mathcal{F}$, the set $D^*(x)$ is connected.

Proof: Assume for a contradiction that $D^*(x)$ is not connected. Let F_i be the free-space component containing x. Since by definition $x \in D^*(x)$, we can find some $y \in D^*(x)$ that is in a different connected component of $D^*(x)$ from x. Since $y \in D^*(x) \subset \mathcal{D}_2(x)$, the distance between x and y is at most 2. Hence, any point on the line segment $\overline{xy}$ is within a distance of 1 of either x or y. Since $x, y \in F_i$, we know that $\overline{xy} \subset \mathcal{W}$, otherwise either x or y would not be in $\mathcal{F}$. We also know that $\overline{xy} \not\subset F_i$, since, otherwise, x and y would not be in different connected components of $D^*(x)$. Because $x, y \in F_i$, by definition there exists a simple path $\pi \subset F_i$ from x to y. Since the workspace is a polygon with finite description complexity, we may assume that π has finite complexity as well, which implies that $\pi \cap \overline{xy}$ is composed of finitely many isolated points and closed segments. See Fig. 1(a) for an illustration.

We now define x', y' as the points on $\pi \cap \overline{xy} \subset D^*(x)$ such that x', y' are in different connected components of $D^*(x)$ and $\|x' - y'\|$ is minimized given the first condition. Let π' be the subpath of π joining x' to y'. Notice that $\pi \cap \overline{x'y'} = \{x', y'\}$. Indeed, if there exists a point $z \in \pi \cap \text{Int}(\overline{x'y'})$, then z must be in a different connected component of $D^*(x)$ than either x' or y', and $\|x' - y'\|$ would not be the minimum. Since π is a simple path, this means that $\lambda \triangleq \pi' \cup \overline{x'y'}$ is a simple closed curve. The area enclosed by λ (including λ) will be referred to as A. We note that $\lambda \subset \mathcal{W}$ since $\pi' \subset \mathcal{F} \subset \mathcal{W}$ and $\overline{x'y'} \subset \overline{xy} \subset \mathcal{W}$. This immediately implies that $A \subset \mathcal{W}$, since $\mathcal{W}$ is a simple polygon.

Let $A^* \triangleq A \backslash \mathcal{F}$. We claim that $A^* \subset \text{Int}(\mathcal{D}_2(x))$, which implies that there exists a path in F_i from x' to y' that goes along $\partial(A^*)$ and is fully contained in $\mathcal{D}_2(x)$. But this contradicts that x' and y' are in different components of $D^*(x)$ and, hence, proves the lemma. It thus remains to prove the claim that $A^* \subset \text{Int}(\mathcal{D}_2(x))$.

Note that for any point $z \in A^*$ and any $w \in \text{obs}(z)$ we have $\overline{zw} \cap \pi' = \emptyset$, since $\pi' \subset \mathcal{F}$. Furthermore, for any $v \in \pi'$ we have $\|w - v\| \geqslant 1$, and as $x', y' \in \pi'$ it follows that $\|w - x'\| \geqslant 1, \|w - y'\| \geqslant 1$. Assume without loss of generality that $\overline{x'y'}$ is vertical and that locally A lies to the right of $\overline{x'y'}$, as in Fig. 1(a). Let K be the circle of radius 1 that passes through x' and y', and whose center lies to the left of $\overline{x'y'}$—such a circle always exists since $\|x' - y'\| \leqslant \|x - y\| \leqslant 2$. (If $\|x' - y'\| = 2$ then the center of the circle lies on $\overline{x'y'}$.) Let ζ be the arc of this circle lying to the right of $\overline{x'y'}$; note that this is the shorter of the two arcs joining x' and y' if they are of different lengths. Then A^* is contained entirely within the area enclosed by ζ and $\overline{x'y'}$. Furthermore, $A^* \subset \text{Int}(A) \cup \text{Int}(\overline{x'y'})$ since $\pi' \subset \mathcal{F}$. Therefore, since $\overline{x'y'}$ is a subsegment of $\overline{xy}$ and ζ cannot cross $\partial(\mathcal{D}_2(x))$, it follows that:

$$A^* \subset (\text{Int}(A) \cup \text{Int}(\overline{x'y'})) \cap \text{Int}(\mathcal{D}_2(x)) \subset \text{Int}(\mathcal{D}_2(x))$$

which finishes the proof of the lemma. □

B. Interference Between Different Connected Components of $\mathcal{F}$

Let F_i, F_j be two distinct components of $\mathcal{F}$, and let $x \in F_i$ be such that $\mathcal{D}_2(x) \cap F_j \neq \emptyset$. We then call x an interference configuration from F_i to F_j, and define the interference set from F_i to F_j as $I_{(i,j)} \triangleq \{x \in F_i : \mathcal{D}_2(x) \cap F_j \neq \emptyset\}$. We also define the mutual interference set of F_i, F_j as $I_{\{i,j\}} \triangleq I_{(i,j)} \cup I_{(j,i)}$. Intuitively, an interference configuration from F_i to F_j is a configuration for a robot in F_i that could block a path in F_j, and the interference set is the set of all such points. The mutual interference set of F_i, F_j is the set of all single-robot configurations in either component that might block a valid single-robot path in the other component.

Lemma 3: For any mutual interference set $I_{\{i,j\}}$ and any two configurations $x_1, x_2 \in I_{\{i,j\}}$ we have $\mathcal{D}_2(x_1) \cap \mathcal{D}_2(x_2) \neq \emptyset$.

Proof: The proof is similar in spirit to the proof of Lemma 2 albeit slightly more involved. Assume for a contradiction that $x_1, x_2 \in I_{\{i,j\}}$ and $\mathcal{D}_2(x_1) \cap \mathcal{D}_2(x_2) = \emptyset$. By definition there exist $y_1 \in \mathcal{D}_2(x_1)$ and $y_2 \in \mathcal{D}_2(x_2)$ such that each pair $\{x_1, y_1\}, \{x_2, y_2\}$ contains one point in F_i and one point in F_j. As shown in the proof for Lemma 2, the segments $\overline{x_1y_1}, \overline{x_2y_2}$ are entirely contained in $\mathcal{W}$. We may assume that $\overline{x_1y_1}$ does not cross $\overline{x_2y_2}$, since if it did the crossing point would be in $\mathcal{D}_2(x_1) \cap \mathcal{D}_2(x_2)$ and we would be done. Therefore, there exists a simple closed curve $\lambda \subset \mathcal{W}$ composed of the union of two simple curves π_i, π_j and two line segments ℓ_1, ℓ_2 such that $\pi_i \subset F_i$ and $\pi_j \subset F_j$, and $\ell_1 \subset \overline{x_1y_1}, \ell_2 \subset \overline{x_2y_2}$. Note that both ℓ_1 and ℓ_2 have one endpoint in F_i and the other in F_j; see Fig. 1(b) for an illustration. The endpoints of ℓ_1 consist of x_1', y_1', such that x_1, x_1' and y_1, y_1' belong to the same connected components, and minimize the distance $\|x_1' - y_1'\|$ (ℓ_2 is similarly defined).

We refer to the region enclosed by λ (including λ) as A. Because $\lambda \subset \mathcal{W}$ and $\mathcal{W}$ is a simple polygon, we know that $A \subset \mathcal{W}$. Furthermore, since $\pi_i, \pi_j \subset \mathcal{F}$, for any $x \in \text{Int}(A)$ and $y \in \text{obs}(x)$ (by definition, $y \in \mathbb{R}^2 \backslash \mathcal{W}$ so $y \notin A$; thus, $\overline{xy} \cap \lambda \neq \emptyset$), we know that $\overline{xy} \cap \pi_i = \overline{xy} \cap \pi_j = \emptyset$. Thus, $\overline{xy} \cap \text{Int}(\ell_1) \neq \emptyset$ or $\overline{xy} \cap \text{Int}(\ell_2) \neq \emptyset$, or both. Let $A^* \triangleq A \backslash \mathcal{F}$ and denote by A_1^* the set of configurations $x \in A^*$ for which there exists $y \in \text{obs}(x)$ such that $\overline{xy} \cap \text{Int}(\ell_1) \neq \emptyset$; the set A_2^* is defined in a similar manner, only that now $\overline{xy} \cap \text{Int}(\ell_2) \neq \emptyset$. Note that $A^* = A_1^* \cup A_2^*$.

We claim that $A_1^* \cap A_2^* \neq \emptyset$. Indeed, if $A_1^* \cap A_2^* = \emptyset$ then there is a path from x_1 to y_1 along $\partial(A_1^*)$ that stays in $A \setminus A^*$ and, hence, stays in $\mathcal{F}$, which would contradict that $x_1 \in F_i$ and $y_1 \in F_j$ for $i \neq j$. Thus, there exists a point $x^* \in A_1^* \cap A_2^*$. We define the unit circles K_1, K_2 whose boundaries lie on the endpoints of ℓ_1, ℓ_2 respectively, and whose centers are located outside A. Thus, we have $A_1^* \subset K_1$ and $A_2^* \subset K_2$. Hence, $x^* \in K_1 \cap K_2$, which implies $x^* \in \mathcal{D}_2(x_1) \cap \mathcal{D}_2(x_2)$, so $\mathcal{D}_2(x_1) \cap \mathcal{D}_2(x_2) \neq \emptyset$, contradicting our initial assumption. □

The next lemma is a generalization of the previous one. Intuitively, instead of considering a cycle of length 2 among interacting free-space components, we now consider larger cycles.

Lemma 4: Let $\{\phi(1), \phi(2), \ldots, \phi(h)\} \subset \{1, 2, \ldots, q\}$, and let $x_1, x_2, \ldots, x_h$ be points such that for all i, $x_i \in I_{\{\phi(i), \phi(i+1)\}}$, where $\phi(h+1) \equiv \phi(1)$. (Thus, the list is circular with respect to its index). Then, there exists some $i \neq j$ such that $\mathcal{D}_2(x_i) \cap \mathcal{D}_2(x_j) \neq \emptyset$.

Proof: This can be proved in a manner completely analogous to the proof of Lemma 3; we will outline the proof here. We assume for a contradiction that $\mathcal{D}_2(x_i) \cap \mathcal{D}_2(x_j) = \emptyset$ for all $i \neq j$. We can argue that we can construct a simple closed curve $\lambda \subset \mathcal{W}$ passing through $F_{\phi(1)}, F_{\phi(2)}, \ldots, F_{\phi(h)}$ (in that order), which is composed of simple closed curves $\pi_i \subset F_{\phi(i)}$ and line segments $\ell_i \subset \mathcal{W}$ with endpoints in $F_{\phi(i)}$ and $F_{\phi(i+1)}$. We then consider the area A enclosed by λ and note that $A \subset \mathcal{W}$. Let $A^* \triangleq A \setminus \mathcal{F}$. If there exists some simple curve $\pi^* \subset A^*$ connecting ℓ_i to ℓ_j for some $i \neq j$, we can show that there exists some k such that $\mathcal{D}_2(x_i) \cap \mathcal{D}_2(x_k) \neq \emptyset$, contradicting our assumption. Therefore no such π^* exists for any $i \neq j$. But this means that there exists some simple path $\pi' \subset A \cap \mathcal{F}$ that joins π_i and π_j for some $i \neq j$, which contradicts the fact that π_i and π_j belong to different components of $\mathcal{F}$. □

IV. Algorithm for a Single Component

In this section, we consider a single component F_i of $\mathcal{F}$. We present an algorithm that solves the problem within F_i, ignoring the possibility that robots in F_i might collide with robots in other

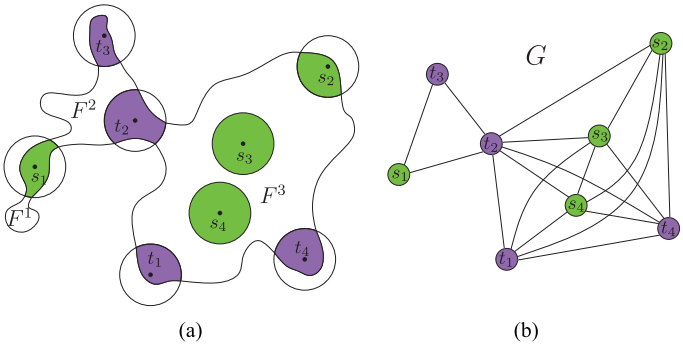


Fig. 2. (a) A partition of a maximal connected component F. The start and target positions consist of the elements $S' = \{s_1, s_2, s_3, s_4\}, T' = \{t_1, t_2, t_3, t_4\}$, respectively, where the areas $D^*(s)$ for $s \in S'$ are drawn in green and $D^*(t)$ for $t \in T'$ are drawn in purple. F^* consists of the parts F^1, F^2, F^3. (b) A motion graph of F.

components F_j. In the next section we will show how to avoid such collisions without changing the motion plans within the individual components. As before we set $S_i \triangleq S \cap F_i$ and $T_i \triangleq T \cap F_i$, and assume $|S_i| = |T_i|$.

A. The Motion Graph

The motion graph G_i of F_i is a graph whose vertices represent start or target configurations, and whose edges represent "adjacencies" between these configurations, as defined more precisely below.

Recall that for any $x \in F_i$ we defined $D^*(x) \triangleq \mathcal{D}_2(x) \cap F_i$ as the part of the collision disc of x inside F_i, and recall from Lemma 2 that $D^*(x)$ is connected. Moreover, for any two distinct configurations $x_1, x_2 \in S_i \cup T_i$ we have $D^*(x_1) \cap D^*(x_2) = \emptyset$, because $\mathcal{D}_2(x_1) \cap \mathcal{D}_2(x_2) = \emptyset$ by the assumption that the start and target positions are well-separated. The vertices of our motion graph G_i correspond to the start and target configurations in $S_i \cup T_i$. From now on, and with a slight abuse of notation we will not distinguish between configurations in $S_i \cup T_i$ and their corresponding vertices in G_i.

Now, consider $F_i^* \triangleq F_i \setminus \bigcup_{x \in S_i \cup T_i} D^*(x)$, the complement of the collision discs of the given start and target configurations in F_i. This complement consists of several connected components, which we denote by $F_i^1, F_i^2, \ldots$. If the motion graph G_i contains an edge (x_1, x_2) then there is a component F_i^ℓ that is adjacent to both $D^*(x_1)$ and $D^*(x_2)$. In other words, two configurations x_1 and x_2 are connected in G_i if there is a path from x_1 to x_2 that stays inside F_i and does not cross the collision disc of any other configuration $x_3 \in S_i \cup T_i$. Fig. 2 illustrates the definition of G_i. The following observation summarizes the main property of the motion graph.

1) Observation 1: Suppose all robots in F_i are located at a start or target configuration in $S_i \cup T_i$, and let (x_1, x_2) be any edge in G_i. Then a robot located at x_1 can move to x_2 without colliding with any of the other robots.

Remark: We could also work with the dual graph of the partitioning of F_i into cells induced by the collision discs. This dual graph would, in addition to vertices representing start and target configurations, also have vertices for the regions F_i^ℓ. For the pebble-motion problem discussed below it is easier to work with the graph as we defined it. This graph may have many more edges, but in the implementation of our algorithm described in Section VI, we avoid computing it explicitly.

B. The Unlabeled Pebble-Motion Problem

Using the motion graph G_i we can view the motion-planning problem within F_i as a pebble-motion problem. (A similar approach was taken in [18], where a sampling-based algorithm for multi-robot motion planning produces multiple pebble problems by random sampling of the configuration space.) To this end, we represent a robot located at configuration $x \in S_i \cup T_i$ by a *pebble* on the corresponding vertex in G_i. The pebbles are indistinguishable, like the robots, and they can move along the edges of the graph. At the start of the pebble-motion problem for a graph with vertex set $S_i \cup T_i$, with $|S_i| = |T_i|$, there is a pebble on every vertex $x \in S_i$. The goal is to move the pebbles such that each pebble ends up in vertex in T_i, under the following conditions: 1) no two pebbles may occupy the same vertex at the same time; and 2) pebbles can only halt at vertices; and 3) at most one pebble may move (that is, be in transit along an edge) at any given time. We call this problem the unlabeled pebble-motion problem. The following lemma follows immediately from Observation 1.

Lemma 5: Any solution to the unlabeled pebble-motion problem on G_i can be translated into a valid collision-free motion plan for the robots in F_i.

Kornhauser [25, Section 3, Lemma 1] proved that the unlabeled pebble-motion problem is, in fact, always solvable, and he gave an algorithm to find a solution. Since he did not analyze the running time of his algorithm, we sketch the solution in the proof of the lemma below.

Lemma 6: For any graph G with vertex set $S \cup T$, where $|S| = |T|$, there exists a solution to the unlabeled pebble-motion problem. Moreover, a solution can be found in $O(|S|^2)$ time [25]

Proof: Let $\mathcal{T}_G$ be a spanning tree of G. The algorithm performs $O(|S|)$ phases. In each phase, one or more pebbles may be moved and one leaf will be removed from $\mathcal{T}_G$, possibly with a pebble on it. After the phase ends, the algorithm continues with the next phase on the modified tree $\mathcal{T}_G$, until all pebbles have been removed and the problem has been solved.

A phase proceeds as follows. If there are leaves v that are target vertices then we select such a leaf v. If v does not yet contain a pebble, we find a pebble closest to v in $\mathcal{T}_G$—this can be done by a simple breadth-first search—and move it to v along the shortest path in G. Note that the vertices on the shortest path cannot contain other pebbles, since we took a closest pebble. We now remove the leaf v, together with the pebble occupying it, and end the phase. If all leaves in $\mathcal{T}_G$ are start vertices, then let v be such a leaf. If v is not occupied by a pebble it can be removed from $\mathcal{T}_G$, and the phase ends. Otherwise a pebble resides in v, which we move away, as follows. We find the closest unoccupied vertex w to v of $\mathcal{T}_G$ and move all pebbles on this shortest path (including the pebble on v) one step closer to w, in order of decreasing distance from w. After we evacuated v we remove it from $\mathcal{T}_G$ to end the phase.

The algorithm produces paths of total length $O(|S|^2)$, and it can easily be implemented to run in $O(|S|^2)$ time. In some cases $\Omega(|S|^2)$ moves are required, e.g., when G is a single path

with all start positions in the first half of the path and all target positions in the second half. □

Lemma 7: Suppose we have an instance of our multi-robot path planning problem, where $|S_i| = |T_i|$ for every component F_i of the free space $\mathcal{F}$. Then, for each F_i there exists a motion plan Π_i that brings the robots in F_i from S_i to T_i, such that they do not collide with the obstacle space nor with the other robots in F_i.

V. Combining Single-Component Plans

We now consider possible interactions between robots contained in different components F_i and F_j of $\mathcal{F}$. As before, we assume that $|S_i| = |T_i|$ for all i. We will show that there exists a permutation $\sigma : \{1, 2, \ldots, \ell\} \to \{1, 2, \ldots, \ell\}$ such that we can independently execute the single-component motion plans for each component F_i as long as we do so in the order $F_{\sigma(1)}, F_{\sigma(2)}, \ldots, F_{\sigma(\ell)}$.

To obtain this order, we define a directed graph representing the structure of $\mathcal{F}$, which we call the directed-interference forest $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where the nodes in $\mathcal{V}$ correspond to the components F_i. We add the directed edge (F_i, F_j) to $\mathcal{E}$ if either there exists a start position $s \in S$ such that $s \in I_{(i,j)}$, or there exists a target position $t \in T$ such that $t \in I_{(j,i)}$. For any $i \in \{1, 2, \ldots, \ell\}$, we additionally define $N^+(i)$ to be the set of indices of the vertices in the out-neighborhood of v_i; similarly, $N^-(i)$ is defined as the set of indices of the vertices in the in-neighborhood of v_i.

Note that by Lemma 3 and since S, T are well separated, we cannot have more than one start or target position in $I_{\{i,j\}}$. This implies that $\mathcal{E}$ cannot contain both (v_i, v_j) and (v_j, v_i). Lemma 4 and the well-separatedness condition additionally imply that $\mathcal{G}$ cannot have an undirected cycle. Thus, $\mathcal{G}$ is a directed forest.

We now produce the desired ordering using $\mathcal{G}$. Consider $F_i \in \mathcal{V}$, and suppose that for all $j \in N^+(i)$, every robot in F_j is at a start position, and for all $j \in N^-(i)$, every robot in F_j is at a target position. Additionally, suppose that for all $j \notin N^+(i) \cup N^-(i)$, every robot in F_j is at a start or target position. Then, by the definition of $\mathcal{G}$, no robot is at a configuration in $I_{\{i,j\}}$ for any $j \neq i$; thus any motion plan for the robots in F_i, such as the one described in Section IV, can be carried out without being blocked by the robots not in F_i. Hence, if we have an ordering $\sigma : \{1, 2, \ldots, \ell\} \to \{1, 2, \ldots, \ell\}$ such that for all (directed) edges $(v_i, v_j) \in \mathcal{E}$, $\sigma^{-1}(i) < \sigma^{-1}(j)$, where σ^{-1} is the inverse permutation of σ, then we can execute the motion plans for the robots in $F_{\sigma(1)}, F_{\sigma(2)}, \ldots, F_{\sigma(\ell)}$ in that order. Since $\mathcal{G}$ is a directed forest such an ordering can be produced using topological sorting on the vertices of $\mathcal{G}$. Thus, combining this result with Lemma 7 we obtain the following.

Theorem 8: Let there be a collection of m unlabeled unit-disc robots in a simple polygonal workspace $\mathcal{W} \subset \mathbb{R}^2$, with start and target configurations S, T that are well-separated. Then, if for every maximal connected component F_i of $\mathcal{F}$ (where $\mathcal{F}$ is the free space for a single unit-disc robot in $\mathcal{W}$) with $|S \cap F_i| = |T \cap F_i|$, there exists a collision-free motion plan for these robots starting at S that terminates with every position of T occupied by a robot.

VI. Algorithmic Details

In this section, we fill in a few missing details in the description of our algorithm. Specifically, we present an efficient method for generating motion graphs and describe a technique for generating configuration-space paths that correspond to edges in the motion graphs. We also consider the complexity of the various subsets of $\mathcal{F}$ used throughout the algorithm.

A. Partitioning $\mathcal{F}$

We analyze the combinatorial complexity of $\mathcal{F}^* \triangleq \mathcal{F} \setminus \bigcup_{x \in S \cup T} D^*(x)$ and $\mathbb{D} \triangleq \bigcup_{x \in S \cup T} D^*(x)$.

Lemma 9: The combinatorial complexity of $\mathcal{F}^*$ and $\mathbb{D}$ is $O(m + n)$.

Proof: We start with $\mathcal{F}^*$ and decompose the complement of the workspace polygon into $O(n)$ trapezoids—this is doable by standard vertical decomposition. We define a set X, which consists of the trapezoids, and in addition a collection of $O(m)$ unit discs that are centered at the start and target positions. We now observe that the regions in X are pairwise interior disjoint (and convex). Hence, it is known [32] that the complexity of the union of the regions in X, each Minkowski-summed with a unit disc, is linear in the number of regions plus the sum of the complexities of the original regions. As the result of the Minkowski sum operation of X with a unit disc is the area $\mathcal{F}^*$, we conclude that the complexity of $\mathcal{F}^*$ is $O(m + n)$.

Due to the fact that every vertex of the boundary of $\mathbb{D}$ is either a vertex of $\mathcal{F}^*$ or a vertex of $\mathcal{F}$, it immediately follows that the complexity of $\mathbb{D}$ is $O(m + n)$ as well. □

Note that this upper bound still holds if we consider instead of $\mathcal{F}^*$ the union of $F_i^* \triangleq F_i \setminus \bigcup_{x \in S_i \cup T_i} D^*(x)$, for all $1 \leqslant i \leqslant q$.

B. Generating the Motion Graphs

We consider a specific component F of $\mathcal{F}$ and construct its motion graph G. Denote $F^* \triangleq F \setminus \bigcup_{x \in (S \cup T) \cap F} D^*(x)$. Note that by the analysis in Section V we can ignore the influence of $\mathcal{D}_2(x)$ on connected components in $\mathcal{F}$ that do not contain x. We assume that F^* breaks into k maximal connected components $F^1, \ldots, F^k$. The construction of G, along with the paths in F that correspond to the edges of G, is carried out in two steps. First, for every F^i, we generate the portion of G, denoted by G^i, whose vertices represent start and target positions that touch the boundary of F^i. Then, we connect between the various parts of G.

We consider a specific connected component F^i of F^* and describe how the respective portion of the motion graph, namely G^i, is generated. We split the start and target positions that share a boundary with F^i into two subsets: B^i are those positions for which the collision disc intersects the boundary of F and H^i are those positions for which the collision disc floats inside F. See Fig. 3. We first handle positions in B^i. Consider the outer boundary Γ^i of $F^i \setminus \bigcup_{x \in S \cup T} D^*(x)$. We argue that each $x \in B^i$ can contribute exactly one piece to Γ^i.

Lemma 10: If $x \in B^i$ then $\partial(\mathcal{D}_2(x)) \cap \partial(F^i)$ consists of a single component.

Proof: By contradiction, assume that the intersection consists of two maximal connected components. Denote by y, y' two configurations on the two components. As F^i consists of a single connected component of F there exists a path $\pi_{yy'} \subset F$ from y to y'. Additionally, as y, y', x belong to the same connected component of $\mathcal{F}$ there exist two paths—π_{xy} from x to y and $\pi_{xy'}$ from x to y'—that lie entirely in $D^*(x)$. Thus, the

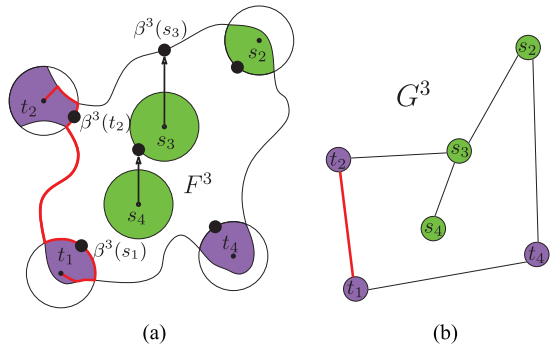


Fig. 3. (a) An illustration of a component F^3 of F^* and the structures used for generating the relevant portion of the motion graph. The boundary positions of F^3 consist of $B^3 := \{s_2, t_1, t_2, t_4\}$, while the hole positions consist of $H^3 := \{s_3, s_4\}$. For every $x \in H^3$ its boundary representative $\beta^3(x)$ is illustrated as a large black dot. A path between t_1 and t_2 is illustrated in red. (b) The motion graph G^3 induced by F^3.

area that is bounded by the three paths $\pi_{yy'}, \pi_{xy}, \pi_{xy'}$ contains a patch of forbidden space, which contradicts the fact the our workspace is a simple polygon. □

For every $x \in B^i$, we arbitrarily select a representative point $\beta^i(x) \in \partial(\mathcal{D}_2(x)) \cap F^i$. We order the points $\beta^i(x)$ clockwise around Γ^i, and store them in a circular list $\mathcal{L}^i$. We now incorporate the remaining start and target positions H^i, namely, those positions x for which $D_2(x) \cap \partial(F) = \emptyset$. Each position in H^i will be connected either to Γ^i or to the boundary of a collision disc of another position in H^i as follows. For each $x \in H^i$ we shoot a vertical ray upwards until it hits $\partial(F^i)$. Denote the point where the ray hits $\partial(F^i)$ by c. If $c \in \partial(\mathcal{D}_2(x'))$ for some $x' \in H^i, x' \neq x$ then an edge between x and x' is added to G^i. Otherwise, we let $\beta^i(x) \triangleq c$ and insert it into the circular list $\mathcal{L}^i$ representing the points $\beta^i(x)$ along Γ^i collected so far. After all positions in H^i have been handled in this manner, for each pair of consecutive points $\beta^i(x'), \beta^i(x'')$ in $\mathcal{L}^i$ (along Γ^i) we add an edge in G^i between the vertices x' and x''. (Notice that some of the positions x whose $\beta^i(x)$ appear in $\mathcal{L}^i$ belong to H^i; for example, s_3 in Fig. 3.) Finally, the connection between portions of the motion graph that represent different parts of F^* is achieved through positions shared between two sets B^i, B^j, for $i \neq j$.

C. Transforming Graph Edges Into Paths in the Free Space

There are three different types of transformations depending on how the edge was created. Let (x, x') be an edge in G_i. Consider Fig. 3 for an illustration. (i) If both x and x' belong to H^i (see (s_4, s_3) in the figure) then the path simply consists of the two straight-line segments $\overline{xc}$ and $\overline{cx'}$. For the remaining two cases we note that if either vertex, say x, is in B^i, then part of the path is a simple curve connecting x to $\beta^i(x)$ within $D^*(x)$ (see the red curves from s_1 and t_2 in the figure). We denote this curve by δ_x. (ii) $x, x' \in B^i$ and the points $\beta^i(x)$ and $\beta^i(x')$ are consecutive along Γ^i (see (t_1, t_2) in the figure). The path corresponding to the edge (x, x') in this case is a concatenation of three sub-paths: δ_x, the portion of Γ^i between $\beta^i(x)$ and $\beta^i(x')$ (not passing though the boundary of any other collision disc), and $\delta_{x'}$. (iii) $x \in H^i$ and $x' \in B^i$ (see (s_3, s_2) in the figure). The path is again a concatenation of three paths: the line segment $\overline{x\beta^i(x)}$, the portion of Γ^i between $\beta^i(x)$ and $\beta^i(x')$ (not passing though the boundary of any other collision disc), and $\delta_{x'}$.

Notice that for all path types above if a robot r moves from x to x', x' is not occupied, and all other robots occupy positions only at $S \cup T \setminus \{x, x'\}$, r will not collide with any other robot during the motion.

D. Complexity Analysis

We provide complexity analysis of our algorithm and show that a solution to the problem can be produced within $O(mn + m^2)$ operations.

Recall that the pebble problem solver (Section IV) operates in $O(m)$ phases, where in each phase a leaf node is removed from the spanning tree of G. We show, using Lemma 9 that each phase can be transformed into a set of movements for the robots whose combinatorial complexity is $O(m+n)$. The crucial observation is that in one phase each edge of the motion graph is used at most once. Thus the set of robot movements in one phase is bounded by the complexity of the movements corresponding to all the edges in the graph together. These comprise $O(m)$ line segments, portions of the boundaries Γ^i (whose complexity is $O(m+n)$ by Lemma 9), and the paths δ_x inside the $D^*(x)'s$ (whose complexity is $O(m+n)$ as well). A path of the latter type, δ_x, might be traversed twice: once for reaching x and once for leaving x. Asymptotically all the movements together have complexity $O(m+n)$. Therefore, the overall complexity of the motions is described by $O(m^2 + mn)$.

The generation of $\mathcal{F}$ can be performed by constructing the medial axis of $\mathcal{W}$ [33] in $O(n)$ time. The construction of $\mathcal{F}^*$ and $\mathbb{D}$ is done straightforwardly by performing the following two operations for every $x \in S \cup T$. First, we check whether $\mathcal{D}_2(x) \cup \partial(\mathcal{F}) \neq \emptyset$ by intersecting $\mathcal{D}_2(x)$ with every edge of $\mathcal{F}$. In case that no intersection is found, we find the connected component of $\mathcal{F}$ that contains x by point-in-polygon test [34]. These two checks take $O(n)$ time for every such x, and $O(mn)$ time in total. In conclusion, we have the following theorem.

Theorem 11: Let $\mathcal{W}$ be a simple polygon with n vertices and let $S = \{s_1, \ldots, s_m\}, T = \{t_1, \ldots, t_m\}$ be two sets of m points in $\mathcal{W}$. Additionally, assume that for every two distinct element x, x' of $S \cup T$ it holds that $\|x - x'\| \geqslant 4$. Then, given m unlabeled unit disc robots, our algorithm can determine whether a path moving the m robots from S to T exists, and find a solution in $O(m^2 + mn)$ time.

VII. Separation and Solvability

Our results from the previous section imply that a separation distance of four ensures that the problem always has a solution, assuming that each connected component contains the same number of start and target positions. However, for smaller separation values this does not have to be the case.

We define a magnitude λ to be the minimum separation between start and target positions of unlabeled discs in a simple polygon, that guarantees that a solution always exist (assuming that each connected component contains the same number of start and target positions). Our work in the previous sections

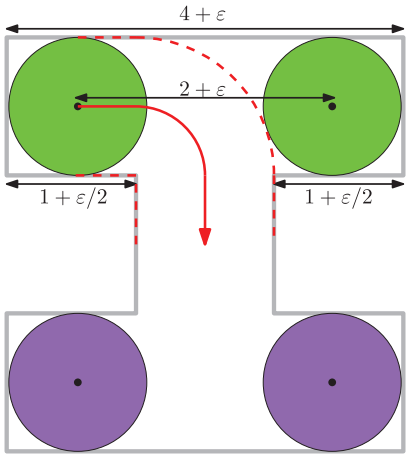


Fig. 4. In the given example, the two robots need to leave through the corridor of width 2 located below their initial positions. This is only possible when $\epsilon \geqslant 4\sqrt{2}-2$, as otherwise the robots will not be able to get to the corridor without colliding with each other. The dashed red segments depict the trace of the boundary of the left robot, while getting to the corridor (whose width is 1). The red arrow depicts the path of the left robot. When $\epsilon < 4\sqrt{2}-4$, the right dashed line will penetrate the robot located in the right start position, and consequently the two robots will be in collision.

shows that $\lambda \leqslant 4$. The following proposition provides a lower bound for the value of λ.

Proposition 12: $\lambda \geqslant 4\sqrt{2}-2 (\approx 3.646)$.

Proof: We describe a concrete example where the value of λ has to be greater than $4\sqrt{2}-2$ to guarantee solvability (see Fig. 4). While the proof is rather technical and straightforward, we mention that the main observation here is that the distance between the left corner of the corridor, around which the left robot has to rotate, has to be at distance of at least three from the start position from the right robot. □

VIII. Open Problems and Future Work

We have studied a basic variant of the multi-robot motion-planning problem, where the goal is to find collision-free motions that bring a given set of indistinguishable unit discs in a simple polygon to a given set of target positions. Under the condition that the start and target positions are separated from each other by a distance of at least four, we developed an algorithm that solves the problem in time polynomial in the complexity of the polygon as well as in the number of discs: quadratic in the number of robots and near-linear in the complexity of the polygon.

Our result should be contrasted with the labeled counterpart of the problem, which is NP-hard [8]. In the NP-hardness proof, the discs have different radii, however, and there is no restriction on the separation of the start and target position. Thus, one of the main open questions resulting from our study is to settle the complexity of the unlabeled problem without this extra separation condition. Very recently, we have made progress in this direction, by showing that the general unlabeled problem is PSPACE-hard [31]. It should be noted that this proof applies to a slightly different setting that consists of unit-square robots translating amidst polygonal obstacles.

A natural question that arises is what happens when the separation of start or target positions is equal to $2+\varepsilon$, where $0 < \varepsilon < 2$? Would it be possible to design an algorithm, whose running time is polynomial in m and n, and also depends in some manner on ε? We believe that the work by Alt *et al.* [35] would be a good starting point for this line of work. Following our discussion in Section VII, it will also be interesting to find the exact threshold above which a problem is always solvable.

References

[1] J. T. Schwartz and M. Sharir, "On the Piano Movers problem: II. General techniques for computing topological properties of real algebraic manifolds," *Adv. Appl. Math.*, vol. 4, no. 3, pp. 298–351, 1983.
[2] J. T. Schwartz and M. Sharir, "On the Piano Movers problem: III. Coordinating the motion of several independent bodies," *Int. J. Robot. Res.*, vol. 2, no. 3, pp. 46–75, 1983.
[3] C.-K. Yap, "Coordinating the motion of several discs," Michigan State Univ., Courant Inst. Math. Sci., New York, NY, USA, Tech. Rep., 1984.
[4] M. Sharir and S. Sifrony, "Coordinated motion planning for two independent robots," *Ann. Math. Artif. Intell.*, vol. 3, no. 1, pp. 107–130, 1991.
[5] B. Aronov, M. de Berg, A. F. van der Stappen, P. Švestka, and J. Vleugels, "Motion planning for multiple robots," *Discrete Comput. Geomet.*, vol. 22, no. 4, pp. 505–525, 1999.
[6] J. van den Berg, J. Snoeyink, M. C. Lin, and D. Manocha, "Centralized path planning for multiple robots: Optimal decoupling into sequential plans," in *Proc. Robot.: Sci. Syst. (RSS)*, Seattle, WA, USA, 2009.
[7] J. E. Hopcroft, J. T. Schwartz, and M. Sharir, "On the complexity of motion planning for multiple independent objects; PSPACE-hardness of the "Warehouseman's problem"," *Int. J. Robot. Res.*, vol. 3, no. 4, pp. 76–88, 1984.
[8] P. G. Spirakis and C.-K. Yap, "Strong NP-hardness of moving many discs," *Inform. Process. Lett.*, vol. 19, no. 1, pp. 55–59, 1984.
[9] L. E. Kavraki, P. Švestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high dimensional configuration spaces," *IEEE Trans. Robot. Autom.*, vol. 12, no. 4, pp. 566–580, Aug. 1996.
[10] J. J. Kuffner and S. M. Lavalle, "RRT-Connect: An efficient approach to single-query path planning," in *Int. Conf. Robot. Autom. (ICRA)*, 2000, pp. 995–1001.
[11] G. Sanchez and J.-C. Latombe, "Using a PRM planner to compare centralized and decoupled planning for multi-robot systems," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, Washington, DC, USA, 2002, pp. 2112–2119.
[12] S. Hirsch and D. Halperin, "Hybrid motion planning: Coordinating two discs moving among polygonal obstacles in the plane," in *Proc. Workshop Algorithmic Foundations of Robot. (WAFR)*, 2002, pp. 239–255.
[13] O. Salzman, M. Hemmer, and D. Halperin, "On the power of manifold samples in exploring configuration spaces and the dimensionality of narrow passages," *IEEE Trans. Autom. Sci. Eng.*, vol. 12, no. 2, pp. 529–538, Apr. 2015.
[14] K. Solovey, O. Salzman, and D. Halperin, "Finding a needle in an exponential haystack: Discrete RRT for exploration of implicit roadmaps in multi-robot motion planning," in *Proc. Algorithmic Found. Robot. XI*, 2014, pp. 591–607.
[15] P. Švestka and M. H. Overmars, "Coordinated path planning for multiple robots," *Robot. Auton. Syst.*, vol. 23, pp. 125–152, 1998.
[16] G. Wagner and H. Choset, "Subdimensional expansion for multirobot path planning," *Artif. Intell.*, vol. 219, pp. 1–24, 2015.
[17] S. Kloder and S. Hutchinson, "Path planning for permutation-invariant multi-robot formations," in *Proc. ICRA*, 2005, pp. 1797–1802.
[18] K. Solovey and D. Halperin, "k-color multi-robot motion planning," *Int. J. Robot. Res.*, 2013, to be published.
[19] M. Turpin, K. Mohta, N. Michael, and V. Kumar, "Goal assignment and trajectory planning for large teams of interchangeable robots," *Auton. Robots*, vol. 37, no. 4, pp. 401–415, 2014.
[20] K. Solovey, J. Yu, O. Zamir, and D. Halperin, "Motion planning for unlabeled discs with optimality guarantees," in *Proc. Robot.: Sci. Syst. (RSS)*, Rome, Italy, 2015.
[21] S. Bereg, A. Dumitrescu, and J. Pach, "Sliding disks in the plane," *Int. J. Comput. Geometry Appl.*, vol. 18, no. 5, pp. 373–387, 2008.
[22] A. Dumitrescu and M. Jiang, "On reconfiguration of disks in the plane and related problems," *Computat. Geometry: Theory Appl.*, vol. 46, no. 3, pp. 191–202, 2013.
[23] O. Goldreich, "Shortest move-sequence in the generalized 15-puzzle is NP-hard," *Manuscript, Laboratory for Computer Sci., MIT*, vol. 1, 1984.

[24] G. Goraly and R. Hassin, "Multi-color pebble motion on graphs," *Algorithmica*, vol. 58, no. 3, pp. 610–636, 2010.
[25] D. Kornhauser, "Coordinating pebble motion on graphs, the diameter of permutation groups, and applications," M.Sc. thesis, Dept. Elect. Eng. Comput. Sci., Mass. Inst. Technol., Cambridge, MA, USA, 1984.
[26] A. Krontiris, R. Luna, and K. E. Bekris, "From feasibility tests to path planners for multi-agent pathfinding," in *Proc. Symp. Combinatorial Search*, Leavenworth, WA, USA, 2013.
[27] C. H. Papadimitriou, P. Raghavan, M. Sudan, and H. Tamaki, "Motion planning on a graph," in *Found. Comput. Sci.*, 1994, pp. 511–520.
[28] J. Yu and S. M. LaValle, "Distance optimal formation control on graphs with a tight convergence time guarantee," in *Proc. IEEE Int. Conf. Decision Control*, 2012, pp. 4023–4028.
[29] T.-R. Hsiang, E. M. Arkin, M. A. Bender, S. P. Fekete, and J. S. Mitchell, "Algorithms for rapidly dispersing robot swarms in unknown environments," in *Algorithmic Foundations of Robotics V*. New York, NY, USA: Springer, 2004, pp. 77–93.
[30] A. Becker, S. P. Fekete, A. Kröller, S. Lee, J. McLurkin, and C. Schmidt, "Triangulating unknown environments using robot swarms," in *Proc. Symp. Comput. Geometry, SoCG'13*, Rio de Janeiro, Brazil, Jun. 17–20, 2013, pp. 345–346.
[31] K. Solovey and D. Halperin, "On the hardness of unlabeled multi-robot motion planning," in *Proc. Robot.: Sci. Syst. (RSS)*, Rome, Italy, 2015.
[32] K. Kedem, R. Livne, J. Pach, and M. Sharir, "On the union of jordan regions and collision-free translational motion amidst polygonal obstacles," *Discrete Comput. Geomet.*, vol. 1, pp. 59–70, 1986.
[33] F. Y. L. Chin, J. Snoeyink, and C. A. Wang, "Finding the medial axis of a simple polygon in linear time," *Discrete Comput. Geomet.*, vol. 21, no. 3, pp. 405–420, 1999.
[34] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars, *Computational Geometry: Algorithms and Applications*, 3rd ed. Berlin, Germany: Springer-Verlag, 2008.
[35] H. Alt, R. Fleischer, M. Kaufmann, K. Mehlhorn, S. Näher, S. Schirra, and C. Uhrig, "Approximate motion planning and the complexity of the boundary of the union of simple geometric figures," *Algorithmica*, vol. 8, no. 1–6, pp. 391–406, 1992.

Aviv Adler received the B.A. degree in mathematics from Princeton University, Princeton, NJ, USA, in 2014. He is currently working towards the Ph.D. degree at the Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, USA.

His primary research interests are algorithms, computational geometry, graph theory, and motion planning.

Mark de Berg received the Ph.D. degree in computer science from Utrecht University, Utrecht, The Netherlands, in 1988.

After receiving the Ph.D. degree, he became a faculty member at Utrecht University. In 2002, he moved to the TU Eindhoven, where he became a Full Professor and Head of the TU/e Algorithms Group. His research area is algorithms and data structures and, in particular, computational geometry. He is (co-)author of two books on computational geometry, one of which has become the standard textbook in the area, and he published over 190 papers in peer-reviewed journals and conferences.

Prof. de Berg was on the program committee of numerous international conferences on computational geometry and on algorithms, and he serves on the editorial board of three international journals.

Dan Halperin received the Ph.D. degree in computer science from Tel-Aviv University, Tel-Aviv, Israel.

He then spent three years at the Computer Science Robotics Laboratory, Stanford University. In 1996, he joined the Department of Computer Science, Tel-Aviv University, where he is currently a Full Professor and for two years was the Department Chair. His main field of research is computational geometry and its applications. A major focus of his work has been in research and development of robust geometric software, principally as part of the CGAL project and library. The application areas he is interested in include robotics and automated manufacturing, and algorithmic motion planning.

Kiril Solovey is currently working towards the Ph.D. degree at the School of Computer Science, Tel-Aviv University, Tel-Aviv, Israel, under the supervision of Prof. Halperin.

His research interests include sampling-based algorithms for motion planning, multirobot motion planning, and computational geometry.

4

Motion Planning for Unlabeled Discs with Optimality Guarantees

Motion Planning for Unlabeled Discs with Optimality Guarantees

Kiril Solovey*, Jingjin Yu†, Or Zamir* and Dan Halperin*

***Abstract*—We study the problem of path planning for unlabeled (indistinguishable) unit-disc robots in a planar environment cluttered with polygonal obstacles. We introduce an algorithm which minimizes the total path length, i.e., the sum of lengths of the individual paths. Our algorithm is guaranteed to find a solution if one exists, or report that none exists otherwise. It runs in time $\tilde{O}\left(m^4 + m^2n^2\right)$, where m is the number of robots and n is the total complexity of the workspace. Moreover, the total length of the returned solution is at most OPT$+4m$, where OPT is the optimal solution cost. To the best of our knowledge this is the first algorithm for the problem that has such guarantees. The algorithm has been implemented in an exact manner and we present experimental results that attest to its efficiency.**

I. Introduction

We study the problem of path planning for unlabeled (i.e., indistinguishable) unit-disc robots operating in a planar environment cluttered with polygonal obstacles. The problem consists of moving the robots from a set of *start* positions to another set of *goal* positions. Throughout the movement, each robot is required to avoid collisions with obstacles and well as with its fellow robots. Since the robots are unlabeled, we only demand that at the end of the motion each position will be occupied by exactly one robot, but do not insist on a specific assignment of robots to goals (in contrast to the standard labeled setting of the problem).

Following the methodology of performing target assignments and path planning concurrently [1, 2, 3] and adopting a graph-based vertex ordering argument from [4], we develop a complete combinatorial algorithm to the unlabeled problem described above. Our algorithm makes two simplifying assumptions regarding the input. First, it assumes that each start or goal position has a (Euclidean) distance of at least $\sqrt{5}$ to any obstacle in the environment (recall that the robot discs have unit radius) . Additionally, it requires that the distance between each start or goal position to any other such position will be at least 4 (see Figure 1). Given that the two separation conditions are fulfilled, out algorithm is guaranteed to generate a solution if one exists, or report that none exists otherwise. It has a running time[1] of $\tilde{O}\left(m^4 + m^2n^2\right)$, where m is the number of robots, and n is the description complexity of the workspace environment, i.e., the number of edges of the polygons. Furthermore, the total length of the returned solution, i.e., the sum of lengths of the individual paths, is at most OPT $+ 4m$, where OPT is the optimal solution cost.

This paper should be juxtaposed with another work by the authors [5] in which we show that a slightly different setting of the unlabeled problem is computationally intractable. In particular, we show that the unlabeled problem of unit-square robots translating amid polygonal obstacles is PSPACE-hard. Our proof relies on a construction of gadgets in which start and goal positions are very close to one another or to the obstacles, i.e., no separation is assumed. This can be viewed as a justification to the assumptions used in the current paper which allow an efficient solution of the problem.

We make several novel contributions. To the best of our knowledge, we are the first to describe a complete algorithm that fully addresses the problem of planning minimum total-distance paths for unlabeled discs. We mention that Turpin et al. [3] describe a complete algorithm for unlabeled discs; however their algorithm minimizes the maximal path length of an individual robot. Furthermore, our algorithm makes more natural assumptions on the input problem than the ones made by Turpin et al. We also mention the work by Adler et al. [6] in which a complete algorithm is described, but it does not guarantee optimality. While the latter work makes the same assumption as we make on a separation of 4 between starts and goals, it does not assume separation from obstacles. However, it requires that the workspace environment will consist of a simple polygon. On the practical side, we provide an exact and efficient implementation which does not rely on non-deterministic procedures, unlike sampling-based algorithms. The implementation will be made publicly available upon the publication of this paper.

Our work is motivated by theoretical curiosity as well as potential real world applications. From a theoretical standpoint, we have seen great renewed interests in developing planning algorithms for multi-robot systems in continuous domains (see, e.g., [3, 7, 8, 9, 10]) and discrete settings (see, e.g., [2, 11]). Whereas significant headway has been made in solving multi-robot motion-planning problems, many challenges persist; the problem studied in this paper—path planning for unlabeled disc robots in a general environment with guarantees on total distance optimality—remains unresolved (until now). On the

*K. Solovey, O. Zamir and D. Halperin are with the Blavatnic School of Computer Science, Tel Aviv University, Israel; Email: {kirilsol,danha}@post.tau.ac.il, orzamir@mail.tau.ac.il; The work of K.S. and D.H. has been supported in part by the Israel Science Foundation (grant no. 1102/11), by the German-Israeli Foundation (grant no. 1150-82.6/2011), and by the Hermann Minkowski-Minerva Center for Geometry at Tel Aviv University.

†J. Yu is with the Computer Science and Artificial Intelligence Lab at the Massachusetts Institute of Technology; Email: jingjin@csail.mit.edu; His work has been supported in part by ONR projects N00014-12-1-1000 and N00014-09-1-1051.

[1]For simplicity of presentation, we omit log factors when stating the complexity of the algorithm, and hence use the $\tilde{O}$ notation.

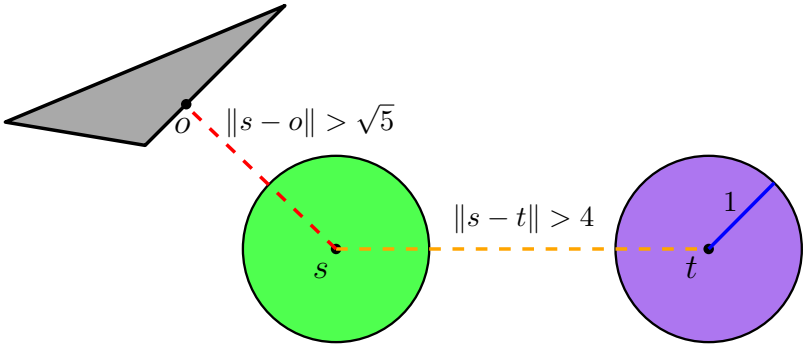


Fig. 1: Illustration of the separation conditions assumed in this work. The green and purple discs represent two unit-disc robots placed in $s \in S, t \in T$, respectively. The blue line represents the unit radius of the robot (for scale). The distance between s and t is at least 4 units (see dashed orange line). The gray rectangle represents an obstacle, and the point o represents the closest obstacle point to s. Notice that the distance between o and s is at least $\sqrt{5}$ units (see dashed red line).

application side, in the past few years, we have witnessed the rapid development and adaptation of autonomous multi-robot and multi-vehicle systems in a wide variety of application domains. The most prominent example is arguably the success of Kiva Systems, now part of Amazon, which developed a warehousing system employing hundreds of mobile robots to facilitate order assembly tasks [12]. More recently, Amazon, DHL, and Google have demonstrated working prototypes of aerial vehicles capable of automated package delivery. Since the vehicles are intended to operate in an autonomous, swarm-like setting, we can foresee in the near future the emerging demand of efficient path planning algorithms designed for such systems. We note that when the warehouse robots or the delivery aerial vehicles do not carry loads, they are effectively unlabeled robots. In such scenarios, planning collision-free, total-distance near-optimal paths translates into allowing the vehicles, as a whole, to travel with minimum energy consumption.

The rest of the paper is organized as follows. In Section II we review related work. In Section III we provide an overview of our algorithm and necessary background material. In Section IV we establish several basic properties of the problem and describe the algorithm in Section V. We report on experimental results in Section VI and conclude with a discussion in Section VII.

II. Related work

The problem of multi-robot motion planning is notoriously challenging as it often involves many degrees of freedom, and consequently a vast search space [13, 14]. In general, each additional robot introduces several more degrees of freedom to the problem. Nevertheless, there is a rich body of work dedicated to this problem. The earliest research efforts can be traced back to the 1980s [15].

Approaches for solving the problem can be typically subdivided into categories. Decoupled techniques (see, e.g., [16, 17, 18, 19, 20, 21]) reduce the size of the search space by partitioning the problem into several subproblems, which are solved separately, and then the different components are combined. In contrast to that, centralized approaches (see, e.g., [22, 23, 24, 25, 26, 27, 28]) usually work in the combined high-dimensional configuration space, and thus tend to be slower than decoupled techniques. However, centralized algorithms often come with stronger theoretical guarantees, such as completeness. Besides these, the multi-robot motion-planning problem has also been attacked using methods based on network flow [8] and mixed integer programming [29], among others.

Multi-robot motion planning can also be considered as a discrete problem on a graph [30]. In this case the robots are pebbles placed on the vertices of the graph and are bound to move from one set of vertices to another along edges. Many aspects of the discrete case are well understood. In particular, for the labeled setting of the problem there exist efficient feasibility-test algorithms [31, 32, 33], as well as complete planners ([9, 34, 35]). For the unlabeled case, there even exist complete and efficient planners that generate the optimal solution [4, 11, 36] under different metrics. While there exists a fundamental difference between the discrete and the continuous setting of the multi-robot problem, the continuous case being exceedingly more difficult, several recent techniques in the continuous domain [3, 6, 37] have employed concepts that were initially introduced in the discrete domain.

As mentioned above, the works by Adler et al. [6] and Turpin et al. [3] solve very similar settings of the unlabeled problem for disc robots to the one treated in this paper, only with different assumptions and goal functions, and thus it is important to elaborate on these two techniques. Adler et al. [6] show that the unlabeled problem in the continuous domain can be transformed into a discrete *pebble-motion on graph* problem. Their construction guarantees that in case a solution to the former exists, it can be generated by solving a discrete pebble problem and adapted to the continuous domain. In particular, a motion of a pebble along an edge is transformed into a motion of a robot along a local path in the free space. Turpin et al. [3] find an assignment between starts and goals which minimize the longest path length traveled by any of the robots. Given such an assignment a shortest path between every start position to its assigned goal is generated. However, such paths still may result in collisions between the robots.

The authors show that collisions can be elegantly avoided by prioritizing the paths. Our current work follows to some extent a similar approach, although in our case some of the robots must slightly stray from the precomputed paths in order to guarantee completeness—a thing which makes our technique much more involved. This follows from the fact that we make milder assumptions on the input. Another difference is that the goal function of our algorithm is to minimize the total path length, which requires very different machinery than the one used by Turpin et al.

III. Preliminaries and algorithm overview

Our problem consists of moving m indistinguishable unit-disc robots in a workspace $\mathcal{W} \subset \mathbb{R}^2$ cluttered with polygonal obstacles, whose overall number of edges is n. We define $\mathcal{O} := \mathbb{R}^2 \setminus \mathcal{W}$ to be the complement of the workspace, and we call $\mathcal{O}$ the *obstacle space*. For given $r \in \mathbb{R}_+, x \in \mathbb{R}^2$, we define $\mathcal{B}_r(x)$ to be the open disc of radius r, centered at x. For given $r \in \mathbb{R}_+, X \subset \mathbb{R}^2$ we also define $\mathcal{B}_r(X) := \bigcup_{x \in X} \mathcal{B}_r(x)$.

We consider the unit-disc robots to be open sets. Thus a robot avoids collision with the obstacle space if and only if its center is at distance at least 1 from $\mathcal{O}$. More formally, the collection of all collision free configurations, termed the *free space*, can be expressed as $\mathcal{F} := \left\{x \in \mathbb{R}^2 : \mathcal{B}_1(x) \cap \mathcal{O} = \emptyset\right\}$. We also require the robots to avoid collisions with each other. Thus for every given two configurations $x, x' \in \mathcal{F}$ two distinct robots can be placed in x and x' only if $\|x - x'\| \geqslant 2$. Throughout this paper the notation $\|\cdot\|$ will indicate the L_2 norm.

In addition to the workspace $\mathcal{W}$ we are given sets $S = \{s_1, s_2, ..., s_m\}$ and $T = \{t_1, t_2, ..., t_m\}$ such that $S, T \subset \mathcal{F}$. These are respectively the sets of *start* and *goal* configurations of our m indistinguishable disc robots. The problem is now to plan a collision-free motion for m unit-disc robots such that each of them starts at a configuration in S and ends at a configuration in T. Since the robots are indistinguishable or *unlabeled*, it does not matter which robot ends up at which goal configuration, as longs as all the goal configurations are occupied at the end of the motion. Formally, we wish to find m paths $\pi_i : [0, 1] \to \mathcal{F}$, for $1 \leqslant i \leqslant m$, such that $\pi_i(0) = s_i$ and $\bigcup_{i=1}^{m} \pi_i(1) = T$. Furthermore, we are interested in finding a set of such paths which minimizes the expression $\sum_{i=1}^{m} |\pi_i|$, where $|\pi_i|$ represents the length of π_i in the L_2 norm.

A. Simplifying assumptions

By making the following simplifying assumptions (see Figure 1) we are able to show that our algorithm is complete and near-optimal. The first assumption that we make is identical to the one used by Adler et al. [6]. It requires that every pair of start or goal positions will be separated by a distance of at least 4:

$$\forall v, v' \in S \cup T, \quad \|v - v'\| \geqslant 4. \tag{1}$$

The motivation for the above assumption is the ability to prove the existence of a *standalone goal*—a goal position that does not block other paths, assuming that the paths minimize the total length of the solution.

We also need the following assumption in order to guarantee that the robots will be able switch targets, in case that a given assignment of robots to goals induces collision between the robots:

$$\forall v \in S \cup T \text{ and } \forall x \in \mathcal{O}, \quad \|v - x\| \geqslant \sqrt{5}. \tag{2}$$

B. Overview of algorithm

Here we provide an overview of our technique. Recall that our problem consists of S, T, which specify the set of start and goal positions for a collection of m unit-disc robots, and a workspace environment $\mathcal{W}$.

We describe the first iteration of our recursive algorithm. For every $s_i \in S, t_j \in T$ we find the shortest path $\gamma_i^j : [0, 1] \to \mathcal{F}$ from s_i to t_j. Among these m^2 paths we select a set of m paths $\Gamma = \{\gamma_1, \ldots, \gamma_m\}$, where $\gamma_i : [0, 1] \to \mathcal{F}$ for every $1 \leqslant i \leqslant m$, such that $\bigcup_{i=1}^{m} \{\gamma_i(0)\} = S, \bigcup_{i=1}^{m} \{\gamma_i(1)\} = T$. Furthermore, we require that Γ will be the path set that minimizes the total length of its paths under these conditions. Note that at this point we only require that the robots will not collide with obstacles, and do not worry about collisions between the robots. The generation of Γ is described in detail in Section V, Theorem 13.

In the next step we find a goal $t \in T$ which does not block paths in Γ that do not lead to t. We call such a t a *standalone goal*. Next, we find a start $s \in S$ from which a robot will be able to move to t without colliding with other robots situated in the rest of the start positions. We carefully select such a start s and generate the respective path to t in order to minimize the cost of the returned solution. We prepare the input to the next iteration of the algorithm by assigning $S := S \setminus \{s\}, T := T \setminus \{t\}$, and by treating the robot placed in t as an obstacle, i.e., $\mathcal{W} := \mathcal{W} \setminus \mathcal{B}_1(t)$.

IV. Theoretical foundations

In this section we establish several basic properties of the problem. Recall that our problem is defined for a workspace $\mathcal{W}$, whose free space for a single unit-disc robot is $\mathcal{F}$. Additionally, we have two sets of start and goal positions $S = \{s_1, \ldots, s_m\}, T = \{t_1, \ldots, t_m\}$, respectively.

The following lemma implies that if a robot moving from some start position $s \in S$ along a given path collides with a region $\mathcal{B}_1(t)$, for some other goal $t \in T$, then there exists another path $\gamma \in \mathcal{F}$ which moves the robot from s to t. In the context of our algorithm this lemma implies that if a path for a robot interferes with some other goal position, then the path can be modified such that it will move the robot to the interfering goal instead.

Lemma 1. *Let $v \in S \cup T$ and $x \in \mathcal{F}$ such that $\mathcal{B}_1(x) \cap \mathcal{B}_1(v) \neq \emptyset$. Then the straight-line path from x to v is contained in $\mathcal{F}$, i.e., $\overline{xv} \subset \mathcal{F}$.*

Proof: Here we use the fact that for every $o \in \mathcal{O}$ it holds that $\|v - o\| \geqslant \sqrt{5}$, which is the second separation assumption. Without loss of generality, assume that the straight-line segment from v to x is parallel to the x-axis. Denote by A and B the bottom points of the unit discs around v and x,

respectively (see Figure 2). Similarly, denote by C and D the top points of these discs. By definition of v, x, we know that $\|v - x\| < 2$. Thus, $\|v - B\| = \|v - D\| < \sqrt{5}$ (see dashed red segment). This implies that the rectangle defined by the points A, B, C, D is entirely contained in $\mathcal{W}$ (see orange square). Thus, the straight-line path $\overline{xv}$ is fully contained in $\mathcal{F}$. ■

Definition 2. *Let Γ be a set of m paths $\{\gamma_1, \ldots, \gamma_m\}$ such that for every $1 \leqslant i \leqslant m$, $\gamma_i : [0,1] \to \mathcal{F}$, $S = \bigcup_{i=1}^{m} \{\gamma_i(0)\}, T = \bigcup_{i=1}^{m} \{\gamma_i(1)\}$. We call Γ the* optimal-assignment path set *for $S, T, \mathcal{F}$, if it minimizes the expression $|\Gamma| := \sum_{i=1}^{m} |\gamma_i|$, over all such path sets.*

Note that Γ is not necessarily a feasible solution to our problem since at this stage we still ignore possible collisions between robots. Let $\Gamma = \{\gamma_1, \ldots, \gamma_m\}$ be an optimal-assignment path set for $S, T, \mathcal{F}$. Without loss of generality, assume that for every $1 \leqslant i \leqslant m$, $\gamma_i(0) = s_i, \gamma_i(1) = t_i$.

Definition 3. *Given an optimal-assignment path set Γ, we call $t_k \in T$ a* standalone goal, *for some $1 \leqslant k \leqslant m$, if for every $\gamma_i \in \Gamma, i \neq k$ it holds that $\mathcal{B}_1(t_k) \cap \mathcal{B}_1(\gamma_i) = \emptyset$.*

Standalone goals play a crucial role in our algorithm. We first show that at least one such goal always exists.

Theorem 4. *Let Γ be an optimal-assignment path set. Then there exists a standalone goal.*

Proof: Assume by contradiction that every goal t_i interferes with some path $\gamma_j \in \Gamma, i \neq j$. This implies that there is a circular interference, i.e., there exist $\ell \leqslant m$ goals, which we denote, for simplicity and without loss of generality, as $t_1, \ldots, t_\ell$ such that for every $1 < i \leqslant \ell$, $\mathcal{B}_1(t_i) \cap \mathcal{B}_1(\gamma_{i-1}) \neq \emptyset$, and $\mathcal{B}_1(t_1) \cap \mathcal{B}_1(\gamma_\ell) \neq \emptyset$. More formally, let $\mathcal{I}$ be a directed graph vertices are T. For every t_i which interferes with a path γ_j, where $j \neq i$ we draw an edge from t_i to t_j. If there exists no standalone goal then $\mathcal{I}$ has a directed cycle of size greater than one. This is due to the fact that if $\mathcal{I}$ were *directed acyclic* then it should have had a node whose *out degree* is zero.

We show that in this case, the paths $\gamma_1, \ldots, \gamma_\ell$ do not induce the minimal assignment for the starts and goals $\{s_1, \ldots, s_\ell\}, \{t_1, \ldots, t_\ell\}$. This would imply that Γ is not the optimal assignment path set. We claim that instead of assigning s_i to t_i we may assign s_i to t_{i+1} for $1 \leqslant i < \ell$ and from s_ℓ to t_1 and get a collection of paths $\gamma'_1, \ldots, \gamma'_\ell$, such that $|\gamma'_i| < |\gamma_i|$ for every $1 \leqslant i \leqslant \ell$. Denote by $x \in \gamma_i$ the first interference point with t_{i+1} along γ_i. Additionally, denote by γ_i^x the subpath of γ_i that starts with s_i and ends with x. Define γ'_i to be the concatenation of γ_i^x and xt_{i+1}.

We need to show first that $\gamma'_i \subset \mathcal{F}$ and that $|\gamma'_i| < |\gamma_i|$. The subpath γ_i^x is obstacle-collision free, and so is $\overline{xt_{i+1}}$ according to Lemma 1. Thus, γ'_i is free as well. Now, note that $\|x - t_{i+1}\| < 2$ and $\|t_{i+1} - t_i\| \geqslant 4$. Thus, by the triangle inequality $\|x - t_i\| \geqslant 2$. This finishes our proof. ■

We introduce the notion of 0-*hop* and 1-*hop* paths. Informally, a 0-hop path is a path assigned to the standalone goal which is not blocked by any start position.

Definition 5. *Let t_k be a standalone goal. The path γ_k is called a 0-hop path if for every $s_i \in S, i \neq k$ it follows that $\mathcal{B}_1(s_i) \cap \mathcal{B}_1(\gamma_k) = \emptyset$.*

Definition 6. *Let t_k be a standalone goal, and suppose that γ_k is not 0-hop. Let $x \in \gamma_k$ be the farthest point along γ_k for which there exists $s_i \in S$ such that $\mathcal{B}_1(s_i) \cap \mathcal{B}_1(x) \neq \emptyset$. The 1-hop path, which is denoted by H_i^k, is a concatenation of the straight-line path $\overline{s_i x}$ and a subpath of γ_k that starts at x and ends in t_k.*

Namelly, the 1-hop path moves the robot situated in s_i, which interferes with γ_k, to t_k (see Figure 3). The following theorem shows that a robot situated in s_i that blocks the 0-hop path can be moved to t_k without inducing collisions with other robots.

Theorem 7. *Let t_k be a standalone goal. Suppose that γ_k is not 0-hop and let H_i^k be the 1-hop path from s_i to t_k. Then it holds that for every $s_j \in S, j \neq i, k$, $\mathcal{B}_1(H_i^k) \cap \mathcal{B}_1(s_j) = \emptyset$.*

Proof: By separation of start positions, and by definition of x (see Definition 6), there exists a single start position s_i for which $\mathcal{B}_1(x) \cap \mathcal{B}_1(s_i) \neq \emptyset$. By Lemma 1, and the separation condition, the path $\overline{s_i x}$, does not induce collisions between a robot moving along it and other robots placed in s_j, for $1 \leqslant j \leqslant m, j \neq i, k$. Finally, by definition of x, the rest of the path H_i^k, which is a subpath of γ_k, is free of collisions with a robot situated in s_j. ■

After moving a robot from s_i to t_k we have to ensure that some other robot can reach t_i. We define the following *switch* path (see Figure 3).

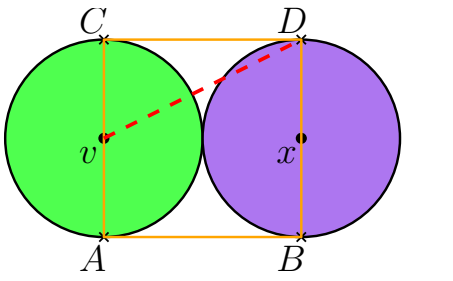


Fig. 2: Illustration of the proof of Lemma 1.

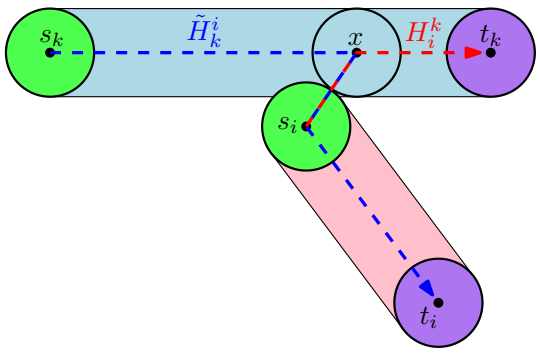


Fig. 3: Example of 1-hop and switch paths. t_k is a standalone goal. The 1-hop path H_i^k is drawn as a dashed red curve, while the switch path $\tilde{H}_k^i$ is drawn as a dashed blue curve.

Definition 8. *Let t_k be a standalone goal and suppose that γ_k is not 0-hop. The* switch *path $\tilde{H}_k^i$ is a concatenation of the following paths: (1) the subpath of γ_k that starts in s_k and ends in x; (2) $\overline{xs_i}$; (3) γ_i.*

Lemma 9. *Let $\tilde{H}_i^k$ be a switch path. Then $\mathcal{B}_1(\tilde{H}_i^k) \cap \mathcal{B}_1(t_k) = \emptyset$.*

Proof: $\tilde{H}_i^k$ can potentially interfere with t_k only along $\overline{xs_i}$, due to the definition of x. For any $y \in \overline{xs_i}$ it follows that $\|y - s_i\| < 2$. If $\|y - t_k\| < 2$ then it follows that $\|s_i - t_k\| < 4$, which is a contradiction. ∎

Corollary 10. *Let $H_i^k, \tilde{H}_k^i$ be the 1-hop and switch paths, respectively. Then $|H_i^k| + |\tilde{H}_k^i| = |\gamma_i| + |\gamma_k| + 4$.*

V. Near-optimal algorithm for unlabeled planning

In this section we describe our algorithm for the unlabeled multi-robot motion planning problem of unit-disc robots moving amid polygonal obstacle and establish its completeness. Additionally, we bound the cost of the returned solution. Finally, we analyze the complexity of the algorithm.

A. The algorithm

We describe our recursive algorithm, which returns a set of m paths Π. Recall that the input consists of S, T and a workspace $\mathcal{W}$, which induces the free space $\mathcal{F}$. The algorithm first produces the optimal-assignment path set Γ. Let OPT be the optimal solution cost and note that $|\Gamma| \leqslant$ OPT since $|\Gamma|$ is a lower bound on the actual cost, as interactions between the robots might increase the traversed distance.

Let t_k be a standalone goal, which exists according to Theorem 4. Suppose that the 0-hop path γ_k is not blocked by any other robot located in a start position. Then, γ_k is added to Π and the algorithm is run recursively on the input $S' := S \setminus \{s_k\}, T' := T \setminus \{t_k\}$, with the workspace $\mathcal{W}' := \mathcal{W} \setminus \mathcal{B}_1(t_k)$, which results in the free space $\mathcal{F}' := \mathcal{F} \setminus \mathcal{B}_2(t_k)$.

Alternatively, in case that γ_k is blocked, i.e., in interference with some $s_i \in S, i \neq k$, the algorithm produces the 1-hop path H_i^k, as described in Theorem 7, and adds it to Π. Then, the algorithm is run recursively on the input $S' := S \setminus \{s_i\}, T' := T \setminus \{t_k\}$, with the workspace $\mathcal{W}' := \mathcal{W} \setminus \mathcal{B}_1(t_k)$, and the free space $\mathcal{F}' := \mathcal{F} \setminus \mathcal{B}_2(t_k)$.

B. Completeness and near-optimality

We first show that the algorithm is guaranteed to find a solution, if one exists, or report that none exists otherwise.

Theorem 11. *Given an input $S, T, \mathcal{W}$, which complies with assumptions 1 ,2 (Section III-A), and for which the number of start and goal positions for every connected component of $\mathcal{F}$ is the same, the algorithm is guaranteed to find a solution for the unlabeled multi-robot motion-planning problem.*

Proof: Consider the first iteration of the algorithm. Let $\Gamma := \{\gamma_1, \ldots, \gamma_m\}$ be the optimal-assignment path set and let t_k be a standalone goal.

Suppose that γ_k is a 0-hop path. Then, for every $j \neq k$, $\gamma_j \subset \mathcal{F} \setminus \mathcal{B}_2(t_k)$, since t_k is a standalone goal. Now suppose that γ_k is not a 0-hop path. By Theorem 7 the 1-hop path H_i^k from s_i to t_k does not collide with any other start position. By Lemma 9 $\tilde{H}_k^i \subset \mathcal{F} \setminus \mathcal{B}_2(t_k)$. Additionally, notice that for every $j \neq i, k$, $\gamma_j \subset \mathcal{F} \setminus \mathcal{B}_2(t_k)$.

Thus, in any of the two situations, the removal of the standalone goal does not separate between start and goal configurations that are in the same connected component of $\mathcal{F}$. Note that in the first level of the recursion the existence of a standalone goal t_k guarantees the existence of a path to t_k which does not collide with the other robots. This is possible due to assumptions 1, 2. Thus, in order to ensure the success of the following recursions, we need to show that these assumptions are not violated. First, note that assumption 1 is always maintained, since we do not move existing start and goal positions. Secondly, when a robot situated in t_k is treated as an obstacle added to the set $\mathcal{O} := \mathcal{O} \cup \mathcal{B}_1(t_k)$ (or conversely removed from $\mathcal{F}$, as described above), the first assumption induces the second. ∎

We proceed to prove the near-optimality of our solution.

Theorem 12. *Let Π be the solution returned by our algorithm and let Γ be the optimal-assignment path set for $S, T, \mathcal{W}$. Then $|\Pi| \leqslant$ OPT $+ 4m$, where OPT is the optimal solution cost.*

Proof: Let $\Gamma := \{\gamma_1, \ldots, \gamma_m\}$ be the optimal-assignment path set for the input $S, T, \mathcal{W}$, and let t_k be a standalone. Additionally, assume that the algorithm generated the 1-hop path H_i^k, since γ_k was blocked (the case when γ_k is not blocked is simpler to analyze). Let $\tilde{H}_k^i$ be the switch path from s_k to t_i. Similarly, denote by Γ' the optimal-assignment path set for $S' := S \setminus \{s_i\}, T' := T \setminus \{t_k\}$ and the workspace $\mathcal{W}' := \mathcal{W} \setminus \mathcal{B}_1(t_k)$, which induces the free space $\mathcal{F}'$.

We will show that $|\Gamma'| + |H_i^k| \leqslant$ OPT $+ 4$. As mentioned in Theorem 11, for every $j \neq i, k$, it follows that for $\gamma_j \in \Gamma$ and $\gamma_j \subset \mathcal{F}'$. We also showed that the same holds for $\tilde{H}_k^i$, i.e., $\tilde{H}_k^i \subset \mathcal{F}'$. Thus, the path set $R := (\Gamma \setminus \{\gamma_i, \gamma_k\}) \cup \{\tilde{H}_k^i\}$ represents a valid assignment for $S', T', \mathcal{F}'$, even though it might not be optimal. This means that $|\Gamma'| \leqslant |R|$. Thus,

$$\begin{aligned} |\Gamma'| + |H_i^k| &\leqslant |R| + |H_i^k| = \sum_{j \neq i,k}^{m} |\gamma_j| + |\tilde{H}_k^i| + |H_i^k| \\ &\leqslant \sum_{k=1}^{m} |\gamma_k| + 4 = |\Gamma| + 4 \leqslant \text{OPT} + 4, \end{aligned}$$

where the third step is due to Corollary 10. Thus, every level of the recursion introduces and additive error factor of 4. Repeating this process for the m iterations we obtain $|\Pi| \leqslant$ OPT $+ 4m$. ∎

C. Complexity Analysis

We analyze the complexity of the algorithm. In order to do so, we have to carefully consider the operations that are performed in every iteration of the algorithm.

Theorem 13. *Given m unlabeled unit-disc robots operating in a polygonal workspace with n vertices, the algorithm described*

above returns a solution, or reports that none exists otherwise, with a running time of $\tilde{O}\left(m^4 + m^2n^2\right)$.

Proof: Let us consider a specific iteration j. The input of this iteration consists of $m-j+1$ start positions S_j and $m-j+1$ goal positions T_j. The workspace region of this iteration is defined to be $\mathcal{W}_j := \mathcal{W} \setminus (\mathcal{B}_1(T \setminus T_j))$, which induces the free space $\mathcal{F}_j$. Note that $S_1 \equiv S, T_1 \equiv T, \mathcal{F}_1 \equiv \mathcal{F}, \Gamma_1 \equiv \Gamma$.

In order to find the optimal-assignment path set Γ_j for $S_j, T_j, \mathcal{F}_j$ one has to first find the shortest path in $\mathcal{F}_j$ for every pair of start and goal positions $s \in S_j, t \in T_j$. Given the costs of all those combinations, the Hungarian algorithm [38][2] finds the optimal assignment, and so Γ_j is produced. In the next step, a standalone goal t_k is identified and it is checked whether the 0-hop path leading to t_k is clear, in which case γ_k is added to Π. If it is not, one needs to find the last start position that interferes with this path and generate the respective 1-hop path, which will be included in Π.

The complexity of finding a shortest path for a disc depends on the complexity of the workspace, which in our case is $O(n+j)$. This task is equivalent to finding a shortest path for a point robot in $\mathcal{F}_j$. The generation of $\mathcal{F}_j$ can be done in $O\left((n+j)\log^2(n+j)\right)$ time complexity, and the overall complexity of this structure would be $O(m+j)$ [39]. A common approach for finding shortest paths in the plane is to construct a *visibility graph* [40] which encapsulates information between every pair of vertices of a given arrangement of segments, while avoiding crossings with the segments. In our case, the arrangement should include $\mathcal{F}_j$ as well as all the points from S_j and T_j. Thus, we would need to generate a visibility graph over a generalized polygon of total complexity $O(m+n)$. This can be done in $\tilde{O}\left((m+n)^2\right) = \tilde{O}(m^2+n^2)$ [40][3]. Given the visibility graph with $O(m+n)$ vertices and $O(m^2+n^2)$ edges we find for each $s \in S_j$ the shortest path to every $t \in T_j$. For each s we run the Dijkstra algorithm which requires $O(m^2+n^2)$ time, and finds the shortest path from the given s to any $t \in T_j$. Since $|S_j| = m-j+1$, the total running time of finding the shortest path from every start to every goal is $O((m-j)(m^2+n^2))$.

To find Γ_j we employ the Hungarian algorithm [38], which runs in $O\left((m-j)^3\right)$ time. Now, given Γ_j we wish to find a standalone goal t_k. We first note that the complexity of each path in Γ_j is bounded by the complexity of $\mathcal{F}_j$, which is $O(n+j)$. For every $t_i \in T_j, \gamma_{i'} \in \Gamma_j, i \neq i'$ we check whether $\mathcal{B}_1(t_i) \cap \mathcal{B}_1(\gamma_{i'}) \neq \emptyset$. This step has a running time of $\tilde{O}((m-j)(m-j)(n+j)) = \tilde{O}((m-j)^2(n+j))$. Finding the last closest blocking start from S_j of the path γ_k takes additional $O((m-j)(n+j))$ time by going over all starts in S_j and comparing the distance of their interference point with γ_k. Thus, the overall complexity of a given iteration j is

$$\tilde{O}\left((m-j)(m^2+n^2)+(m-j)^3+(m-j)^2(n+j)+(m-j)^2\right) = \tilde{O}\left((m-j)(m^2+n^2)\right).$$

Note that the cost of the different components is absorbed in the cost of calculating the shortest paths. We conclude with the running time of the entire algorithm:

$$\tilde{O}\left(\sum_{j=1}^{m}(m-j)(m^2+n^2)\right) = \tilde{O}\left(m^4+m^2n^2\right).$$

■

VI. Experimental results

We implemented the algorithm and evaluated its performance on various challenging scenarios.

A. Implementation details

First, we wish to emphasize that our implementation deals with geometric primitives, e.g., polygons and discs, and does not rely on any graph discretization of the problem. The implementation relies on exact geometric methods that are provided by CGAL [41]. As such, it is complete, robust and deterministic. In addition to that, the implementation is parameter-free.

We implemented the algorithm in C++ and relied heavily on CGAL for geometric computing, and in particular on the `Arrangement_2` package [42]. Generation of the free space was done using *Minkowski sums*, while shortest paths were generated using *visibility graphs* [40]. In order to find the optimal assignment, we used a C++ implementation of the *Hungarian algorithm* [38], available at [43].

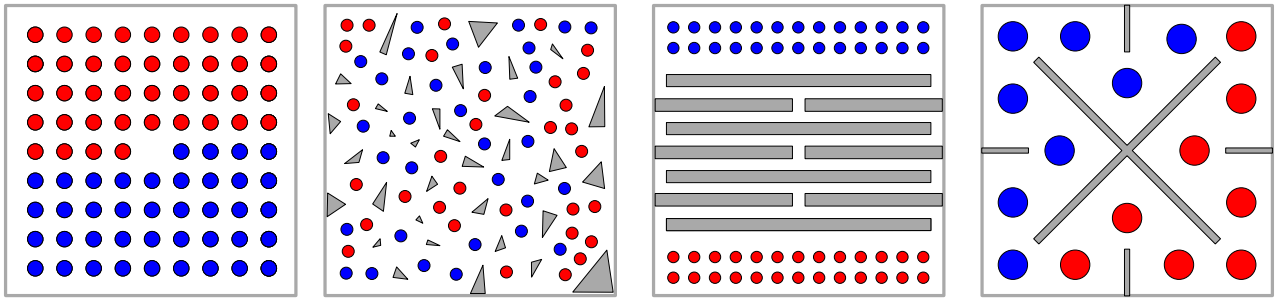
B. Test scenarios

We report in Table I on the running time of the algorithm for the four scenarios depicted in Figure 4. The grid scenario (Figure 4a) is used to illustrate the performance of the algorithm in a sterile obstacles-free workspace. The triangles and cross scenarios (Figure 4b,4d) include multiple obstacles, which have a tremendous affect on the performance of the algorithm. The maze scenario (Figure 4c), which also includes multiple obstacles, has several narrow passages.

It is evident that the running time is dominated by shortest-path calculations (see Theorem 13). The second largest contributor to the running time of the algorithm, is the calculation and maintenance of the configuration space, which includes the update of the free space for every iteration. A interesting relation is found between the overall complexity of finding the optimal assignment ($O(m^4)$) and its modest contribution to the actual running time in practice, when compared to the other components. This can be explained by the fact that the implementation of the Hungarian algorithm [43] employs floating-point arithmetic, while the geometric operations, e.g., shortest pathfinding, rely on exact geometric kernels, which have unlimited precision [44]. It is noteworthy that the algorithm produces solutions whose cost is very close to the optimal cost.

[2]Also known as the Kuhn-Munkres algorithm.

[3]We note that in our setting the arrangement consists not only of straight-line segments, but also of circular arcs which are induced by robots situated in target positions. Yet, the algorithm constructing the visibility graph can treat such cases as well, while still guaranteeing the (near-)quadratic run-time complexity mentioned above.



(a) grid (b) triangles (c) maze (d) cross

Fig. 4: Test scenarios. Obstacles are represented as gray polygons. Discs represent robots placed in start (red) and goal (blue) positions. (a) Grid scenario with 40 robots. (b) Triangles scenario with 32 robots and multiple triangular obstacles. (c) Maze scenario with with 26 robots; robots need to pass through a collection of narrow passages in order to reach their goals. (d) Cross scenario with 8 robots and several wall obstacles.

To gather a better understanding of the running time of the algorithm, we also report on the running time of each iteration for the triangles scenario (see Figure 5a). For every iteration $1 \leqslant j \leqslant 32$ we report on the running time of maintaining the configuration space, and finding shortest paths.

Another important aspect of the algorithm is the relation between the number of robots and the performance. To test this relation we performed the following experiment, which is based on the triangles scenario as well. We report in Figure 5b how the performance is affected by an increase or decrease in the number of robots. In order to maintain a similar level of density between the various tests we increase the radius of the robots to the maximal allowed radius which abides by the separation assumptions.

The algorithm and its current implementation can deal with rather complex scenarios. However, it is clear that there is a limit to the number of robots or workspace complexity with which it can cope, due the relatively high degree of the polynomial, in the running time complexity.

VII. Discussion and Future Work

A. Relaxing the separation conditions

The algorithm presented in this paper requires that two conditions will hold. First it is assumed that every pair of start or goal position will be separated by a distance of at least 4. The second condition requires that every start or goal positions will be separated from an obstacle by a distance of at least $\sqrt{5} \approx 2.236$. We believe that the obstacles-separation factor can be lowered to $\sqrt{13 - 6\sqrt{3}} \approx 1.614$ using a tighter mathematical analysis. While the proof for Lemma 1 would change, the rest of the proofs will not require any special modification. In the near future we aim to consider less strict separation requirements such as reducing the robots separation to 3 and removing the requirement that start positions will be separated from goal positions. However, it seems that the machinery described here will not suffice for the tighter setting and different tools will be required.

We note that without adequate obstacle clearance, it is in fact impossible for any algorithm to guarantee a maximum $O(m)$ deviation from the cost of an optimal-assignment path set, regardless of the separation between start and goals. For instance, see the gadget in Figure 6. The two pairs of (blue) start and (red) goal positions satisfy the 4 separation requirement. The disc starting at the bottom left, which we call disc 1, is matched by the optimal assignment with the goal on the top left of the figure, and the disc starting in the middle, which we call disc 2, is matched with the goal on the right. This is due to the fact that disc 1 and disc 2 reside in two disjoint components of the free space. For disc 1 to go through the lower tunnel, disc 2 must move non-trivially (e.g., to the location of the green discs). Let this nontrivial distance be δ. Then, as disc 1 passes through the upper tunnel, another nontrivial move of distance δ must be performed by disc 2 to clear the way. Now, we remove disc 1 from the gadget and stack $m/2$ of the resulting gadget vertically. If we require the rest $m/2$ discs to pass through the stacked construction, a total distance penalty of $\Omega(m^2\delta)$ is incurred.

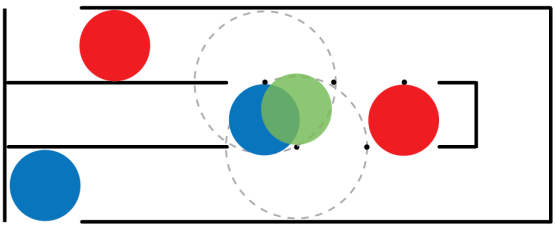
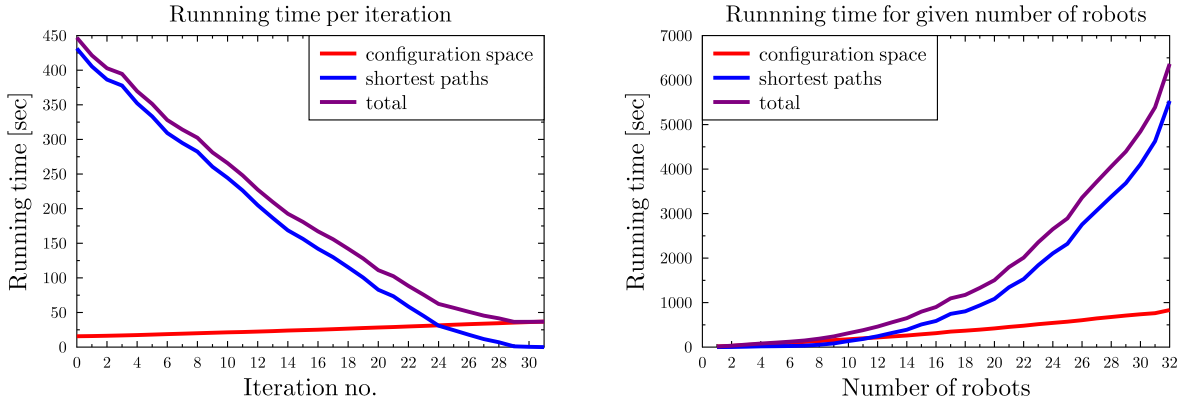


Fig. 6: Illustration of a scenario in which every solution has a deviation of $\Omega(m^2)$ from the cost of the optimal-assignment path set.

B. Improving the algorithm

It is evident from our experiments that most of the running time of the algorithm in practice is devoted to finding shortest paths. To be specific, in each iteration the algorithm performs a shortest-path search between every pair of start and target positions. This entire process is performed only for the sake of finding a single standalone goal. It will be interesting to investigate whether more information can be extracted from the optimal assignment returned by the Hungarian algorithm.



(a) (b)

Fig. 5: Graphs depicting the running time of maintaining the configuration space, and finding shortest path, for the triangles scenario (Figure 4b). (a) Running time is reported separately for every iteration of the algorithm. (b) The behavior of the algorithm for a growing number of robots in the triangles scenario is depicted.

		Scenarios			
		Grid	Triangles	Maze	Cross
scenario properties	robots	40	32	26	8
	workspace vertices	4	118	56	44
	robot radius	0.05	0.034	0.035	0.09
running time (sec.)	configuration space	261	832	141	17
	shortest paths	50	5532	658	1
	optimal assignment	0.016	0.001	0.016	0
	standalone goal	0.001	0.016	0.119	0
	total	**311**	**6394**	**800**	**19**
cost	lower bound	36.33	10.41	154.86	12.11
	actual cost	37.16	10.69	155.21	12.4
algorithm's behavior	0-hop paths	25	32	14	7
	1-hop paths	1	0	12	1

TABLE I: Results of our algorithm for the scenarios depicted in Figure 4. We first describe the properties of each scenario, which consist of the number of robots, the complexity of the workspace (n), and the robot's radius (every scenario is bounded by the $[-1, 1]^2$ square). Then, we report the running time (in seconds) of the different components of the algorithm: construction and maintenance of the configuration space; calculation of shortest paths; calculation of the optimal assignment; search for a standalone goal. Then, we report the lower-bound cost of the solution and the cost of the actual solution returned by the algorithm. Finally, we report the number of iterations for which the algorithm returned a 0-hop or a 1-hop path.

For instance, could there be two or more standalone goals to which there are separate non-interfering paths?

C. Implementation in 3D

For simplicity of presentation, our algorithm is discussed for the planar setting. Yet, its completeness and near-optimality guarantees hold also for the three-dimensional case with ball robots under the two separation assumptions used in this paper. A main requirement for a correct implementation of the algorithm is the ability to produce shortest paths for a single robot moving amid static obstacles. While this task does not pose particular challenges in the planar case, it becomes extremely demanding in 3D. In particular, finding a shortest path for a polyhedral robot translating amid polyhedral obstacles is known to be NP-Hard [45]. It would be interesting to investigate whether our algorithm can be extended to work with approximate shortest paths.

References

[1] G. Calinescu, A. Dumitrescu, and J. Pach, "Reconfigurations in graphs and grids," *SIAM Journal on Discrete Mathematics*, vol. 22, no. 1, pp. 124–138, 2008.

[2] J. Yu and S. M. LaValle, "Multi-agent path planning and network flow," in *Workshop on the Algorithmic Foundations of Robotics (WAFR), MIT, Cambridge, Massachusetts, USA*, 2012, pp. 157–173.

[3] M. Turpin, N. Michael, and V. Kumar, "Concurrent assignment and planning of trajectories for large teams of interchangeable robots," in *International Conference on Robotics and Automation (ICRA)*, 2013.

[4] J. Yu and S. M. LaValle, "Distance optimal formation control on graphs with a tight convergence time guarantee," in *Proceedings IEEE Conference on Decision & Control*, 2012, pp. 4023–4028.

[5] K. Solovey and D. Halperin, "On the hardness of unlabeled multi-robot motion planning," in *Robotics: Science and Systems (RSS)*, 2015, these proceedings.

[6] A. Adler, M. de Berg, D. Halperin, and K. Solovey, "Efficient multi-robot motion planning for unlabeled discs in simple polygons," in *Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2014.

[7] M. Turpin, N. Michael, and V. Kumar, "Trajectory planning and assignment in multirobot systems," in *Workshop on the Algorithmic Foundations of Robotics (WAFR), MIT, Cambridge, Massachusetts, USA*, 2012, pp. 175–190.

[8] I. Karamouzas, R. Geraerts, and A. F. van der Stappen, "Space-time group motion planning," in *Workshop on the Algorithmic Foundations of Robotics (WAFR), MIT, Cambridge, Massachusetts, USA*, 2012, pp. 227–243.

[9] A. Krontiris, R. Luna, and K. E. Bekris, "From feasibility tests to path planners for multi-agent pathfinding," in *Symposium on Combinatorial Search, (SOCS), Leavenworth, Washington, USA*, 2013.

[10] M. Turpin, K. Mohta, N. Michael, and V. Kumar, "Goal assignment and trajectory planning for large teams of aerial robots," in *Robotics: Science and Systems*, 2013.

[11] J. Yu and S. M. LaValle, "Planning optimal paths for multiple robots on graphs," in *International Conference on Robotics and Automation (ICRA)*, 2013, pp. 3612–3617.

[12] P. R. Wurman, R. D'Andrea, and M. Mountz, "Coordinating hundreds of cooperative, autonomous vehicles in warehouses," *AI Magazine*, vol. 29, no. 1, pp. 9–19, 2008.

[13] J. E. Hopcroft, J. T. Schwartz, and M. Sharir, "On the complexity of motion planning for multiple independent objects; PSPACE-hardness of the "Warehouseman's problem"," *International Journal of Robotics Research*, vol. 3, no. 4, pp. 76–88, 1984.

[14] P. G. Spirakis and C.-K. Yap, "Strong NP-hardness of moving many discs," *Information Processing Letters*, vol. 19, no. 1, pp. 55–59, 1984.

[15] J. T. Schwartz and M. Sharir, "On the piano movers' problem: III. coordinating the motion of several independent bodies: the special case of circular bodies moving amidst polygonal barriers," *International Journal of Robotics Research*, vol. 2, no. 3, pp. 46–75, 1983.

[16] M. A. Erdmann and T. Lozano-Pérez, "On multiple moving objects," in *International Conference on Robotics and Automation (ICRA)*, 1986, pp. 1419–1424.

[17] S. M. LaValle and S. A. Hutchinson, "Optimal motion planning for multiple robots having independent goals," *IEEE Transactions on Robotics & Automation*, vol. 14, no. 6, pp. 912–925, 1998.

[18] J. Peng and S. Akella, "Coordinating multiple robots with kinodynamic constraints along specified paths," in *Algorithmic Foundations of Robotics V*, J.-D. Boissonat, J. Burdick, K. Goldberg, and S. Hutchinson, Eds. Berlin: Springer-Verlag, 2002, pp. 221–237.

[19] J. van den Berg and M. H. Overmars, "Prioritized motion planning for multiple robots," in *International Conference on Intelligent Robots and Systems (IROS)*, 2005, pp. 430 – 435.

[20] R. Ghrist, J. M. O'Kane, and S. M. LaValle, "Computing Pareto Optimal Coordinations on Roadmaps," *International Journal of Robotics Research*, vol. 24, no. 11, pp. 997–1010, 2005.

[21] J. van den Berg, J. Snoeyink, M. Lin, and D. Manocha, "Centralized path planning for multiple robots: Optimal decoupling into sequential plans," in *Robotics: Science and Systems*, 2009.

[22] P. A. O'Donnell and T. Lozano-Pérez, "Deadlock-free and collision-free coordination of two robot manipulators," in *International Conference on Robotics and Automation (ICRA)*, 1989, pp. 484–489.

[23] P. Švestka and M. H. Overmars, "Coordinated path planning for multiple robots," *Robotics and Autonomous Systems*, vol. 23, no. 3, pp. 125–152, 1998.

[24] B. Aronov, M. de Berg, A. F. van der Stappen, P. Švestka, and J. Vleugels, "Motion planning for multiple robots," *Discrete & Computational Geometry*, vol. 22, no. 4, pp. 505–525, 1999.

[25] S. Kloder and S. Hutchinson, "Path planning for permutation-invariant multirobot formations," *IEEE Transactions on Robotics*, vol. 22, no. 4, pp. 650–665, 2006.

[26] O. Salzman, M. Hemmer, and D. Halperin, "On the power of manifold samples in exploring configuration spaces and the dimensionality of narrow passages," in *Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2012, pp. 313–329.

[27] K. Solovey, O. Salzman, and D. Halperin, "Finding a needle in an exponential haystack: Discrete RRT for exploration of implicit roadmaps in multi-robot motion planning," in *Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2014.

[28] G. Wagner and H. Choset, "Subdimensional expansion for multirobot path planning," *Artif. Intell.*, vol. 219, pp. 1–24, 2015.

[29] E. J. Griffith and S. Akella, "Coordinating multiple droplets in planar array digital microfluidic systems," *International Journal of Robotics Research*, vol. 24, no. 11, pp. 933–949, 2005.

[30] D. Kornhauser, G. Miller, and P. Spirakis, "Coordinating pebble motion on graphs, the diameter of permutation groups, and applications," in *Foundations of Computer Science (FOCS)*. IEEE Computer Society, 1984, pp. 241–250.

[31] V. Auletta, A. Monti, M. Parente, and P. Persiano, "A linear time algorithm for the feasibility of pebble motion on trees," in *Scandinavian Symposium and Workshops on Algorithm Theory (SWAT)*, 1996, pp. 259–270.

[32] G. Goraly and R. Hassin, "Multi-color pebble motion on graphs," *Algorithmica*, vol. 58, no. 3, pp. 610–636, 2010.

[33] J. Yu, "A linear time algorithm for the feasibility of pebble motion on graphs," *CoRR*, vol. abs/1301.2342, 2013.

[34] D. Kornhauser, "Coordinating pebble motion on graphs, the diameter of permutation groups, and applications," M.Sc. Thesis, Department of Electrical Engineering and Computer Scienec, Massachusetts Institute of Technology, 1984.

[35] R. Luna and K. E. Bekris, "An efficient and complete approach for cooperative path-finding," in *Conference on Artificial Intelligence, San Francisco, California, USA*, 2011.

[36] M. Katsev, J. Yu, and S. M. LaValle, "Efficient formation path planning on large graphs," in *2013 IEEE International Conference on Robotics and Automation, Karlsruhe, Germany*, 2013, pp. 3606–3611.

[37] K. Solovey and D. Halperin, "*k*-Color multi-robot motion planning," *International Journal of Robotic Research*, vol. 33, no. 1, pp. 82–97, 2014.

[38] E. L. Lawler, *Combinatorial optimization: networks and matroids*. Courier Dover Publications, 1976.

[39] K. Kedem, R. Livne, J. Pach, and M. Sharir, "On the union of jordan regions and collision-free translational motion amidst polygonal obstacles," *Discrete & Computational Geometry*, vol. 1, pp. 59–70, 1986.

[40] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars, *Computational Geometry: Algorithms and Applications*, 3rd ed. Springer-Verlag, 2008.

[41] "CGAL, Computational Geometry Algorithms Library," http://www.cgal.org.

[42] E. Fogel, D. Halperin, and R. Wein, *CGAL Arrangements and Their Applications - A Step-by-Step Guide*, ser. Geometry and computing. Springer, 2012, vol. 7.

[43] "Implementation of the Kuhn-Munkres algorithm," https://github.com/saebyn/munkres-cpp.

[44] L. Kettner, K. Mehlhorn, S. Pion, S. Schirra, and C. Yap, "Classroom examples of robustness problems in geometric computations," *Comput. Geom.*, vol. 40, no. 1, pp. 61–78, 2008.

[45] J. F. Canny and J. H. Reif, "New lower bound techniques for robot motion planning problems," in *28th Annual Symposium on Foundations of Computer Science, Los Angeles, California, USA*, 1987, pp. 49–60.

[46] J. Ayala and H. Rubinstein, "A geometric approach to shortest bounded curvature paths," *arXiv preprint arXiv:1403.4899*, 2014.

5

Finding a Needle in an Exponential Haystack: Discrete RRT for Exploration of Implicit Roadmaps in Multi-Robot Motion Planning

Article

The International Journal of Robotics Research
2016, Vol. 35(5) 501–513

DOI: 10.1177/0278364915615688
ijr.sagepub.com
SAGE

Finding a needle in an exponential haystack: Discrete RRT for exploration of implicit roadmaps in multi-robot motion planning

Kiril Solovey*, Oren Salzman* and Dan Halperin

Abstract

We present a sampling-based framework for multi-robot motion planning. which combines an implicit representation of roadmaps for multi-robot motion planning with a novel approach for pathfinding in geometrically embedded graphs tailored for our setting. Our pathfinding algorithm, discrete-RRT (dRRT)*, is an adaptation of the celebrated RRT algorithm for the discrete case of a graph, and it enables a rapid exploration of the high-dimensional configuration space by carefully walking through an implicit representation of the tensor product of roadmaps for the individual robots. We demonstrate our approach experimentally on scenarios that involve as many as 60 degrees of freedom and on scenarios that require tight coordination between robots. On most of these scenarios our algorithm is faster by a factor of at least 10 when compared to existing algorithms that we are aware of.*

Keywords

Multi-robot motion planning, sampling-based motion planning

1. Introduction

Multi-robot motion planning is a fundamental problem in robotics and has been extensively studied for several decades. In this work we are concerned with finding paths for a group of robots, operating in the same workspace, moving from start to target positions while avoiding collisions with obstacles, as well as with each other. We consider the continuous formulation of the problem, where the robots and obstacles are geometric entities and the robots operate in a configuration space, e.g. $\mathbb{R}^d$ (as opposed to the discrete variant, sometimes called the *pebble motion* problem, where the robots move on a graph (Auletta et al., 1996; Goraly and Hassin, 2010; Kornhauser, 1984; Luna and Bekris, 2011)). Moreover, we assume that each robot has its own start and target positions, as opposed to the unlabeled case (Adler et al., 2015; Kloder and Hutchinson, 2005; Solovey and Halperin, 2014, 2015; Solovey et al., 2015; Turpin et al., 2014).

1.1. Previous work

We assume familiarity with the basic terminology of motion planning. For background, see Choset et al. (2005) or LaValle (2006). Initial work on motion planning aimed to develop *complete* algorithms, which guarantee to find a solution when one exists or report that none exists otherwise. Such algorithms for the multi-robot case exist (Schwartz and Sharir, 1983; Sharir and Sifrony, 1991; Yap, 1984), yet are exponential in the number of robots. The exponential running time, which may be unavoidable (Hearn and Demaine, 2005; Hopcroft et al., 1984; Solovey et al., 2015; Spirakis and Yap, 1984) can be attributed to the high number of *degrees of freedom* (*DOFs*)—the sum of the DOFs of the individual robots. Aronov et al. (1999) showed that for two or three robots, the number of DOFs may be slightly reduced by constructing a path where the robots move while maintaining contact with each other. A more general approach to reduce the number of DOFs was suggested by van den Berg et al. (2009). In their work, the multi-robot motion-planning problem is decomposed into subproblems, each consisting of a subset of robots, where every subproblem can be solved separately and the results can be combined into a solution for the original problem.

Decoupled planners are an alternative to complete planners trading completeness for efficiency. Typically, decoupled planners solve separate problems for individual robots and combine the individual solutions into a global solution (Leroy et al., 1999; van den Berg and Overmars, 2005).

*Kiril Solovey and Oren Salzman contributed equally to this paper.
Blavatnik School of Computer Science, Tel-Aviv University, Israel

Corresponding author:
Kiril Solovey, Blavatnik School of Computer Science, Tel-Aviv University, PO Box 39040, Tel-Aviv 69978, Israel.
Email: kirilsol@post.tau.ac.il

Although efficient in some cases, the approach usually works only for a restricted set of problems.

The introduction of *sampling-based* algorithms such as the *probabilistic roadmap method (PRM)* by Kavraki et al. (1996), *rapidly-exploring random trees (RRT)* by LaValle and Kuffner (1999), and their many variants, had a significant impact on the field of motion planning due to their efficiency, simplicity and applicability to a wide range of problems. Sampling-based algorithms attempt to capture the connectivity of the *configuration space (C-space)* by sampling collision-free configurations and constructing a *roadmap*—a graph data structure where the free configurations are vertices and the edges represent collision-free paths between nearby configurations. Although these algorithms are not complete, most of them are *probabilistically complete*, that is, they are guaranteed to find a solution, if one exists, given a sufficient amount of time. Recently, Karaman and Frazzoli (2011) introduced several variants of these algorithms such that, with high probability, they produce paths that are *asymptotically optimal* with respect to some quality measure.

Sampling-based algorithms can be easily extended to the multi-robot case by considering the fleet of robots as one composite robot (Sanchez and Latombe, 2002). Such a naïve approach suffers from inefficiency as it overlooks aspects that are unique to the multi-robot problem. More tailor-made sampling-based techniques have been proposed for the multi-robot case by Hirsch and Halperin (2002), Salzman et al. (2015a) and Solovey and Halperin (2014). Particularly relevant to our efforts is the work of Švestka and Overmars (1998) who suggested constructing a *composite* roadmap which is a Cartesian product of roadmaps of the individual robots. Due to the exponential nature of the resulting roadmap, this technique is only applicable to problems that involve a modest number of robots. Recent work by Wagner and Choset (2015) suggests that the composite roadmap does not necessarily have to be explicitly represented. Instead, they maintain an implicitly represented composite roadmap, and apply their M* algorithm to efficiently retrieve paths, while minimizing the explored portion of the roadmap. The resulting technique is able to cope with a large number of robots, for certain types of scenarios. Additional information on these two approaches is provided in Section 2.

1.2. Contribution

We present a sampling-based algorithm for the multi-robot motion-planning problem called *multi-robot discrete RRT (MRdRRT)*. Similar to the approach of Wagner and Choset (2015), we maintain an implicit representation of the composite roadmap. We propose an alternative highly efficient technique for pathfinding in the roadmap, which can cope with scenarios that involve tight coordination of the robots. Our new approach to pathfinding on geometrically embedded graphs, which we call dRRT, is an adaptation of the celebrated RRT algorithm (LaValle and Kuffner, 1999) for the discrete case of a graph, embedded in Euclidean space. dRRT traverses the composite roadmap, which may have exponentially many neighbors (exponential in the number of robots that need to be coordinated). The efficient traversal is achieved by retrieving only partial information on the explored roadmap. Specifically, it considers a single neighbor of a visited vertex at each step. dRRT rapidly explores the C-space represented by the implicit graph. Integrating the implicit representation of the roadmap allows us to solve multi-robot problems while exploring only a small portion of the C-space.

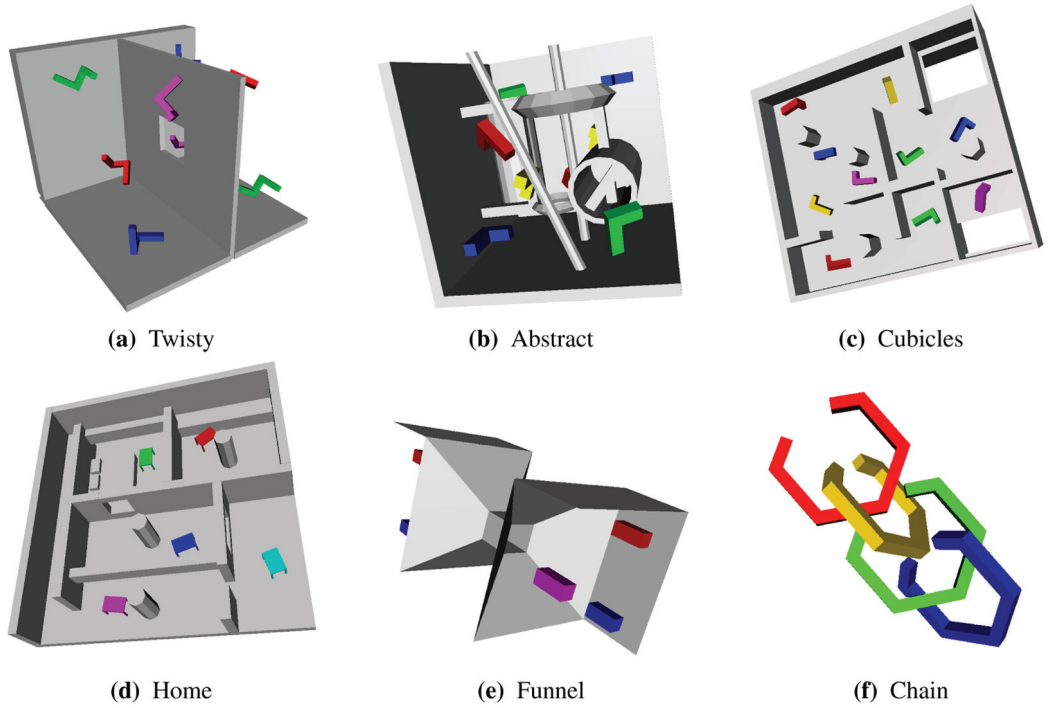
We demonstrate the capabilities of MRdRRT on the setting of polyhedral robots translating and rotating in space amidst polyhedral obstacles. We provide experimental results on several challenging scenarios, for most of which MRdRRT is faster by a factor of at least 10 when compared to existing algorithms that we are aware of. We show that we manage to solve problems of up to 60 DOFs, and scenarios that require tight coordination between the robots (see Figure 1).

The organization of this paper is as follows. In Section 2 we elaborate on two sampling-based multi-robot motion-planning algorithms, namely the composite roadmap approach by Švestka and Overmars (1998) and the work on subdimensional expansion and M* by Wagner and Choset (2015). In Section 3 we introduce the dRRT algorithm. For clarity of exposition, we first describe it as a general pathfinding algorithm for geometrically embedded graphs. In Section 4 we describe the MRdRRT method where dRRT is used in the setting of multi-robot motion planning for the exploration of the implicitly represented composite roadmaps. We show in Section 5 experimental results for the algorithm on different scenarios. We provide a discussion on the advantages and shortcomings of our algorithm in Section 6 and conclude with possible future research directions. On the theoretical side, we provide rigorous proofs for the *probabilistic completeness* of dRRT and MRdRRT in Sections 3 and 4, respectively.

Remark. We mention that we are not the first to consider RRTs in discrete domains. Branicky et al. (2003) applied the RRT algorithm to a discrete graph. However, a key difference between the approaches is that we assume that the graph is *geometrically embedded*, hence we use *random points* as samples while they use nodes of the graph as samples. Additionally, their technique requires that all the neighbors of a visited vertex will be considered—a costly operation in our setting, as mentioned above.

2. Composite roadmaps for multi-robot motion planning

We describe the composite roadmap approach introduced by Švestka and Overmars (1998). Here, a Cartesian product of PRM roadmaps of individual robots is considered as a means of devising a roadmap for the entire fleet of



(a) Twisty **(b)** Abstract **(c)** Cubicles

(d) Home **(e)** Funnel **(f)** Chain

Fig. 1. 3D environments with robots that are allowed to rotate and translate (6 DOFs each). In scenarios (a), (b), (c) and (e) robots of the same color need to exchange positions. (a) Twisty scenario with eight corkscrew-shaped robots, in a room with a barrier. (b) Abstract scenario with eight L-shaped robots. (c) Cubicles scenario with 10 L-shaped robots. (d) Home scenario with five table-shaped robots that are placed in different rooms. The goal is to change rooms in a clockwise order. (e) The Funnel scenario consists of six robots where three robots are located on each side of the funnel. The goal is to move every robot from one side of the funnel to the other. (f) The Chain scenario consists of four C-shaped robots located at the four corners of a room with no obstacles. They need to move to the center to create a highly coupled chain-like formation. Scenarios (a)–(d) are based on meshes provided by the Open Motion Planning Library (OMPL 0.10.2, Sucan et al., 2012) distribution.

robots. However, since they consider an explicit construction of this roadmap, their technique is applicable to scenarios that involve only a small number of robots. To overcome this, Wagner and Choset (2015) suggest representing the roadmap *implicitly* and describe a novel algorithm to find paths on this implicit graph.

Let $r_1, \ldots, r_m$ be m robots operating in a workspace W with robot r_i having start and target configurations s_i, t_i. We wish to find paths for every robot from start to target, while avoiding collision with obstacles as well as with the other robots. Let $G_i = (V_i, E_i)$ be a PRM roadmap for r_i, $|V_i| = n$, and let k denote the maximal degree of a vertex in any G_i. In addition, assume that $s_i, t_i \in V_i$, and that s_i, t_i reside in the same connected component of G_i. Given such a collection of roadmaps $G_1, \ldots, G_m$ a composite roadmap can be defined in two different ways—one is the result of a *Cartesian product* of the individual roadmaps while in the other a *tensor product* is used.

The *composite roadmap* $\mathbb{G} = (\mathbb{V}, \mathbb{E})$ is defined as follows. The vertices $\mathbb{V}$ represent all combinations of collision-free placements of the m robots. Formally, a set of m robot configurations $C = (v_1, \ldots, v_m)$ is a vertex of $\mathbb{G}$ if for every i, $v_i \in V_i$, and in addition, when every robot r_i is placed in v_i the robots are pairwise collision-free. The Cartesian and tensor products differ in the type of edges in the resulting roadmap. If the Cartesian product is used, then $(C, C') \in \mathbb{E}$, where $C = (v_1, \ldots, v_m), C' \in (v_1', \ldots, v_m')$, if there exists i such that $(v_i, v_i') \in E_i$, for every $j \neq i$ it holds that $v_j = v_j'$, and r_i does not collide with the other robots stationed at $v_j = v_j'$ while moving from v_i to v_i'. A tensor product generates many more edges. Specifically, $(C, C') \in \mathbb{E}$ if $(v_i, v_i') \in E_i$ for every i, and the robots remain collision-free while moving on the respective single-graph edges.[1]

Remark. Throughout this work we only consider the *tensor* product composite roadmap.

Note that by the definition of G_i and $\mathbb{G}$ it holds that $S, T \in \mathbb{V}$, where $S = (s_1, \ldots, s_m), T = (t_1, \ldots, t_m)$. The following observation immediately follows (for both product types).

Observation 1. *Let $C_1, \ldots, C_h$ be a sequence of h vertices of $\mathbb{G}$ such that $S = C_1$, $T = C_h$. If for every two consecutive vertices $(C_i, C_{i+1}) \in \mathbb{E}$, then, there exists a path for the robots from S to T.*

Thus, given a composite roadmap $\mathbb{G}$, it is left to find such a path between S and T. Unfortunately, standard pathfinding techniques, which require the full representation of the graph, are prohibitively expensive to use since the number

of vertices of $\mathbb{G}$ alone may reach $O(n^m)$. One may consider the A* algorithm (Pearl, 1984), or its variants, as appropriate for the task, since it may not need to traverse all the vertices of graph. A central property of A* is that it needs to consider all the neighbors of a visited vertex in order to guarantee that it will find a path eventually. Alas, in our setting, this turns out to be a significant hurdle, since the number of neighbors of every vertex is $O(k^m)$.

Wagner and Choset (2015) introduced an adaptation of A* to the case of composite roadmaps called M*. Their approach exploits the observation that only the motion of some robots has to be coordinated in typical scenarios. Thus, planning in the joint C-space is only required for robots whose motion has to be coordinated, while the motion of the remaining robots can be planned individually. Hence, their method dynamically explores low-dimensional search spaces embedded in the full C-space, instead of the joint high-dimensional C-space. This technique is highly effective for scenarios with a low degree of coordination, and can cope with large fleets of robots in such settings. However, when the degree of coordination was increased, we observed a sharp increase in the running time of this algorithm, as it has to consider many neighbors of a visited vertex. This makes M* less effective when the motion of many robots needs to be tightly coordinated.

3. Discrete RRT

We describe a technique which we call *discrete RRT (dRRT)* for pathfinding in implicit graphs embedded in Euclidean space. For clarity of exposition, we first describe dRRT without the technicalities related to motion planning. We add these details in the subsequent section. As the name suggests, dRRT is an adaptation of the RRT algorithm (LaValle and Kuffner, 1999) for the purpose of exploring discrete geometrically embedded graphs, instead of a continuous space.

Since the graph serves as an approximation of some relevant portion of the Euclidean space, traversal of the graph can be viewed as a process of exploring the subspace. The dRRT algorithm rapidly explores the graph by biasing the search towards vertices embedded in unexplored regions of the space.

Let $G = (V, E)$ be a graph where every $v \in V$ is embedded in a point in the Euclidean space $\mathbb{R}^d$ and every edge $(v, v') \in E$ is a line segment connecting the points. Given two vertices $s, t \in V$, dRRT searches for a path in G from s to t. For simplicity, assume that the graph is embedded in $[0, 1]^d$.

Similarly to its continuous counterpart, dRRT grows a tree rooted in s and attempts to connect it to t to form a path from s to t. As in RRT, the growth of the tree is achieved by extending it towards random samples in $[0, 1]^d$. In our case though, vertices and edges that are added to the tree are taken from G, and we do not generate new vertices and edges along the way.

As G is represented implicitly, the algorithm uses an oracle to retrieve information regarding neighbors of visited vertices. We first describe this oracle and then proceed with a full description of the dRRT algorithm. Finally, we show that this technique is *probabilistically complete*, namely guaranteed to find a solution given a sufficient number of samples.

3.1. Oracle to query the implicit graph

In order to retrieve partial information regarding the neighbors of visited vertices, dRRT consults an oracle described below. We start with several basic definitions.

Given two points $v, v' \in [0, 1]^d$, denote by $\rho(v, v')$ the ray that starts in v and goes through v'. Given three points $v, v', v'' \in [0, 1]^d$, denote by $\angle_v(v', v'')$ the (smaller) angle between $\rho(v, v')$ and $\rho(v, v'')$.

Definition 1. *Given a vertex $v \in V$, and a point $u \in [0, 1]^d$ we define*

$$\mathcal{O}_{\mathrm{D}}(v, u) := \operatorname*{argmin}_{v'} \left\{ \angle_v(u, v') \,|\, (v, v') \in E \right\}$$

In other words, the direction oracle returns the neighbor v' of v in G such that the direction from v to v' is the closest to the direction from v to u, than any other neighbor v'' of v.

3.2. Description of dRRT

At a high level, dRRT (Algorithm 1) proceeds similarly to RRT: it grows a tree $\mathcal{T}$ which is a subgraph of G and is rooted in s (line 1). The growth of $\mathcal{T}$ (line 3) is achieved by an expansion towards random samples. Additionally, an attempt to connect $\mathcal{T}$ with t is made (line 4). The algorithm terminates when this operation succeeds and a solution path is generated (line 6), otherwise the algorithm goes to the next expansion iteration (line 2).

Expansion of $\mathcal{T}$ is performed by the EXPAND operation (Algorithm 2) which performs N iterations that consist of the following steps: A point q_{rand} is sampled uniformly from $[0, 1]^d$ (line 2). Then, a node q_{near} of $\mathcal{T}$ that is the closest to the sample (in Euclidean distance), is selected (line 3). q_{near} is extended towards the sample by locating the vertex $q_{\mathrm{new}} \in V$, that is the neighbor of q_{near} in G in the direction of q_{rand} (by the direction oracle $\mathcal{O}_{\mathrm{D}}$). Once q_{new} is found (line 4), it is added to the tree (line 6) with the edge $(q_{\mathrm{near}}, q_{\mathrm{new}})$ (line 7). An illustration of this process can be seen in Figure 2. This is already different from the standard RRT as we cannot necessarily proceed exactly in the direction of the random point.

After the expansion, dRRT attempts to connect the tree $\mathcal{T}$ with t using the CONNECT_TO_TARGET operation (Algorithm 3). For every vertex q of $\mathcal{T}$, which is one of the K nearest neighbors of t in $\mathcal{T}$ (line 1), an attempt is made to connect q to t using the method LOCAL_CONNECTOR (line 2) which is a crucial part of the dRRT algorithm (see Section 3.3). Finally, given a path from some node q of $\mathcal{T}$

Algorithm 1 dRRT_PLANNER (s, t)

1: $\mathcal{T}$.init(s)
2: **loop**
3: EXPAND($\mathcal{T}$)
4: $\Pi \leftarrow$ CONNECT_TO_TARGET($\mathcal{T}, t$)
5: **if** not_empty(Π) **then**
6: **return** RETRIEVE_PATH($\mathcal{T}, \Pi$)

Algorithm 2 EXPAND ($\mathcal{T}$)

1: **for** $i = 1 \rightarrow N$ **do**
2: $q_{\text{rand}} \leftarrow$ RANDOM_SAMPLE()
3: $q_{\text{near}} \leftarrow$ NEAREST_NEIGHBOR($\mathcal{T}, q_{\text{rand}}$)
4: $q_{\text{new}} \leftarrow \mathcal{O}_{\text{D}}(q_{\text{near}}, q_{\text{rand}})$
5: **if** $q_{\text{new}} \notin \mathcal{T}$ **then**
6: $\mathcal{T}$.add_vertex(q_{new})
7: $\mathcal{T}$.add_edge($q_{\text{near}}, q_{\text{new}}$)

Algorithm 3 CONNECT_TO_TARGET($\mathcal{T}, t,$)

1: **for** $q \in$ NEAREST_NEIGHBORS($\mathcal{T}, t, K$) **do**
2: $\Pi \leftarrow$ LOCAL_CONNECTOR(q, t)
3: **if** not_empty(Π) **then**
4: **return** Π
5: **return** $\emptyset$

to t the method RETRIEVE_PATH (Algorithm 1, line 6) returns the concatenation of the path from s to q, with Π.

We note that the only two parameters needed by our algorithm are the number of iterations N for which to expand the roadmap, and the number K of nearest neighbors used to connect the tree to the target. As we discuss in Section 5, one can set $K = \log i$ at the ith iteration.

3.3. Local connector

We show in the following subsection that it is possible that $\mathcal{T}$ will eventually reach t during the EXPAND stage, and therefore an application of LOCAL_CONNECTOR will not be necessary. However, in practice this is unlikely to occur within a short time frame, especially when G is large. Thus, we employ a heavy duty technique, which, given two vertices q_0, q_1 of G tries to find a path between them. We mention that it is common to assume in sampling-based algorithms that connecting nearby samples will require less effort than solving the original problem and here we make a similar assumption. We assume that a local connector is effective only on *restricted* pathfinding problems, thus in the general case it cannot be applied directly on s, t, as it may be too costly (unless the problem is easy). A concrete example of a local connector is provided in the next section.

3.4. Probabilistic completeness of dRRT

Recall that an algorithm is *probabilistically complete* if the probability that it finds a solution tends to one as the number of iterations tends to infinity, when such a solution exists. For simplicity, the proof does not make use of the local connector, i.e. the entire run of the dRRT algorithm consists of a single call to the EXPAND procedure (Algorithm 2), which terminates when the number of iterations reaches N, or when $q_{\text{new}} = t$ is added to $\mathcal{T}$. With a slight abuse of notation we use the vertices and their embedding interchangeably.

Let $G = (V, E)$ be a connected graph, embedded in $\mathbb{R}^d$, with a fixed number of vertices. Let $V' \subset V$ be a connected subset in G. For a given $v \in V'$, denote by $\text{Vor}(v, V')$ the *Voronoi cell* (de Berg et al., 2008) of the site v, in the Euclidean (standard) Voronoi diagram of point sites in $\mathbb{R}^d$, where the sites are point-embeddings of the vertices V' (Figure 2(b)). Now, denote by $\text{nbr}(v)$ the set of neighbors of v in G. For every $v^* \in \text{nbr}(v)$ denote by $\text{Vor}'(v, v^*)$ the Voronoi cell of $\rho(v, v^*)$, in the Voronoi diagram of the ray sites $\{\rho(v, v') \,|\, v' \in \text{nbr}(v)\}$ (see Figure 2(c)).

Lemma 2. *For a given G there exists $\xi > 0$ such that for every $V' \subseteq V, v \in V', v^* \in \text{nbr}(v)$, it holds that $|\text{Vor}(v, V') \cap \text{Vor}'(v, v^*)| \geq \xi$, where $|\cdot|$ denotes the* Lebesgue measure.

Proof. For every $V' \subseteq V$ and $v \in V'$ it holds that $\text{Vor}(v, V) \subseteq \text{Vor}(v, V')$, as adding a site u to the Voronoi diagram either leaves a cell intact or removes a portion of the cell $\text{Vor}(v, V')$: the portion inside the halfspace determined by the bisector of v and u, on the side of u. Consequently

$$\text{Vor}(v, V) \cap \text{Vor}'(v, v^*) \subseteq \text{Vor}(v, V') \cap \text{Vor}'(v, v^*)$$

for every $v^* \in \text{nbr}(v)$. Now, we set $v \in V, v^* \in \text{nbr}(v)$ to be the vertices for which the area of $\text{Vor}(v, V) \cap \text{Vor}'(v, v^*)$ is minimized and set

$$\xi := |\text{Vor}(v, V) \cap \text{Vor}'(v, v^*)|$$

For the remainder of the proof we assume that every two distinct vertices of G are embedded into two distinct points in $\mathbb{R}^d$, and every pair of edges of G incident to the same (embedded) vertex does not overlap in their interior. If several vertices collapse into a single Voronoi site, one can choose one of the vertices with equal probability, by slightly modifying the NEAREST_NEIGHBOR routine, in Algorithm 2. Overlapping edges can be treated in a similar manner.

It remains to show that $\xi > 0$. Let $v \in V, v^* \in \text{nbr}(v)$ be the vertices from which the value ξ is attained. As vertices are embedded into distinct points, and as edges do not overlap, we can deduce that $|\text{Vor}(v, V)| > 0$ and $|\text{Vor}'(v, v^*)| > 0$. In addition, the intersection between the two cells is clearly non-empty: There is a ball with radius $r > 0$ whose center is v and is completely contained in

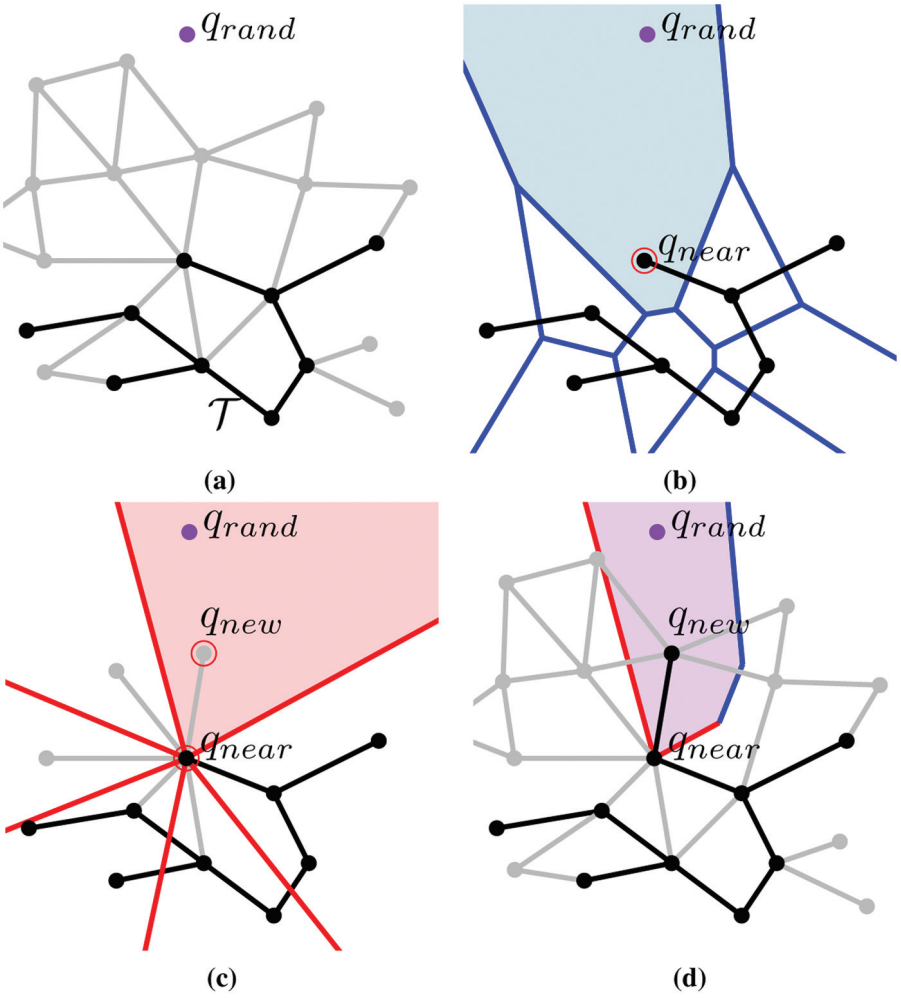


Fig. 2. An illustration of the expansion step of dRRT. The tree $\mathcal{T}$ is drawn with black vertices and edges, while the gray elements represent the unexplored portion of the implicit graph G. (a) A random point q_{rand} (purple) is drawn uniformly from $[0,1]^d$. (b) The vertex q_{near} of $\mathcal{T}$ that is the Euclidean nearest neighbor of q_{rand} is extracted. (c) The neighbor q_{new} of q_{near}, such that its direction from q_{near} is the closest to the direction of q_{rand} from q_{near}, is identified. (d) The new vertex and edge are added to $\mathcal{T}$. *Additional information for Theorem 2*: In (b) the Voronoi diagram of the vertices of $\mathcal{T}$ is depicted in blue, and the Voronoi cell of q_{near} ,$\text{Vor}(q_{\text{near}}, V')$, is shaded light blue, where V' denotes the vertices of $\mathcal{T}$. In (c) the Voronoi diagram of the rays that leave q_{near} and pass through its neighbors is depicted in red, and the Voronoi cell of $\rho(q_{\text{near}}, q_{\text{new}})$, $\text{Vor}'(q_{\text{near}}, q_{\text{new}})$, is shaded pink. The purple region in (d) represents $\text{Vor}(q_{\text{near}}, V') \cap \text{Vor}'(q_{\text{near}}, q_{\text{new}})$.

$\text{Vor}(v, V)$; similarly, there is a cone of solid angle $\alpha > 0$ with apex at v fully contained in $\text{Vor}'(v, v^*)$. Hence, it holds that $\xi > 0$, otherwise v and v^* are embedded in the same point. □

Theorem 3. *Let G be a connected graph embedded in $\mathbb{R}^d$, and denote by $s, t \in V$ the start and target vertices, respectively. The probability that dRRT finds a path from s to t converges to 1 as $N \to \infty$, where N represents the number of iterations (steps) performed by dRRT in the EXPAND procedure.*

Proof. Denote by $L := (s = v_1, v_2, \ldots, v_\ell = t)$ the shortest path between $s, t \in V$ in G, in terms of the number of vertices, and let $\mathcal{V}_j := \text{Vor}(v_j, V) \cap \text{Vor}'(v_j, v_{j+1})$ for $1 \le j \le \ell - 1$. Additionally, denote by $q_1, \ldots, q_N$ the samples drawn consecutively by dRRT, and by $V_i^{\mathcal{T}}$ the vertices of $\mathcal{T}$ at step $1 \le i \le N$.

The main observation here is that for a given step i of dRRT, if $q_i \in \mathcal{V}_j$ and $v_j \in V_i^{\mathcal{T}}$, for $1 \le j \le \ell - 1$ then the edge (v_j, v_{j+1}) will be added to $\mathcal{T}$ at the end of this step. Thus, with probability at least $\xi > 0$ dRRT will make an advancement along L toward t (Lemma 2).

The process of uncovering L by dRRT can be modeled as a *Markov chain*, where for every $v_j \in L$ we have a state S_j, which represents the event that the farthest vertex along L that was reached is v_j. Given that we arrived at state S_j, the probability that we will move to state S_{j+1} given the next sample is denoted by $\beta_j > \xi$. We stay put at S_j with probability at most $1 - \beta_j$. Note that once $t = v_\ell$ is reached, dRRT will not proceed to explore G for the remainder of

the iterations. Thus, the probability of leaving the state S_ℓ is equal to 0. The process can be viewed as an *absorbing* Markov chain with the absorbing state S_ℓ, in which every state can reach S_ℓ, but once S_ℓ is entered, it cannot be left. It is known that the probability of reaching an absorbing state within N steps tends to 1 as N tends to ∞ (see Theorem 11.3 in Chapter 11.2 of Grinstead and Snell, 2012). □

Note that this is true only when the values of the transition probabilities do not depend on N, which is indeed the case in our setting. Specifically, as the number of vertices of G, and the dimension in which it is embedded, are assumed to be fixed, ξ is fixed as well. The same applies for the length of L, namely the value ℓ.

4. Multi-robot motion planning with dRRT

In this section we describe the MRdRRT algorithm. Specifically, we discuss the adaptation of dRRT for pathfinding in a composite roadmap $\mathbb{G}$, which is embedded in the joint C-space of m robots. In particular, we show an implementation of the oracle $\mathcal{O}_\mathrm{D}$, which relies solely on the representation of $G_1, \ldots, G_m$. Additionally, we discuss an implementation of the local-connector component, which takes advantage of the fact that $\mathbb{G}$ represents a set of valid positions and movements of multiple robots. Finally, we discuss the probabilistic completeness of our entire approach to multi-robot motion planning.

4.1. Direction oracle $\mathcal{O}_\mathrm{D}$

Recall that given $C \in \mathbb{V}$ and a random sample q, $\mathcal{O}_\mathrm{D}(C, q)$ returns $C' \in \mathbb{V}$ that is a neighbor of C in $\mathbb{G}$, and for every other neighbor C'' of C, $\rho(C, q)$ forms a smaller angle with $\rho(C, C')$ than with $\rho(C, C'')$, where ρ is as defined in Section 3.4.

Denote by $\mathcal{C}(r_i)$ the C-space of r_i. Let $q = (q_1, \ldots, q_m)$ where $q_i \in \mathcal{C}(r_i)$, and let $C = (c_1, \ldots, c_m)$ where $c_i \in V_i$. To find a suitable neighbor for C we first find the most suitable neighbor for every individual robot and combine the m single-robot neighbors into a candidate neighbor for C. We denote by $c_i' = \mathcal{O}_\mathrm{D}(c_i, q_i)$ the neighbor of c_i in G_i that is in the direction of q_i. Notice that the implementation of the oracle for individual roadmaps is trivial—for example, by traversing all the neighbors of c_i in G_i. Let $C' = (c_1', \ldots, c_m')$ be a candidate for the result of $\mathcal{O}_\mathrm{D}(C, q)$. If (C, C') represents a valid edge in $\mathbb{G}$, i.e. no robot–robot collision occurs, we return C'. Otherwise, $\mathcal{O}_\mathrm{D}(C, q)$ returns $\emptyset$. In this case, the new sample is ignored and another sample is drawn in the EXPAND phase (Algorithm 2).

The completeness proof of the dRRT (Theorem 3) for this specific implementation of $\mathcal{O}_\mathrm{D}$, is straightforward. Notice that in order to extend $C = (c_1, \ldots, c_m)$ to $C' = (c_1', \ldots, c_m')$ the sample $q = (q_1, \ldots, q_m)$ must obey the following restriction: For every robot r_i, q_i must lie in $\mathrm{Vor}(c_i, V_i) \cap \mathrm{Vor}'(c_i, c_i')$ (where in the original proof we required that q will lie in $\mathrm{Vor}(C, \mathbb{V}) \cap \mathrm{Vor}'(C, C')$).

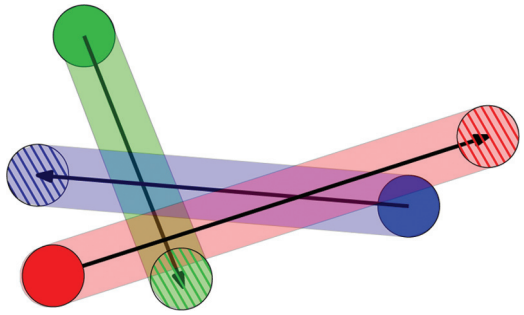


Fig. 3. Visualization of the local connector used to connect two vertices of the graph based on the method described by van den Berg et al. (2009). Three disk robots (red, green and blue) need to move from a start placement to a target placement (depicted by shaded disks and dashed disks, respectively) by following a straight line. At every given point of time, one robot moves while the other robots are stationary either in their start or target configurations. Note that the path of the red robot is blocked both by the start position of the blue robot and the target position of the green robot. Thus, to allow for a collision-free path based on prioritizing the movements, the red robot should move *before* the green robot moves, and *after* the blue one moves.

4.2. Local-connector implementation

Recall that in the general dRRT algorithm the local connector is used for connecting two given vertices of a graph. For our local connector we rely on a framework described by van den Berg et al. (2009). Given two vertices $\mathbb{V} = (v_1, \ldots, v_m), \mathbb{V}' = (v_1', \ldots, v_m')$ of $\mathbb{G}$ we find for each robot i a path π_i on G_i from v_i to v_i'. The connector attempts to find an ordering of the robots such that robot i does not leave its start position on π_i until robots with higher priority reached their target positions on their respective path, and of course that it also avoids collisions. When these robots reach their destination, robot i moves along π_i from $\pi_i(0)$ to $\pi_i(1)$. During the movement of this robot the other robots stay put.

The priorities are assigned according to the following rule: if moving robot i along π_i causes a collision with robot j that is placed in v_j then robot i should move *after* robot j. Similarly, if i collides with robot j that is placed in v_j' then robot i should move *before* robot j. This prioritization induces a directed graph $\mathcal{I}$. In case this graph is acyclic we generate a solution according to the prioritization of the robots. Otherwise, we report failure. For a visualization of the method, see Figure 3. We decided to use this simple technique in our experiments due to its low cost, in terms of running time.

We remark that we also experimented with M* as a local connector. The motivation here is the common assumption in sampling-based motion planning that connecting close-by configurations is often an easy problem. As the local connector is applied multiple times, it should be simple and extremely efficient on easy instances, even at the cost of not finding a complex solution when one exists. Indeed,

for easy problems with a *low degree of coordination* M* meets the above criteria. Now, to use M* as a local connector, one needs to modify it such that when a local path is not easy to find, the running time and the memory consumption of M* will not be high. Obviously, this comes at the price of relaxing the completeness of M*. Thus, we bounded both the degree of coordination that M* could consider (to avoid considering exponentially many neighbors for a certain node) and the number of nodes that M* could consider (to bound the running time of M*). While very efficient at times, this approach introduced the need to tune the above parameters and for certain scenarios did not yield satisfactory results. The ordering algorithm of van den Berg et al. (2009) turned out to be considerably more efficient and obviously it is parameter-free.

4.3. Probabilistic completeness of MRdRRT

In order for the motion-planning framework to be probabilistically complete, we need to show that (i) as the number of samples used for each single-robot roadmap tends to ∞, the probability that the composite roadmap contains a path tends to 1, and (ii) that the proof of Theorem 3 still holds when the size of the graph tends to ∞.

A motion-planning problem is said to be *robustly feasible* (Karaman and Frazzoli, 2011), if there exists a solution with positive clearance. In the setting of multi-robot motion planning, robust feasibility implies that there exists a solution in which the robots do not move in contact with obstacles or with the other robots. Švestka and Overmars show that the composite-roadmap approach is probabilistically complete, given that the problem is robustly feasible, and assuming that the underlying graph-search algorithm is complete.

Theorem 4. *(Švestka and Overmars, 1998) Given a robustly feasible problem, the probability that the composite roadmap, constructed from PRM roadmaps with n vertices each, denoted by* $\mathbb{G}(n)$*, contains a solution that tends to* 1 *as n tends to* ∞*, i.e.*

$$\lim_{n\to\infty} \Pr[\mathbb{G}(n) \text{ contains a solution}] = 1$$

In our setting we employ a probabilistic algorithm to query $\mathbb{G}$, namely dRRT, whose success rate tends to 1 as the number of its samples N tends to ∞, under the assumption that n is fixed (see Theorem 3). We first prove a weaker property of MRdRRT, and then proceed to the main theorem.

Proposition 5. *Given a robustly feasible problem, for every* $0 < p < 1$ *there exists* $n_0 \in \mathbb{N}^+$ *such that for every* $n > n_0$ *it holds that*

$$\lim_{N\to\infty} \Pr[\text{MRdRRT finds a solution using } \mathbb{G}(n) \text{ and } N \text{ samples}] \geq p$$

Proof. Let $0 < p < 1$ be some constant. By Theorem 4, $\mathbb{G}(n)$ contains a solution with probability 1 as n tends to ∞. Then, there also exists $n_0 \in \mathbb{N}^+$ such that for every $n > n_0$ it follows that

$$\Pr[\mathbb{G}(n) \text{ contains a solution}] \geq p$$

Denote by $\mathcal{P}(n, N)$ the event that MRdRRT finds a solution using $\mathbb{G}(n)$ and N samples. We fix some $n_1 > n_0$. From Theorem 3 it follows that

$$\begin{aligned}\lim_{N\to\infty} \Pr[\mathcal{P}(n_1, N)] &= \lim_{N\to\infty} \Pr[\mathcal{P}(n_1, N) \,|\, \mathbb{G}(n_1) \\ &\quad \text{contains a solution}] \cdot \Pr[\mathbb{G}(n_1) \text{ contains a solution}] \\ &= \lim_{N\to\infty} \Pr[\mathcal{P}(n_1, N) \,|\, \mathbb{G}(n_1) \text{ contains a solution}] \cdot p = p\end{aligned}$$

□

Theorem 6. *Given a robustly feasible problem, the probability of MRdRRT to find a solution tends to* 1*, as n and N tend to* ∞*.*

Proof. For every positive integer i, define $\varepsilon_i := 1/i$. Let n_i be the minimal value such that for every $n \geq n_i$ there exists N such that $\Pr[\mathcal{P}(n, N)] \geq 1 - \varepsilon_i$. Note that such n_i exists by Proposition 5. For a given n, let $i(n)$ be such that $n_i \leq n < n_{i+1}$. Also, let $N(n)$ be the minimal value for which $\Pr[\mathcal{P}(n, N(n))] \geq 1 - \varepsilon_{i(n)}$. Then, it follows that

$$\lim_{n\to\infty} \Pr[\mathcal{P}(n, N(n)] \geq \lim_{n\to\infty} 1 - \varepsilon_{i(n)} = 1$$

□

5. Experimental results

We implemented MRdRRT for the case of polygonal and polyhedral robots translating and rotating among polygonal and polyhedral obstacles, respectively. We compared the performance of MRdRRT with RRT and an improved (recursive) version of M* that appears in Wagner and Choset (2015). To make the comparison as equitable as possible, we used the *inflated* version of M* which has relaxed optimality guarantees (as dRRT does not take into consideration the quality of the solution).

5.1. Implementation details

The algorithms were implemented in C++. The experiments were conducted on a laptop with an Intel i5-3230M 2.60 GHz processor with 16 GB of memory, running 64-bit Windows 7. We implemented a generic framework for multi-robot motion planning based on composite roadmaps.

The implementation relies on the CGAL Project (2011) and on PQP (see `http://gamma.cs.unc.edu/SSV/` for collision detection in the two- and three-dimensional workspaces, respectively, and performs nearest-neighbor queries using the Fast Library for Approximate Nearest Neighbors (FLANN) by Muja and Lowe (2009). Metrics,

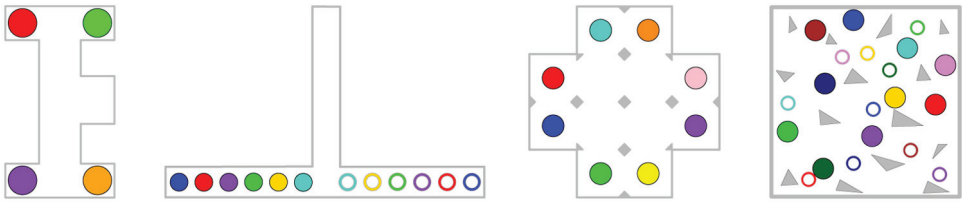


Fig. 4. Highly coupled settings in planar environments used for the experiments. (a) The red and green robots (at the top) need to exchange positions with the purple and orange robots (at the bottom), (b) the robots need to reverse their order by entering the narrow corridor (start and target positions are depicted by shaded disks and annuli, respectively), (c) opposite robots need to interchange positions, (d) 10 randomly placed robots that move to random goals (start and target locations are depicted by shaded disks and annuli, respectively).

sampling and interpolation in the 3D environments follow the guidelines of Kuffner (2004). To eliminate the dependence of dRRT on parameters we assigned them according to the number of iterations the algorithm performed so far, i.e. the number of times that the main loop has been repeated. Specifically, in the ith iteration each EXPAND (Algorithm 2) call performs 2^i iterations ($N = 2^i$), while CONNECT_TO_TARGET uses $K = i$ candidates that are connected with t. Let k_{nn} denote the number of nearest neighbors that is used to construct the roadmaps of the individual robots. We used the values $k_{\text{nn}} = 15$ and $k_{\text{nn}} = 12$ for the two- and three-dimensional workspaces, respectively.

5.2. *Test scenarios*

We report results for two sets of tests. In the first set we test M* and MRdRRT on challenging scenarios (Figures 1 and 4) to illustrate the performance of the two techniques. In the second set of tests we study how the difficulty of the scenarios, in terms of the necessary amount of coordination between the robots, affects the running time of M* and MRdRRT.

Basic scenarios. The two-dimensional workspaces (see Figure 4) represent scenarios where the motion of the robot needs to be tightly coordinated. In all the scenarios a collection of between 4 and 10 identical disk robots move in a polygonal workspace with relatively little clearance. The 3D workspaces used (see Figure 1) are constructed using meshes provided by the OMPL. The Twisty scenario (Figure 1(a)) contains eight corkscrew-shaped robots, in a room with a barrier. There are four robots on each side of the barrier and each robot needs to pass through the small hole in the barrier to reach the other side. This results in a tight coordination of the robots' motion in the vicinity of the barrier. The Abstract scenario (Figure 1(b)) and the Cubicles scenario (Figure 1(c)) contain 8 and 10 L-shaped robots, respectively. While the shape of the obstacles differ, these two scenarios are similar as both require coordinated motion of the robots due to the large size of the robots and the obstacles with respect to the small size of the room. The Home scenario (Figure 1(d)) contains five table-shaped robots that are placed in different rooms. The goal is to move each table to a different room. This problem contains little coordination as only one robot needs to pass in every narrow passage connecting the rooms. Next, the Funnel scenario (Figure 1(e)) consists of six robots where three robots are located on each side of the funnel and the goal is to move every robot from one side of the funnel to the other. Finally, the Chain scenario (Figure 1(f)) consists of four C-shaped robots located at the four corners of a room with no obstacles. In this "bug trap"-like scenario the robots need to move to the center to create a highly coupled chain-like formation.

We report in Table 1 the running times of M* and dRRT for the scenarios. We ran each of the 3 algorithms 10 times on each scenario. We remark that RRT proved incapable of solving any of the test scenarios, running for several 10s of minutes until terminating due to exceeding the memory limits. We believe that RRT as-is is unsuitable for multi-robot motion planning in cases where the scenarios consist of more than a couple of robots. M* exhibited better performance than RRT. However, for almost all of the attempts in all 2D scenarios, it failed to find a solution (even in scenarios with a relatively small number of robots). Only one solution was found for the Random scenario, with running times exceeding those of MRdRRT by roughly 50%. For the Twisty, Funnel and Chain scenarios, which involve multiple robots and require a substantial amount of coordination, M* failed to find a solution. For the Abstract and the Cubicles scenarios, it never exceeded a success rate of 40%. In particular, it often ran out of memory or ran for a very long duration, and was terminated if its running time exceeded 10× the running time of MRdRRT. On the other hand, MRdRRT was stable in its results and managed to solve all the scenarios, except for the Chain scenario, for each of the 10 attempts. When M* did manage to solve 1 of the first 3 scenarios, it explored between 2.5 to 10× the number of vertices that dRRT explored. For the Home scenario the results of MRdRRT and M* were comparable and in general we found M* more suitable for situations where only a small number of robots have to be coordinated at any

given time. We mention that MRdRRT was unable to solve scenarios that consist of a substantially larger number of robots than we used in our experiments. We believe that it would be beneficial to consider a stronger *local connector* in such cases.

As mentioned, both algorithms failed to find a solution for the Chain scenario. We believe that this is due to the fact that this problem can be interpreted as a bug trap, where a search algorithm may benefit from performing *bidirectional search*. Indeed, we implemented such a variant (see Section 7), which solved the problem (results presented in Table 1). We mention that such an approach may be applicable to M* as well.

Gradual increase of coordination. We studied how the difficulty of scenarios affects the performance of M* and MRdRRT. For this purpose we fixed a workspace environment and the collection of start and target positions of the robots, and gradually increased the size of the robots (Figure 5). The increase gradually eliminated pathways through which robots can avoid one another. As a result, the motion of robots became increasingly coupled. The two algorithms exhibited similar performance for the two most simple cases, in which a scale of 1 and 1.3 was used for the robots. However, using a scaling factor of 1.7 and higher, MRdRRT outperformed M* in terms of the running time. Additionally, the success rate of the former stands on 100% throughout the entire test set, while the latter's success rate was at %50 for a scaling factor of 2.1, and decreased even more for the harder cases. In particular, for scaling 3.7, M* had a success rate of 0%.

6. Discussion

In this section we review the advantages and limitations of MRdRRT. Recall that the implicitly represented composite roadmap $\mathbb{G}$ results from a tensor product of m PRM roadmaps $G_1, \ldots, G_m$. The reliance on the precomputed individual roadmaps eliminates the need to perform additional collision checking between robots and obstacles while querying $\mathbb{G}$. This has a substantial impact on the performance of MRdRRT as it is often the case that checking whether m robots collide with obstacles is much more costly than checking whether the m robots collide between themselves. This is in contrast with more naïve approaches, such as RRT, which consider the group of robots as one large robot. In such cases, checking whether a configuration (or an edge) is collision-free requires checking for the two types of collisions simultaneously.

The M* algorithm, which also uses the underlying structure of $\mathbb{G}$, performs very well in situations where only a small subset of the robots need to coordinate. In these situations it can cope, almost effortlessly, with several 10s of robots while outperforming our framework. This points indeed to a weakness of dRRT in easy scenarios. However, in scenarios where a substantial amount of coordination is required between the robots (see, e.g. Figure 4(b)), M* suffers from a disadvantage, since it is forced to consider exponentially many neighbors when performing the search on $\mathbb{G}$. In contrast, dRRT performs a "minimalistic" search and advances in small steps, little by little, regardless of the difficulty of the problem at hand. Moreover, dRRT strives to reach unknown regions in $\mathbb{G}$ while avoiding spending too much time in the exploration of regions that are in the vicinity of explored vertices. This is done via the Voronoi bias, as shown in the proof of Theorem 3. This is extremely beneficial when working on $\mathbb{G}$ since it contains vertices which densely cover the space, and thus considering many vertices within a small region would not lead to a better understanding of the problem at hand. To justify this claim, consider the following example. Suppose that for every robot r_i, v_i is a vertex of V_i that has k neighbors in G_i at distance at most ε. Then the vertex $(v_1, ..., v_m) \in \mathbb{V}$ might have as many as k^m neighbors that are at distance at most $\varepsilon\sqrt{m}$ in $\mathbb{G}$.

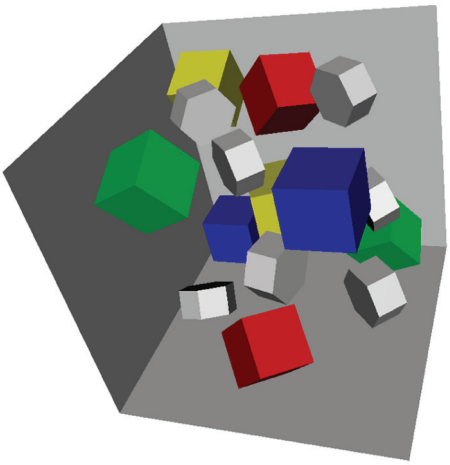
7. Future work

Towards optimality. Currently, our algorithmic framework is concerned with finding *some* solution. Our immediate future goal is to modify it to provide a solution with quality guarantees, possibly by taking an approach similar to the continuous RRT* algorithm (Karaman and Frazzoli, 2011), which is known to be asymptotically optimal. A fundamental difference between RRT* and the original formulation of RRT is in a rewiring step, where the structure of the tree is revised to improve previously examined paths. Specifically, when a new node is added to the tree, it is checked as to whether it will be more beneficial for some of the existing nodes to point to the new vertex instead of their current parent in the tree. This can be adapted, to some extent, to the discrete case, although it is not clear how to efficiently perform rewiring on an implicitly represented graph.

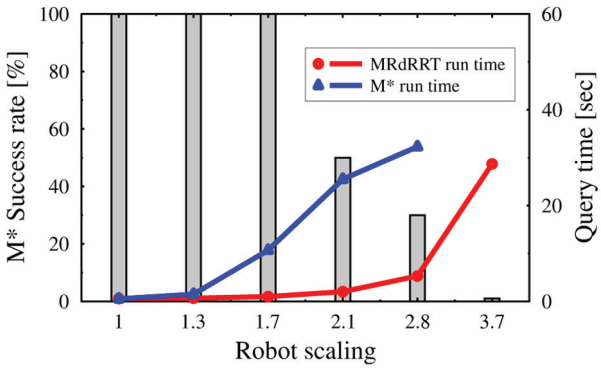
dRRT in other settings of motion planning. In this paper we combined the dRRT algorithm with implicit composite roadmaps to provide an efficient algorithm for multi-robot motion planning. One of the benefits of our framework comes from the fact that it reuses some of the already computed information to avoid performing costly operations. In particular, it refrains from checking collisions between robots and obstacles by forcing the individual robots to move on precalculated individual roadmaps (i.e. G_i). A similar approach can be used in other settings of motion planning. In particular, we have recently developed a dRRT-based approach for motion planning of free-flying multi-link robots (Salzman et al., 2015b). The new approach generates an implicitly represented roadmap, which encapsulates information on configurations and paths that do not induce self-intersections for the robot, while ignoring the existence of obstacles. Then, we overlay this roadmap on the workspace, an operation which invalidates some of the nodes and edges of the roadmap. Thus, we know only which configurations are self-collision-free, but not obstacle collision-free. Then we employ dRRT for pathfinding

Table 1. Results for M* and MRdRRT on the scenarios depicted in Figures 1 and 4. We first report the number of vertices used in the construction of the single-robot PRM roadmaps and the elapsed time (all times are reported in seconds). Then we report the number of visited vertices, the averaged total running time over the number of successful attempts (out of 10), the standard deviation, and the success rate of M*. A similar report is given for dRRT, but we also specify the duration of the connection phase (using the local connector) and the expansion phase. The running times and the number of explored vertices are averaged over the number of successful attempts. The results for MRdRRT in the Chain scenario are presented for a bidirectional variant of dRRT. See Section 7.

		2D Scenarios					3D Scenarios				
		Bay	T-shape	Room	Rand	Twisty	Abstract	Cubicles	Home	Funnel	Chain
PRM	Vertices number	300	300	300	300	8k	10k	10k	5k	500	15k
	Generation time	0.1 s	0.1 s	0.1 s	0.1 s	10 s	24.8 s	16.2 s	10.1 s	17 s	8.7 s
M*	Visited vertices	∞	∞	∞	450k	∞	300k	27k	2k	∞	∞
	Total time (avg.)	∞	∞	∞	106 s	∞	267 s	31 s	4 s	∞	∞
	Total time (s.d.)	∞	∞	∞	0	∞	205	13	1.8	∞	∞
	Success rate	0%	0%	0%	10%	0%	30%	40%	100%	0%	0%
MRdRRT	Visited vertices	380k	5.3k	320k	330k	8k	34k	12k	8k	120k	600
	Connect time	5.4 s	3.1 s	9.7 s	7.9 s	3.3 s	30.4 s	16.3 s	1.5 s	110 s	3.3 s
	Expand time	119 s	1.2 s	27 s	69.4 s	6.7 s	25.5 s	36.8 s	2.9 s	373 s	5.4 s
	Total time (avg.)	124.4 s	4.3 s	36.7 s	77.3 s	11 s	55.9 s	53.1 s	4.4 s	500 s	17.4 s
	Total time (s.d.)	57.3	1.8	3.3	69.8	2.0	38	78	1.6	221	9.3
	Success rate	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%



(a) Gradual coordination scenario



(b) Comparison of results

Fig. 5. Comparing the affect of a gradual increase in the degree of coordination on the performance of MRdRRT and M*. (a) Eight cubical robots, capable of translating and rotating, are placed in a scenario cluttered with polyhedral obstacles (drawn in gray). Robots of the same color need to exchange positions. The robots' size is gradually increased, which results in an increase in the difficulty of the problem. In this figure, the most difficult setting is depicted, where the robots are scaled by a factor of 3.7 from their original size. (b) Results for the different scaling factors for the scenario. Success rate for M* is depicted by shaded bars (MRdRRT had a 100% success rate in all cases). The average query time for MRdRRT and M* is depicted by colored graphs. A run of M* was considered a failure if its running time exceeded 10× the average running time of MRdRRT, or it ran out of memory.

on the new roadmap, while avoiding self-collision tests and while exploring a small portion of a massive roadmap. A key benefit of our approach is the ability to cache local-planner motions, which are particularly costly to compute, when the robot at hand has many DOFs. A visualization of the approach can be seen in Figure 6.

dRRT variants. Our dRRT algorithm conceptually mimics the celebrated RRT algorithm for the case of a discrete graph embedded in some geometric space. However, the motion-planning literature contains many more algorithms, some more suitable for specific problem instances. For example, the RRT-Connect by Kuffner and LaValle (2000) which grows two RRT-trees—one rooted at the source and one at the target. As mentioned in Section 5, the Chain scenario is an example of such a problem. We implemented a simple variant of dRRT that performs a bidirectional search similar to RRT-Connect. While M* or dRRT did not find a solution, when we used the bidirectional variant of

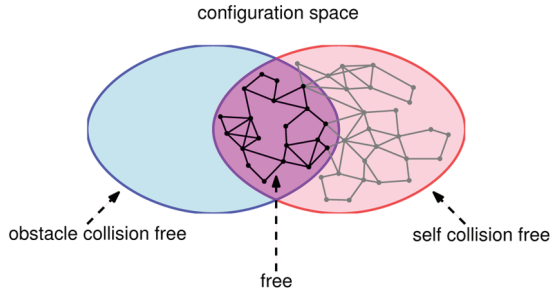


Fig. 6. Visualization of the dRRT-based approach for motion planning of a free-flying multi-link robot. The entire configuration space of the problem is illustrated as the bounding box. The red ellipse represents the set of configurations which do not induce self-collisions. Similarly, the blue ellipse represents the set of configurations which do not induce collisions between the robot and the workspace obstacles. The purple area represents the set of configurations which are fully free. In our dRRT-based approach for this setting we generate an implicit roadmap which provides a tilling of the self collision-free space, and using dRRT we uncover the portion of this roadmap which is also obstacle collision-free. Note that in this case the implicit roadmap is generated for a specific type of robot, and is independent of the structure of the workspace.

dRRT, a solution was quickly found (see results in Table 1). Other motion-planning algorithms can be adapted to a discrete variant and we give two examples to demonstrate how one may transform a continuous motion-planning algorithm to search in a geometrically embedded roadmap. Consider the Expansive Space-Tree planner (EST) by Hsu et al. (1999) and the Resolution Independent Density Estimation for Motion Planning in High-Dimensional Spaces planner (STRIDE) by Gipson et al. (2013). Both algorithms maintain a tree as a roadmap and expand the tree by selecting a node from the tree. While the details on how to select a node differs between algorithms, they both rely on a geometric embedding of the tree's vertices to perform the selection. Once a node x is chosen from the tree, a random configuration y in the vicinity of x is selected (again, the algorithms differ in the way the node is selected). The planners then attempt to connect x to y. To apply each of the above algorithms for the case of a discrete graph $\mathcal{G}$, one needs to simply replace y with the vertex $y' \in \mathcal{G}$ such that y' is the neighbor of x which is the closest to y.

Note

1. There is wide consensus on the term *tensor product* as defined here, and less so on the term *Cartesian product*. As the latter has already been used before in the context of motion planning, we will keep using it here as well.

Acknowledgements

We wish to thank Glenn Wagner for advising on the M* algorithm and providing helpful additional information. We also thank Ariel Felner for advice regarding pathfinding algorithms on graphs. We note that the title name "Finding a Needle in an Exponential Haystack" has been previously used in a talk by Joel Spencer in a different context.

Funding

This work was supported in part by the 7th Framework Programme for Research of the European Commission (FET-Open grant number 255827, CGL—Computational Geometry Learning), by the Israel Science Foundation (grant number 1102/11), by the German–Israeli Foundation (grant number 1150-82.6/2011), and by the Hermann Minkowski–Minerva Center for Geometry at Tel-Aviv University.

References

Adler A, de Berg M, Halperin D and Solovey K (2015) Efficient multi-robot motion planning for unlabeled discs in simple polygons. *IEEE Transactions on Automation Science and Engineering* 12(4): 1309–1317. DOI: 10.1109/TASE.2015.2470096.

Aronov B, de Berg M, van der Stappen AF, Švestka P and Vleugels J (1999) Motion planning for multiple robots. *Discrete & Computational Geometry* 22(4): 505–525.

Auletta V, Monti A, Parente M and Persiano P (1996) A linear time algorithm for the feasibility of pebble motion on trees. In: Karlsson R and Lingas A (eds) *Algorithm Theory – SWAT'96*. New York: Springer, vol. 1097, pp. 259–270.

Branicky MS, Curtiss MM, Levine JA and Morgan SB (2003) RRTs for nonlinear, discrete, and hybrid planning and control. In: *42nd IEEE conference on decision and control*, Maui, HI, 9–12 December 2003, vol. 1, pp. 656–663. Piscataway: IEEE Press.

Choset H, Lynch K, Hutchinson S, Kantor G, Burgard G, Kavraki L et al. (2005) *Principles of Robot Motion: Theory, Algorithms, and Implementations*. Cambridge: MIT Press.

de Berg M, van Kreveld M, Overmars M and Schwarzkopf O (2008) *Computational Geometry: Algorithms and Applications*. Berlin, Germany: Springer-Verlag.

Gipson B, Moll M and Kavraki L (2013) Resolution independent density estimation for motion planning in high-dimensional spaces. In: *2003 IEEE international conference on robotics and automation (ICRA'03)*, Karlsruhe, Germany, 6–10 May 2013, pp. 2437–2443. Piscataway: IEEE Press. Berlin, Germany.

Goraly G and Hassin R (2010) Multi-color pebble motion on graphs. *Algorithmica* 58(3): 610–636.

Grinstead C and Snell J (2012) *Introduction to Probability*. Providence, RI: American Mathematical Society. Providence, RI, USA.

Hearn R and Demaine E (2005) PSPACE-completeness of sliding-block puzzles and other problems through the nondeterministic constraint logic model of computation. *Theoretical Computer Science* 343(1–2): 72–96.

Hirsch S and Halperin D (2002) Hybrid motion planning: coordinating two discs moving among polygonal obstacles in the plane. In: Boissonnat J-D, Burdick J, Goldberg K and Hutchinson S (eds) *Algorithmic Foundations of Robotics V*. New York: Springer, pp. 239–255.

Hopcroft J, Schwartz J and Sharir M (1984) On the complexity of motion planning for multiple independent objects; PSPACE-hardness of the "Warehouseman's problem". *International Journal of Robotics Research* 3(4): 76–88.

Hsu D, Latombe JC and Motwani R (1999) Path planning in expansive configuration spaces. *International Journal of Computational Geometry & Applications* 9(4–5): 495–512.

Karaman S and Frazzoli E (2011) Sampling-based algorithms for optimal motion planning. *International Journal of Robotics Research* 30(7): 846–894.

Kavraki LE, Švestka P, Latombe JC and Overmars M (1996) Probabilistic roadmaps for path planning in high dimensional configuration spaces. *IEEE Transactions on Robotics and Automation* 12(4): 566–580.

Kloder S and Hutchinson S (2005) Path planning for permutation-invariant multi-robot formations. In: *2005 IEEE international conference on robotics and automation (ICRA'05)*, Barcelona, Spain, 18–22 April 2005, pp. 1797–1802. Piscataway: IEEE Press.

Kornhauser D (1984) *Coordinating pebble motion on graphs, the diameter of permutation groups, and applications*. MSc Thesis, Massachusetts Institute of Technology, USA.

Kuffner J (2004) Effective sampling and distance metrics for 3D rigid body path planning. In: *2004 IEEE international conference on robotics and automation (ICRA'04)*, New Orleans, LA, 26 April–1 May 2004, pp. 3993–3998. Piscataway: IEEE Press.

Kuffner J and LaValle S (2000) RRT-Connect: An efficient approach to single-query path planning. In: *2000 IEEE international conference on robotics and automation (ICRA'00)*, April 24–28, 2000, San Francisco, CA, USA. pp. 995–1001. Piscataway: IEEE Press.

LaValle SM (2006) *Planning Algorithms*. Cambridge: Cambridge University Press.

LaValle SM and Kuffner J (1999) Randomized kinodynamic planning. In: *1999 IEEE international conference on robotics and automation (ICRA'99)*, Detroit, MI, 10–15 May 1999, pp. 473–479. Piscataway: IEEE Press.

Leroy S, Laumond JP and Simeon T (1999) Multiple path coordination for mobile robots: A geometric algorithm. In: *16th international joint conference on artificial intelligence (IJCAI'99)*, Barcelona, Catalonia, Spain, 16–22 July 2011, pp. 1118–1123.

Luna R and Bekris KE (2011) Push and swap: Fast cooperative path-finding with completeness guarantees. In: *22nd international joint conference on artificial intelligence (IJCAI'11)*, Barcelona, Catalonia, Spain, July 16–22, 2011. pp. 294–300.

Muja M and Lowe DG (2009) Fast approximate nearest neighbors with automatic algorithm configuration. In: *4th international conference on computer vision theory and applications (VISAPP'09)*, Lisbon, Portugal, 5–8 February 2009, pp. 331–340. Lisbon, Portugal: INSTICC Press.

Pearl J (1984) *Heuristics: Intelligent Search Strategies for Computer Problem Solving*. Boston, MA: Addison-Wesley.

Salzman O, Hemmer M and Halperin D (2015a) On the power of manifold samples in exploring configuration spaces and the dimensionality of narrow passages. *IEEE Transactions on Automation Science and Engineering* 12(2): 529–538.

Salzman O, Solovey K and Halperin D (2015b) Motion planning for multi-link robots by implicit configuration-space tiling. *Computing Research Repository*, abs/1504.06631.

Sanchez G and Latombe JC (2002) Using a PRM planner to compare centralized and decoupled planning for multi-robot systems. In: *2002 IEEE international conference on robotics and automation (ICRA'02)*, Washington, DC, 11–15 May 2002, pp. 2112–2119. Piscataway: IEEE Press.

Schwartz JT and Sharir M (1983) On the piano movers' problem: III. Coordinating the motion of several independent bodies. *International Journal of Robotics Research* 2(3): 46–75.

Sharir M and Sifrony S (1991) Coordinated motion planning for two independent robots. *Annals of Mathematics and Artificial Intelligence* 3(1): 107–130.

Solovey K and Halperin D (2014) *k*-Color multi-robot motion planning. *International Journal of Robotics Research* 33(1): 82–97.

Solovey K and Halperin D (2015) On the hardness of unlabeled multi-robot motion planning. In: *Robotics: science and systems XI* (ed LE Kavraki, D Hsu and J Buchli), Rome, Italy, 13–17 July 2015.

Solovey K, Yu J, Zamir O and Halperin D (2015) Motion planning for unlabeled discs with optimality guarantees. In: *Robotics: science and systems XI* (ed LE Kavraki, D Hsu and J Buchli), Rome, Italy, 13–17 July 2015.

Spirakis P and Yap C (1984) Strong NP-hardness of moving many discs. *Information Processing Letters* 19(1): 55–59.

Sucan IA, Moll M and Kavraki LE (2012) The open motion planning library. *IEEE Robotics & Automation Magazine* 19(4): 72–82.

The CGAL Project (2011) *CGAL User and Reference Manual*. xxx: CGAL Editorial Board, edition 3.9.

Turpin M, Mohta K, Michael N and Kumar V (2014) Goal assignment and trajectory planning for large teams of interchangeable robots. *Autonomous Robots* 37(4): 401–415.

van den Berg J and Overmars M (2005) Prioritized motion planning for multiple robots. In: *2005 IEEE/RSJ international conference on intelligent robots and systems (IROS'05)*, pp. 430–435. Piscataway: IEEE Press.

van den Berg J, Snoeyink J, Lin M and Manocha D (2009) Centralized path planning for multiple robots: Optimal decoupling into sequential plans. In: *Robotics: science and systems V* (ed J Trinkle, Y Matsuoka and JA Castellanos), Seattle, USA, 28 June–1 July 2009. Cambridge: MIT Press.

Švestka P and Overmars M (1998) Coordinated path planning for multiple robots. *Robotics and Autonomous Systems* 23(3): 125–152.

Wagner G and Choset H (2015) Subdimensional expansion for multirobot path planning. *Artificial Intelligence* 219: 1–24.

Yap C (1984) Coordinating the motion of several discs. Technical Report 105, Courant Institute of Mathematical Sciences, New York, USA.

6

New Perspective on Sampling-based Motion Planning via Random Geometric Graphs

Journal Title
XX(X):1–18

DOI: 10.1177/ToBeAssigned
www.sagepub.com/
SAGE

New perspective on sampling-based motion planning via random geometric graphs

Kiril Solovey[1], Oren Salzman[2] and Dan Halperin[1]

Abstract

Roadmaps constructed by many sampling-based motion planners coincide, in the absence of obstacles, with standard models of random geometric graphs (RGGs). Those models have been studied for several decades and by now a rich body of literature exists analyzing various properties and types of RGGs. In their seminal work on optimal motion planning Karaman and Frazzoli (2011) conjectured that a sampling-based planner has a certain property if the underlying RGG has this property as well. In this paper we settle this conjecture and leverage it for the development of a general framework for the analysis of sampling-based planners. Our framework, which we call *localization-tessellation*, allows for easy transfer of arguments on RGGs from the free unit-hypercube to spaces punctured by obstacles, which are geometrically and topologically much more complex. We demonstrate its power by providing alternative and (arguably) simple proofs for probabilistic completeness and asymptotic (near-)optimality of probabilistic roadmaps (PRMs). Furthermore, we introduce three variants of PRMs, analyze them using our framework, and discuss the implications of the analysis.

Keywords

motion planning, sampling-based algorithms, probabilistic roadmaps, random geometric graphs, probabilistic completeness, asymptotic optimality

1 Introduction

Motion planning is a fundamental research area in robotics with applications in diverse domains such as graphical animation, surgical planning, computational biology and computer games. For an overview of the subject and its applications, see, e.g., Choset et al. (2005); Latombe (1991); LaValle (2006); Halperin et al. (2016a,b).

The basic problem of motion planning is concerned with finding a collision-free path for a robot in a *workspace* cluttered with static obstacles. The spatial pose of the robot, or its *configuration*, is uniquely defined by its degrees of freedom (DOFs). The set of all configurations $\mathcal{C}$ is termed the *configuration space* of the robot, and decomposes into the disjoint sets of free and forbidden configurations, namely $\mathcal{F}$ and $\mathcal{C} \setminus \mathcal{F}$, respectively. Thus, given start and target configurations, the problem can be restated as the task of finding a continuous curve in $\mathcal{F}$ connecting the two configurations. This can be very challenging, as $\mathcal{F}$ can be exponentially complex (see, e.g., Canny (1988); Reif (1979); Solovey and Halperin (2015)) in the number of DOFs.

The high computational complexity of exact solutions to motion planning have led to the development of sampling-based planners. These algorithms, which trade completeness with applicability in practical settings, aim to capture the connectivity of $\mathcal{F}$ in a graph data structure, called a roadmap, by randomly sampling $\mathcal{C}$. Most of the theoretical properties of these algorithms are stated in terms of their *asymptotic* behavior, i.e., assuming that the number of samples is sufficiently large: The property of *probabilistic completeness* indicates that a given algorithm will eventually find a solution (if one exists); algorithms that are known to be *asymptotically optimal* also return a solution whose cost converges to the optimum.

Interestingly, roadmaps constructed by many sampling-based planners coincide, in the absence of obstacles, with standard models of random geometric graphs (RGGs). These models have been studied for several decades and by now a rich body of literature exists analyzing various

[1]Blavatnik School of Computer Science, Tel Aviv University, Israel
[2] Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA

Corresponding author:
Kiril Solovey, Blavatnik School of Computer Science, Tel Aviv University, Israel.
Email: kirilsol@post.tau.ac.il

properties and types of RGGs. Indeed, in their seminal work on optimal motion planning, Karaman and Frazzoli (2011) observed this relation. They employed techniques that were initially developed for the analysis of RGGs to the study of sampling-based planners. Subsequent proofs regarding completeness and optimality of new planners (see, e.g., Gammell et al. (2015); Janson et al. (2015); Salzman and Halperin (2014)) rely, to some extent, on the proofs in Karaman and Frazzoli (2011). Karaman and Frazzoli conjectured that a sampling-based planner possesses a certain property if the underlying RGG has this property as well (see (Karaman and Frazzoli 2011, Section 6)). The validity of this conjecture, which is settled in this paper, allows to import existing results on RGGs directly to the corresponding sampling-based planners.

Contribution. We introduce the *localization-tessellation* framework for the analysis of sampling-based algorithms in motion planning. Our framework facilitates the extension of properties of RGGs to sampling-based techniques in motion planning. This is done using conceptually simple ideas and elementary tools in probability theory. The underlying result of the framework is that RGGs demonstrate similar behavior in the absence as well as in the presence of obstacles. The framework consists of two main components. First we show through *localization* that RGGs maintain their properties in arbitrarily-small, yet fixed, neighborhoods. The *tessellation* stage extends these properties to complex domains which can be viewed as free spaces of motion-planning problems. Namely, the configuration space punctured by obstacles.

We demonstrate the power of the framework by providing conditions for probabilistic completeness and asymptotic (near-)optimality of Probabilistic Roadmaps (PRMs) Kavraki et al. (1996). Our proofs are (arguably) much simpler than the original proofs of Karaman and Frazzoli (2011).

Furthermore, we introduce three variants of PRMs called Soft-PRM, Bluetooth-PRM and Embedded-PRM, which perform connections in a randomized fashion, and analyze them using our framework. Using Soft-PRMs and Bluetooth-PRM we show that the standard PRM still maintains its favorable properties even when implemented using *approximate* nearest-neighbor search queries.

Organization. In Section 2 we review related work. In Section 3 we provide formal definitions of several types of RGGs and describe their properties, which will be employed by our localization-tessellation framework. In Section 4 we describe the *localization* component of the framework, that is, we show that RGGs maintain a wide range of their properties in arbitrarily-small neighborhoods. In Section 5 we focus on the two specific properties of connectivity and bounded stretch and show that they hold in general domains via a *tessellation* argument. In Section 6 we make the transition to motion planning: we describe several planners—including the standard PRM—and study their asymptotic behavior using the framework. In Section 7 we show empirically that the theoretical results obtained by the framework also hold in practice and compare the Soft-PRM and PRM algorithms.. We conclude the paper with a discussion and state several future research directions (Section 8). In the appendix we give a proof of Theorem 3, which appears in Section 3.

2 Related work

We review related work in the area of sampling-based algorithms for motion planning and random geometric graphs.

2.1 Sampling-based motion planning

Sampling-based algorithms, such as PRMs by Kavraki et al. (1996), Expansive Space Trees (EST) by Hsu et al. (1999) and Rapidly-exploring Random Trees (RRT) by Kuffner and LaValle (2000), as well as their many variants, have proven to be effective tools for motion planning. These algorithms, and others were shown to be probabilistically complete. While this is a desirable property of any algorithm, in certain applications stronger guarantees are required.

In recent years we have seen an increasing interest in *high-quality*[*] motion planning. The literature contains many examples of planners that are shown empirically to produce high-quality paths (for a partial list see Amato et al. (1998); Geraerts and Overmars (2007); Lien et al. (2003); Luna et al. (2013); Raveh et al. (2011); Siméon et al. (2000); Urmson and Simmons (2003)). Unfortunately, they are not backed by rigorous proofs pertaining to the quality of the solution produced by the algorithm. A complementary work by Nechushtan et al. (2010) proves theoretically that in certain settings RRT can produce paths of arbitrarily-poor quality.

Karaman and Frazzoli (2011) develop the first rigorous analysis of quality in the setting of sampling-based motion planning: They provide conditions under which existing planners are not asymptotically optimal. More importantly, they introduce two new variants of RRT and PRM, termed RRT* and PRM*, which are shown to be asymptotically optimal, under the right choice of parameters. Following this exposition, several asymptotically-optimal algorithms have emerged (see e.g., Alterovitz et al. (2011); Arslan and Tsiotras (2013); Gammell et al. (2015); Janson et al. (2015); Salzman and Halperin (2015)). To reduce the running time

[*]Quality can be measured in terms of length, clearance, smoothness, energy, to mention a few criteria. However, in this paper we will restrict our focus to the standard length measure.

of such algorithms several asymptotically *near*-optimal planners have been suggested, which trade the quality of the solution with speed of computation (see e.g., Dobson and Bekris (2014); Littlefield et al. (2013); Salzman and Halperin (2014); Salzman et al. (2014)).

Although the focus of this paper is on the simplified "geometric" setting of motion planning, we mention that some planners can cope with more complex robotic systems in which uncertainty and physical constraints come into play (see, e.g., Karaman and Frazzoli (2010); Ladd and Kavraki (2004); Li et al. (2014); Perez et al. (2012); Şucan and Kavraki (2012); Webb and van den Berg (2013); Xie et al. (2015); Schmerling et al. (2015a,b)). Some of these planners can also produce high-quality paths.

2.2 Random geometric graphs

The study of random geometric graphs (RGGs) was initiated by Gilbert (1961) who considered the following model: a collection of points is sampled at random in a given subspace of $\mathbb{R}^d$, and a graph is formed by drawing edges between points that are closer than a given threshold $r > 0$, called the *connection radius*.

An immediate question that follows is for which values of r the graph is connected (with high probability). Several works have addressed this question and showed that it is both necessary and sufficient that the connection radius will be proportional to $\left(\frac{\log n}{n}\right)^{1/d}$, where n is the number of points and the points are sampled from the unit hypercube $[0,1]^d$ (see, e.g., Appel and Russo (2002); Kozma et al. (2010); Penrose (1997)). Penrose (1999) established that connectivity occurs approximately when the graph has no isolated vertices. His monograph (Penrose (2003)) studies many more properties of RGGs, including vertex degree, clique size and coloring. The reader is also referred to a survey on the subject by Walters (2011).

In recent years RGGs have attracted much attention as a tool for modeling large-scale communication networks, and in particular sensor networks: the vertices of the graph represent sensors and an edge is drawn between two sensors that are in the communication range. Gupta and Kumar (1999) used this analogy in order to deduce the transmission power necessary for the network to be connected. An important parameter that arises in this context is the number of transmitters a message has to traverse in order to establish a broadcast between two given transmitters. Several works have established that this parameter is proportional to the Euclidean distance between the two nodes (see, e.g., Bradonjic et al. (2010); Díaz et al. (2015); Ellis et al. (2007); Friedrich et al. (2013); Mitsche and Perarnau (2012); Muthukrishnan and Pandurangan (2010)).

Various alternative connection strategies for RGGs have been proposed over the years, the most studied of which is the k-nearest model (see, e.g., Bagchi and Bansal (2008); Balister et al. (2009); Xue and Kumar (2004)). More complex models assign edges between vertices in a randomized fashion (see, e.g., Broutin et al. (2014); Frieze and Pegden (2014); Penrose (2016)). Some models introduce an ordering on the sampled points (see, e.g., Bhatt and Roy (2004); Penrose and Wade (2010a,b); Schulte and Thaele (2014); Wade (2009)) which results in a directed graph that resembles the aforementioned RRT.

3 Preliminaries

We describe several models of random geometric graphs (RGGs) and mention useful properties that will be used throughout the paper. When possible, we follow the notation and conventions in the standard literature of RGGs (see, e.g., Penrose (2003)). Let $\mathcal{X}_n = \{X_1, \ldots, X_n\}$ be n points chosen independently and uniformly at random from the Euclidean d-dimensional Euclidean cube $[0,1]^d$. We assume that the dimension d of the domain is fixed and greater than one. Let $\|x-y\|_2$ denote the Euclidean distance between two points $x, y \in \mathbb{R}^d$ and θ_d denote the Lebesgue measure of the unit ball in $\mathbb{R}^d$. Finally, denote by $\mathcal{B}_r(x)$ be the d-dimensional ball of radius $r > 0$ centered at $x \in \mathbb{R}^d$ and $\mathcal{B}_r(\Gamma) = \bigcup_{x\in\Gamma} \mathcal{B}_r(x)$ for any $\Gamma \subseteq \mathbb{R}^d$. Similarly, given a curve $\sigma : [0,1] \to \mathbb{R}^d$ denote $\mathcal{B}_r(\sigma) = \bigcup_{\tau\in[0,1]} \mathcal{B}_r(\sigma(\tau))$.

Throughout the paper we will use the standard notation for asymptotic bounds: Let $f = f(n), g = g(n)$ be two functions. The notation $f = \omega(g)$ indicates that $\lim_{n\to\infty} f/g \to \infty$, and $f = o(g)$ indicates that $\lim_{n\to\infty} f/g \to 0$. Let $A_1, A_2, \ldots$ be random variables in some probability space and let B be an event depending on A_n. We say that B occurs *asymptotically almost surely* (a.a.s., in short) if $\lim_{n\to\infty} \Pr[B(A_n)] = 1$. Finally, all logarithms are at base e.

Our first main definition is concerned with the most basic model of RGGs, sometimes called a random disk graph.

Definition 1. (Penrose (2003)) Given $r_n \in \mathbb{R}^+$, the *random geometric graph* (RGG) $\mathcal{G}^{\text{disk}}(\mathcal{X}_n; r_n)$ is an undirected graph with the vertex set $\mathcal{X}_n$. For any two given vertices $x, y \in \mathcal{X}_n$ the graph contains the edge (x, y) if $\|x-y\|_2 \leqslant r_n$.

We use the term RGG to refer both to the family of random geometric graphs and to the specific model described in Definition 1. This slight abuse of notation is introduced to be consistent with existing literature and the exact meaning of RGG will be clear from the context.

The following definition is concerned with a more complex structures called *random Bluetooth graphs*, also known as *random irrigation graphs*.

Definition 2. (Broutin et al. (2014)) Let $2 \leqslant c_n \leqslant n$ be a positive integer and $r_n \in \mathbb{R}^+$. The random *Bluetooth* graph (RBG) $\mathcal{G}_n^{\text{BT}} = \mathcal{G}^{\text{BT}}(\mathcal{X}_n; r_n; c_n)$ is an undirected graph with the vertex set $\mathcal{X}_n$. For every $x \in \mathcal{X}_n$ let $E(x, r_n)$ denote the set of points within maximal distance r_n from x, i.e.,

$$E(x, r_n) = \{(x, y) : y \in \mathcal{X}_n \setminus \{x\}, \|x - y\|_2 \leqslant r_n\}.$$

For every $x \in \mathcal{X}_n$ we pick randomly and independently c_n edges from $E(x, r_n)$, and denote this set of edges by $E(x, r_n, c_n)$. The edge set of $\mathcal{G}_n^{\text{BT}}$ is defined to be $\bigcup_{x \in \mathcal{X}_n} E(x, r_n, c_n)$.

The following model is also a generalization of RGGs. Here a pair of vertices are connected by an edge with a probability that depends on the length of the edge.

Definition 3. (Penrose (2016)) Let $r_n \in \mathbb{R}^+$, and $\phi_n : \mathbb{R}^+ \to [0, 1]$ is a probability measure over $[0, r_n]$. The *soft* random geometric graph (SRGG) $\mathcal{G}^{\text{soft}}(\mathcal{X}_n; r_n; \phi_n)$ is an undirected graph with the vertex set $\mathcal{X}_n$. Denote by E the edge set of this graph. For a pair of vertices $x, y \in \mathcal{X}_n$ such that $\|x - y\|_2 \leqslant r_n$ it holds that $\Pr[(x, y) \in E] = \phi_n(\|x - y\|_2)$, independently for each edge.

The following model can be viewed as a special case of SRGG where $r_n = \infty$ and ϕ_n is constant.

Definition 4. (Frieze and Pegden (2014)) The randomly-*embedded* geometric graph (REGG) $\mathcal{G}^{\text{embed}}(\mathcal{X}_n; p_n)$ is an undirected graph with the vertex set $\mathcal{X}_n$. For every two distinct vertices $x, y \in \mathcal{X}_n$, the graph contains the edge (x, y) with probability p_n, and independently from the other edges.

Throughout the text we will omit the superscript indicating the graph type, and use instead the notation $\mathcal{G}_n$, if the type in question is clear from the context.

3.1 Connectivity

Recall that for every undirected graph $\mathcal{G}$, two vertices u and v are called *connected* if $\mathcal{G}$ contains a path from u to v. A graph is said to be connected if every pair of its vertices is connected. Later on in the paper we will show that a sampling-based planner is probabilistically complete if the underlying RGG is connected a.a.s.

We mention three results related to the connectivity of the RGG,RBG, and SRGG models.

Theorem 1. (Broutin et al. (2014)) *Let* $\mathcal{G}_n = \mathcal{G}^{\text{disk}}(\mathcal{X}_n, r_n)$ *and* $r_n = \gamma \left(\frac{\log n}{n}\right)^{1/d}$. *Then*

$$\lim_{n \to \infty} \Pr[\mathcal{G}_n \text{ is connected}] = \begin{cases} 0 & \text{if } \gamma < \gamma^*, \\ 1 & \text{if } \gamma > \gamma^*, \end{cases}$$

where $\gamma^* = 2(2d\theta_d)^{-1/d}$.

Theorem 2. (Broutin et al. (2014)) *Let* $\mathcal{G}_n = \mathcal{G}^{\text{BT}}(\mathcal{X}_n; r_n; c_n)$, *where* $d \geqslant 2$, *and* $r_n = \gamma \left(\frac{\log n}{n}\right)^{1/d}$, *where* $\gamma > \gamma^{**}$ *for* $\gamma^{**} = d \cdot 2^{1+1/d}$. *Then*

$$\lim_{n \to \infty} \Pr[\mathcal{G}_n \text{ is connected}] = \begin{cases} 0 & \text{if } c_n < c_n^*, \\ 1 & \text{if } c_n > c_n^*, \end{cases}$$

where $c_n^* = \sqrt{\frac{2 \log n}{\log \log n}}$.

Recently, Penrose (2016) developed a general characterization of the necessary condition over r_n and ϕ_n so that $\mathcal{G}^{\text{soft}}(\mathcal{X}_n; r_n; \phi_n)$ will be connected. We chose to focus here on a specific range of values which can be of interest to motion planning. The following theorem is proven in the appendix.

Theorem 3. *Let* $\mathcal{G}_n = \mathcal{G}^{\text{soft}}(\mathcal{X}_n; r_n; \phi_n)$ *and* $r_n = \gamma \left(\frac{\log n}{n}\right)^{1/d}$. *Set* $\gamma > (d+1)^{1/d}\gamma^*$ *(see Theorem 1), and define* $\phi_n(z) = 1 - z/r_n$, *for any* $z \in \mathbb{R}^+$. *Then* $\mathcal{G}_n$ *is connected a.a.s.*

3.2 Bounded stretch

Let $\mathcal{G}$ be a graph whose vertices are embedded in $\mathbb{R}^d$, and the edge weights correspond to the Euclidean distance between the edges' endpoints. For every two vertices $x, y \in \mathcal{G}$ denote their *weighted graph distance*, i.e., the sum of lengths of the shortest path from x to y, by $\text{dist}(\mathcal{G}, x, y)$. Throughout the paper we will use the term *stretch* to denote the ratio between $\text{dist}(\mathcal{G}, x, y)$ and the length of the shortest path between x, y in the domain in which the graph is embedded. For instance, if this domain is convex, then for every $x, y \in \mathcal{G}$ the stretch is defined to be $\text{dist}(\mathcal{G}, x, y)/\|x - y\|_2$.

In the setting of motion planning, we will use the graph distance to bound the asymptotic path length of sampling-based planners.

Theorem 4. (Friedrich et al. (2013)) *Let* $\mathcal{G}_n = \mathcal{G}^{\text{disk}}(\mathcal{X}_n; r_n)$ *with* $r_n = \gamma \left(\frac{\log n}{n}\right)^{1/d}$ *where* $\gamma > \gamma^*$ *(see Theorem 1). Then there exists a constant* ζ *such that for every two vertices* x, y *in the same connected component of* $\mathcal{G}_n$, *with* $\|x - y\|_2 = \omega(r_n)$, *it holds that* $\text{dist}(\mathcal{G}_n, x, y)$ *is at most* $\zeta\|x - y\|_2$ *a.a.s.*

Theorem 5. (Frieze and Pegden (2014)) *Let* $\mathcal{G}_n = \mathcal{G}^{\text{embed}}(\mathcal{X}_n; p_n)$ *and* $p_n = \omega\left(\frac{\log^d n}{n}\right)$. *Then for every two vertices* $x, y \in \mathcal{X}_n$ *it holds that* $\text{dist}(\mathcal{G}_n, x, y)$ *is at most* $\|x - y\|_2 + o(1)$ *a.a.s.*

Notice that this statement does not condition the existence of a short graph path on the event that the two vertices are in the same connected component of the graph. Due to this fact it also follows that the graph is connected with high probability.

4 Localization of monotone properties of RGGs

In this section we discuss graph properties and their asymptotic behavior, when focusing on a subset of the domain $[0,1]^d$. A property $\mathcal{A}$ is *monotone* if for every $G=(V,E)$ and $H=(V,E')$ such that $E\subseteq E'$, it holds that $G\in\mathcal{A} \implies H\in\mathcal{A}$. Note that connectivity (Section 3.1) and bounded stretch (Section 3.2) are monotone.†

The following four lemmata state that if an RGG a.a.s. possesses a certain monotone property, then the restriction of this to a local domain a.a.s. has the aforementioned property as well. These will serve as main ingredients in the extension of existing properties of RGGs to more complex domains than $[0,1]^d$ (see Section 5).

Definition 5. Let $\mathcal{G}=(X,E)$ be a graph embedded in $[0,1]^d$, i.e., the vertices X represent points in $[0,1]^d$ and edges represent straight-line paths between the corresponding vertices. Given $\Gamma\subset[0,1]^d$ we denote by $\mathcal{G}(\Gamma)$ the graph obtained from the intersection of $\mathcal{G}$ and Γ. This graph consists of the vertex set $X\cap\Gamma$ and all the edges in E that are fully contained in Γ.

Definition 6. Let $\mathcal{G}_n$ be any RGG, RBG, SRGG or REGG, defined over the vertex set $\mathcal{X}_n$. Then $\mathcal{G}_n$ is *localizable* for a property $\mathcal{A}$ if for **any given** constant $0<\varepsilon\leqslant 1$ and d-dimensional axis-aligned cube $B_\varepsilon\subseteq[0,1]^d$ with side length of ε it holds that $\mathcal{G}_n(B_\varepsilon)\in\mathcal{A}$ a.a.s.

Lemma 1. *Let $\mathcal{A}$ be a monotone property and $\gamma_{\mathcal{A}}$ some constant. Let $\mathcal{G}_n=\mathcal{G}^{\text{disk}}(\mathcal{X}_n;r_n)$ be an RGG such that $\mathcal{G}_n\in\mathcal{A}$ a.a.s., for $r_n=\gamma\left(\frac{\log n}{n}\right)^{1/d}$, where $\gamma>\gamma_{\mathcal{A}}$. Then for any fixed $\eta>1$, $\mathcal{G}^{\text{disk}}(\mathcal{X}_n;\eta r_n)$ is localizable for $\mathcal{A}$.*

Proof. For simplicity of presentation, we will use the notation $\mathcal{G}$ to refer to $\mathcal{G}^{\text{disk}}$ throughout the proof. Recall that $\mathcal{X}_n$ is a collection of n points chosen independently and uniformly at random from $[0,1]^d$. We will also use $\mathcal{Y}_m^\varepsilon=\{Y_1,\ldots,Y_m\}$ to denote a collection of m points chosen independently and uniformly at random from B_ε. Without loss of generality, assume that $B_\varepsilon=[0,\varepsilon]^d$.

Observe that

$$\begin{aligned}&\Pr[\mathcal{G}(\mathcal{X}_n\cap B_\varepsilon;\eta r_n)\notin\mathcal{A}]\\&\quad=\sum_{m=0}^{n}\Pr\left[\mathcal{G}\left(\mathcal{X}_n\cap B_\varepsilon;\eta r_n\right)\notin\mathcal{A}\;\middle|\;|\mathcal{X}_n\cap B_\varepsilon|=m\right]\\&\quad\cdot\Pr\left[|\mathcal{X}_n\cap B_\varepsilon|=m\right]\\&\quad=\sum_{m=0}^{n}\Pr\left[\mathcal{G}\left(\mathcal{Y}_m^\varepsilon;\eta r_n\right)\notin\mathcal{A}\right]\cdot\Pr\left[|\mathcal{X}_n\cap B_\varepsilon|=m\right].\end{aligned}$$

Denote

$$\sigma(i,j)=\sum_{m=i}^{j}\Pr\left[\mathcal{G}\left(\mathcal{Y}_m^\varepsilon;\eta r_n\right)\notin\mathcal{A}\right]\cdot\Pr\left[|\mathcal{X}_n\cap B_\varepsilon|=m\right],$$

and by definition we have that for $1\leqslant\ell\leqslant n$

$$\begin{aligned}&\Pr\left[\mathcal{G}(\mathcal{X}_n\cap B_\varepsilon;\eta r_n)\notin\mathcal{A}\right]\\&\quad=\sigma(0,n)=\sigma(0,\ell-1)+\sigma(\ell,n).\end{aligned}$$

We show that for $\ell=\eta^{-d}\varepsilon^d n$ both $\lim_{n\to\infty}\sigma(0,\ell-1)=0$ and $\lim_{n\to\infty}\sigma(\ell,n)=0$ which will conclude the proof of the lemma (for simplicity we assume that $\ell\in\mathbb{N}$). We start with the former expression:

$$\begin{aligned}\sigma(0,\ell-1)&=\sum_{m=0}^{\ell-1}\Pr\left[\mathcal{G}\left(\mathcal{Y}_m^\varepsilon;\eta r_n\right)\notin\mathcal{A}\right]\cdot\Pr\left[|\mathcal{X}_n\cap B_\varepsilon|=m\right]\\&\leqslant\sum_{m=0}^{\ell-1}1\cdot\Pr\left[|\mathcal{X}_n\cap B_\varepsilon|=m\right]\\&=\Pr\left[|\mathcal{X}_n\cap B_\varepsilon|<\ell\right]\\&=\Pr\left[|\mathcal{X}_n\cap B_\varepsilon|<\eta^{-d}\mathbb{E}[|\mathcal{X}_n\cap B_\varepsilon|]\right]\\&\leqslant\exp\left\{-n\varepsilon^d(1-\eta^{-d})^2\right\}.\end{aligned}$$

The last inequality follows from the fact that $|\mathcal{X}_n\cap B_\varepsilon|$ is a *binomial random variable* with n elements, success rate of $|B_\varepsilon|=\varepsilon^d$ per trial, and a mean value of $\mathbb{E}[|\mathcal{X}_n\cap B_\varepsilon|]=n\varepsilon^d$. This in turn enables the use of *Chernoff inequality* (see, e.g., (Dubhashi and Panconesi 2009, Theorem 1.1)), the application of which is made possible due to the η^{-d} factor.

We now focus on showing that $\lim_{n\to\infty}\sigma(\ell,n)=0$. For any two integers n,m such that $\ell\leqslant m\leqslant n$ we have that

$$\begin{aligned}\Pr[\mathcal{G}(\mathcal{Y}_m^\varepsilon;\eta r_n)\notin\mathcal{A}]&\overset{(1)}{=}\Pr[\mathcal{G}(\mathcal{X}_m;\eta\varepsilon^{-1}r_n)\notin\mathcal{A}]\\&\overset{(2)}{\leqslant}\Pr[\mathcal{G}(\mathcal{X}_m;r_m)\notin\mathcal{A}],\end{aligned}$$

where the transitions are made possible due to (1) a scaling of the graph from $[0,\varepsilon]^d$ to $[0,1]^d$; (2) the monotonicity of $\mathcal{A}$ and the fact that $r_m\leqslant\frac{\eta}{\varepsilon}r_n$. To show that indeed $r_m\leqslant\frac{\eta}{\varepsilon}r_n$, note that $\eta\varepsilon^{-1}>1$ and that $\lim_{n\to\infty}r_n=0$. Thus,

$$\begin{aligned}r_m\leqslant r_\ell=\eta^{-1}\eta r_n&=\eta^{-1}\gamma\left(\frac{\log\eta^{-d}\varepsilon^d n}{\eta^{-d}\varepsilon^d n}\right)^{1/d}\\&=\varepsilon^{-1}\gamma\left(\frac{\log\eta^{-d}\varepsilon^d n}{n}\right)^{1/d}\\&\leqslant\varepsilon^{-1}\gamma\left(\frac{\log n}{n}\right)^{1/d}=\frac{\eta}{\varepsilon}r_n.\end{aligned}$$

† Additional examples of monotone properties for a graph $\mathcal{G}$ are: $\mathcal{G}$ is Hamiltonian, $\mathcal{G}$ contains a clique of size t, $\mathcal{G}$ is not planar, the clique number of $\mathcal{G}$ is larger than that of its complement, the diameter of $\mathcal{G}$ is at most s, etc.

Now, set $m^* = \text{argmax}_{m\in[\ell,n]} \left(\Pr\left[\mathcal{G}(\mathcal{X}_m, r_m) \notin \mathcal{A}\right]\right)$. It follows that

$$\begin{aligned}\sigma(\ell,n) &= \sum_{m=\ell}^{n} \Pr\left[\mathcal{G}\left(\mathcal{Y}_m^{\varepsilon}; \eta r_n\right) \notin \mathcal{A}\right] \cdot \Pr\left[|\mathcal{X}_n \cap B_\varepsilon| = m\right] \\ &\leqslant \sum_{m=\ell}^{n} \Pr\left[\mathcal{G}(\mathcal{X}_m, r_m) \notin \mathcal{A}\right] \cdot \Pr\left[|\mathcal{X}_n \cap B_\varepsilon| = m\right] \\ &\leqslant \Pr\left[\mathcal{G}(\mathcal{X}_{m^*}, r_{m^*}) \notin \mathcal{A}\right] \sum_{m=\ell}^{n} \Pr\left[|\mathcal{X}_n \cap B_\varepsilon| = m\right] \\ &= \Pr\left[\mathcal{G}(\mathcal{X}_{m^*}, r_{m^*}) \notin \mathcal{A}\right] \cdot \Pr\left[|\mathcal{X}_n \cap B_\varepsilon| \geqslant \ell\right] \\ &\leqslant \Pr\left[\mathcal{G}(\mathcal{X}_{m^*}, r_{m^*}) \notin \mathcal{A}\right].\end{aligned}$$

Note that $\lim_{n\to\infty} \Pr\left[\mathcal{G}(\mathcal{X}_{m^*}, r_{m^*}) \notin \mathcal{A}\right] = 0$, which concludes the proof. □

The following are the RBG, SRGG and REGG equivalents of Lemma 1.

Lemma 2. *Let $\mathcal{A}$ be a monotone property and $\gamma_{\mathcal{A}}$ some constant. Let $\mathcal{G}_n = \mathcal{G}^{\text{BT}}(\mathcal{X}_n; r_n; c_n)$ be an RBG such that $\mathcal{G}_n \in \mathcal{A}$ a.a.s., for every $r_n = \gamma_{\mathcal{A}} \left(\frac{\log n}{n}\right)^{1/d}$, where $\gamma > \gamma_{\mathcal{A}}$, and c_n is non-decreasing. Then for any fixed $\eta > 1$, $\mathcal{G}^{\text{BT}}(\mathcal{X}_n; \eta r_n; c_n)$ is localizable for $\mathcal{A}$.*

Proof. We only prove the following inequality, as the rest of the proof proceeds in a manner similar to that of Lemma 1. We keep the notation from the previous proof. Recall that $\ell = \eta^{-d}\varepsilon^d n$. For any two integers m, n such that $\ell \leqslant m \leqslant n$ we have that

$$\begin{aligned}\Pr[\mathcal{G}(\mathcal{Y}_m^{\varepsilon}; \eta r_n; c_n) \notin \mathcal{A}] &= \Pr[\mathcal{G}(\mathcal{X}_m; \eta\varepsilon^{-1} r_n; c_n) \notin \mathcal{A}] \\ &\leqslant \Pr[\mathcal{G}(\mathcal{X}_m; r_m; c_n) \notin \mathcal{A}] \\ &\leqslant \Pr[\mathcal{G}(\mathcal{X}_m; r_m; c_m) \notin \mathcal{A}].\end{aligned}$$

Here we used the fact that $c_n \geqslant c_m$. □

Lemma 3. *Let $\mathcal{A}$ be a monotone property and let $\gamma_{\mathcal{A}}$ some constant. Let $\mathcal{G}_n = \mathcal{G}^{\text{soft}}(\mathcal{X}_n; r_n; \phi_n)$ be an SRGG such that $\mathcal{G}_n \in \mathcal{A}$ a.a.s., where $r_n = \gamma \left(\frac{\log n}{n}\right)^{1/d}$ for some $\gamma > \gamma_{\mathcal{A}}$, and for every $z \in \mathbb{R}^+$, the function $\phi_n(z)$ is increasing. Then for any fixed $\eta > 1$, $\mathcal{G}^{\text{disk}}(\mathcal{X}_n; \eta r_n; \phi_n)$ is localizable for $\mathcal{A}$.*

Proof. The proof is identical to that of Lemma 2. One only needs to replace c_n with ϕ_n. □

Lemma 4. *Let $\mathcal{A}$ be a monotone property and let $\mathcal{G}_n = \mathcal{G}^{\text{embed}}(\mathcal{X}_n; p_n)$ be an REGG such that $\mathcal{G}_n \in \mathcal{A}$ a.a.s., where p_n is non-decreasing. Then $\mathcal{G}_n$ is localizable for $\mathcal{A}$.*

Proof. The proof follows very similar lines of the proof of Lemma 1. The main observation here is that for every $m < n$ it follows that

$$\Pr[\mathcal{G}(\mathcal{X}_m; p_n) \notin \mathcal{A}] \leqslant \Pr[\mathcal{G}(\mathcal{X}_m; p_m) \notin \mathcal{A}]$$

due to the monotonicity of $\mathcal{A}$ and the fact that p_n is non-decreasing. □

5 Properties of RGGs in general domains via tessellation

In the previous section we considered four models of RGGs defined over the *convex* domain $[0,1]^d$. We discussed the **sufficient** conditions such that random graphs will be localizable for any monotone property $\mathcal{A}$. In this section we consider the specific monotone properties of *connectivity* and *bounded stretch* for general domains.

A region $\Gamma \subset [0,1]^d$ is said to be *ρ-safe* for some $\rho > 0$ if $\mathcal{B}_\rho(\Gamma) \subset [0,1]^d$, namely if the Minkowski sum of Γ with an origin-centered ball of radius ρ is contained in $[0,1]^d$.

5.1 Connectivity

Denote by $\mathcal{A}_{\text{conn}}$ the connectivity property. We show that for any random graph $\mathcal{G}_n$ which is an RGG, RBG, SRGG or REGG that is localizable for $\mathcal{A}_{\text{conn}}$ it also holds that $\mathcal{G}_n$ is connected over any ρ-safe region $\Gamma \subset [0,1]^d$. Note that we make no additional assumptions on Γ in this section.

Theorem 6. *Let $\Gamma \subset [0,1]^d$ be a ρ-safe region for some constant $\rho > 0$ independent of n and let $\mathcal{G}_n$ be a random graph that is localizable for $\mathcal{A}_{\text{conn}}$. Then any two points $x, y \in \Gamma \cap \mathcal{X}_n$ that lie in the same connected component of Γ are connected in $\mathcal{G}_n(\mathcal{B}_\rho(\Gamma))$ a.a.s.*

In the proof of Theorem 6 we will place two partially-overlapping grids over Γ and use the localization of $\mathcal{G}_n$ in each grid cell (see Fig. 1). We now proceed to define the grids and state several of their properties which, in turn, will allow us to formally prove Theorem 6.

Let $\mathbb{H}_\varepsilon$ be a grid partition of $[0,1]^d$ into axis-aligned hypercubes with side length of $\varepsilon = \frac{2}{3\sqrt{d}}\rho$. Furthermore, denote by $\mathbb{H}_\varepsilon(\Gamma)$ the subset of cells of $\mathbb{H}_\varepsilon$ whose intersection with Γ is non-empty. Namely, $\mathbb{H}_\varepsilon(\Gamma) = \{H \in \mathbb{H}_\varepsilon \mid H \cap \Gamma \neq \emptyset\}$. Let $\tilde{\mathbb{H}}_\varepsilon$ be a grid partition of $[0,1]^d$ into axis-aligned hypercubes with side length of ε obtained by shifting $\mathbb{H}_\varepsilon$ by $\varepsilon/2$ along every axis and let $\tilde{\mathbb{H}}_\varepsilon(\Gamma) = \{H \in \tilde{\mathbb{H}}_\varepsilon \mid H \cap \mathbb{H}_\varepsilon \neq \emptyset\}$. We have the following claim.

Claim 1. *Let $H \in \mathbb{H}_\varepsilon(\Gamma) \cup \tilde{\mathbb{H}}_\varepsilon(\Gamma)$. Then $H \subset \mathcal{B}_\rho(\Gamma)$.*

Proof. Consider a hypercube $H \in \mathbb{H}_\varepsilon(\Gamma)$. By the definition of $\mathbb{B}_\varepsilon(\Gamma)$, H intersects Γ and let $x \in H \cap \Gamma$ be some intersection point. Since $x \in \Gamma$, we have that $\|x - y\|_2 > \rho$ for any point $y \notin \mathcal{B}_\rho(\Gamma)$. Recall that H is an axis-aligned

hypercube with side length of $\varepsilon = \frac{2}{3\sqrt{d}}\rho$. Thus, the maximal distance between any two points in H is $\varepsilon\sqrt{d} = \frac{2}{3}\rho$. Using the triangle inequality we have for every point $x' \in H$ and for any point $y \notin \mathcal{B}_\rho(\Gamma)$,

$$\|x' - y\|_2 \geqslant \|x - y\|_2 - \|x - x'\|_2 > \rho - \frac{2}{3}\rho = \frac{1}{3}\rho > 0,$$

which implies that $x' \in \mathcal{B}_\rho(\Gamma)$.

The proof for a hypercube $\tilde{H} \in \tilde{\mathbb{H}}_\varepsilon(\Gamma)$ follows similar lines using the fact that for any point $x' \in \tilde{H}$ and any point $x \in H$ such that $H \cap \tilde{H} \neq \emptyset$ we have that $\|x - x'\|_2 \leqslant \frac{3}{2}\varepsilon\sqrt{d} = \rho$. □

We introduce some more terminology. Every two cells $H, H' \in \mathbb{H}_\varepsilon(\Gamma)$ are called *neighbors* if they share a $(d-1)$-dimensional face. We now consider a refinement of each grid cell H of $\mathbb{H}_\varepsilon(\Gamma)$ (or of $\tilde{\mathbb{H}}_\varepsilon(\Gamma)$) into 2^d sub-cells obtained by splitting H by two along each axis through the middle point of H. This induces the set of (refined) grid cells $\mathbb{H}_{\varepsilon/2}(\Gamma)$ (or $\tilde{\mathbb{H}}_{\varepsilon/2}(\Gamma)$, respectively). Note that the number of cells in $\mathbb{H}_{\varepsilon/2}(\Gamma)$ and $\tilde{\mathbb{H}}_{\varepsilon/2}(\Gamma)$ is fixed for the given d, ρ, Γ, and does not depend on n.

Claim 2. *Let $H \in \mathbb{H}_\varepsilon(\Gamma) \cup \tilde{\mathbb{H}}_\varepsilon(\Gamma)$ (similarly for $H \in \mathbb{H}_{\varepsilon/2}(\Gamma) \cup \tilde{\mathbb{H}}_{\varepsilon/2}(\Gamma)$). Then $\mathcal{X}_n \cap H \neq \emptyset$, a.a.s.*

Proof. We show the proof for $\mathbb{H}_\varepsilon(\Gamma)$, and the proof for $\tilde{\mathbb{H}}_\varepsilon(\Gamma)$ follows similar lines. We start by showing that the probability that a specific cell $H \in \mathbb{H}_\varepsilon(\Gamma)$ does not contain a sample of $\mathcal{X}_n$ tends to 0:

$$\Pr[\mathcal{X}_n \cap H = \emptyset] = (1 - |H|)^n = (1 - \varepsilon^d)^n \leqslant e^{-n\varepsilon^d}.$$

Using the union bound, we deduce,

$$\Pr[\exists H \in \mathbb{H}_\varepsilon(\Gamma) : \mathcal{X}_n \cap H = \emptyset] \leqslant \sum_{H \in \mathbb{H}_\varepsilon(\Gamma)} \Pr[\mathcal{X}_n \cap H = \emptyset] \leqslant be^{-n\varepsilon^d},$$

where b denotes the number of cells in $\mathbb{B}_\varepsilon(\Gamma)$. As b is independent of n the last expression tends to 0 as n tends to ∞. □

We are ready for the main proof.

Proof (Theorem 6). Recall that $\mathcal{G}_n$ is localizable for $\mathcal{A}_{\text{conn}}$. As $\bigcup_{H \in \mathbb{H}_\varepsilon(\Gamma)} \subset \mathcal{B}_\rho(\Gamma)$, and since x and y are in the same connected component of Γ, there exists a sequence of hypercubes $H_1, \ldots, H_k \in \mathbb{H}_\varepsilon(\Gamma)$ such that (i) $x \in H_1$, (ii) $y \in H_k$ and (iii) H_i and H_{i+1} are neighbors for $1 \leqslant i < k$. By Claim 1 each H_i is contained in $\mathcal{B}_\rho(\Gamma)$.

Claim 2 ensures, using the fact that Γ is ρ-safe, that each H_i contains a vertex of $\mathcal{G}_n$ a.a.s. Let $x = x_1, \ldots, x_k = y$ denote such a set of vertices where $x_i \in H_i$. We will show (using the localization of monotone properties) that x_i and x_{i+1} are connected in $\mathcal{G}_n(\mathcal{B}_\rho(\Gamma))$ which will conclude our proof.

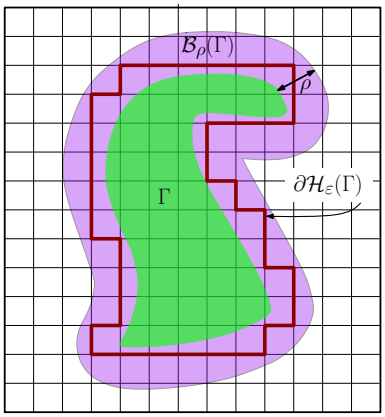


Figure 1. Visualization of Γ (green), $\mathcal{B}_\rho(\Gamma)$ (purple) and the grid $\mathbb{H}_\varepsilon$ used for the proof of Theorem 6. The boundary of the set of grid cells $\mathbb{H}_\varepsilon(\Gamma)$ is depicted using dark red lines.

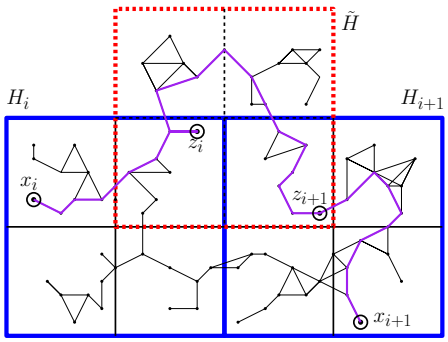


Figure 2. Visualization of the proof of Theorem 6. Hypercubes H_i, H_{i+1} and $\tilde{H}$ of side length ε are depicted in solid blue lines and dashed red lines, respectively. A path connecting $x_i \in H_i$ to $x_{i+1} \in H_{i+1}$ via intermediate points $z_i \in H_i \cap \tilde{H}$ and $z_{i+1} \in H_{i+1} \cap \tilde{H}$ is depicted by a purple line.

Let $\tilde{H} \in \tilde{\mathbb{H}}_\varepsilon(\Gamma)$ be a hypercube that intersects both H_i and H_{i+1} (there are always 2^{d-1} such hypercubes). By Claim 2, both $\tilde{H} \cap H_i$ and $\tilde{H} \cap H_{i+1}$ contain a vertex of $\mathcal{G}_n$ a.a.s., since both of these intersection represent hypercubes in $\mathbb{H}_{\varepsilon/2}(\Gamma)$. Let z_i and z_{i+1} be these vertices, respectively (see Fig. 2).

Now, using Lemmata 1-4 we have that x_i and z_i are connected in H_i, that z_i and z_{i+1} are connected in $\tilde{H}$, and that z_{i+1} and x_{i+1} are connected in H_{i+1} a.a.s. This must hold for every $1 \leqslant i < k$ in order to ensure that x and y are connected in $\mathcal{G}_n(\mathcal{B}_\rho(\Gamma))$. Due to the fact that k can be at most the number of cells in $\mathbb{H}_\varepsilon(\Gamma_\rho)$, which is independent of n, we deduce that indeed x, y are connected in $\mathcal{G}_n(\mathcal{B}_\rho(\Gamma))$ a.a.s. □

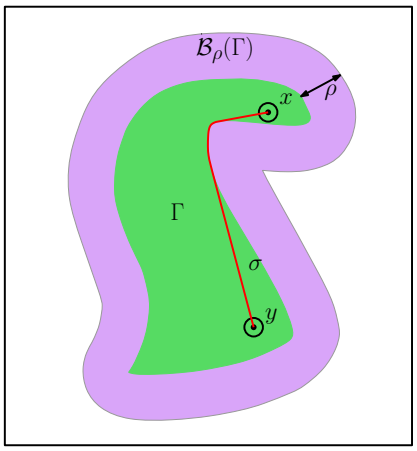


Figure 3. Visualization of proof of Theorem 7: The shortest path σ (red) connecting two points x, y within Γ (green).

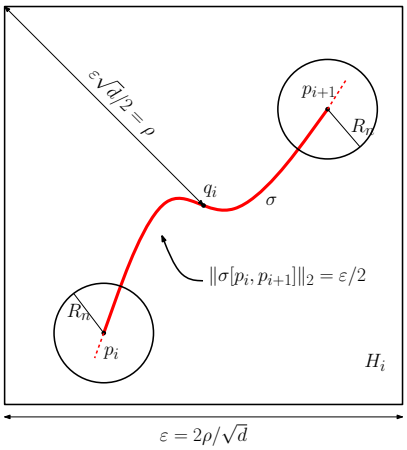


Figure 4. Visualization of proof of Theorem 7: Points p_i and p_{i+1} along path p_i are connected such that the distance between the points along σ is exactly $\varepsilon/2$. Hypercube H_i of side length ε centred at q_i (midpoint between p_i and p_{i+1}).

5.2 Bounded stretch

Given $\zeta \geqslant 1$ denote by $\mathcal{A}^{\zeta}_{\text{str}}$ the property indicating that a given geometrically-embedded graph has a *bounded stretch* of ζ, for any two vertices. Formally, let $\mathcal{G}$ be a graph defined over a vertex set $X \subset [0,1]^d$. The notation $\mathcal{G} \in \mathcal{A}^{\zeta}_{\text{str}}$ indicates that for every $x, y \in X$ it holds that $\text{dist}(\mathcal{G}, x, y) \leqslant \zeta\|x - y\|_2$. The proof of the following theorem is very similar to that of Theorem 6.

Theorem 7. *Let $\Gamma \subset [0,1]^d$ be a ρ-safe region for some constant $\rho > 0$ independent of n. Let $\mathcal{G}_n$ be a random graph that is localizable for $\mathcal{A}^{\zeta}_{\text{str}}$, for some $\zeta \geqslant 1$. Additionally, let $x, y \in \mathcal{X}_n$ be two points that lie in the same connected component of Γ. Then $\text{dist}(\mathcal{G}_n(\mathcal{B}_\rho(\Gamma)), x, y) \leqslant \zeta\|x - y\|_\Gamma + o(1)$ a.a.s., where $\|x - y\|_\Gamma$ denotes the length of the shortest path between x and y that is fully contained in Γ.*

Proof. Let σ be the shortest path connecting x and y, which is entirely contained in Γ (see Fig. 3). We define a sequence of b points $p_1, \ldots, p_b$ that are equally spaced along σ (here b plays a similar role to the number of hypercubes used in the proof of Theorem 6). Next, we show that for every p_i there is a vertex x_i of $\mathcal{X}_n$ that is sufficiently close to p_i. Moreover, we show that for every $1 \leqslant i < b$, the points x_i, x_{i+1} are contained in a hypercube H_i whose size is independent of n, and which is contained in $\mathcal{B}_\rho(\Gamma)$. This allows to exploit the fact that $\mathcal{G}_n$ is localizable for $\mathcal{A}^{\zeta}_{\text{str}}$ and show that $\mathcal{G}_n$ contains a path from x_i to x_{i+1} that is similar in length to the subpath of σ connecting p_i to p_{i+1}.

Set $R = R_n := \theta_d^{-1/d}\left(\frac{\log n}{n}\right)^{1/d}$ and $\varepsilon = 2\rho/\sqrt{d}$. Note that for sufficiently large n it follows that $R < 2\rho$. Consider the sequence of b points $P = p_1, \ldots, p_b$ in $[0,1]^d$ along σ such that (i) $p_1 = x$, (ii) $p_b = y$ and (iii) the subpath of σ between points p_i and p_{i+1} has length exactly $\varepsilon/2$ (except for, possibly, the last subpath). Note that $b = 2|\sigma|/\varepsilon$ is finite and independent of n. See Fig. 4.

For every $p_i \in P$ define $x_i = \text{argmin}_{x \in \mathcal{X}_n} \|x - p_i\|_2$, namely the closest point from $\mathcal{X}_n$ to p_i. We show that a.a.s. for every $1 \leqslant i \leqslant b$ it holds that $\|x_i - p_i\|_2 \leqslant R_n$. Similarly to the proof of Claim 2, for a given $1 \leqslant i \leqslant b$ we have that

$$\begin{aligned}\Pr[\mathcal{X}_n \cap \mathcal{B}_{R_n}(x_i) = \emptyset] &= (1 - |\mathcal{B}_{R_n}(x_i)|)^n \\ &= (1 - \theta_d R_n^d)^n \leqslant e^{-\log n} \leqslant 1/n.\end{aligned}$$

Then, we use the union bound to establish the bound

$$\Pr[\exists 1 \leqslant i \leqslant b, \mathcal{X}_n \cap \mathcal{B}_{R_n}(x_i) = \emptyset] \leqslant b/n,$$

which converges to 0 as $n \to \infty$.

Now, set q_i to be the point midway between p_i and p_{i+1} on σ. Additionally, define H_i to be the axis-aligned hypercube of side-length ε centred at q_i. Note that for sufficiently large n it holds that $x_i, x_{i+1} \in H_i$ a.a.s. Moreover, it can be shown using an argument similar to the one used in Claim 1 that $H_i \subset \mathcal{B}_\rho(\Gamma)$. This, combined with the fact that $\mathcal{G}_n$ is localizable for $\mathcal{A}^{\zeta}_{\text{str}}$, yields that the following holds a.a.s.:

$$\begin{aligned}\text{dist}(\mathcal{G}_n(\mathcal{H}_i), x_i, x_{i+1}) &\leqslant \zeta\|x_i - x_{i+1}\|_2 + o(1) \\ &\leqslant \zeta(\|p_i - p_{i+1}\|_2 + 2R_n) + o(1) \\ &= \zeta\|p_i - p_{i+1}\|_2 + o(1) \\ &= \zeta\|p_i - p_{i+1}\|_\Gamma + o(1).\end{aligned}$$

Recall that b is finite and independent of n. Thus, the following inequality, which concludes the proofs, holds

a.a.s.:

$$\begin{aligned}\mathrm{dist}(\mathcal{G}_n(\mathcal{B}_\rho(\Gamma), x, y) &\leqslant \sum_{i=1}^{b-1} \mathrm{dist}(\mathcal{G}_n(H_i), x_i, x_{i+1}) \\ &\leqslant \sum_{i=1}^{b-1} \zeta \|p_i - p_{i+1}\|_\Gamma + o(1) \\ &\leqslant \zeta \|x - y\|_\Gamma + o(1).\end{aligned}$$

□

Remark 1. The proof of Theorem 6 could be altered to use the same arguments presented for the proof of Theorem 7, i.e., follow a specific path instead of constructing a grid over the entire Γ.

6 Application to sampling-based motion planning

We now move to the setting of motion planning in which a robot operates in the configuration space $\mathcal{C} = [0,1]^d$, and whose free space is denoted by $\mathcal{F} \subseteq \mathcal{C}$. Recall that the problem consists of finding a continuous path between two configurations (points) $s, t \in \mathcal{F}$, that is fully contained in $\mathcal{F}$.

The reason why we cannot apply results on RGGs to motion planning directly is that $\mathcal{F}$ is not the full hypercube $[0,1]^d$ but rather could be a geometrically and topologically very complicated subset of this hypercube. However, the localization-tessellation approach that we have devised enables us to fairly directly adapt results from the theory of RGGs to this more involved setting, as we do in this section.

Specifically, we start by introducing the Soft-PRM, Bluetooth-PRM and Embedded-PRM algorithms, which are extensions of SRGG, RBG and REGG to the setting of motion planning. We then continue to provide proofs for probabilistic completeness and asymptotic optimality of these methods. We note that Soft-PRM and Bluetooth-PRM are very similar to a technique that was studied experimentally by McMahon et al. (2012); here we provide theoretical analysis for it. Furthermore, we describe in Section 8 an interesting connection between the algorithms Soft-PRM, Bluetooth-PRM and an implementation of the standard PRM, which employs *approximate* nearest-neighbor search.

6.1 Motion-planning algorithms

We introduce the Soft-PRM algorithm. The description of Bluetooth-PRM and Embedded-PRM immediately follow, as they are special cases of Soft-PRM. Recall that SRGG is defined for a connection radius r_n and the function $\phi_n : \mathbb{R}^+ \to [0,1]$: two vertices $x, y \in \mathcal{X}_n$ for which $\|x - y\|_2 \leqslant r_n$ are connected with an edge with probability $\phi_n(\|x - y\|_2)$.

We use the following standard procedures: `sample`(n) returns n configurations that are sampled uniformly and randomly from $\mathcal{C}$; `nearest_neighbors`(x, V, r) returns all the configurations from V that are found within a distance of r from x; `collision_free`(x, y) tests whether the straight-line segment connecting x and y is contained in $\mathcal{F}$; `random_variable`$()$ selects uniformly at random a real number in the range $[0,1]$.

The *preprocessing phase* of Soft-PRM is described in Alg. 1. In lines 1-4, n configurations are sampled (note that this slightly differs from some PRM descriptions in which the samples are assumed to be collision free) and for each sample, Soft-PRM retrieves the neighboring samples which are within a distance of at most r_n from it. For each sample point x and each candidate neighbor y it decides with probability $\phi_n(\|x - y\|_2)$ whether to attempt the connection (lines 5-6). If this is the case, the edge (x, y) is tested for being collision free (line 7), and added accordingly to E.

The (standard) PRM and Embedded-PRM are identical to Alg. 1 using the parameters r_n and $\phi_n = 1$ for PRM and $r_n = \infty$ and $\phi_n = p_n$ for Embedded-PRM. Note that in the implementation of Embedded-PRM there is no need to maintain a nearest-neighbor data structure (line 3) as every pair of vertices $x, y \in \mathcal{X}_n$ is chosen with probability p_n. Bluetooth-PRM can be obtained by replacing lines 5,6 in Alg. 1 with a suitable procedure which uniformly samples a subset of c_n neighbors from a given collection.

In the *query stage* each of the three algorithms is given two configurations s, t, which are then connected to their neighbors in the underlying roadmap by executing `nearest_neighbors` with the connection radius $r_{\text{query}} = \gamma \left(\frac{\log n}{n}\right)^{1/d}$, where $\gamma > \gamma^* = 2(2d\theta_d)^{-1/d}$. Naturally, every connection is tested for collision. Finally, the underlying graph is searched for the shortest path from s to t and the respective path in $\mathcal{F}$ is returned (if exists).

Observation 1. *Denote by $\mathcal{G}_n$ the* Soft-PRM *roadmap produced for n samples and the connection radius r_n. Then $\mathcal{G}_n = \mathcal{G}_n^{\text{soft}}(\mathcal{X}_n; r_n; \phi_n) \cap \mathcal{F}$. The same applies for the relation of the underlying roadmaps of* PRM*,*Bluetooth-PRM*,* Embedded-PRM*, and RGG, RBG, REGG, respectively.*

6.2 Probabilistic completeness

Let $(\mathcal{F}, s, t)$ be a motion-planning problem that consists of the free space $\mathcal{F} \subset [0,1]^d$, and $s, t \in \mathcal{F}$ are the start and target configurations, respectively. We provide the definition of *probabilistic completeness* and state the conditions under which the aforementioned algorithms posses this property.

Definition 7. Let $\sigma : [0,1] \to \mathcal{F}$ be a continuous path, and let $\delta > 0$. The path σ is δ-robust if $\mathcal{B}_\delta(\sigma) \subseteq \mathcal{F}$.

Algorithm 1 Soft-PRM(n, r_n, ϕ_n)

$V \leftarrow \{x_{\text{init}}\} \cup \texttt{sample}(n)$; $E \leftarrow \emptyset$; $\mathcal{G} \leftarrow (V, E)$
for all $x \in V$ **do**
  $U \leftarrow \texttt{nearest\_neighbors}(x, V, r_n)$
  for all $y \in U$ **do**
    $\xi \leftarrow \texttt{random\_variable}()$
    if $\xi \leqslant \phi_n(\|x - y\|_2)$ **then**
      if $\texttt{collision\_free}(x, y)$ **then**
        $E \leftarrow E \cup (x, y)$
return $\mathcal{G}$

Definition 8. A motion-planning problem $(\mathcal{F}, s, t)$ is *robustly feasible* if there exists a δ-robust path σ connecting s to t, for some fixed $\delta > 0$.

Definition 9. A planner ALG is probabilistically complete if for any robustly-feasible $(\mathcal{F}, s, t)$, the probability that ALG finds a solution with n samples converges to 1 as n tends to ∞.

Lemma 5. *Let* ALG *be any of the algorithms* PRM*,* Bluetooth-PRM*,* Soft-PRM*, or* Embedded-PRM*, with a selection of parameters for which the corresponding random graph $\mathcal{G}_n$ be* localizable for connectivity. *Then* ALG *is probabilistically complete.*

Proof. Suppose that $(\mathcal{F}, s, t)$ is robustly feasible. By definition, there exists a path σ connecting s to t and $\delta > 0$ for which $\mathcal{B}_\delta(\sigma) \subseteq \mathcal{F}$.

Let $\delta' := \delta/2$ and let $\Gamma := \mathcal{B}_{\delta'}(\sigma)$. Note that $s, t \in \Gamma$ and also $\mathcal{B}_{\delta'}(\Gamma) = \mathcal{B}_\delta(\sigma) \subseteq \mathcal{F} \subset [0,1]^d$, which implies that Γ is δ'-safe. By the fact that ALG is localizable for connectivity, and by Theorem 6, we have that for every $x, y \in \mathcal{X}_n \cap \Gamma$ it follows that x, y are connected in $\mathcal{G}_n(\mathcal{B}_{\delta'}(\Gamma))$ a.a.s.

It remains to show that during the query stage s and t are connected to $\mathcal{G}_n(\mathcal{B}_{\delta'}(\Gamma))$. It is not hard to verify that $\mathcal{X}_n \cap \mathcal{B}_{r_{\text{query}}}(s) \neq \emptyset, \mathcal{X}_n \cap \mathcal{B}_{r_{\text{query}}}(t) \neq \emptyset$, a.a.s., which implies connectivity. □

Theorem 8. *Recall that*

$$\gamma^* = 2(2d\theta_d)^{-1/d}, \gamma^{**} = d \cdot 2^{1+1/d},$$

and that $d \geqslant 2$. Additionally, let $\eta > 1$ be any fixed value. Then the following algorithms are probabilistically complete:

(i) PRM(ηr_n)*, where* $r_n = \gamma \left(\frac{\log n}{n}\right)^{1/d}$*, for* $\gamma > \gamma^*$*;*

(ii) Bluetooth-PRM($\eta r_n; c_n$)*, where* $r_n = \gamma \left(\frac{\log n}{n}\right)^{1/d}$*, for* $\gamma > \gamma^{**}$ *and* $c_n > \sqrt{\frac{2 \log n}{\log \log n}}$*;*

(iii) Soft-PRM($\eta r_n; \phi_n$)*, where* $r_n = \gamma \left(\frac{\log n}{n}\right)^{1/d}$*, for* $\gamma > (d+1)^{1/d}\gamma^*$ *and* $\phi_n(z) = 1 - z/r_n$*, for any* $z \in \mathbb{R}^+$*;*

(iv) Embedded-PRM(p_n)*, where* $p_n = \omega\left(\frac{\log^d n}{n}\right)$.

Item **(i)** follows from combining Theorem 1 with the localization lemma for RGGs (i.e., Lemma 1), and Lemma 6. The other items similarly follow.

Remark 2. The conditions in **(i)** are not only sufficient but also necessary, according to Theorem 1.

Remark 3. The connection radius in **(i)** is smaller by a factor of $2^{-1/d}$ than the one obtained by Janson et al. (2015), and smaller by a factor of $2^{-1/d}(d+1)^{-1/d}$ than the connection radius proposed by Karaman and Frazzoli (2011) when no obstacles are present. We also mention that, similarly to these two works, r_n can be reduced by a factor of $|\mathcal{F}|^{1/d}$, with a slight modification to Theorem 6.

6.3 Asymptotic (near-)optimality

Given a path σ denote its length by $|\sigma|$. We define the property of asymptotic near-optimality and state the conditions under which PRM and Embedded-PRM have this property.

Definition 10. Suppose that $(\mathcal{F}, s, t)$ is robustly feasible. A path σ^* connecting s to t is *robustly optimal* if it is a shortest path for which the following holds: for any $\varepsilon > 0$ there exists a δ-robust path σ such that $|\sigma| \leqslant (1+\varepsilon)|\sigma^*|$ for some fixed $\delta > 0$.

Definition 11. A sampling-based planner ALG is *asymptotically ζ-optimal*, for a given $\zeta \geqslant 1$, if for every robustly-feasible problem $(\mathcal{F}, s, t)$ it follows that $|\sigma_n| \leqslant \zeta|\sigma^*| + o(1)$ a.a.s., where σ_n denotes the solution returned by ALG with n samples. A planner that is asymptotically 1-optimal is simply called *asymptotically optimal*.

Lemma 6. *Let* ALG *be any of the algorithms* PRM*,* Bluetooth-PRM*,* Soft-PRM*, or* Embedded-PRM*, with a selection of parameters for which the corresponding random graph $\mathcal{G}_n$ be localizable for $\mathcal{A}^\zeta_{\text{str}}$, for $\zeta \geqslant 1$. Then* ALG *is asymptotically ζ-optimal.*

Proof. Suppose that $(\mathcal{F}, s, t)$ is robustly feasible. Denote by σ^* the robustly-optimal path. By definition, for every $\varepsilon > 0$ there exists $\delta > 0$ and a δ-robust path σ_ε such that $|\sigma_\varepsilon| \leqslant (1+\varepsilon)|\sigma^*|$. For now we will consider a fixed $\varepsilon > 0$ and the corresponding path σ_ε.

Similarly to the proof of Lemma 6, Let $\delta' := \delta/2$ and let $\Gamma := \mathcal{B}_{\delta'}(\sigma_\varepsilon)$. The query stage will succeed a.a.s. and s, t will be connected to some two vertices $x, y \in \mathcal{X}_n \cap \Gamma$ such

that $\|s-x\|_2, \|t-y\|_2 \leqslant R_n$. Observe that (a.a.s.)

$$\begin{aligned}\|x-y\|_\Gamma &\leqslant \|s-t\|_\Gamma + 2R_n \leqslant |\sigma_\varepsilon| + 2R_n \\ &\leqslant (1+\varepsilon)|\sigma^*| + o(1).\end{aligned}$$

Using this observation, together with Theorem 7, and with the fact that $\mathcal{G}_n$ is localizable, we deduce that ALG finds a solution σ_n, which is contained in $\mathcal{B}_{\delta'}(\Gamma)$, such that $|\sigma_n| \leqslant \zeta(1+\varepsilon)|\sigma^*| + o(1)$ a.a.s.

We will now eliminate the ε factor from the distance bound. For a given $\varepsilon > 0$ and n, denote by $\mathcal{P}(n,\varepsilon)$ the event $\text{dist}(\mathcal{G}_n(\Gamma), s, t) \leqslant \zeta(1+\varepsilon)|\sigma^*| + o(1)$. For every positive integer i define $\varepsilon_i := 1/i$. Let n_i be the minimal integer such that for every $n \geqslant n_i$ it follows that $\Pr[\mathcal{P}(n,\varepsilon_i)] \geqslant 1-\varepsilon_i$. For a given n, let $i(n)$ be such that $n_i \leqslant n < n_{i+1}$. It follows that

$$\lim_{n\to\infty} \Pr[\mathcal{P}(n,\varepsilon_{i(n)})] \geqslant \lim_{n\to\infty} 1-\varepsilon_{i(n)} = 1.$$

As $\varepsilon_{i(n)} = o(1)$ we may deduce that ALG is asymptotically ζ-optimal. □

Theorem 9. *For $d \geqslant 2$ we have the following results:*

(i) PRM(r_n) *is asymptotically ζ-optimal for $r_n = \gamma\left(\frac{\log n}{n}\right)^{1/d}$, where $\gamma > \gamma^*$, and some constant ζ;*

(ii) Embedded PRM(p_n) *is asymptotically optimal for $p_n = \omega\left(\frac{\log^d n}{n}\right)$;*

Remark 4. The conditions in **(i)** are not only sufficient but also necessary, as without them PRM will be incomplete.

7 Evaluation

In this section we present experiments demonstrating the behavior of RGGs and SRGGs in the absence and in the presence of obstacles. We then proceed to compare the Soft-PRM and PRM algorithms. For each model, and each algorithm, we use the minimal parameters that are required in order to ensure connectivity. For our experiments we used the Open Motion Planning Library (OMPL) Şucan et al. (2012) with Randomly Transformed Grids (RTG) Aiger et al. (2014) as our nearest-neighbor (NN) data structure. RTG were shown to outperform other NN libraries for several motion-planning algorithms Kleinbort et al. (2015). All experiments were run on a 2.8GHz Intel Core i7 processor with 8GB of memory. Results are averaged over 100 runs, and computed for dimensions $d = 2, 6, 9$ and 12. Additionally, when results for different dimensions behave similarly, we present only plots for $d = 2$ and $d = 12$.

Connectivity and stretch in the unit cube. We begin by reporting the number of nodes that are *not* in the largest connected component (CC) for RGG and SRGG in the absence of obstacles. Clearly, when the graph is connected, this number is zero. One can see (Fig. 5) that as the number of nodes increases, the number of nodes not in the largest CC approaches zero. Additionally, the two models exhibit very similar trends.

We continue to asses how increasing the number of nodes affects the stretch of the graphs. For each such n, we sampled $m = 50$ vertices and computed the stretch for every pair of sampled vertices. We then report on the maximal stretch obtained among all $O(m^2)$ pairs of nodes which gives a rough approximation of the average stretch of the graph. Results are depicted in Fig. 5. Observe that typically the stretch decreases as the number of nodes increases and that RGG and SRRG behave very similarly. Notice that each point along the plot is an average of 100 different runs. In addition, the RGG's used for each time-step are independent. Thus, it is probable that a graph drawn with n_1 vertices is connected while a graph drawn with $n_2 > n_1$ vertices is disconnected.

Connectivity and stretch of RGGs in general domains. The set of experiments come to demonstrate Theorems 6 and 7. Namely, that the asymptotic behavior of RGGs with respect to connectivity and stretch is maintained in the presence of obstacles. We constructed the following toy-scenario where we subdivided the d-dimensional unit hypercube by halfing it along each axis. In the center of each one of the 2^d sub-cubes, we inserted an axis-aligned hypercube as an obstacle. The size of the obstacle was chosen such that the obstacles covered 25% of the unit hypercube. See Figure 6 for a visualization in two and three dimensions.

We report on the results for RGGs (Fig. 7) and note that similar results were observed for SRGGs. Stretch was computed between the origin $(0,\ldots,0)$ and the center $(0.5,\ldots,0.5)$. Observe that for all dimensions, the graph is asymptotically connected and the stretch tends to one.

Motion-planning algorithms. Finally, we compare PRM and Soft-PRM as sampling-based planners for rigid-body motion planning on the Home scenario (Fig. 8) provided by the OMPL distribution.[‡] This six-dimensional configuration space, SE3, includes both translational and rotational degrees of freedom. Thus, it is not clear if the theoretic results presented in this paper still hold in this non-Euclidean space.

[‡]We used a robot which was scaled down to 80% the size of the robot provided by the OMPL distribution.

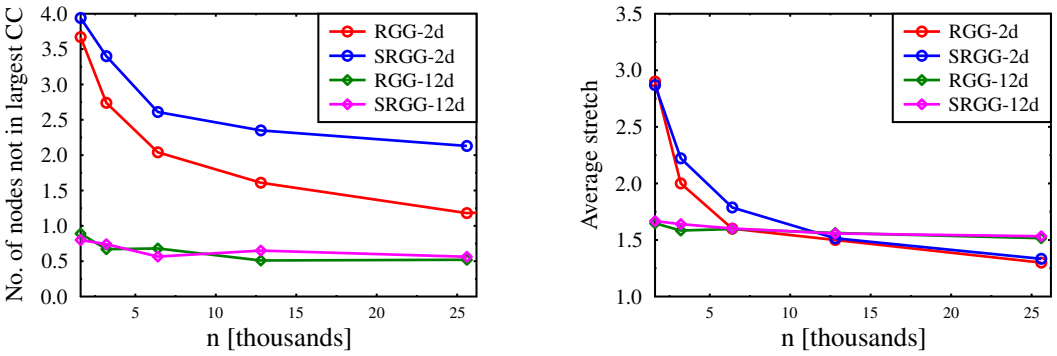


Figure 5. Connectivity (left) and stretch (right) in the absence of obstacles.

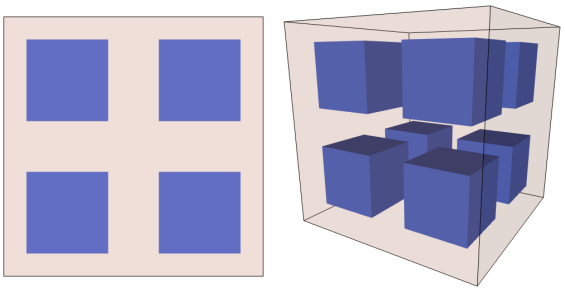


Figure 6. Visualization of toy scenario for two (left) and three (right) dimensions. Obstacles are depicted in blue.

To apply the results, a key question one has to address is how to choose the connection radius when using a non-Euclidean metric. Let x be a point sufficiently far from the boundary and let $r_n = \gamma \left(\log n/n\right)^{1/d}$ be the connection radius used. When using the Euclidean metric, the average number of neighbors of x is $\text{nbr}(n) = \left(2^{d-1}/d\right) \cdot \log n$. Thus, for each value of n, we sampled 100 random points and, for each one, computed the radius r for which the point had $\text{nbr}(n)$ neighbors. Finally, we used the average value over all such points in the experiments.

Figure 8 presents the cost of the solution produced by each algorithm as a function of the running time. Similar to the previous tests, both algorithms exhibit similar behavior, and the cost obtained approaches the optimum as the number of nodes increases.

8 Discussion

We conclude this paper by describing a connection between the Bluetooth-PRM and Soft-PRM algorithms and approximate *nearest-neighbor (NN) search* in sampling-based motion planning, which can dominate the running time of the planner in certain settings (see, Kleinbort et al. (2016)). We then proceed to describe future research directions that follow from our work.

Approximate NN search in motion planning. NN search is a key ingredient in the implementation of sampling-based planners (see, e.g., line 3 in Alg. 1). Typically, exact NN computation, where all the neighbors of a query point in a given area are reported, tends to be slow in high dimensions, due to the "curse of dimensionality" Har-Peled et al. (2012). Thus, most implementations of motion planners involve approximate NN libraries (see, e.g., Arya et al. (1998); Kleinbort et al. (2015); Muja and Lowe (2009)), which are only guaranteed to return a subset of the neighbors of a given query point (see, e.g., Aiger et al. (2014); Bentley (1975); Friedman et al. (1977); Indyk and Motwani (1998)).

However, existing proofs of probabilistic completeness and asymptotic optimality of standard planners (see, e.g., Janson et al. (2015); Karaman and Frazzoli (2011); Kavraki et al. (1996)) assume that NNs are computed exactly. Without these assumptions, the proofs no longer hold (although it may be possible to modify them to take this into account).

The analysis given in this paper bridges this gap: PRM, when implemented with approximate NN search, can be modeled as a Bluetooth-PRM or Soft-PRM. Thus, the former algorithm is probabilistically complete by using the probabilistic completeness of the latter (see Theorem 8).

Future work. The literature of RGGs is rich and encompasses many models which were not addressed in this work (see, e.g., Balister et al. (2009); Wade (2009)). Such models can be used to analyze existing planners and might lead to the development of novel planners.

In this work our focus was on Euclidean configuration spaces and the standard Euclidean distance. We mention that several works on RGGs consider different metrics in the Euclidean space (see, e.g., Appel and Russo (2002); Penrose (1999)). Such results can be imported

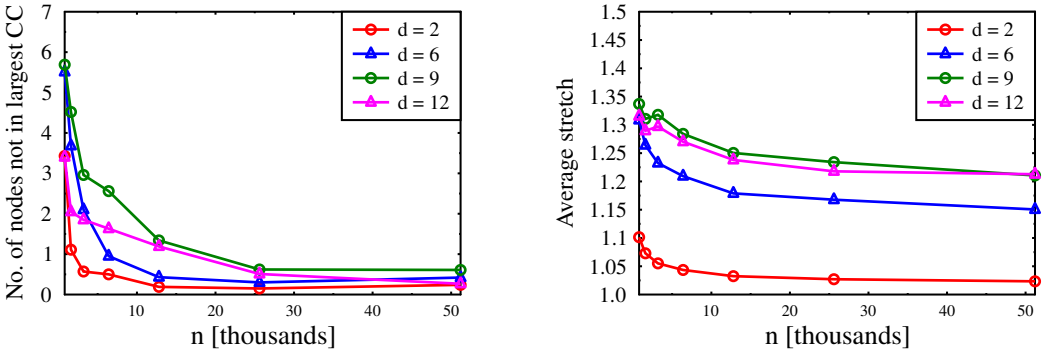
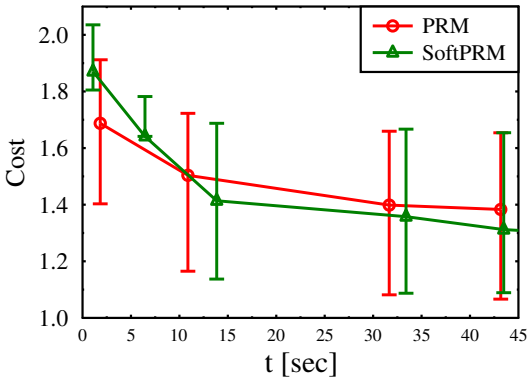


Figure 7. Connectivity (left) and stretch (right) for RGGs in the toy scenarios.



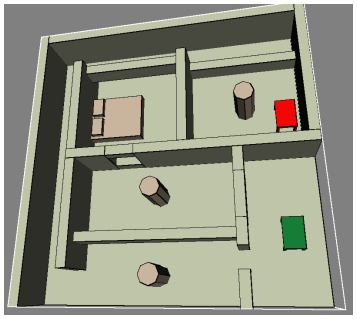


Figure 8. Average quality obtained by the PRM and Soft-PRM algorithms as a function of the running times (top) in the Home scenario (bottom). Error bars denote the 20'th and 80'th percentile. Cost is normalized such that the unit cost represents the optimum.

to the setting of motion planning using our framework, with slight modification of the proofs. We do mention our recent paper (Solovey and Halperin (2016)) where we consider sampling-based motion planning in the presence of continuous cost maps, where the goal is to obtain the minimal *bottleneck cost*, i.e., find a path that minimizes the maximal value along it. In the current work we show using the localization-tessellation framework, presented in the current paper, that connectivity of RGGs implies asymptotic optimality in the setting of bottleneck pathfinding.

Perhaps a more urgent issue involves the analysis of exiting planners in complex configuration spaces. To the best of our knowledge the behavior of standard planners such as PRM and RRT* is not well understood for non-Euclidean spaces, even for the simple case of a rigid-body robot translating and rotating in a three-dimensional workspace. We believe that several results involving RGGs in complex domains can shed light on this question. For instance, Penrose (1997) considers the case where points are sampled on a torus, whereas Penrose and Yukich (2013) study the setting of points on a manifold embedded in Euclidean space.

Funding

Dan Halperin and Kiril Solovey are supported in part by the Israel Science Foundation (grant no. 1102/11), by the German-Israeli Foundation (grant no. 1150-82.6/2011), by the Hermann Minkowski-Minerva Center for Geometry at Tel Aviv University, and by the Clore Israel Foundation. Oren Salzman is supported in part by the National Science Foundation IIS (#1409003), by Toyota Motor Engineering & Manufacturing (TEMA), and by the Office of Naval Research.

Appendix: Connectivity of SRGGs

Recall that an SRGG $\mathcal{G}_n := \mathcal{G}^{\text{soft}}(\mathcal{X}_n; r_n; \phi_n)$ represents a graph such that for every two $x, y \in \mathcal{X}_n$, there is an edge in $\mathcal{G}_n$ with probability $\phi_n(\|x-y\|_2)$ if $\|x-y\|_2 \leqslant r_n$. We refer to $\phi_n : \mathbb{R}^+ \to [0,1]^d$ as the *connection function*.

This appendix is devoted to the proof of Theorem 3 in Section 3. Namely, we show that for

$$r_n = \gamma \left(\frac{\log n}{n} \right)^{1/d}, \quad \gamma > (d+1)^{1/d} \gamma^*$$

and for every $z \in \mathbb{R}^+$ the connection function is defined to be

$$\phi_n(z) = \begin{cases} 0 & \text{if } z > r_n, \\ 1 - \frac{z}{r_n} & \text{if } z \leqslant r_n, \end{cases}$$

the SRGG $\mathcal{G}_n$ is connected a.a.s. For the simplicity of presentation we will use the notation $r := r_n, \phi := \phi_n$, which will be fixed throughout the text. Furthermore, we assume that n is large enough so that $r < 1/2$.

An ingredient of the proof is the following statement by Penrose (2016). Define

$$I_n = n \int_{x \in [0,1]^d} \exp\left(-n \int_{y \in [0,1]^d} \phi(y-x)\, \mathrm{d}y \right) \mathrm{d}x.$$

If $\lim_{n\to\infty} I_n = 0$ then $\mathcal{G}_n$ is connected a.a.s. Thus, it suffices to show that I_n converges to 0.

As is often the case with random geometric graphs (see e.g. Penrose (2003, 2016)), we will have to carefully consider boundary effects, which result from samples that lie close to the boundary of $[0,1]^d$. Thus, before computing I_n for our connection functions, we introduce some notations that will allow us to handle such cases more easily.

Recall that the d-dimensional Euclidean ball of radius r, centered at $x \in \mathbb{R}^d$ is denoted by $\mathcal{B}_r(x) = \left\{ y \in \mathbb{R}^d \middle| \|x-y\|_2 \leqslant r \right\}$. Given $x \in [0,1]^d$ define $\tilde{\mathcal{B}}_r(x) = \mathcal{B}_r(x) \cap [0,1]^d$. Additionally, set $\nu_d = \theta_d r^d$ to be the volume of $\mathcal{B}_r$.

We subdivide $[0,1]^d$ into the regions $R_0, \ldots, R_d$: for $0 \leqslant j \leqslant d$ let R_j to be the set of points for which $B_r(x)$ intersects the boundary of $[0,1]^d$ along exactly j axis-aligned hyper-planes. We bound the volume of $\tilde{\mathcal{B}}_r(x)$ according to the region R_j such that $x \in R_j$. For instance, when $d = 2$, the regions R_0, R_1, R_2 are the set of points x where the disk $B_r(x)$ does not intersect $[0,1]^d$ at all, intersects $[0,1]^d$ along exactly one line, or intersects $[0,1]^d$ along two lines, respectively (see Figure 9).

Claim 3. *For every $0 \leqslant j \leqslant d, x \in R_j$ we have that $\nu_d / 2^j \leqslant |\tilde{\mathcal{B}}_r(x)|$.*

Proof. $\tilde{\mathcal{B}}_r(x)$ attains the minimal volume when x is located on the intersection of exactly j d-dimensional hyperplanes that form the boundary of $[0,1]^d$. Every such hyperplane cuts the volume of ν_d in half. Thus, $\nu_d/2^j \leqslant |\tilde{\mathcal{B}}_r(x)|$. □

Claim 4. *For every $0 \leqslant j \leqslant d$ we have that $|R_j| \leqslant c_j r^j$, where c_j is some positive constant.*

Proof. We fix a specific j. The number of maximal connected components of R_j only depends on d. Denote this number by c_j. The volume of each such component can be expressed as $(1-2r)^{d-j} r^j$ since the number of d-dimensional hyperplanes that are in contact with the r-radius ball is exactly j. Thus, $|R_j| = c_j (1-2r)^{d-j} r^j \leqslant c_j r^j$. See Figure 9. □

Claim 5. *For every $0 \leqslant j \leqslant d, x \in R_j$ we have that*

$$\int_{y \in [0,1]^d} \phi(y-x)\, \mathrm{d}y \geqslant \frac{\nu_d}{(d+1)2^j}.$$

Proof. We use the notation $\mathcal{S}_r(x) = \left\{ y \in \mathbb{R}^d \middle| \|x-y\|_2 = r \right\}$ for the d-dimensional r-sphere.

$$\begin{aligned}
&\int_{y \in [0,1]^d} \phi(y-x)\, \mathrm{d}y \\
&\quad = \int_{y \in \tilde{\mathcal{B}}_r(x)} \phi(y-x)\, \mathrm{d}y \\
&\quad \overset{(1)}{\geqslant} \frac{1}{2^j} \int_{y \in \mathcal{B}_r(x)} \phi(y-x)\, \mathrm{d}y \\
&\quad = \frac{1}{2^j} \int_{y \in \mathcal{B}_r(x)} \frac{r - \|y-x\|_2}{r}\, \mathrm{d}y \\
&\quad = \frac{1}{2^j} \left(\int_{y \in \mathcal{B}_r(x)} \mathrm{d}y - \frac{1}{r} \int_{y \in \mathcal{B}_r(x)} \|y-x\|_2\, \mathrm{d}y \right) \\
&\quad = \frac{1}{2^j} \left(\nu_d - \frac{1}{r} \int_{y \in \mathcal{B}_r(x)} \|y-x\|_2\, \mathrm{d}y \right) \\
&\quad \overset{(2)}{=} \frac{1}{2^j} \left(\nu_d - \frac{1}{r} \int_{\rho \in [0,r]} |\mathcal{S}_\rho(x)| \rho\, \mathrm{d}\rho \right) \\
&\quad \overset{(3)}{=} \frac{1}{2^j} \left(\nu_d - \frac{1}{r} \int_{\rho \in [0,r]} \frac{d}{\rho} |\mathcal{B}_\rho(x)| \rho\, \mathrm{d}\rho \right) \\
&\quad = \frac{1}{2^j} \left(\nu_d - \frac{d}{r} \int_{\rho \in [0,r]} |\mathcal{B}_\rho(x)|\, \mathrm{d}\rho \right) \\
&\quad \overset{(4)}{=} \frac{1}{2^j} \left(\nu_d - \frac{d}{r} \cdot \frac{r}{d+1} \nu_d \right) \\
&\quad = \frac{\nu_d}{(d+1)2^j}.
\end{aligned}$$

Explanation for the non-trivial transitions: (1) Claim 3 and the fact that ϕ is non-negative. (2) Changing integrating parameters: instead of integrating over all distances $\|x-y\|_2$, we integrate over all radii $\rho \in [0,r]$. For each such radius, we multiply the volume of the sphere $\mathcal{S}_\rho(x)$ by ρ. (3) The relation that for every dimension d we have $|S_\rho(x)| = \frac{d}{\rho} |\mathcal{B}_\rho(x)|$. (4) The fact that $|\mathcal{B}_\rho(x)| = c\rho^d$ for some constant $c > 0$, which reduces the integration to a polynomial. □

We are now ready to tame the beast.

Proof of Theorem 3. Recall that it suffices to show that I_n converges to 0. Then

$$\begin{aligned}
I_n &= n \int_{x\in[0,1]^d} \exp\left(-n \int_{y\in[0,1]^d} \phi(y-x)\,\mathrm{d}y\right)\,\mathrm{d}x \\
&= n \sum_{j=0}^{d} \int_{x\in R_j} \exp\left(-n \int_{y\in[0,1]^d} \phi(y-x)\,\mathrm{d}y\right)\,\mathrm{d}x \\
&\overset{(*)}{\leqslant} n \sum_{j=0}^{d} \int_{x\in R_j} \exp\left(-n \cdot \frac{\nu_d}{(d+1)2^j}\right)\,\mathrm{d}x \\
&= n \sum_{j=0}^{d} |R_j| \exp\left(-n \cdot \frac{\nu_d}{(d+1)2^j}\right) \\
&= n \sum_{j=0}^{d} |R_j| \exp\left(-n \cdot \frac{\theta_d r_n^d}{(d+1)2^j}\right) \\
&= n \sum_{j=0}^{d} |R_j| \exp\left(-n \cdot \frac{\theta_d}{(d+1)2^j} \cdot \frac{\gamma^d \log n}{n}\right) \\
&= n \sum_{j=0}^{d} |R_j| \exp\left(-\frac{\theta_d \gamma^d}{(d+1)2^j} \cdot \log n\right),
\end{aligned}$$

where $(*)$ is due to Claim 5. Denote $a := \frac{\theta_d \gamma^d}{(d+1)2^j}$. We have that

$$\begin{aligned}
I_n &\leqslant n \sum_{j=0}^{d} |R_j| \exp\left(-a \log n\right) = n \sum_{j=0}^{d} |R_j| n^{-a} \\
&\overset{(**)}{\leqslant} n \sum_{j=0}^{d} c_j r^j n^{-a} = n \sum_{j=0}^{d} c_j \gamma^j (\log n)^{j/d} n^{-j/d} n^{-a} \\
&= \sum_{j=0}^{d} c_j \gamma^j n^{1-j/d-a} (\log n)^{j/d},
\end{aligned}$$

where $(**)$ is due to Claim 4. We now show that for $\gamma > (d+1)^{1/d}\gamma^*$ we have that $j/d + a > 1$, which implies that that I_n converges to 0.

$$\begin{aligned}
\frac{j}{d} + a &= \frac{j}{d} + \frac{\theta_d \gamma^d}{(d+1)2^j} > \frac{j}{d} + \frac{\theta_d (d+1)(\gamma^*)^d}{(d+1)2^j} \\
&= \frac{j}{d} + \frac{\theta_d 2^d}{2d\theta_d 2^j} = \frac{j + 2^{d-j-1}}{d} \geqslant 1.
\end{aligned}$$

□

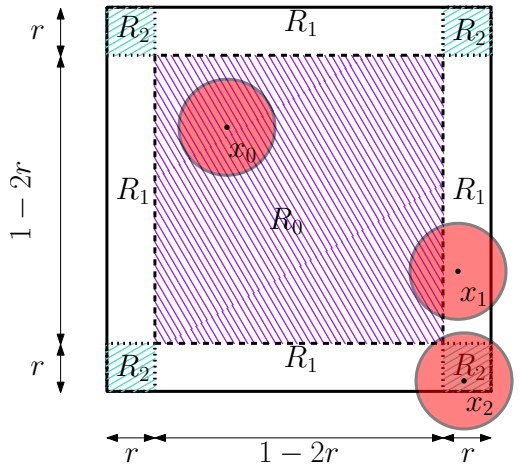


Figure 9. Visualization of regions used in proof of Theorem 3 for the planar case. The regions R_0, R_1 and R_2 are depicted by blue, white and cyan regions. For each region R_j one point x is shown together with a red disk surrounding it. Notice that the disk intersects exactly j lines supporting the unit cube $[0,1]^2$.

References

Aiger D, Kaplan H and Sharir M (2014) Reporting neighbors in high-dimensional Euclidean space. *SIAM J. Comput.* 43(4): 1363–1395.

Alterovitz R, Patil S and Derbakova A (2011) Rapidly-exploring roadmaps: Weighing exploration vs. refinement in optimal motion planning. In: *IEEE International Conference on Robotics and Automation (ICRA)*. pp. 3706–3712.

Amato NM, Bayazit OB, Dale LK, Jones C and Vallejo D (1998) OBPRM: A obstacle-based PRM for 3D workspaces. In: *International Workshop on the Algorithmic Foundations of Robotics (WAFR)*. pp. 155–168.

Appel MJ and Russo RP (2002) The connectivity of a graph on uniform points on $[0,1]^d$. *Statistics & Probability Letters* 60(4): 351–357.

Arslan O and Tsiotras P (2013) Use of relaxation methods in sampling-based algorithms for optimal motion planning. In: *IEEE International Conference on Robotics and Automation (ICRA)*. pp. 2413–2420.

Arya S, Mount DM, Netanyahu NS, Silverman R and Wu AY (1998) An optimal algorithm for approximate nearest neighbor searching in fixed dimensions. *J. ACM* 45(6): 891–923.

Bagchi A and Bansal S (2008) Nearest-neighbor graphs on random point sets and their applications to sensor networks. In: *ACM Symposium on Principles of Distributed Computing*. New York, NY, USA: ACM, pp. 434–434.

Balister PN, Bollobás B, Sarakar A and Walters M (2009) A critical constant for the k-nearest-neighbour model. *Advances in Applied Probability* 41(1): 1–12.

Bentley JL (1975) Multidimensional binary search trees used for associative searching. *Commun. ACM* 18: 509–517.

Bhatt AG and Roy R (2004) On a random directed spanning tree. *Advances in Applied Probability* 36(1): 19–42.

Bradonjic M, Elsässer R, Friedrich T, Sauerwald T and Stauffer A (2010) Efficient broadcast on random geometric graphs. In: *ACM-SIAM Symposium on Discrete Algorithms (SODA)*. pp. 1412–1421.

Broutin N, Devroye L, Fraiman N and Lugosi G (2014) Connectivity threshold of bluetooth graphs. *Random Struct. Algorithms* 44(1): 45–66.

Canny J (1988) *The complexity of robot motion planning*. MIT press.

Choset H, Lynch K, Hutchinson S, Kantor G, Burgard W, Kavraki LE and Thrun S (2005) *Principles of Robot Motion: Theory, Algorithms, and Implementations*. MIT Press.

Díaz J, Mitsche D, Perarnau G and Giménez XP (2015) On the relation between graph distance and Euclidean distance in random geometric graphs. *Advances in Applied Probability* To appear.

Dobson A and Bekris KE (2014) Sparse roadmap spanners for asymptotically near-optimal motion planning. *International Journal of Robotics Research* 33(1): 18–47.

Dubhashi DP and Panconesi A (2009) *Concentration of measure for the analysis of randomized algorithms*. Cambridge University Press.

Ellis RB, Martin JL and Yan C (2007) Random geometric graph diameter in the unit ball. *Algorithmica* 47(4): 421–438.

Friedman JH, Bentley JL and Finkel RA (1977) An algorithm for finding best matches in logarithmic expected time. *ACM Trans. Math. Softw.* 3(3): 209–226.

Friedrich T, Sauerwald T and Stauffer A (2013) Diameter and broadcast time of random geometric graphs in arbitrary dimensions. *Algorithmica* 67(1): 65–88.

Frieze A and Pegden W (2014) Traveling in randomly embedded random graphs. *arXiv:1411.6596* .

Gammell JD, Srinivasa SS and Barfoot TD (2015) Batch informed trees (BIT*): Sampling-based optimal planning via the heuristically guided search of implicit random geometric graphs. In: *IEEE International Conference on Robotics and Automation (ICRA)*. pp. 3067–3074.

Geraerts R and Overmars MH (2007) Creating high-quality paths for motion planning. *International Journal of Robotics Research* 26(8): 845–863.

Gilbert EN (1961) Random plane networks. *Journal of the Society for Industrial & Applied Mathematics* 9(4): 533–543.

Gupta P and Kumar P (1999) Critical power for asymptotic connectivity in wireless networks. In: McEneaney WM, Yin G and Zhang Q (eds.) *Stochastic Analysis, Control, Optimization and Applications*, Systems & Control: Foundations & Applications. Birkhuser Boston, pp. 547–566.

Halperin D, Kavraki L and Solovey K (2016a) Robotics. In: Goodman JE, O'Rourke J and Toth CD (eds.) *Handbook of Discrete and Computational Geometry*, 3rd edition, chapter 51. CRC Press LLC. URL `http://www.csun.edu/~ctoth/Handbook/HDCG3.html`.

Halperin D, Salzman O and Sharir M (2016b) Algorithmic motion planning. In: Goodman JE, O'Rourke J and Toth CD (eds.) *Handbook of Discrete and Computational Geometry*, 3rd edition, chapter 50. CRC Press LLC. URL `http://www.csun.edu/~ctoth/Handbook/HDCG3.html`.

Har-Peled S, Indyk P and Motwani R (2012) Approximate nearest neighbor: Towards removing the curse of dimensionality. *Theory of Computing* 8(1): 321–350.

Hsu D, Latombe JC and Motwani R (1999) Path planning in expansive configuration spaces. *International Journal of Computational Geometry and Applications* 9(4/5).

Indyk P and Motwani R (1998) Approximate nearest neighbors: Towards removing the curse of dimensionality. In: *ACM Symposium on the Theory of Computing (STOC)*. pp. 604–613.

Janson L, Schmerling E, Clark AA and Pavone M (2015) Fast marching tree: A fast marching sampling-based method for optimal motion planning in many dimensions. *International Journal of Robotics Research* 34(7): 883–921.

Karaman S and Frazzoli E (2010) Optimal kinodynamic motion planning using incremental sampling-based methods. In: *IEEE Conference on Decision and Control (CDC)*. pp. 7681–7687.

Karaman S and Frazzoli E (2011) Sampling-based algorithms for optimal motion planning. *International Journal of Robotics Research* 30(7): 846–894.

Kavraki LE, Svestka P, Latombe JC and Overmars MH (1996) Probabilistic roadmaps for path planning in high dimensional configuration spaces. *IEEE Transactions on Robotics* 12(4): 566–580.

Kleinbort M, Salzman O and Halperin D (2015) Efficient high-quality motion planning by fast all-pairs r-nearest-neighbors. In: *IEEE International Conference on Robotics and Automation (ICRA)*. pp. 2985–2990.

Kleinbort M, Salzman O and Halperin D (2016) Collision detection or nearest-neighbor search? On the computational bottleneck in sampling-based motion planning. In: *Workshop on the Algorithmic Foundations of Robotics (WAFR)*.

Kozma G, Lotker Z and Stupp G (2010) On the connectivity threshold for general uniform metric spaces. *Information Processing Letters* 110(10): 356–359.

Kuffner JJ and LaValle SM (2000) RRT-Connect: An efficient approach to single-query path planning. In: *IEEE International Conference on Robotics and Automation (ICRA)*. pp. 995–1001.

Ladd AM and Kavraki LE (2004) Fast tree-based exploration of state space for robots with dynamics. In: *Algorithmic Foundations of Robotics*. pp. 297–312.

Latombe JC (1991) *Robot Motion Planning*. Norwell, MA, USA: Kluwer Academic Publishers.

LaValle SM (2006) *Planning Algorithms*. Cambridge, U.K.: Cambridge University Press.

Li Y, Littlefield Z and Bekris KE (2014) Sparse methods for efficient asymptotically optimal kinodynamic planning. In: *Algorithmic Foundations of Robotics*. pp. 263–282.

Lien JM, Thomas SL and Amato NM (2003) A general framework for sampling on the medial axis of the free space. In: *IEEE International Conference on Robotics and Automation (ICRA)*. pp. 4439–4444.

Littlefield Z, Li Y and Bekris KE (2013) Efficient sampling-based motion planning with asymptotic near-optimality guarantees for systems with dynamics. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. pp. 1779–1785.

Luna R, Şucan IA, Moll M and Kavraki LE (2013) Anytime solution optimization for sampling-based motion planning. In: *IEEE International Conference on Robotics and Automation (ICRA)*. pp. 5053–5059.

McMahon T, Jacobs SA, Boyd B, Tapia L and Amato NM (2012) Local randomization in neighbor selection improves PRM roadmap quality. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. pp. 4441–4448.

Mitsche D and Perarnau G (2012) On the treewidth and related parameters of random geometric graphs (stacs). In: *International Symposium on Theoretical Aspects of Computer Science*. pp. 408–419.

Muja M and Lowe DG (2009) Fast approximate nearest neighbors with automatic algorithm configuration. In: *International Conference on Computer Vision Theory and Applications (VISSAPP)*. INSTICC Press, pp. 331–340.

Muthukrishnan S and Pandurangan G (2010) Thresholding random geometric graph properties motivated by ad hoc sensor networks. *Journal of Computer and System Sciences* 76(7): 686–696.

Nechushtan O, Raveh B and Halperin D (2010) Sampling-diagram automata: A tool for analyzing path quality in tree planners. In: *International Workshop on the Algorithmic Foundations of Robotics (WAFR)*. pp. 285–301.

Penrose MD (1997) The longest edge of the random minimal spanning tree. *The annals of applied probability* : 340–361.

Penrose MD (1999) On k-connectivity for a geometric random graph. *Random Structures & Algorithms* 15(2): 145–164.

Penrose MD (2003) *Random geometric graphs*. Oxford University Press.

Penrose MD (2016) Connectivity of soft random geometric graphs. *Ann. Appl. Probab.* 26(2): 986–1028.

Penrose MD and Wade A (2010a) Random directed and online networks. In: Kendall WS and Molchanov I (eds.) *New Perspectives in Stochastic Geometry*. Oxford University Press, pp. 248–264.

Penrose MD and Wade AR (2010b) Limit theorems for random spatial drainage networks. *Advances in Applied Probability* 42(3): 659–688.

Penrose MD and Yukich JE (2013) Limit theory for point processes in manifolds. *The Annals of Applied Probability* 23(6): 2161–2211.

Perez A, Platt R, Konidaris G, Kaelbling L and Lozano-Pérez T (2012) LQR-RRT*: Optimal sampling-based motion planning with automatically derived extension heuristics. In: *IEEE International Conference on Robotics and Automation (ICRA)*. pp. 2537–2542.

Raveh B, Enosh A and Halperin D (2011) A little more, a lot better: Improving path quality by a path-merging algorithm. *IEEE Transactions on Robotics* 27(2): 365–371.

Reif JH (1979) Complexity of the movers problem and generalization. In: *IEEE Symposium on Foundations of Computer Science (FOCS)*. pp. 421–427.

Salzman O and Halperin D (2014) Asymptotically near-optimal RRT for fast, high-quality, motion planning. In: *IEEE International Conference on Robotics and Automation (ICRA)*. pp. 4680–4685.

Salzman O and Halperin D (2015) Asymptotically-optimal motion planning using lower bounds on cost. In: *IEEE International Conference on Robotics and Automation (ICRA)*. pp. 4167–4172.

Salzman O, Shaharabani D, Agarwal PK and Halperin D (2014) Sparsification of motion-planning roadmaps by edge contraction. *International Journal of Robotics Research* 33(14): 1711–1725.

Schmerling E, Janson L and Pavone M (2015a) Optimal sampling-based motion planning under differential constraints: The drift case with linear affine dynamics. In: *IEEE Conference on Decision and Control (CDC)*. pp. 2574–2581.

Schmerling E, Janson L and Pavone M (2015b) Optimal sampling-based motion planning under differential constraints: The driftless case. In: *IEEE International Conference on Robotics and Automation (ICRA)*. pp. 2368–2375.

Schulte M and Thaele C (2014) Central limit theorems for the radial spanning tree. *arXiv preprint arXiv:1412.7462* .

Siméon T, Laumond J and Nissoux C (2000) Visibility-based probabilistic roadmaps for motion planning. *Advanced Robotics* 14(6): 477–493.

Solovey K and Halperin D (2015) On the hardness of unlabeled multi-robot motion planning. In: *Robotics: Science and Systems (RSS)*.

Solovey K and Halperin D (2016) Sampling-based bottleneck pathfinding with applications to Fréchet matching. In: *European Symposium on Algorithms (ESA)*. pp. 76:1–76:16.

Şucan IA and Kavraki L (2012) A sampling-based tree planner for systems with complex dynamics. *IEEE Transactions on Robotics* 28(1): 116–131.

Şucan IA, Moll M and Kavraki LE (2012) The open motion planning library. *IEEE Robotics & Automation Magazine* 19(4): 72–82.

Urmson C and Simmons RG (2003) Approaches for heuristically biasing RRT growth. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. pp. 1178–1183.

Wade AR (2009) Asymptotic theory for the multidimensional random on-line nearest-neighbour graph. *Stochastic Processes and their Applications* 119(6): 1889 – 1911.

Walters M (2011) Random geometric graphs. In: Chapman R (ed.) *Surveys in Combinatorics 2011*, chapter 8. Cambridge University Press, pp. 365–401.

Webb DJ and van den Berg JP (2013) Kinodynamic RRT*: Asymptotically optimal motion planning for robots with linear dynamics. In: *IEEE Conference on Robotics and Automation (ICRA)*. pp. 5054–5061.

Xie C, van den Berg JP, Patil S and Abbeel P (2015) Toward asymptotically optimal motion planning for kinodynamic systems using a two-point boundary value problem solver. In: *IEEE International Conference on Robotics and Automation (ICRA)*. pp. 4187–4194.

Xue F and Kumar PR (2004) The number of neighbors needed for connectivity of wireless networks. *Wireless Networks* 10(2): 169–181.

7

Sampling-Based Bottleneck Pathfinding with Applications to Fréchet Matching

Sampling-Based Bottleneck Pathfinding with Applications to Fréchet Matching*

Kiril Solovey[1] and Dan Halperin[2]

1 **Blavatnik School of Computer Science, Tel Aviv University, Tel Aviv, Israel**
kirilsol@post.tau.ac.il
2 **Blavatnik School of Computer Science, Tel Aviv University, Tel Aviv, Israel**
danha@post.tau.ac.il

Abstract

We describe a general probabilistic framework to address a variety of Fréchet-distance optimization problems. Specifically, we are interested in finding minimal *bottleneck*-paths in d-dimensional Euclidean space between given start and goal points, namely paths that minimize the maximal value over a continuous cost map. We present an efficient and simple sampling-based framework for this problem, which is inspired by, and draws ideas from, techniques for robot motion planning. We extend the framework to handle not only standard bottleneck pathfinding, but also the more demanding case, where the path needs to be monotone in all dimensions. Finally, we provide experimental results of the framework on several types of problems.

1998 ACM Subject Classification I.3.5 Computational Geometry and Object Modeling

Keywords and phrases Computational geometry, Fréchet distances, sampling-based algorithms, random geometric graphs, bottleneck pathfinding

Digital Object Identifier 10.4230/LIPIcs.ESA.2016.76

1 Introduction

This paper studies the problem of finding near-optimal paths in d-dimensional Euclidean space. Specifically, we are interested in *bottleneck* paths which minimize the maximal value the path obtains over a generally-defined continuous cost map. As an example, suppose that one wishes to plan a hiking route in a mountainous region between two camping grounds, such that the highest altitude along the path is minimized [17]. In this case, the map assigns to each two-dimensional point its altitude. A similar setting, albeit much more complex, requires to find a pathway of low energy for a given protein molecule (see, e.g., [37]).

Our main motivation for studying bottleneck optimization over cost maps is its tight relation to the Fréchet distance (or matching), which is a popular and widely studied similarity measure in computational geometry. The problem has applications to various domains such as path simplification [19], protein alignment [27], handwritten-text search [48], and signature verification [53]. The Fréchet distance, which was initially defined for curves, is often considered to be a more informative measure than the popular *Hausdorff* distance as it takes into consideration not only each curve as a whole but also the location and the ordering of points along it. Usually one is interested not only in the Fréchet distance between two given curves, but also in the parametrization which attains the optimal alignment.

* This work has been supported in part by the Israel Science Foundation (grant no. 1102/11), by the German-Israeli Foundation (grant no. 1150-82.6/2011), and by the Hermann Minkowski-Minerva Center for Geometry at Tel Aviv University. Kiril Solovey is also supported by the Clore Israel Foundation.

Since its introduction by Alt and Godau [2] in 1995, a vast number of works has been devoted to the subject, and many algorithms have been developed to tackle various settings of the problem. However, from a practical standpoint the problem is far from being solved: for many natural extensions of the Fréchet problem only prohibitively-costly algorithms are known. Furthermore, in some cases it was shown, via hardness proofs, that efforts for finding polynomial-time algorithms are doomed to fail. For some variants of the problem efficient algorithms are known to exist, however their implementation requires complex geometric machinery that relies on geometric kernels with infinite precision [30].

Contribution. We describe a generic, efficient and simple algorithmic framework for solving pathfinding optimization problems over cost maps. The framework is inspired by, and draws ideas from, sampling-based methods for robot motion planning. We provide experimental results of the framework on various scenarios. Furthermore, we theoretically analyze the framework and show that the cost of the obtained solution converges to the optimum, as the number of samples increases. We also consider the more demanding case, where paths need to be monotone in all dimensions.

Organization. In Section 2 we review related work. In Section 3 we provide a formal definition of the bottleneck pathfinding problem. In Section 4 we describe an algorithmic framework for solving this problem. In Section 5 we provide an analysis of the method. Finally, in Section 6 we report on experimental results.

2 Related work

This section is devoted to related work on Fréchet distance and robot motion planning.

2.1 Fréchet distance

The *Fréchet distance* between two curves is often described by an analogy to a person walking her dog: each of the two creatures is required to walk along a predefined path and the person wishes to know the length of the shortest *leash* which will make this walk possible. In many cases one also likes to know how to advance along the path given the short leash.

Formally, let $\sigma_1, \sigma_2 : [0,1] \to \mathbb{R}^d$ be two continuous curves. We wish to find a traversal along the two curves which minimizes the distance between the two traversal points. The traversal is defined by two continuous parametrizations $\alpha_1, \alpha_2 : [0,1] \to [0,1]$ of σ_1, σ_2 respectively, where for a given point in time $\tau \in [0,1]$, the positions of the person and her dog are specified by $\sigma_1(\alpha_1(\tau))$ and $\sigma_2(\alpha_2(\tau))$, respectively. The *Fréchet distance* between σ_1, σ_2 is defined by the expression

$$\min_{\alpha_1,\alpha_2:[0,1]\to[0,1]} \max_{\tau\in[0,1]} \|\sigma_1(\alpha_1(\tau)) - \sigma_2(\alpha_2(\tau))\|_2.$$

Alt and Godau [2] described an $O(n^2 \log n)$-time algorithm for the setting of two polygonal curves, where n is the number of vertices in each of the two curve. Buchin et al. [12] described a different method for solving this problem for the same running time. Recently, Buchin et al. [10] developed an algorithm with a slightly improved running time $O(n^2 \log^2 \log n)$. Har-Peled and Raichel [22] introduced a simpler randomized algorithm with running time of $O(n^2 \log n)$. Bringmann [6] showed that an algorithm with running time of $O(n^{2-\delta})$, for some constant $\delta > 0$, does not exist, unless a widely accepted conjecture, termed

SETH [25], is wrong. In a following work [7] this conditional lower bound was extended to $(1+\varepsilon)$-approximation algorithms of the Fréchet problem, where $\varepsilon \leq 0.399$.

The notion of Fréchet distance can be extended to k curves in various ways. One natural extension can be described figuratively as having a pack of k dogs, where each of the dogs has to walk along a predefined path, and every pair of dogs is connected with a leash. The goal now is to find a parametrization which minimizes the length of the longest leash. Dumitrescu and Rote [18] introduced a generalization of the Alt-Godau algorithm to this case, which runs in $O(kn^k \log n)$ time, i.e., exponential in the number of input curves. They also describe a 2-approximation algorithm with a much lower running time of $O(k^2 n^2 \log n)$. In the work of Har-Peled and Raichel [22] mentioned above they also consider the case of k input curves and devise an $O(n^k)$ algorithm. Notably, their technique is flexible enough to cope with different Fréchet-type goal functions over the k curves. Furthermore, their algorithm is also applicable when the k curves are replaced with k *simplicial complexes*, and the problem is to find k curves—one in each complex—which minimize the given goal function. A recent work [9], which extends the conditional lower bound mentioned earlier for the setting of multiple curves, suggests that a running time that is exponential in the number of curves is unavoidable.

The notion of Fréchet distance can be generalized to more complex objects. Buchin et al. [13] considered the problem of finding a mapping between two simple polygons, which minimizes the maximal distance between a point and its image in the other polygon. More formally, given two simple polygons $P, Q \subset \mathbb{R}^2$ the problem consists of finding a mapping $\delta : P \to Q$ which minimizes the expression $\max_{p \in P} \|p - \delta(p)\|_2$, subject to various constraints on δ. They introduced a polynomial-time algorithm for this case. In a different paper, Buchin et al. [11] showed that the decision problem is NP-hard for more complex geometric objects, e.g., pairs of polygons with holes in the plane or pairs of two-dimensional terrains. Another interesting NP-hard problem that was studied by Sherette and Wenk [41] is *curve embedding* in which one wishes to find an embedding of a curve in $\mathbb{R}^3$ to a given plane, which minimizes the Fréchet distance with the curve. In a similar setting Meulemans [34] showed that it is NP-hard to decide whether there exists a simple cycle in a plane-embedded graph that has at most a given Fréchet distance to a simple closed curve.

The Fréchet distance between curves in the presence of obstacles have earned some attention. Cook and Wenk [16] studied the *geodesic* variant, which consists of a simple polygon and two polygonal curves inside it. As in the standard formulation, the main goal is to minimize the length of the leash, but now the leash may wrap or bend around obstacles. Their algorithm has running time of $O(m + n^2 \log mn \log n)$, where m is the complexity of the polygon and n is defined as the total complexity of the two curves, as before. The more complex *homotopic* setting is a special case of the aforementioned geodesic setting, with the additional constraint that the leash must continuously deform. Chambers et al. [14] considered this problem for the specific setting of two curves in planar environment with polygonal obstacles. They developed an algorithm whose running time is $O(N^9 \log N)$, where $N = n + m$ for n and m as defined above.

2.2 Motion planning

Motion planning is a fundamental problem in robotics. In its most basic form, the problem consists of finding a collision-free path for a robot $\mathcal{R}$ in a workspace environment $\mathcal{W}$ cluttered with obstacles. Typically, the problem is approached from the configuration space $\mathcal{C}$—the set of all robot configurations. The problem can be reformulated as finding a continuous curve in $\mathcal{C}$, which entirely consists of collision-free configurations and represents a path for the robot

from a given start configuration to another, target, configuration. An important attribute of the problem is the number of *degrees of freedom* of $\mathcal{R}$, using which one can specify every configuration in $\mathcal{C}$. Typically the dimension of $\mathcal{C}$ equals the number of degrees of freedom.

For some cases of the problem, which involve a small number of degrees of freedom, efficient and exact analytical techniques exist (see, e.g., [4, 21, 40]), which are guaranteed to find a solution if one exists, or report that none exists otherwise. Recently, it was shown [46, 1, 50] that efficient and complete techniques can be developed for the *multi-robot* motion-planning problem, which entails many degrees of freedom, by making several simplifying assumptions on the separation of the start and target positions. However, it is known that the general setting of the motion-planning problem is computationally intractable (see, e.g., [38, 23, 47, 43]) with respect to the number of degrees of freedom.

Sampling-based algorithms for motion planning, which were first described about two decades ago, have revolutionized the field of robotics by providing simple yet effective tools to cope with challenging problems involving many degrees of freedom. Such algorithms (see, e.g., PRM by Kavraki et al. [29], RRT by Kuffner and LaValle [32], and EST by Hsu et al. [24]) explore the high-dimensional configuration space by random sampling and connecting nearby samples, which result in a graph data structure that can be viewed as an approximation of the *free space*—a subspace of $\mathcal{C}$, which consists entirely of collision-free configurations. While such techniques have weaker theoretical guarantees than analytical methods, many of them are *probabilistically complete*, i.e., guaranteed to find a solution if one exists, given sufficient processing time. More recently, *asymptotically optimal* sampling-based algorithms, whose solution converges to the optimum, for various criteria, have started to emerge: Karaman and Frazzoli introduced the RRT* and PRM* [28] algorithms, which are asymptotically optimal variants of RRT and PRM. Following their footsteps Arslan and Tsiotras introduced RRT# [3]. A different approach was taken by Janson and Pavone who introduced the FMT* algorithm [26], which was later refined by Salzman and Halperin [39].

3 Problem statement

In this section we describe the general problem of bottleneck pathfinding over a given cost map, to which we describe an algorithmic framework in Section 4. We conclude this section we several concrete examples of the problems that will be used for experiments in Section 6.

We start with several basic definitions. Given $x, y \in \mathbb{R}^d$, for some fixed dimension $d \geq 2$, let $\|x-y\|_2$ denote the Euclidean distance between two points. Denote by $\mathcal{B}_r(x)$ the d-dimensional Euclidean ball of radius $r > 0$ centered at $x \in \mathbb{R}^d$ and $\mathcal{B}_r(\Gamma) = \bigcup_{x\in\Gamma} \mathcal{B}_r(x)$ for any $\Gamma \subseteq \mathbb{R}^d$. We will use the terms "path" and "curve" interchangeably, to refer to a continuous curve in $\mathbb{R}^d$ parametrized over $[0, 1]$. Given a curve $\sigma : [0, 1] \to \mathbb{R}^d$ define $\mathcal{B}_r(\sigma) = \bigcup_{\tau\in[0,1]} \mathcal{B}_r(\sigma(\tau))$. Additionally, denote the image of a curve σ by $\mathrm{Im}(\sigma) = \bigcup_{\tau\in[0,1]}\{\sigma(\tau)\}$. Let $A_1, A_2, \ldots$ be random variables in some probability space and let B be an event depending on A_n. We say that B occurs *almost surely* (a.s., in short) if $\lim_{n\to\infty} \Pr[B(A_n)] = 1$.

Let $\mathcal{M} : [0, 1]^d \to \mathbb{R}$ be a *cost map* that assigns to each point in $[0, 1]^d$ a real value. For simplicity, we assume that the domain of $\mathcal{M}$ is a d-dimensional unit hypercube. Let $S, T \in [0, 1]^d$ denote the start and target points. Denote by $\Sigma(S, T)$ the collection of paths that start in S and end in T. Formally, every $\sigma \in \Sigma(S, T)$ is a continuous path $\sigma : [0, 1] \to [0, 1]^d$, where $\sigma(0) = S, \sigma(1) = T$. Given a path σ we use the notation $\mathcal{M}(\sigma) = \max_{\tau\in[0,1]} \mathcal{M}(\sigma(\tau))$ to represent its bottleneck cost.

In some applications, monotone paths are desired. For instance, in the classical problem of Fréchet matching between two curves it is often the case that backward motion along the

curves is forbidden. Here we consider monotonicity in all d coordinates of points along the path. Formally, given two points $p, p' \in \mathbb{R}^d$, where $p = (p_1, \ldots, p_d), p' = (p'_1, \ldots, p'_d)$, we use the notation $p \preceq p'$ to indicate that $p_i \leq p'_i$, for every $1 \leq i \leq d$. A path $\sigma \in \Sigma(S, T)$ is said to be *monotone* if for every $0 \leq \tau \leq \tau' \leq 1$ it holds that $\sigma(\tau) \preceq \sigma(\tau')$.

▶ **Definition 1.** Given the triplet $\langle \mathcal{M}, S, T \rangle$, the *bottleneck-pathfinding problem* (BPP, for short) consists of finding a path $\sigma \in \Sigma(S, T)$ which minimizes the expression $\max_{\tau \in [0,1]} \mathcal{M}(\sigma(\tau))$. A special case of the bottleneck pathfinding problem, termed *strong*-BPP, requires that the path will be *monotone*.

3.1 Examples

We provide three examples of BPPs, which will be used for experiments in Section 6. Each example is paired with the d-dimensional configuration space $\mathcal{C} := [0, 1]^d$, start and target points $S, T \in \mathcal{C}$, and a cost map $\mathcal{M} : [0, 1]^d \to \mathbb{R}$. The examples below are defined for two-dimensional input objects, but can generalized to higher dimensions.

Problem 1: We start with the classical *Fréchet distance among k curves* (see, e.g., [22]). Let $\sigma_1, \ldots, \sigma_k : [0, 1] \to [0, 1]^2$ be k continuous curves embedded in Euclidean plane. Here $\mathcal{C} = [0, 1]^k$ is defined as the Cartesian product of the various positions along the k curves. Namely, a point $P = (p_1, \ldots, p_k) \in \mathcal{C}$ describes the location $\sigma_i(p_i)$ along σ_i, for each $1 \leq i \leq k$. To every such P we assign the cost $\mathcal{M}(P) = \max_{1 \leq i < j \leq k} \|\sigma_i(p_i) - \sigma_j(p_j)\|_2$. We note that more complex formulations of $\mathcal{M}$ can be used, depending on the exact application. The start and target positions are defined to be $S = (\sigma_1(0), \ldots, \sigma_k(0)), T = (\sigma_1(1), \ldots, \sigma_k(2))$.

Problem 2: We introduce the problem of *Fréchet distance with visibility*, whose basis is similar to **P1** with $k = 3$. In addition to the curves, we are given a subspace $\mathcal{F} \subseteq [0, 1]^2$. The goal is to find a traversal of the curves which minimizes $\mathcal{M}$ as defined in **P1**, with the additional constraint that the traversal point along σ_1 must be "seen" by one of the traversal points of σ_2, σ_3. Formally, for every $P = (p_1, p_2, p_3) \in \mathcal{C}$ it must hold that $p_1 p_2 \subset \mathcal{F}$ *or* $p_1 p_3 \subset \mathcal{F}$ (but not necessarily both), where $p_i p_j$ is the straight-line path from p_i to p_j.

Problem 3: In *curve embedding* (see, [41, 34]), the input consists of a curve $\sigma : [0, 1] \to [0, 1]^2$, a subspace $\mathcal{F} \subseteq [0, 1]^2$ and a pair of two-dimensional points $s, t \in \mathcal{F}$. A point $P = (p_1, p_2, p_3) \in \mathcal{C} = [0, 1]^3$ describes the location $\sigma(p_1)$ along σ and the point $(p_2, p_3) \in \mathcal{F}$. The BPP is defined for the start and target points $S = (0, s), T = (1, t) \in \mathcal{C}$ and the cost map $\mathcal{M}(P) = \|\sigma(p_1) - (p_2, p_3)\|_2$.

4 Algorithmic framework

In this section we describe an algorithmic framework that will be used for solving standard and strong regimes of BPP (Definition 1). The framework can be viewed as a variant of the PRM algorithm [28], and we chose to describe it here in full detail for completeness. However, the analysis provided in Section 5 is brand new.

The framework consists of three conceptually simple steps: In the first step, we construct a random graph embedded in $[0, 1]^d$, whose vertices consist of the start and target points S, T, and of a collection of randomly sampled points; the edges connect points that are separated by a distance of at most a given connection threshold r_n. In the second step the edges of the graph are assigned with weights corresponding to their bottleneck cost over $\mathcal{M}$.

In the third and final step, the discrete graph is searched for a path connecting S to T which minimizes the bottleneck cost.

Before proceeding to a more elaborate description of the framework we provide a formal definition of the random graphs that are at the heart of the technique. Let $\mathcal{X}_n = \{X_1, \ldots, X_n\}$ be n points chosen independently and uniformly at random from the Euclidean d-dimensional cube $[0,1]^d$. The following definition corresponds to the standard and well-studied model of random geometric graphs (see, e.g., [36, 51, 5] and the literature review in [45]).

▶ **Definition 2.** The *random geometric graph* (RGG) $\mathcal{G}_n = \mathcal{G}(\mathcal{X}_n; r_n)$ is a *directed* graph with vertex set $\mathcal{X}_n$ and edge set $\{(x,y) : x \neq y,\ x, y \in \mathcal{X}_n,\ \|x-y\|_2 \leq r_n\}$.

We are ready to describe the framework, which has two parameters: n represents the number of samples generated and r_n defines the Euclidean connection radius used in the construction of the graphs. In the next section we show that for a range of values of r_n, which is a function of the number of samples n, the cost of the returned solution converges to the optimum, as n tends to infinity. The framework consists of the following steps:

Step I: We construct the RGG $\mathcal{G}_n = (\mathcal{X}_n \cup \{S,T\}; r_n)$. For the purpose of generating $\mathcal{G}_n$ a collection of n samples $\mathcal{X}_n$ is generated and a nearest-neighbor structure is employed to find for every $x \in \mathcal{X}_n \cup \{S,T\}$ the set of samples that located within a Euclidean distance of r_n from it.

Step II: We assign to each edge of the graph the bottleneck cost of the straight-line path connecting its endpoints under $\mathcal{M}$. In particular, for the standard BPP, for every edge (x,y) the cost $\max_{\tau\in[0,1]} \mathcal{M}(x+\tau(y-x))$ is assigned. The same applies for strong-BPP, unless $x \not\preceq y$, in which case the value $+\infty$ is assigned.

Step III: For the final step we find a path over $\mathcal{G}_n$ from S to T which minimizes the bottleneck cost. Several efficient algorithms solving this problem exist (see, e.g., [52, 15]).

5 Theoretical foundations

We study the behavior of the framework for the standard and the strong case of BPP (Definition 1). Recall the framework uses the two parameters n and r_n, which specify the number of samples and the connection radius.We establish a range of connection radii, r_n, for which the cost of the returned solution is guaranteed to converge to a relaxed notion of the optimum.

The analysis below does not restrict itself to a specific type of cost maps $\mathcal{M}$, e.g., continuous or smooth. Thus, due to the stochastic nature of the framework, and the general definition of $\mathcal{M}$, we cannot guarantee that the returned solution will tend to the absolute optimum. As an example consider the cost map $\mathcal{M}$ such that for a given $x = (x_1, x_2)$, $\mathcal{M}(x) = 0$ if $x_1 = x_2$, and $\mathcal{M}(x) = 1$ otherwise. For the start and target points $S = (0.1, 0.1), T = (0.9, 0.9)$ the optimal solution is a subset of the diagonal. Obviously, the probability of having a single point of $\mathcal{X}_n$, let alone a whole path in $\mathcal{G}_n$, that lie on the diagonal is equal to 0.

We can however guarantee convergence to a *robustly-optimal* path, which is defined below. Informally, such paths have "well-behaved" neighborhoods, in terms of the value of $\mathcal{M}$. We provide below a formal definition of this notion for the bottleneck cost function. Recall that given a path σ the notation $\mathcal{M}(\sigma)$ represents its bottleneck cost.

▶ **Definition 3.** Given the triplet $\langle \mathcal{M}, S, T \rangle$, a path $\sigma \in \Sigma(S,T)$ is called *robust* if for every $\varepsilon > 0$ there exists $\delta > 0$ such that $\mathcal{M}(\sigma') \leq (1+\varepsilon)\mathcal{M}(\sigma)$, for any $\sigma' \in \Sigma(S,T)$ such that $\mathrm{Im}(\sigma') \subset \mathcal{B}_\delta(\sigma)$. A path that attains the infimum cost, over all robust paths, is termed *robustly optimal*.

5.1 (Standard) Bottleneck cost

For a given triplet $\langle \mathcal{M}, S, T \rangle$ representing an instance of BPP, denote by σ^* a robustly-optimal solution. Note that we do not require here that σ^* or the returned solution will be monotone. We obtain the following result. All logarithms stated henceforth are to base e.

▶ **Theorem 4.** *Let $\mathcal{G}_n = \mathcal{G}(\mathcal{X}_n \cup \{S,T\}; r_n)$ be an RGG with*

$$r_n = \gamma \left(\frac{\log n}{n} \right)^{1/d}, \quad \gamma > 2(2d\theta_d)^{-1/d},$$

where θ_d denotes the Lebesgue measure of a unit ball in $\mathbb{R}^d$. Then $\mathcal{G}_n$ contains a path $\sigma_n \in \Sigma(S,T)$ such that $\mathcal{M}(\sigma_n) = (1+o(1))\mathcal{M}(\sigma^*)$, a.s.

We mention that this connection radius is also essential for connectivity of RGGs, i.e., a smaller radius results in a graph that is disconnected with high probability (see, e.g.,[8]). This fact is instrumental to our proof. We also mention that a result similar to Theorem 4 can be obtained through a different proof technique [28], albeit with a larger value of the constant γ.

For simplicity, we assume for the purpose of the proof that exists a finite constant $\delta' > 0$ such that $\mathcal{B}_{\delta'}(\sigma^*) \subset [0,1]^d$, namely the robustly-optimal solution is at least δ' away from the boundary of the domain $[0,1]^d$. This constraint can be easily relaxed by transforming $\langle \mathcal{M}, S, T \rangle$ into an equivalent instance $\langle \mathcal{M}', S', T' \rangle$ where this condition is met. In particular the original input can be embedded to a cube of side length $1-\varepsilon$ for some constant $\varepsilon > 0$, which is centered in the middle of $[0,1]^d$. The cost along the boundaries of the smaller cube should be extended to the remaining parts of the $[0,1]^d$ cube.

Given an RGG $\mathcal{G}_n$ and a subset $\Gamma \subset [0,1]^d$ denote by $\mathcal{G}_n(\Gamma)$ the graph obtained from the intersection of $\mathcal{G}_n$ and Γ: it consists of the vertices of $\mathcal{G}_n$ that are contained in Γ and the subset of edges of $\mathcal{G}_n$ that are fully contained in Γ. Each edge is considered as a straight-line segment connecting its two end points.

A main ingredient in the proof of Theorem 4 is the following Lemma. We employ the *localization-tessellation* framework [45], which was developed by the authors. The framework allows to extend properties of RGGs to domains with complex geometry and topology.

▶ **Lemma 5.** *Let $\mathcal{G}_n$ be the RGG defined in Theorem 4. Additionally let $\Gamma \subset [0,1]^d$ be a fixed subset, where $S, T \in \Gamma$, and let $\rho > 0$ be some fixed constant, such that $\mathcal{B}_\rho(\Gamma) \subset [0,1]^d$. Then S, T are connected in $\mathcal{G}_n(\mathcal{B}_\rho(\Gamma))$ a.s.*

Proof. We rely on the well-known result that $\mathcal{G}_n$ is connected a.s. in the domain $[0,1]^d$ for the given connection radius r_n (see, e.g., [45, Theorem 1]). We then use Lemma 1 and Theorem 6 in [45] which state that if $\mathcal{G}_n$ is connected a.s., and is *localizable* (see, Definition 6 therein), then S, T are connected a.s. over $\mathcal{G}_n(\mathcal{B}_\rho(\Gamma))$. ◀

Proof of Theorem 4. We first show that for any $\varepsilon > 0$ it follows that $\mathcal{M}(\sigma_n) \leq (1+\varepsilon)\mathcal{M}(\sigma^*)$ a.s. Fix some $\varepsilon > 0$. Due to the fact that σ^* is robustly optimal, there exists $\delta_\varepsilon > 0$ independent of n such that for every $\sigma \in \Sigma(S,T)$ such that $\mathrm{Im}(\sigma) \subset \mathcal{B}_{\delta_\varepsilon}(\sigma^*)$ we have

that $\mathcal{M} \leq (1+\varepsilon)\mathcal{M}(\sigma^*)$ a.s. Additionally, recall that there exists some $\delta' > 0$ such $\mathcal{B}_{\delta'}(\sigma^*) \subset [0,1]^d$.

Set $\delta = \min\{\delta_\varepsilon, \delta'\}$ and define the sets $\Gamma_{\delta/2} = \mathcal{B}_{\delta/2}(\sigma^*), \Gamma_\delta = \mathcal{B}_\delta(\sigma^*)$ and notice that $S, T \in \Gamma_{\delta/2}$. By Lemma 5 we have that S, T are connected in $\mathcal{G}_n(\Gamma_\delta)$. Moreover, a path connecting S, T in $\mathcal{G}_n(\Gamma_\delta)$ must a have a bottleneck cost of at most $(1+\varepsilon)\mathcal{M}(\sigma^*)$.

We have shown that for any fixed $\varepsilon > 0$, $\mathcal{M}(\sigma_n) \leq (1+\varepsilon)\mathcal{M}(\sigma^*)$ a.s. By defining the sequence $\varepsilon_i = 1/i$ one can extend the previous result and show that $\mathcal{M}(\sigma_n) \leq (1+o(1))\mathcal{M}(\sigma^*)$. This part is technical and its details are omitted (see a similar proof in [44, Theorem 6]). This concludes the proof. ◀

5.2 Strong bottleneck cost

We now focus on the strong case of the problem, where the solution is restricted to paths that are monotone in each of the d coordinates. Denote by $\vec{\sigma}^*$ the robustly-optimal monotone solution for a given instance $\langle \mathcal{M}, S, T \rangle$.

▶ **Theorem 6.** *Let $\mathcal{G}_n = \mathcal{G}(\mathcal{X}_n \cup \{S,T\}; r_n)$ be an RGG with $r_n = \omega(1)\left(\frac{\log n}{n}\right)^{1/d}$. Then $\mathcal{G}_n$ contains a monotone path $\vec{\sigma}_n \in \Sigma(S,T)$ such that $\mathcal{M}(\vec{\sigma}_n) = (1+o(1))\mathcal{M}(\vec{\sigma}^*)$, a.s.*

Let $x, x' \in [0,1]^d$ be two points such that $x \preceq x'$. For a given $\delta > 0$ the notation $x \preceq_\delta x'$ indicates that $\delta = \min\{x'_i - x_i\}_{i=1}^d$, where $x = (x_1, \ldots, x_d), x' = (x'_1, \ldots, x'_d)$. Given two points $x, x' \in [0,1]^d$, such that $x \preceq x'$, denote by $\mathcal{H}(x, x')$ the d-dimensional box $[x_1, x'_1] \times \ldots \times [x_d, x'_d]$. In addition to the assumption that the robustly-optimal solution $\vec{\sigma}^*$ is separated from the boundary of $[0,1]^d$ that we have taken in the previous analysis, we also assume that there exists a constant $0 < \delta'' \leq 1$ such that $S \preceq_{\delta''} T$.

In preparation for the main proof we prove the following lemma.

▶ **Lemma 7.** *Choose any[1] $f_n \in \omega(1)$ and set $r_n = \omega(1)\left(\frac{\log n}{n}\right)^{1/d}$. Let $q, q' \in [0,1]^d$ be two points such that $q \preceq_\delta q'$, where δ is independent of n. Then a.s. there exist $X, X' \in \mathcal{X}_n$ with the following properties: (i) $\|X - q\|_2 \leq r_n/2, \|X' - q'\| \leq r_n/2$; (ii) $q \preceq X, X' \preceq q'$; (iii) X, X' are connected in $\mathcal{G}_n$ with a monotone path.*

Proof. We apply a tessellation argument similar to the one used to show that the standard (and undirected) RGG is connected (see, e.g., [51, Section 2.4]). Set $\ell = \left\lceil \frac{2\|q'-q\|_2}{r_n} \right\rceil$ and observe that $\ell \leq 2\sqrt{d}/r_n$. Define the normalized vector $\vec{v} = \frac{q'-q}{\|q'-q\|_2}$ and let $H_1, \ldots, H_\ell$ be a sequence of ℓ hyperboxes, where

$$H_j = \mathcal{H}\left(q + (j-1)\cdot\frac{r_n}{2}\cdot\vec{v}, q + j\cdot\frac{r_n}{2}\cdot\vec{v}\right),$$

for every $1 \leq j \leq \ell$ (see Figure 1). Observe that for every $1 \leq j < \ell$ and every $X_j \in H_j, X_{j+1} \in H_{j+1}$, we have

$$X_j \preceq X_{j+1}, \|X_{j+1} - X_j\|_2 \leq r_n. \tag{1}$$

We show that for every $1 \leq j \leq \ell$ it follows that $\mathcal{X}_n \cap H_j \neq \emptyset$, a.s. We start by bounding the volume of H_j. Denote by $c_1, \ldots, c_d$ the side lengths of H_j, and denote by $\delta_1, \ldots, \delta_d$ the

[1] For instance, f_n can be either one of the following functions: $\log n, \log^* n$, or the inverse Ackerman function $\alpha(n)$.

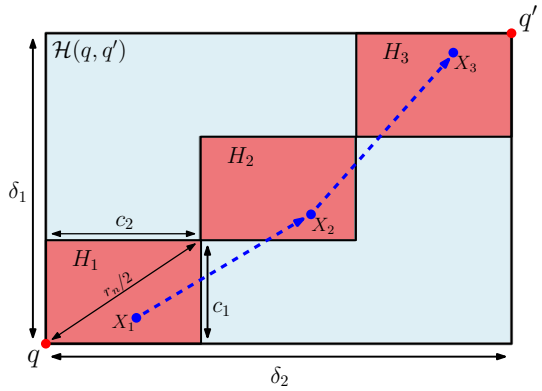


Figure 1 Visualization of the proof of Lemma 7 for $d = 2$. The blue rectangle represents $\mathcal{H}(q, q')$ and the three red rectangles represent $H_1, \ldots, H_\ell$ for $\ell = 3$ (the small value of ℓ was selected for the clarity of visualization and in reality $r_n \ll \delta_1$). The length of the largest diagonal in each of the small rectangles is $r_n/2$, which implies that a distance between $X_j \in H_j, X_{j+1} \in H_{j+1}$ is at most r_n. The blue dashed arrows represent the directed graph edges $(X_1, X_2), (X_2, X_3)$ which correspond to a monotone path connecting X_1 to X_3.

side lengths of $\mathcal{H}(q, q')$. Note that δ_i is independent of n and $c_i = \delta_i/\ell$. Consequently, we can represent $c_i = \alpha_i r_n$, where $\alpha_i > 0$ is constant, for every $1 \leq i \leq d$. Thus, $|H_j| = c r_n^d$ for some constant $c > 0$. Now,

$$\Pr\left[\mathcal{X}_n \cap H_j = \emptyset\right] = (1 - |H_j|)^n \leq \exp\left\{-n|H_j|\right\} = \exp\left\{-\omega(1) \cdot c \log n\right\} \leq n^{-1}.$$

In the last transition we used the fact that the function $f_n \in \omega(1)$ can "absorb" any constant c. We are ready to show that every H_i contains a point from $\mathcal{X}_n$ a.s.:

$$\begin{aligned}\Pr\left[\exists H_j : \mathcal{X}_n \cap H_j = \emptyset\right] &\leq \sum_{j=1}^{\ell} \Pr\left[\mathcal{X}_n \cap H_i = \emptyset\right] \\ &\leq \ell \cdot n^{-1} \leq \frac{2\sqrt{d}}{r_n} \cdot n^{-1} \\ &= \frac{2\sqrt{d}}{\omega(1) \cdot n^{1-1/d} \log^{1/d} n}.\end{aligned}$$

Thus, a.s. there exists for every $1 \leq j \leq \ell$ a point $x_j \in H_j$. Observe that $X := X_1, X' := X_\ell$ satisfy (i),(ii). Condition (iii) follows from Equation 1. ◀

Proof of Theorem 6. Similarly to the proof of Theorem 4, we fix $\varepsilon > 0$ and select $\delta \leq \min\{\delta', \delta''\}$ such that $\mathcal{M}(\vec{\sigma}) \leq (1+\varepsilon)\mathcal{M}(\vec{\sigma}^*)$ for every $\vec{\sigma} \in \vec{\Sigma}(S, T)$ with $\mathrm{Im}(\vec{\sigma}) \subset \mathcal{B}_\delta(\vec{\sigma}^*) \subset [0,1]^d$.

The crux of this proof is that there exists a sequence of k points $q_1, \ldots, q_k \in \mathrm{Im}(\vec{\sigma}^*)$, where $S = q_1, T = q_k$, such that $q_j \prec_{\delta/2} q_{j+1}$ for every $1 \leq j < k$ (see Figure 2). Moreover, due to fact that $\vec{\sigma}^*$ is monotone we can determine that such k is finite and independent of n. Thus, by Lemma 7, for every $1 \leq j < k$ there exist $X_j, X'_j \in \mathcal{X}_n$ which satisfy the following conditions a.s.: (i) $\|X_j - q_j\|_2 \leq r_n/2, \|X'_j - q_{j+1}\|_2 \leq r_n/2$; (ii) $q_j \preceq X_j, X'_j \preceq q_{j+1}$; (iii) X_j, X'_j are connected in $\mathcal{G}_n$. By conditions (i),(ii), for every $1 \leq j < k$ the graph $\mathcal{G}_n$ contains the edge (X'_j, X_{j+1}). Combined with condition (iii) this implies that S is connected to T in $\mathcal{G}_n$ a.s.

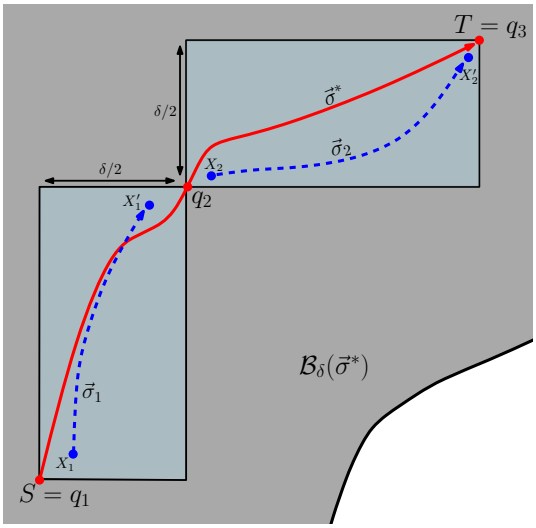


Figure 2 Visualization of the proof of Theorem 6 for $d = 2$ and $k = 3$. The red curve represents $\vec{\sigma}^*$, on which lie the points q_1, q_2, q_3 such that $q_1 \prec_{\delta/2} q_2 \prec_{\delta/2} q_3$. The dashed blue curves represent $\vec{\sigma}_1, \vec{\sigma}_2$. The gray area represents $\mathcal{B}_\delta(\vec{\sigma}^*)$.

It remains to show that the path constructed above has a cost of at most $(1+\varepsilon)\mathcal{M}(\vec{\sigma}^*)$. For every $1 \leq j \leq k$ denote by $\vec{\sigma}_j$ the path induced by Lemma 7 from X_j to X'_j, i.e., $\vec{\sigma}_j(0) = X_j, \vec{\sigma}_j(1) = X'_j$ and $\mathrm{Im}(\vec{\sigma}_j) \subset H_i$. Additionally, for every $1 \leq j < k$ denote by $\vec{\sigma}'_j$ the straight-line segment (sub-path) from X'_j to X_{j+1}. Now, define $\vec{\sigma}$ to be a concatenation of $\vec{\sigma}_1, \vec{\sigma}'_1, \ldots, \vec{\sigma}_{k-1}, \vec{\sigma}'_{k-1}, \vec{\sigma}_k$. We showed in the previous paragraph that such a path exists in $\mathcal{G}_n$ a.s. Observe that for every $1 \leq j \leq k$ it holds that $\vec{\sigma}_j \subset \mathcal{H}(q_j, q_{j+1})$, where $\mathcal{H}(q_j, q_{j+1}) \subset \mathcal{B}_{\delta/2}(\vec{\sigma}^*)$. This implies that $\mathcal{M}(\vec{\sigma}_i) \leq (1+\varepsilon)\mathcal{M}(\vec{\sigma}^*)$. Additionally, recall that for every $1 \leq j < k$ it holds that $\|X'_j - q_{j+1}\|_2 \leq r_n/2, \|X_{j+1} - q_{j+1}\|_2 \leq r_n/2$, which implies that $\mathrm{Im}(\vec{\sigma}'_j) \subset \mathcal{B}_{r_n}(q_{j+1}) \subset \mathcal{B}_\delta(\vec{\sigma}^*)$, and consequently $\mathcal{M}(\sigma'_j) \leq (1+\varepsilon)\mathcal{M}(\vec{\sigma}^*)$. Finally, $\mathcal{M}(\vec{\sigma}_n) \leq \mathcal{M}(\vec{\sigma}) \leq (1+\varepsilon)\mathcal{M}(\vec{\sigma}^*)$. This concludes the proof. ◀

6 Experimental results

In this section we validate the theoretical results that were described in the previous section. We observe that the framework can cope with complex scenarios involving two or three degrees of freedom ($d \in \{2, 3\}$), and converges quickly to the optimum.

Before proceeding to the results we provide details regrading the implementation. We implemented the framework in C++, and tested it on scenarios involving two-dimensional objects. Nearest-neighbor search, which is used for the construction of RGGs, was implemented using FLANN [35]. We note that other nearest-neighbor search data structures that are tailored for the implementation of RGGs exist (see, e.g., [31]). Geometric objects, such as points, curves, and polygons were represented with CGAL [49]. For the representation of graphs and related algorithms we used BOOST [42]. Experiments were conducted on a PC with Intel i7-2600 3.4GHz processor with 8GB of memory, running a 64-bit Windows 7 OS.

We proceed to describe the implementation involving the computation of non-trivial cost maps. For curve embedding we used PQP [20] for collision detection, i.e., determining whether a given point lies in the forbidden region $[0,1]^2 \setminus \mathcal{F}$. Finally, the cost of an edge

with respect to a given cost map was approximated by dense sampling along the edge, as is customary in motion planning (see, e.g., [33]).

The majority of running time (over %90) in the experiments below is devoted to the computation of $\mathcal{M}$ for given point samples or edges. Thus, we report only the overall running time in the following experiments. We mention that we also implemented a simple grid-based method for the purpose of comparison with the framework. However, it performed poorly in easy scenarios and did not terminate in hard cases. Thus, we chose to omit theses results here.

Unless stated otherwise, we use in the experiments the connection radius which is described in Theorem 4, and denote it by r_n^*. This applies both to the standard and strong regimes of the problem. A discussion regarding the connection radius in the strong regime appears below in Section 6.3.

6.1 Various scenarios

In this set of experiments we demonstrate the flexibility of the framework and test it on the three different scenarios. We emphasize that we employ a shared code framework to solve these three problems and the ones described later. The only difference in the implementation lies in the type of cost function used. The following problems are solved using a planner for the *strong* case of BPP.

Figure 3 (left) depicts an instance of **P1** (see Section 3.1), which consists of two geometrically-identical curves (red and blue). The curves are bounded in $[0,1]^2$ and the red curve is translated by $(0.05, 0.05)$ from the blue curve. The optimal solution has a cost of 0.07, in which the curves are traversed identically. Our program was able to produce a solution of cost 0.126 in 27 seconds and $n = 100{,}000$ samples. Results reported throughout this section are the averaged over 10 trials.

Figure 4 (left) depicts an instance of **P2**. The goal is to find a traversal of the three curves such that the traversal point along the purple curve is visible from either the blue or red curve, while of course minimizing the lengths of the leashes between the three curves. Note that the view can be obstructed by the gray rectangular obstacles. A trivial, albeit poor, solution is to move the point along the purple curve from start to end, while the traversal point of, say, the red curve stays put in the start position. A much better solution, which maintains short leashes, is described as follows: we move along the purple curve until reaching the first resting point, indicated by the leftmost black disc. Then we move along the red curve until we reach to the position directly below the black circle. Only then we move along the blue curve from start until reaching the point directly below the first black disc. We use a similar parametrization with respect to the second "pit stop", and so on. Such a solution was obtained by our program in 11 seconds using $n = 20{,}000$ samples.

Figure 4 (right) depicts an instance of **P3**. The input consists of a curve (depicted in red), and polygonal obstacles (depicted in gray). The solution obtained by our program after 600 seconds with $n = 100{,}000$, is drawn in blue.

6.2 Increasing difficulty

Here we focus on **P1** for two curves in the standard regime. We study how the difficulty of the problem affects the running time and the rate of convergence of the returned cost. We start with a base scenario, depicted in Figure 3 (right), and gradually increase its difficulty. In the depicted scenario the bottom (blue) curve consists of five circular loops of radius 0.15, where the entrance and exit point to each circle is indicated by a bullet. The top curve

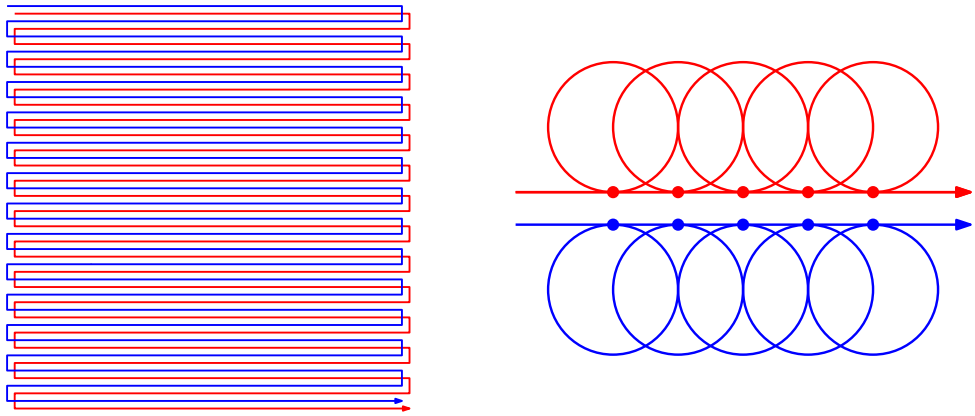


Figure 3 Scenarios involving two curve.

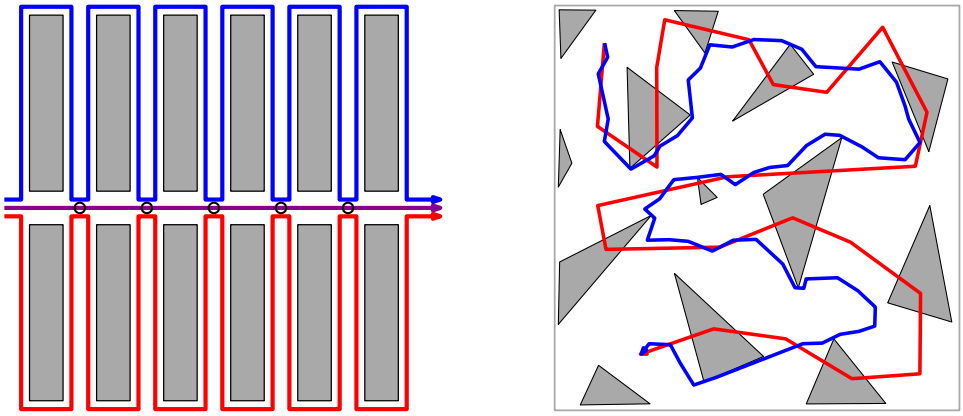


Figure 4 Scenarios involving curves and obstacles.

is similarly defined, and the two curves are separated by a vertical distance of 0.04. The optimal matching of cost 0.34 is obtained in the following manner: when a given circle of the red curve is traversed, the position along the blue curve is fixed to the entrance point of the circle directly below the traversed circle, and vice versa. In a similar fashion we construct scenarios with 10,20,40 and 80 loops in each curve.

In Figure 5 we report for each of the scenarios the cost of the obtained solution as a function of the number of samples n. We set $n = 2^i$ for the integer value i between 12 and 18. For $i = 12$ and $i = 18$ the running times were roughly 2 and 66 seconds, respectively. In between, the values were linearly proportional to the number of samples (results omitted). Observe that as the difficulty of the problem increases the convergence rate of the cost slightly decreases, but overall a value near the optimum is reached fairly quickly.

6.3 Connection radius in the strong regime

Here we consider the *strong* regime and study the behavior of the framework for varying connection radii. For this purpose, we use the two-curves scenario with 20 loops that was described in Section 6.2. We set the connection radius to $r_n := g_n \cdot r_n^*$, where r_n^* is the radius of the standard regime (see Theorem 4). We set $g_n \in \{1, 1.1, \log\log n + 1, \sqrt{\log n}\}$. Results are depicted in Figure 6.

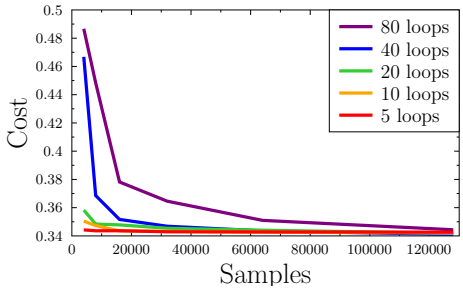


Figure 5 Results for scenarios of increasing difficulty, as described in Section 6.2.

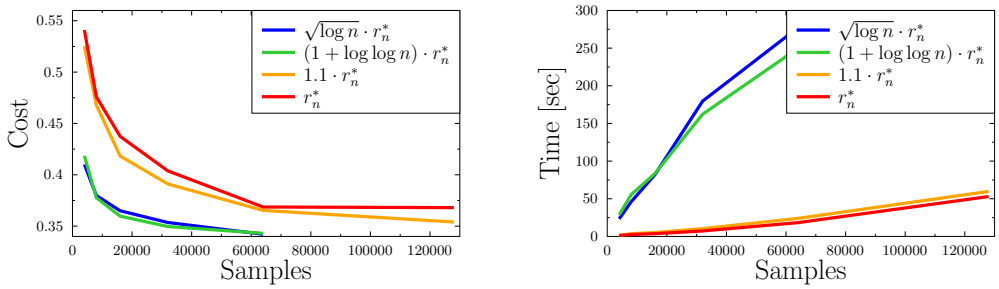


Figure 6 Results for varying connection radii in the strong regime, as described in Section 6.3.

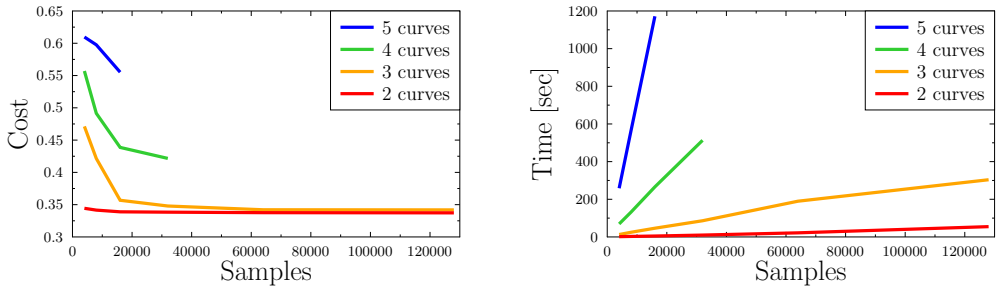


Figure 7 Figures for the first set of experiments, as described in Section 6.4.

Not surprisingly, larger values of r_n lead to quicker convergence, in terms of the number of samples required, to the optimum. However, this comes at the price of a denser RGG, which results in poor running times. Note that the program terminated due to lack of space for the two largest functions of g_n for $n = 128{,}000$. Interestingly, the connection radius r_n^* of the standard regime seems to converge to the optimum, albeit slowly. This leads to the question whether such a function also results in connectivity in the strong regime. Note that our proof of the convergence in the strong regime requires a larger value of r_n (see Theorem 6).

6.4 Increasing dimensionality

We test how the dimension of the configuration space d affects the performance. For this purpose we study the behavior of the framework on *weak* k-curve Fréchet distance with k ranging from 2 to 5. For $k = 2$ we use the scenario described in Section 6.2 with 10 loops. For $k = 3$ we add another copy of the blue curve, for $k = 4$ an additional copy of the red

curve, and another blue curve for $k = 5$. We report running time and cost in Figure 7 for various values of n, as described earlier.

Note that that for $k = 4$ the program ran out of memory for $n = 64{,}000$, and for $k = 5$ around $n = 32{,}000$. This phenomena occurs since the connection radius obtained in Theorem 4 grows exponentially in d. In particular, for $r_n = \gamma \left(\frac{\log n}{n}\right)^{1/d}$, where $\gamma = 2(2d\theta_d)^{-1/d}$, each sample has in expectancy $\Theta(2^d \log n)$ neighbors in the obtained RGG.

References

1 Aviv Adler, Mark de Berg, Dan Halperin, and Kiril Solovey. Efficient multi-robot motion planning for unlabeled discs in simple polygons. *IEEE Trans. Automation Science and Engineering*, 12(4):1309–1317, 2015.

2 Helmut Alt and Michael Godau. Computing the Fréchet distance between two polygonal curves. *Int. J. Comput. Geometry Appl.*, 5:75–91, 1995.

3 Oktay Arslan and Panagiotis Tsiotras. Use of relaxation methods in sampling-based algorithms for optimal motion planning. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 2421–2428. IEEE, 2013.

4 Francis Avnaim, Jean-Daniel Boissonnat, and Bernard Faverjon. A practical exact motion planning algorithm for polygonal objects amidst polygonal obstacles. In *Robotics and Automation, 1988. Proceedings., 1988 IEEE International Conference on*, pages 1656–1661. IEEE, 1988.

5 Paul Balister, Amites Sarkar, and Béla Bollobás. Percolation, connectivity, coverage and colouring of random geometric graphs. In Béla Bollobás, Robert Kozma, and Dezso Miklós, editors, *Handbook of Large-Scale Random Networks*, pages 117–142. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

6 Karl Bringmann. Why walking the dog takes time: Fréchet distance has no strongly subquadratic algorithms unless SETH fails. In *Foundations of Computer Science*, pages 661–670, 2014.

7 Karl Bringmann and Wolfgang Mulzer. Approximability of the discrete Fréchet distance. *Journal of Computational Geometry*, 7(2):46–76, 2016.

8 Nicolas Broutin, Luc Devroye, Nicolas Fraiman, and Gábor Lugosi. Connectivity threshold of bluetooth graphs. *Random Struct. Algorithms*, 44(1):45–66, 2014.

9 Kevin Buchin, Maike Buchin, Maximilian Konzack, Wolfgang Mulzer, and André Schulz. Fine-grained analysis of problems on curves. In *EuroCG, Lugano, Switzerland*, 2016.

10 Kevin Buchin, Maike Buchin, Wouter Meulemans, and Wolfgang Mulzer. Four soviets walk the dog – with an application to Alt's conjecture. In *ACM-SIAM Symposium on Discrete Algorithms*, pages 1399–1413, 2014.

11 Kevin Buchin, Maike Buchin, and André Schulz. Fréchet distance of surfaces: Some simple hard cases. In *European Symposium of Algorithms*, pages 63–74, 2010.

12 Kevin Buchin, Maike Buchin, Rolf van Leusden, Wouter Meulemans, and Wolfgang Mulzer. Computing the Fréchet distance with a retractable leash. In *European Symposium of Algorithms*, pages 241–252, 2013.

13 Kevin Buchin, Maike Buchin, and Carola Wenk. Computing the Fréchet distance between simple polygons. *Comput. Geom.*, 41(1-2):2–20, 2008.

14 Erin W. Chambers, Éric Colin de Verdière, Jeff Erickson, Sylvain Lazard, Francis Lazarus, and Shripad Thite. Homotopic Fréchet distance between curves or, walking your dog in the woods in polynomial time. *Comput. Geom.*, 43(3):295–311, 2010.

15 Shiri Chechik, Haim Kaplan, Mikkel Thorup, Or Zamir, and Uri Zwick. Bottleneck paths and trees and deterministic graphical games. In *Symposium on Theoretical Aspects of Computer Science*, pages 27:1–27:13, 2016.

16 Atlas F. Cook and Carola Wenk. Geodesic Fréchet distance inside a simple polygon. *ACM Transactions on Algorithms*, 7(1):9, 2010.

17 Mark de Berg and Marc J. van Kreveld. Trekking in the alps without freezing or getting tired. *Algorithmica*, 18(3):306–323, 1997.

18 Adrian Dumitrescu and Günter Rote. On the Fréchet distance of a set of curves. In *Canadian Conference on Computational Geometry*, pages 162–165, 2004.

19 Chenglin Fan, Omrit Filtser, Matthew J. Katz, Tim Wylie, and Binhai Zhu. On the chain pair simplification problem. In *Symposium on Algorithms and Data Structures*, pages 351–362, 2015.

20 GAMMA group. PQP – a proximity query package, 1999. University of North Carolina at Chapel Hill, USA.

21 Dan Halperin and Micha Sharir. A near-quadratic algorithm for planning the motion of a polygon in a polygonal environment. *Discrete & Computational Geometry*, 16(2):121–134, 1996.

22 Sariel Har-Peled and Benjamin Raichel. The Fréchet distance revisited and extended. *ACM Transactions on Algorithms*, 10(1):3, 2014.

23 John E. Hopcroft, Jacob T. Schwartz, and Micha Sharir. On the complexity of motion planning for multiple independent objects; PSPACE-hardness of the "Warehouseman's problem". *International Journal of Robotics Research*, 3(4):76–88, 1984.

24 David Hsu, Jean-Claude Latombe, and Rajeev Motwani. Path planning in expansive configuration spaces. *Int. J. Comput. Geometry Appl.*, 9(4/5):495–512, 1999.

25 Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.*, 63(4):512–530, 2001.

26 Lucas Janson, Edward Schmerling, Ashley A. Clark, and Marco Pavone. Fast marching tree: A fast marching sampling-based method for optimal motion planning in many dimensions. *I. J. Robotic Res.*, 34(7):883–921, 2015.

27 Minghui Jiang, Ying Xu, and Binhai Zhu. Protein structure–structure alignment with discrete Fréchet distance. *Journal of bioinformatics and computational biology*, 6(01):51–64, 2008.

28 Sertac Karaman and Emilio Frazzoli. Sampling-based algorithms for optimal motion planning. *International Journal of Robotics Research*, 30(7):846–894, 2011.

29 Lydia E. Kavraki, Petr Švestka, Jean-Claude Latombe, and Mark H. Overmars. Probabilistic roadmaps for path planning in high dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12(4):566–580, 1996.

30 Lutz Kettner, Kurt Mehlhorn, Sylvain Pion, Stefan Schirra, and Chee-Keng Yap. Classroom examples of robustness problems in geometric computations. *Comput. Geom.*, 40(1):61–78, 2008.

31 Michal Kleinbort, Oren Salzman, and Dan Halperin. Efficient high-quality motion planning by fast all-pairs r-nearest-neighbors. In *IEEE International Conference on Robotics and Automation*, pages 2985–2990, 2015.

32 James J. Kuffner and Steven M. LaValle. RRT-Connect: An efficient approach to single-query path planning. In *International Conference on Robotics and Automation (ICRA)*, pages 995–1001, 2000.

33 S. M. LaValle. *Planning algorithms*. Cambridge University Press, 2006.

34 Wouter Meulemans. Map matching with simplicity constraints. *CoRR*, abs/1306.2827, 2013.

35 M. Muja and D. G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *VISSAPP*, pages 331–340. INSTICC Press, 2009.

36 Mathew Penrose. *Random geometric graphs*, volume 5. Oxford University Press, 2003.

37 Barak Raveh, Angela Enosh, Ora Schueler-Furman, and Dan Halperin. Rapid sampling of molecular motions with prior information constraints. *PLoS Computational Biology*, 5(2), 2009.

38 John H Reif. Complexity of the mover's problem and generalizations: Extended abstract. In *Foundations of Computer Science*, pages 421–427, 1979.

39 Oren Salzman and Dan Halperin. Asymptotically-optimal motion planning using lower bounds on cost. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4167–4172, 2015.

40 Micha Sharir. Algorithmic motion planning. In Jacob E. Goodman and Joseph O'Rourke, editors, *Handbook of Discrete and Computational Geometry, Second Edition.*, pages 1037–1064. Chapman and Hall/CRC, 2004.

41 Jessica Sherette and Carola Wenk. Simple curve embedding. *CoRR*, abs/1303.0821, 2013.

42 J. G. Siek, L.-Q. Lee, and A. Lumsdaine. *The Boost Graph Library User Guide and Reference Manual*. Addison-Wesley, 2002.

43 Kiril Solovey and Dan Halperin. On the hardness of unlabeled multi-robot motion planning. In *Robotics: Science and Systems (RSS)*, 2015.

44 Kiril Solovey, Oren Salzman, and Dan Halperin. Finding a needle in an exponential haystack: Discrete RRT for exploration of implicit roadmaps in multi-robot motion planning. *International Journal of Robotics Research*, 35(5):501–513, 2016.

45 Kiril Solovey, Oren Salzman, and Dan Halperin. New perspective on sampling-based motion planning via random geometric graphs. In *Robotics: Science and Systems (RSS)*, Ann Arbor, Michigan, 2016. doi:10.15607/RSS.2016.XII.003.

46 Kiril Solovey, Jingjin Yu, Or Zamir, and Dan Halperin. Motion planning for unlabeled discs with optimality guarantees. In *Robotics: Science and Systems (RSS)*, 2015.

47 Paul G. Spirakis and Chee-Keng Yap. Strong NP-hardness of moving many discs. *Information Processing Letters*, 19(1):55–59, 1984.

48 R Sriraghavendra, K Karthik, and Chiranjib Bhattacharyya. Fréchet distance based approach for searching online handwritten documents. In *Document Analysis and Recognition*, volume 1, pages 461–465. IEEE, 2007.

49 The CGAL Project. *CGAL user and reference manual*. CGAL editorial board, 4.8 edition, 2016. URL: http://www.cgal.org/.

50 Matthew Turpin, Nathan Michael, and Vijay Kumar. Concurrent assignment and planning of trajectories for large teams of interchangeable robots. In *International Conference on Robotics and Automation (ICRA)*, pages 842–848, 2013.

51 Mark Walters. Random geometric graphs. In Robin Chapman, editor, *Surveys in Combinatorics 2011*, chapter 8, pages 365–401. Cambridge University Press, 2011.

52 Virginia Vassilevska Williams. *Efficient Algorithms for Path Problems in Weighted Graphs*. Ph.D. thesis, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, USA, 2008.

53 Jianbin Zheng, Xiaolei Gao, Enqi Zhan, and Zhangcan Huang. Algorithm of on-line handwriting signature verification based on discrete Fréchet distance. In *Advances in Computation and Intelligence*, pages 461–469. Springer, 2008.

8

Efficient Sampling-based Bottleneck Pathfinding over Cost Maps

2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
September 24–28, 2017, Vancouver, BC, Canada

Efficient Sampling-based Bottleneck Pathfinding over Cost Maps

Kiril Solovey* and Dan Halperin*

***Abstract*— We introduce a simple yet effective sampling-based planner that is tailored for *bottleneck pathfinding*: Given an implicitly-defined cost map $\mathcal{M} : \mathbb{R}^d \to \mathbb{R}$, which assigns to every point in space a real value, we wish to find a path connecting two given points, which minimizes the maximal value with respect to $\mathcal{M}$. We demonstrate the capabilities of our algorithm, which we call *bottleneck tree* (BTT), on several challenging instances of the problem involving multiple agents, where it outperforms the state-of-the-art cost-map planning technique T-RRT*. In addition to its efficiency, BTT requires the tuning of only a single parameter: the number of samples. On the theoretical side, we study the asymptotic properties of our method and consider the special setting where the computed trajectories must be monotone in all coordinates. This constraint arises in cases where the problem involves the coordination of multiple agents that are restricted to forward motions along predefined paths.**

I. Introduction

Motion planning is a widely studied problem in robotics. In its basic form it is concerned with moving a robot between two given configurations while avoiding collisions with obstacles. Typically, we are interested in paths of high quality, which lead to lower energy consumption, shorter execution time, or safer execution for the robot and its surrounding. One of the most studied quality measures is path length, which was first studied from the combinatorial and geometric perspective [1] and has recently gained popularity with the introduction of PRM*, RRT* [2] and subsequent work. Another popular measure is the clearance (see, e.g, [3]) of a path: the *clearance* of a particular robot configuration is the distance to the closest forbidden configuration. The goal here is to find a path that maximizes the minimum clearance of the configurations along it.

The latter example is a special case of the *bottleneck pathfinding* problem: we are given an implicitly-defined cost map $\mathcal{M} : \mathcal{C} \to \mathbb{R}$ which assigns to every configuration $c \in \mathcal{C}$ of the robot a value; the goal is to find, given start and target configurations, a continuous path $\nu : [0, 1] \to \mathcal{C}$, which minimizes the expression $\max_{\tau \in [0,1]}\{\mathcal{M}(\nu(\tau))\}$.

Bottleneck pathfinding can also arise in settings involving multiple robots. However, in most cases multi-robot motion planning is computationally intractable (see, e.g., [4], [5]), even when one is only concerned with finding *a* solution. Thus, *decoupled* planners (see, e.g., [6], [7]) usually construct a set of d paths—one for each robot—and attempt to coordinate between the robots along their predefined paths in order to avoid collisions. Now suppose that we also wish to find the *safest* coordination—the one which keeps the robots in a maximal distance apart. This problem again, can be reformulated as a bottleneck-pathfinding problem, where d is the dimension of the effective search space. In particular, denote by $\sigma_i : [0, 1] \to \mathcal{C}$ the path assigned for robot i, where $1 \leq i \leq d$, and define for a given d-dimensional point $x = (x_1, \ldots, x_m) \in [0, 1]^d$ the cost map $\mathcal{M}(x) := 1/\max_{1 \leq i < j \leq c} \|\sigma_i(x_i) - \sigma_j(x_j)\|$. In other cases, the robots are restricted a priori to predefined paths, as in factory assembly lines [8], aviation routes [9] and traffic intersections [10].

An interesting complication that typically arises from the examples of the previous paragraph, is that each robot must move forward along its path and cannot backtrack. This restriction induces paths that are *monotone* in each of the coordinates in the search space $[0, 1]^d$. This is reminiscent of the notion of *Fréchet matching*, which can be viewed as another instance of bottleneck pathfinding. Fréchet matching is a popular similarity measure between curves which has been extensively studied by the computational-geometry community (see, e.g., [11], [12]) and recently was used in robotics [13], [14], [15]. This matching takes into consideration not only the "shape" of curves but also the order in which the points are arranged along them.

Contribution. Rather than tackling each problem individually, we take a sampling-based approach and develop an effective planner that can cope with complex instances of the bottleneck-pathfinding problem. We demonstrate the capabilities of our technique termed *bottleneck tree* (BTT) on a number of challenging problems, where it outperforms the state-of-the-art technique T-RRT* by several orders of magnitude. In addition to its efficiency, BTT requires the tuning of only a single parameter: the number of samples. On the theoretical side, we study the asymptotic properties of our method and consider the special setting where the returned paths must be monotone in all coordinates.

Notice that the monotonicity requirement makes the problem harder, particularly in the sampling-based setting. Consider for example a PRM G that is constructed using a connection radius r_n and n samples. By removing all the non-monotone edges from G we obtain a much sparser graph G': informally, G' may retain only a $1/2^d$ fraction of the edges of G, where d is the dimension of the problem. Fortunately, our analysis of BTT indicates that in order to guarantee optimality in the monotone regime, the connection radius does not have to be drastically increased over its value

* Kiril Solovey and Dan Halperin are with the Blavatnik School of Computer Science, Tel Aviv University, Israel. Email: {kirilsol,danha}@post.tau.ac.il.

This work has been supported in part by the Israel Science Foundation (grant no. 1102/11), by the Blavatnik Computer Science Research Fund, and by the Hermann Minkowski–Minerva Center for Geometry at Tel Aviv University. Kiril Solovey is also supported by the Clore Israel Foundation.

in the non-monotone case of say PRM.

We have already considered sampling-based bottleneck pathfinding in a previous work [16]. However, there we employed a PRM (or PRM*)-based approach which is far inferior to BTT. Already when working in a four-dimensional C-space, the PRM-base approach, which seeks to explore the *entire* C-space, becomes prohibitively expensive. In contrast, BTT quickly produces low-cost solutions even in a seven-dimensional space. We also provide here stronger theoretical analysis of our new technique for the monotone case. These differences are discussed in more detail in Sections V and VI.

In Section II we review related work, including existing sampling-based planners which work with an underlying cost map. In Section III we provide a formal definition of bottleneck pathfinding. In Section IV we describe our BTT algorithm for the problem. In Section V we provide an asymptotic analysis of the method in the monotone case. In Section VI we report on experimental results and conclude with a discussion and future work in Section VII.

II. Related work

Particular instances of the basic motion-planning problem involving a small number of degrees of freedom can be solved efficiently in a complete manner [17]. Complete planners can even cope with scenarios involving multiple robots, if some assumptions are made about the input (see, e.g., [18], [19], [20]. However, in general motion planning is computationally intractable [4], [5], [21]. Thus, most of the recent efforts in this area are aimed at the development of sampling-based planners, which attempt to capture the connectivity of the free space using random sampling.

We continue our literature review with general-purpose sampling-based planners, with an emphasis on planners that are applicable to settings involving cost maps. We then proceed to specific examples of bottleneck pathfinding that were studied from the combinatorial-algorithmic perspective.

A. Sampling-based motion planning

Early sampling-based planners such as PRM [22] and RRT [23] have focused on finding *a* solution. Although some efforts were made to understand the quality of paths produced by such planners [24], [25], [26] this issue remained elusive until recently. In their influential work Karaman and Frazzoli [2] introduced the planners PRM* and RRT*, which were shown to be *asymptotically optimal*, i.e., guaranteed to return a solution whose cost converges to the optimum. Several asymptotically-optimal planners that focus on the path-length cost have later emerged [27], [28], [29]. In our recent work [30] we develop a general framework for the analysis of theoretical properties of a variety of sampling-based planners through a novel connection with the theory of random geometric graphs.

We now proceed to discuss sampling-based planning in the presence of an underlying cost map $\mathcal{M}$, as in bottleneck pathfinding. RRT* theoretically guarantees to converge to the optimum even when the cost of the path is computed with respect to $\mathcal{M}$. However, in practice this convergence is very slow, since RRT* does not take into account the cost of $\mathcal{M}$ while exploring $\mathcal{C}$, and puts too much effort in the traversal of regions of high cost. On the other hand, T-RRT [31] equips RRT with a *transition test*, which biases the exploration towards low-cost regions of $\mathcal{M}$. However, T-RRT does not take path quality into consideration and cannot improve the cost of a solution that was already obtained. Consequently, this planner provides no optimality guarantees. The T-RRT* algorithm [32] combines RRT* with T-RRT, which yields an asymptotically-optimal planner that also works well in practice. The authors of [32] test T-RRT* for the case of *integral cost*, which sums the cost of the configurations along the path, and *mechanical cost*, which sums the positive cost variations along the path. We mention that T-RRT* is also applicable to the bottleneck cost, although this setting was not tested in that work. We note that we are not aware of planners that are suitable for bottleneck pathfinding in the presence of a cost map, other than PRM* (see Section I), or RRT* and T-RRT* that we just mentioned. We do mention that the recently-introduced FMT* planner can be extended to work with an underlying cost map, although in such a case it requires the roadmap to be known a priori (see, [28, Appendix]), which is unlikely to occur in our setting.

B. Bottleneck pathfinding and other problems

Cost maps typically arise as *implicit* underlying structures in many problems involving motion and path planning. de Berg and van Kreveld [33] consider the problem of path planning over a *known* mountainous region. They describe data structures that can efficiently obtain, given two query points (which are the start and goal of the desired path), paths with various properties, such as those that strictly descend and paths that minimize the maximal height obtained, i.e., bottleneck paths.

We have already mentioned the problem of high-clearance paths. For the simple case of a disc robot moving amid polygonal obstacles this problem can be solved efficiently by constructing a Voronoi diagram over the sites that consist of the obstacles [34]. There is an obvious trade-off between clearance and path length, a short paths tend to get very close to the obstacles. The work by Wein et al. [3] describes a data structure that given a clearance threshold $\delta > 0$ returns the shortest path with that clearance. Agarwal et al. [35] consider a particular cost function suggested by Wein et al. [36] that combines path length and clearance and develop an efficient approximation algorithm for this case.

Perhaps the most popular example of bottleneck pathfinding from the recent years is *Fréchet matching*, which is concerned with quantifying the similarity of two given curves: the Fréchet distance between two given curves can be described as the shortest leash that allows a person to walk her dog, where the former is restricted to move along the first curve and latter along the other. It is generally assumed that this quantity tends to be more informative when only *forward motions* along the curves are permitted, which transform into *monotone* paths in the search space. Fréchet matching has been extensively studied for the case of two

curves: several efficient techniques that solve the problem exist (see, e.g., [11], [37]). The problem can be extended to multiple curves although only algorithms that are exponential in the number of curves are known [12], [38]. Several papers consider extensions of the problem involving more complex input objects [39], [40] or cases where additional constraints such as obstacles are imposed on the "leash" [41], [42]. Finally, we mention our recent work [16] where we apply a sampling-based PRM-like technique for solving several Fréchet-type problems.

III. Preliminaries

We provide a formal definition of the bottleneck-pathfinding problem. For some fixed dimension $d \geq 2$, let $\mathcal{M} : [0,1]^d \to \mathbb{R}$ be a cost map that assigns to every point in $[0,1]^d$ a value in $\mathbb{R}$. We will refer from now on to $[0,1]^d$ as the *parameter space* in order to distinguish it from the configuration space of the underlying problem.

Notice that the definition of $\mathcal{M}$ does not imply that we restrict our discussion to the setting of a single robot whose configuration space is $[0,1]^d$. For instance, $[0,1]^d$ can represent the parameter space of d curves of the form $\sigma_i : [0,1] \to \mathcal{C}_i$, where for every $1 \leq i \leq d$, σ_i describes that motion of robot i in the configuration space $\mathcal{C}_i$. Typically in such settings involving multiple robots having predefined paths the robots are only permitted to move forward along their paths and never backtrack. This restriction induces *monotone* motions in $[0,1]^d$. Given two points $x = (x_1, \ldots, x_d), y = (y_1, \ldots, y_d) \in \mathbb{R}^d$ the notation $x \preceq y$ implies that for every $1 \leq i \leq d$ it holds that $x_i \leq y_i$. Additionally, we use the notation $x \preceq_\delta y$ for $\delta > 0$ to indicate that $x \preceq y$ and for every $1 \leq i \leq d$ it holds that $y_i - x_i \geq \delta$.

For simplicity we will assume that the goal consists of planning paths between the points $\mathbb{0}, \mathbb{1} \in [0,1]^d$, where $\mathbb{0} = (0, \ldots, 0), \mathbb{1} = (1, \ldots, 1)$.

Definition 1: A plan $\nu : [0,1] \to [0,1]^d$ is a continuous path in $[0,1]^d$ that satisfies the following two constraints: (i) $\nu(0) = \mathbb{0}, \nu(1) = \mathbb{1}$; (ii) it is monotone in each of the d coordinates, i.e., for every $0 \leq \tau \leq \tau' \leq 1$ and $\nu(\tau) \preceq \nu(\tau')$.

Given a path (or a plan) ν, its *bottleneck cost* is defined to be $\mathcal{M}(\nu) = \max_{\tau \in [0,1]} \mathcal{M}(\nu(\tau))$. In Section VI we will address specific examples of the following problem:

Definition 2: Given a cost map $\mathcal{M} : [0,1]^d \to \mathbb{R}$ the (monotone) bottleneck-pathfinding problem consists of finding a plan ν which minimizes $\mathcal{M}(\nu)$.

IV. Bottleneck-tree planner

In Algorithm 1 we describe our sampling-based technique for bottleneck pathfinding, which we call bottleneck tree (BTT). The algorithm can be viewed as a lazy version of PRM which explores the underlying graph in a Dijkstra fashion, according to cost-to-come, with respect to the bottleneck cost over $\mathcal{M}$. Thus, it bears some resemblance to FMT* [28] since the two implicitly explore an underlying PRM. However, FMT* behaves very differently from BTT since it minimizes the path-length cost.

Algorithm 1 BOTTLENECK-TREE(n, r_n)

```
1:  V := {0, 1} ∪ sample(n);
2:  for all x ∈ V do
3:      c(x) := ∞; p(x) := null
4:  c(0) := M(0)
5:  while ∃x ∈ V such that c(x) < ∞ do
6:      z := get_min(V)
7:      if z == 1 then
8:          return path(T, 0, 1)
9:      V := V \ {z}
10:     N_z := near(z, V, r_n)
11:     for all x ∈ N_z do
12:         if z ⪯ x and c(z) < c(x) then
13:             c_new := max{c(z), M(z, x)}
14:             if c_new < c(x) then
15:                 c(x) := c_new; p(x) := z
16: return ∅
```

BTT accepts the number of samples n and the connection radius r_n as parameters. It maintains a directed minimum spanning tree $\mathcal{T}$ of an underlying *monotone* PRM or a random geometric graph (RGG) [30][1]:

Definition 3: Given $n \in \mathbb{N}_+$ denote by $\mathcal{X}_n = \{X_1, \ldots, X_n\}$, n points chosen independently and uniformly at random from $[0,1]^d$. For a connection radius $r_n > 0$ denote by $\mathcal{G}_n = \mathcal{G}(\{\mathbb{0}, \mathbb{1}\} \cup \mathcal{X}_n; r_n)$ the *monotone* RGG defined over the vertex set $\{\mathbb{0}, \mathbb{1}\} \cup \mathcal{X}_n$. $\mathcal{G}_n$ has a directed edged (x, y) for $x, y \in \{\mathbb{0}, \mathbb{1}\} \cup \mathcal{X}_n$ iff $x \preceq y$ and $\|x - y\| \leq r_n$, where $\|\cdot\|$ represents the standard Euclidean distance.

The algorithm keeps track of the unvisited vertices of $\mathcal{G}_n$ using the set V. It maintains for each vertex x the cost-to-come over the visited portion of $\mathcal{G}_n$ and its parent for this specific path, are denoted by $c(x)$ and $p(x)$, respectively.

In line 1 we initialize V with $\mathbb{0}, \mathbb{1}$, and a set of n uniform samples in $[0,1]^d$, which are generated using `sample`. In lines 2-3 the cost-to-come and the parent attributes are initialized. In each iteration of the **while** loop, which begins in line 5, the algorithm extracts the vertex $z \in V$ which minimizes $c(z)$, using `get_min` (line 6). If z turns out to be $\mathbb{1}$ (line 7), then a path from $\mathbb{0}$ to $\mathbb{1}$ is returned. This is achieved using `path`, which simply follows the ancestors of $\mathbb{1}$ stored in the p attribute until reaching $\mathbb{0}$. Otherwise, z is removed from V (line 9) and a subset of its neighbors in $\mathcal{G}_n$ that are also members of V are obtained using `near` (line 10). For each such neighbor x (line 11) it is checked whether $z \preceq x$ and whether the cost-to-come of x can potentially improve by arriving to it from z (line 12). An alternative cost-to-come for x which is induced by a path that reaches from z is calculated (line 13). In particular, this cost is the maximum between the cost-to-come of z and the cost of the edge[2] from z to x with respect to $\mathcal{M}$. If the alternative option improves its cost then the parents and the

[1]PRMs and RGGs are equivalent in our context.

[2]The cost of an edge with respect to a given cost map can be approximated by dense sampling along the edge, as is customary in motion planning.

cost of x are updated accordingly (line 15).

In preparation for the following section, where we study the asymptotic behavior of BTT, we mention here that given that the underlying graph $\mathcal{G}_n$ contains a path from $\mathbb{0}$ to $\mathbb{1}$, BTT finds the minimum bottleneck path over $\mathcal{G}_n$, namely the path whose maximal $\mathcal{M}$ value is minimized. This follows from the observation that BTT explores the implicitly defined graph $\mathcal{G}_n$ in a Dijkstra fashion, albeit for the bottleneck cost; see, e.g., [43].

V. Asymptotic analysis

In the previous section we have argued that BTT returns a solution that minimizes the bottleneck cost over a fixed graph $G \in \mathcal{G}_n$. A major question in the analysis of sampling-based motion-planning algorithms is under what conditions does a discrete graph structure captures the underlying continuous space. For this purpose, we study the asymptotic properties of $\mathcal{G}_n = \mathcal{G}(\{\mathbb{0}, \mathbb{1}\} \cup \mathcal{X}_n; r_n)$ (Definition 3). To the best of our knowledge, existing proofs concerning asymptotic optimality or completeness of sampling-based planners (see, e.g., [2], [28]) only apply to the non-monotone (and standard) setting. Such results cannot be extended as-are to the analysis of $\mathcal{G}_n$. The remainder of this section is dedicated to strengthening our our analysis concerning the special and harder case that imposes the monotonicity requirement.

Previously [16], we were able to give a convergence guarantee in the monotone case by introducing a connection radius (r_n) of the form $f(n)\left(\frac{\log n}{n}\right)^{1/d}$, where $f(n)$ is any function that diminishes as n tends to ∞. Unfortunately, this statement does not indicate which specific function should be used in practice, and an uncareful selection of f could lead to undesired behavior of BTT. In particular, if f grows too slowly with n the graph becomes disconnected and BTT may not be able to find a solution at all. On the other hand, f that grows too fast with n could result in a needlessly dense graph which will take very long time to traverse. In this paper, we replace this somewhat abstract definition with a connection radius of the form $\gamma\left(\frac{\log n}{n}\right)^{1/d}$, where γ is a constant depending only on d. This is a more typical structure of bounds in the context of random geometric graphs and sampling-based planners. Furthermore, we show in Section VI that this formulation of r_n leads to a quick convergence of BTT to the optimum.

For the purpose of the analysis, we define a supergraph of $\mathcal{G}_n$, denoted by $\mathcal{L}_n = \mathcal{L}(\{\mathbb{0}, \mathbb{1}\} \cup \mathcal{X}_n; r_n)$, which corresponds to the standard and non-monotone random geometric graph (see [30]). In particular, it connects every pair of vertices $x, y \in \{\mathbb{0}, \mathbb{1}\} \cup \mathcal{X}_n$ with an edge if and only if $\|x - y\| \leq r_n$, regardless of whether the monotonicity constraint is satisfied.

In order to guarantee asymptotic optimality, one typically has to make some assumptions with respect to the solution one hopes to converge to. First, we introduce basic definitions. Denote by $\mathcal{B}_r(x)$ the d-dimensional Euclidean ball of radius $r > 0$ centered at $x \in \mathbb{R}^d$ and $\mathcal{B}_r(\Gamma) = \bigcup_{x \in \Gamma} \mathcal{B}_r(x)$ for any $\Gamma \subseteq \mathbb{R}^d$. Given a curve $\nu : [0,1] \to \mathbb{R}^d$ define $\mathcal{B}_r(\nu) = \bigcup_{\tau \in [0,1]} \mathcal{B}_r(\nu(\tau))$. Additionally, denote the image of a curve ν by $\mathrm{Im}(\nu) = \bigcup_{\tau \in [0,1]} \{\nu(\tau)\}$.

Definition 4: Given $\mathcal{M} : [0,1]^d \to \mathbb{R}$, a plan $\nu := [0,1] \to [0,1]^d$ is called *robust* if for every $\varepsilon > 0$ there exists $\delta > 0$ such that for any plan ν' for which $\mathrm{Im}(\nu) \subset \mathcal{B}_\delta(\nu)$ it follows that $\mathcal{M}(\nu') \leq (1+\varepsilon)\mathcal{M}(\nu)$. A plan that attains the infimum cost, over all robust plans, is termed *robustly optimal* and is denoted by ν^*.

The following theorem is the main theoretical contribution of this paper. The constant γ, which is defined below, was first obtained in [28], for a different problem, and involving non-monotone connections.

Theorem 1: Suppose that $\mathcal{L}_n = \mathcal{L}(\mathcal{X}_n \cup \{\mathbb{0}, \mathbb{1}\}; r_n)$ with $r_n = \gamma\left(\frac{\log n}{n}\right)^{1/d}$, where $\gamma = (1+\eta)2(d\theta_d)^{-1/d}$ with θ_d representing the Lebesgue measure of the unit Euclidean ball, and $\eta > 0$ is any positive constant. Then $\mathcal{L}_n$ contains a path ν_n connecting $\mathbb{0}$ to $\mathbb{1}$ which has the following properties a.s.: **(i)** $\mathcal{M}(\nu) = (1 + o(1))\mathcal{M}(\nu^*)$; **(ii)** at most $o(1)$ fraction of the edges along ν_n are non-monotone.

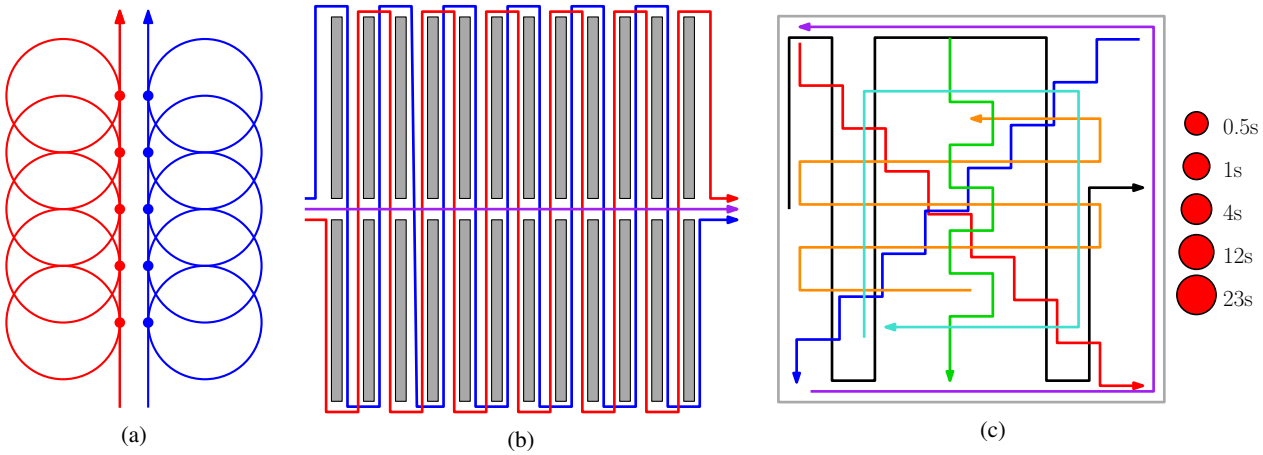
This theorem implies that $\mathcal{L}_n$ contains a path ν that is *almost* entirely monotone and its cost converges to the optimum. Notice that in theory this does not necessarily mean that BTT is asymptotically optimal, as the formulation of the algorithm in Alg. 1 is concerned with paths that are *entirely* monotone. However, our experiments (Section VI) suggest that this theoretical gap can be bridged.

A. Sketch of proof for Theorem 1

Due to space constraints, we provide here a sketch of the proof, whereas the full proof can be found in the extended version of the paper which is available online [44]. Our proof relies on some components that were previously described in the works of Janson et al. [28], Karaman and Frazzoli [2], and our own work [16]. However, we note that several ideas employed here are brand new and so is the final result.

Set r_n to be the connection radius defined in Theorem 1. We show that there exists a sequence of L_n points $q_1, \ldots, q_{L_n}$ along ν^* such that $q_1 = \mathbb{0}, q_{L_n} = \mathbb{1}$ and $\|q_i - q_{i+1}\| < 0.5 \cdot r_n$ for any $1 \leq i < L_n$. We prove that the following event occurs a.s.: There exist a sequence of L_n samples $p_1, \ldots, p_{L_n} \in \mathcal{X}_n$ such that for *every* $1 \leq i \leq L_n$ it holds that $p_i \in \mathcal{B}_{1/2 \cdot r_n}(q_i)$. This ensures that $\mathcal{L}_n$ contains a path ν_n from $\mathbb{0}$ to $\mathbb{1}$ in the vicinity of ν^*, which implies that ν_n has a cost that converges to the optimum, since r_n tends to 0 as $n \to \infty$. In particular, note that for every $1 \leq i < L_n$ it holds that $p_i \in \mathcal{B}_{r_n}(p_{i+1})$, which implies that there is an edge between p_i and p_{i+1} in $\mathcal{L}_n$.

The heart of the proof lies in the following two argument: Firstly, there exists a constant $\beta \in (0, 1/2)$ such that $q_i \preceq_{\beta r_n} q_{i+1}$ for any $1 \leq i < L_n$. Secondly, there exists a subset of indices $I_n \subseteq \{1, \ldots, L_n\}$ such that $|I_n| \geq (1 - o(1))L_n$, and for every $i \in I_n$ it holds that $p_i \in \mathcal{B}_{1/2 \cdot \beta \cdot r_n}(q_i), p_{i+1} \in \mathcal{B}_{1/2 \cdot \beta \cdot r_n}(q_{i+1})$. Observe that if $i \in I_n$ then $p_i \preceq p_{i+1}$, which concludes the sketch of proof.



(a) (b) (c)

Fig. 1: (a) Scenario for **(P1)**, for $d = 2$: the right (blue) curve consists of five circular loops of radius 0.15, where the entrance and exit point to each circle is indicated by a bullet. The left red curve is similarly defined, and the two curves are separated by a vertical distance of 0.04. The optimal matching of cost 0.34 is obtained in the following manner: when a given circle of the red curve is traversed, the position along the blue curve is fixed to the entrance point of the circle directly to the right of the traversed circle, and vice versa. (b) Scenario for **(P2)**: two agents moving along the red and blue curves, respectively, must minimize the distance to the leader on the horizontal purple curve while maintaining visibility with the leader. (c) Scenario for **(P3)**: finding the safest coordination between $d = 7$ moving agents whose routes are depicted as red, blue, turquoise, orange, green, black, and purple curves. The red circles on the right represent the amount of separation obtained by BTT for the specified running time: given a plan returned by BTT, it induces a maximal radius of a disc, such that when d copies of this disc are centered on the moving agents, they are guaranteed to remain collision-free.

VI. Experiments

In this section we report on experimental results of applying BTT to several challenging problems. We also compare its performance with T-RRT*, which is the state-of-the-art for planning on cost maps. Since BTT is concerned with optimality, we compare it against T-RRT* rather than with its non-optimal original version T-RRT. Our technique is faster than T-RRT* by several orders of magnitude. We mention that in a previous work [16] we experimented on similar problems but with a simpler PRM*-flavor technique, which is significantly slower than BTT. Due to this fact we do not compare against it. We also mention that we considered the standard RRT* in the experiments, which was found inferior to T-RRT*, in terms of quality. Thus we only mention the performance of RRT* very briefly below. We refer the reader to Section II for a discussion of sampling-based techniques in the presence of an underlying cost map. Finally, we note that we compute fully monotone paths in all the experiments reported below using the formulation described in Alg. 1. That is, even though our theory does not guarantee the existence of almost entirely-monotone paths, experimentally we see that our formulation of the connection radius suffices for full monotonicity, at least in the tested scenarios.

We implemented BTT and T-RRT* in C++, and tested them on scenarios involving two-dimensional objects. BTT has only two parameters, which are the number of samples n and the connection radius r_n. The latter was set to the value described in Theorem 1 with $\eta = 1$. We note that even though this theorem only suggests that this value suffices for asymptotic optimality in the monotone case, the experiments support this claim also after finite, relatively short, running time. In particular, BTT obtains an initial solution fairly quickly and converges to the optimum, in scenarios where the optimum is known.

Our implementation of T-RRT* is based on its OMPL [45] implementation. T-RRT* uses a connection radius, which we set according to [2]. T-RRT* also has the parameters T, T_{rate}, which are set according to the guidelines in [32]. The length of the `extend` routine, and the rate of target bias, were set according to the OMPL implementation of T-RRT*. Nearest-neighbor search calls in T-RRT* were performed using FLANN [46]. As the set of samples of BTT is generated in one batch, we use the efficient RTG data structure (see, e.g., [47]) for nearest-neighbor search. Geometric objects were represented with CGAL [48]. Experiments were conducted on a PC with Intel i7-2600 3.4GHz processor with 8GB of memory, running a 64-bit Windows 7 OS.

A. Scenarios

We test BTT on several challenging problems, which are described below. We mention that the underlying implementation of BTT and T-RRT* does not change between the different scenarios and we only change the subroutine responsible for the computation of cost-map values.

(P1) Fréchet matching: The goal is to find a traversal which *minimizes* the pairwise distance between the traversal points along $d \in \{2, 3, 4\}$ curves. Recall that given $d \geq 2$ curves $\sigma_1, \ldots, \sigma_d : [0, 1] \to \mathbb{R}^2$, the cost map $\mathcal{M} : [0, 1]^d \to \mathbb{R}_+$ induced by the problem of Fréchet matching [11] is defined to be $\mathcal{M}(\tau_1, \ldots, \tau_d) = \max_{1 \leq i < j \leq d} \|\sigma_i(\tau_i) - \sigma_j(\tau_j)\|$ for $\tau_1, \ldots, \tau_d \in [0, 1]$. The scenario for $d = 2$ is depicted in Fig. 1a. For $d = 3$ we add an identical blue curve, and for

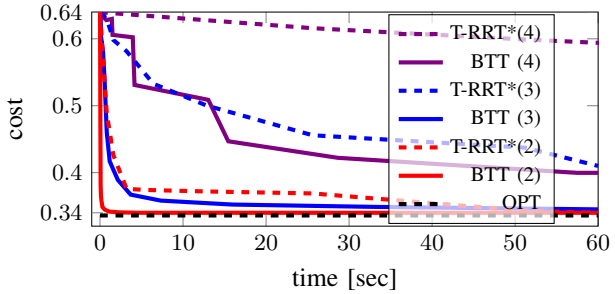


Fig. 2: Results of BTT and T-RRT* on **(P1)** for increasing number of curves $2 \leq d \leq 4$, indicated in parenthesis. The black dashed line represents the optimal cost 0.34. In order to display the quality of the solution of BTT as a function of the running time, rather than the number of samples, we ran BTT on an increasing number of samples, with ten runs for each number. For each n we plot the average cost and the average running time. We note that the standard deviation of the running time is very low, and so the average accurately captures the typical running time. For instance, for $d = 3$ the standard deviation ranges between 0.002 and 0.01.

$d = 4$ another red curve. Note that the optimum is the same for all values of d here.

(P2) Leader following: The problem consists of three curves, where the purple one (Fig. 1b) represents the motion of a "leader", whereas the blue and the red curves represent the "followers". The goal is to plan the motion of the three agents such that at least one of the followers sees the (current position of the) leader at any given time, where the view of each agent can be obstructed by the gray walls. In addition, we must minimize the distance between the leader to its closest follower at a given time. The optimal solution is attained in the following manner: while the leader passes between two vertically-opposite walls the blue follower keeps an eye on it while the red follower goes around a similar set of walls. Then, when the leader becomes visible to the red agent, the blue agent catches up with the other two agents in an analogous fashion, and this process is repeated.

(P3) Safest coordination: This problem consists of finding the *safest coordination* among $d = 7$ agents moving along predefined routes, which represents a complex traffic intersection. The goal is to find the traversal which *maximizes* the pairwise distance between the d traversal points. Each route is drawn in a different color, where the direction is indicated by an arrow (see Fig. 1c).

B. Results

We report on the running times and the obtained cost values of BTT and T-RRT*, after averaging over ten runs in each setting. We start with **(P1)**. For $d \in \{2, 3\}$, BTT rapidly converges to the optimum (see Fig. 2). For $d = 4$ the convergence is slower, although a value relatively close to the optimum is reached (observe that the trivial solution in which the curves proceed simultaneously induces a cost of 0.64). Note that BTT's performance for $d = 3$ is better than that of T-RRT* for $d = 2$. A similar phenomenon occurs for

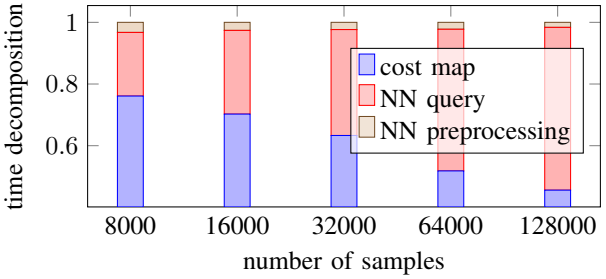


Fig. 3: Time percentage for each of the main components of BTT for **(P3)**.

BTT with $d = 4$ and T-RRT* for $d = 3$. Lastly, for $d = 4$ T-RRT* manages to find a solution that is only slightly better than the trivial one after 60s.

In **(P2)** BTT attains the (near-)optimal solution described earlier after 13s on average. Here T-RRT* is unable to obtain any solution within 60s. In **(P3)**, again, T-RRT* is unable to obtain a solution within 60s. In contrast, BTT obtains a solution in less than a second. Observe that the solution obtained by it for 22 seconds maintains a rather large safety distance between the agents in such a complex setting (see caption of Fig. 1c). Both in **(P2)** and **(P3)** T-RRT*'s transition test leads to an extremely slow exploration of the parameter space. We mention that in these cases the standard RRT* [2] is able to find an initial solution quite rapidly, albeit one of poor quality. Moreover, RRT* hardly manages to improve over the solution obtained initially.

Now we analyze in greater detail the running time of BTT as a function of the number of samples n for the specific scenario in **(P3)**. The running time is dominated by three components: (i) construction of the RTG NN-search data structure; (ii) NN-search calls; (iii) computation of the cost map, which corresponds to collision detection in standard motion planning. BTT's proportion of running time spent between these components changes with n (see Fig. 3). In particular, as n increases so does the proportion of NN calls. This trend can be observed in other planners as well [47].

VII. Discussion and Future Work

We have introduced BTT for sampling-based bottleneck pathfinding over cost maps. We showed that it manages to cope with complex scenarios of the problem and outperforms T-RRT* in all tests. In addition to its efficiency, BTT requires the tuning of only a single parameter: while the number of samples n cannot be determined a priori, the connection radius r_n specified in Theorem 1 proves to be sufficient for all tested cases. We also provide theoretical justification for this phenomenon in the same theorem. Note that this connection radius is widely used in non-monotone settings of motion planning (see OMPL [45]). Thus, we believe that r_n can be fixed to the aforementioned value. In contrast, T-RRT* requires the tuning of two more parameters that are unique to that planner, additionally to those inherited from RRT and RRT*.

The main obstacles to using BTT in more complex settings of bottleneck pathfinding, e.g., higher dimensions, is the

high memory consumption of the RTG data structure. In particular, for $n = 10^6$ and $d = 7$ the program may exceed the 4GB limit provided by the OS. Interestingly, in all the tested cases BTT examines only a small portion of the actual vertex set of $\mathcal{G}_n$ (at most 5%). Moreover, the examined vertices are usually grouped together in contiguous regions of the parameter space. This calls for incremental versions of BTT and RTG which generate samples and preprocess them in an online fashion.

On the theoretical side, we aim to refine the statement made in Theorem 1. Currently we can only ensure that a solution that is "almost monotone" exists, but does a "fully-monotone" solution over $\mathcal{G}_n$ exists as well? We conjecture that this statement is true.

REFERENCES

[1] J. S. B. Mitchell, "Shortest paths and networks," in *Handbook of Discrete and Computational Geometry, Second Edition.*, 2004, pp. 607–641.

[2] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *I. J. Robotic Res.*, vol. 30, no. 7, pp. 846–894, 2011.

[3] R. Wein, J. P. van den Berg, and D. Halperin, "The visibility-voronoi complex and its applications," *Comput. Geom.*, vol. 36, no. 1, pp. 66–87, 2007.

[4] J. E. Hopcroft, J. T. Schwartz, and M. Sharir, "On the complexity of motion planning for multiple independent objects; PSPACE-hardness of the "Warehouseman's problem"," *I. J. Robotic Res.*, vol. 3, no. 4, pp. 76–88, 1984.

[5] K. Solovey and D. Halperin, "On the hardness of unlabeled multi-robot motion planning," *I. J. Robotics Res.*, vol. 35, no. 14, pp. 1750–1759, 2016.

[6] J. P. van den Berg and M. H. Overmars, "Prioritized motion planning for multiple robots," in *IROS*, 2005, pp. 430–435.

[7] F. Altché, X. Qian, and A. de La Fortelle, "Time-optimal coordination of mobile robots along specified paths," *CoRR*, vol. abs/1603.04610, 2016.

[8] D. Spensieri, R. Bohlin, and J. S. Carlson, "Coordination of robot paths for cycle time minimization," in *CASE*, 2013, pp. 522–527.

[9] S. Cafieri and N. Durand, "Aircraft deconfliction with speed regulation: new models from mixed-integer optimization," *Journal of Global Optimization*, vol. 58, no. 4, pp. 613–629, 2014.

[10] P. Dasler and D. M. Mount, "On the complexity of an unregulated traffic crossing," in *WADS*, 2015, pp. 224–235.

[11] H. Alt and M. Godau, "Computing the Fréchet distance between two polygonal curves," *Int. J. Comput. Geometry Appl.*, vol. 5, pp. 75–91, 1995.

[12] S. Har-Peled and B. Raichel, "The Fréchet distance revisited and extended," *ACM Transactions on Algorithms*, vol. 10, no. 1, p. 3, 2014.

[13] C. Voss, M. Moll, and L. E. Kavraki, "A heuristic approach to finding diverse short paths," in *ICRA*, 2015, pp. 4173–4179.

[14] F. Pokorny, K. Goldberg, and D. Kragic, "Topological trajectory clustering with relative persistent homology," in *ICRA*, 2016, pp. 16–23.

[15] R. Holladay and S. Srinivasa, "Distance metrics and algorithms for task space path optimization," in *IROS*, 2016, pp. 5533–5540.

[16] K. Solovey and D. Halperin, "Sampling-based bottleneck pathfinding with applications to Fréchet matching," in *ESA*, 2016, pp. 76:1–76:16.

[17] M. Sharir, "Algorithmic motion planning," in *Handbook of Discrete and Computational Geometry, Second Edition.*, J. E. Goodman and J. O'Rourke, Eds. Chapman and Hall/CRC, 2004, pp. 1037–1064.

[18] K. Solovey, J. Yu, O. Zamir, and D. Halperin, "Motion planning for unlabeled discs with optimality guarantees," in *RSS*, 2015.

[19] A. Adler, M. de Berg, D. Halperin, and K. Solovey, "Efficient multi-robot motion planning for unlabeled discs in simple polygons," *T-ASE*, vol. 12, no. 4, pp. 1309–1317, 2015.

[20] M. Turpin, N. Michael, and V. Kumar, "Concurrent assignment and planning of trajectories for large teams of interchangeable robots," in *ICRA*, 2013, pp. 842–848.

[21] J. H. Reif, "Complexity of the movers problem and generalizations: Extended abstract," in *FOCS*, 1979, pp. 421–427.

[22] L. E. Kavraki, P. Švestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high dimensional configuration spaces," *Trans. Robotics*, vol. 12, no. 4, pp. 566–580, 1996.

[23] J. J. Kuffner and S. M. LaValle, "RRT-Connect: An efficient approach to single-query path planning," in *ICRA*, 2000, pp. 995–1001.

[24] B. Raveh, A. Enosh, and D. Halperin, "A little more, a lot better: Improving path quality by a path-merging algorithm," *Trans. Robotics*, vol. 27, no. 2, pp. 365–371, 2011.

[25] O. Nechushtan, B. Raveh, and D. Halperin, "Sampling-diagram automata: A tool for analyzing path quality in tree planners," in *WAFR*, 2010, pp. 285–301.

[26] R. Geraerts and M. Overmars, "Creating high-quality paths for motion planning," *I. J. Robotics Res.*, vol. 26, no. 8, pp. 845–863, 2007.

[27] O. Arslan and P. Tsiotras, "Use of relaxation methods in sampling-based algorithms for optimal motion planning," in *ICRA*, 2013, pp. 2421–2428.

[28] L. Janson, E. Schmerling, A. A. Clark, and M. Pavone, "Fast marching tree: A fast marching sampling-based method for optimal motion planning in many dimensions," *I. J. Robotic Res.*, vol. 34, no. 7, pp. 883–921, 2015.

[29] O. Salzman and D. Halperin, "Asymptotically near-optimal RRT for fast, high-quality motion planning," *Trans. Robotics*, vol. 32, no. 3, pp. 473–483, 2016.

[30] K. Solovey, O. Salzman, and D. Halperin, "New perspective on sampling-based motion planning via random geometric graphs," in *RSS*, 2016.

[31] L. Jaillet, J. Cortés, and T. Siméon, "Sampling-based path planning on configuration-space costmaps," *Trans. Robotics*, vol. 26, no. 4, pp. 635–646, 2010.

[32] D. Devaurs, T. Siméon, and J. Cortés, "Optimal path planning in complex cost spaces with sampling-based algorithms," *T-ASE*, vol. 13, no. 2, pp. 415–424, 2016.

[33] M. de Berg and M. J. van Kreveld, "Trekking in the alps without freezing or getting tired," *Algorithmica*, vol. 18, no. 3, pp. 306–323, 1997.

[34] C. Ó'Dúnlaing and C. Yap, "A "retraction" method for planning the motion of a disc," *J. Algorithms*, vol. 6, no. 1, pp. 104–111, 1985.

[35] P. K. Agarwal, K. Fox, and O. Salzman, "An efficient algorithm for computing high-quality paths amid polygonal obstacles," in *SODA*, 2016, pp. 1179–1192.

[36] R. Wein, J. P. van den Berg, and D. Halperin, "Planning high-quality paths and corridors amidst obstacles," *I. J. Robotic Res.*, vol. 27, no. 11-12, pp. 1213–1231, 2008.

[37] K. Buchin, M. Buchin, W. Meulemans, and W. Mulzer, "Four soviets walk the dog - with an application to Alt's conjecture," in *SODA*, 2014, pp. 1399–1413.

[38] K. Buchin, M. Buchin, M. Konzack, W. Mulzer, and A. Schulz, "Fine-grained analysis of problems on curves," in *EuroCG*, 2016.

[39] K. Buchin, M. Buchin, and C. Wenk, "Computing the Fréchet distance between simple polygons," *Comput. Geom.*, vol. 41, no. 1-2, pp. 2–20, 2008.

[40] K. Buchin, M. Buchin, and A. Schulz, "Fréchet distance of surfaces: Some simple hard cases," in *ESA*, 2010, pp. 63–74.

[41] A. F. Cook and C. Wenk, "Geodesic Fréchet distance inside a simple polygon," *ACM Transactions on Algorithms*, vol. 7, no. 1, p. 9, 2010.

[42] E. W. Chambers, É. C. de Verdière, J. Erickson, S. Lazard, F. Lazarus, and S. Thite, "Homotopic Fréchet distance between curves or, walking your dog in the woods in polynomial time," *Comput. Geom.*, vol. 43, no. 3, pp. 295–311, 2010.

[43] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms (3. ed.)*. MIT Press, 2009.

[44] K. Solovey and D. Halperin, "Efficient sampling-based bottleneck pathfinding over cost maps," *CoRR*, vol. abs/1608.00261, 2016.

[45] I. A. Şucan, M. Moll, and L. E. Kavraki, "The Open Motion Planning Library," *IEEE Robotics & Automation*, vol. 19, no. 4, pp. 72–82, 2012, http://ompl.kavrakilab.org.

[46] M. Muja and D. G. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration," in *VISSAPP*. INSTICC Press, 2009, pp. 331–340.

[47] M. Kleinbort, O. Salzman, and D. Halperin, "Efficient high-quality motion planning by fast all-pairs r-nearest-neighbors," in *ICRA*, 2015, pp. 2985–2990.

[48] The CGAL Project, *CGAL user and reference manual*, 4.8 ed. CGAL editorial board, 2016. [Online]. Available: http://www.cgal.org/

9

The Critical Radius in Sampling-based Motion Planning

Robotics: Science and Systems 2018
Pittsburgh, PA, USA, June 26-30, 2018

The Critical Radius in Sampling-based Motion Planning

Kiril Solovey and Michal Kleinbort
Blavatnik School of Computer Science, Tel Aviv University, Israel

***Abstract*—We develop a new analysis of sampling-based motion planning in Euclidean space with uniform random sampling, which significantly improves upon the celebrated result of Karaman and Frazzoli (2011) and subsequent work. Particularly, we prove the existence of a critical connection radius proportional to $\Theta(n^{-1/d})$ for n samples and d dimensions: Below this value the planner is guaranteed to fail (similarly shown by the aforementioned work, ibid.). More importantly, for larger radius values the planner is asymptotically (near-)optimal. Furthermore, our analysis yields an explicit lower bound of $1 - O(n^{-1})$ on the probability of success. A practical implication of our work is that asymptotic (near-)optimality is achieved when each sample is connected to only $\Theta(1)$ neighbors. This is in stark contrast to previous work which requires $\Theta(\log n)$ connections, that are induced by a radius of order $\left(\frac{\log n}{n}\right)^{1/d}$. Our analysis is not restricted to PRM and applies to a variety of "PRM-based" planners, including RRG, FMT* and BTT. Continuum percolation plays an important role in our proofs.**

I. Introduction

Motion planning is a fundamental problem in robotics concerned with allowing autonomous robots to efficiently navigate in environments cluttered with obstacles. Although motion planning has originated as a strictly theoretical problem in computer science [12], nowadays it is applied in various fields. Notably, motion planning arises in coordination of multiple autonomous vehicles [8], steering surgical needles [3], and planning trajectories of spacecrafts in orbit [32], to name just a few examples.

Motion planning is notoriously challenging from a computational perspective due to the continuous and high-dimensional search space it induces, which accounts for the structure of the robot, the physical constraints that it needs to satisfy, and the environment in which it operates.

Nowadays the majority of practical approaches for motion planning capture the connectivity of the free space by sampling (typically in a randomized fashion) configurations and connecting nearby samples, to form a graph data structure. Although such *sampling-based planners* are inherently incomplete, i.e., cannot detect situations in which a solution (collision-free path) does not exist, most have the desired property of being able to find a solution *eventually*, if one exists. That is, a planner is *probabilistically complete* (PC) if the probability of finding a solution tends to 1 as the number of samples n tends to infinity. Moreover, some recently-introduced sampling-based techniques are also guaranteed to return high-quality[1] solutions that tend to the optimum as n diverges—a property called *asymptotic optimality* (AO).

An important attribute of sampling-based planners, which dictates both the running time and the quality of the returned solution, is the number of neighbors considered for connection for each added sample. In many techniques this number is directly affected by a connection radius r_n: Decreasing r_n reduces the number of neighbors. This in turn reduces the running time of the planner for a given number of samples n, but may also reduce the quality of the solution or its availability altogether. Thus, it is desirable to come up with a radius r_n that is small, but not to the extent that the planner loses its favorable properties of PC and AO.

A. Contribution

We develop a new analysis of `PRM` [16] for uniform random sampling in Euclidean space, which relies on a novel connection between sampling-based planners and *continuum percolation* (see, e.g., [7]). Our analysis is tight and proves the existence of a *critical connection* radius $r_n^* = \gamma^* n^{-1/d}$, where $\gamma^* > 0$ is a constant[2], and $d \geqslant 2$ is the dimension: If $r_n < r_n^*$ then `PRM` is guaranteed to fail, where d is the dimension. Above the threshold, i.e., when $r_n > r_n^*$, `PRM` is AO for the bottleneck cost, and *asymptotically near optimal*[3] (AnO) with respect to the path-length cost. Furthermore, our analysis yields concrete bounds on the probability of success, which is lower-bounded by $1-O(n^{-1})$. Notice that this bound is comparable to the one obtained in [33] (see Section II) although we show this for a much smaller radius.

Our analysis is not restricted to `PRM` and applies to a variety of planners that maintain `PRM`-like roadmaps, explicitly or implicitly. For instance, when r_n is above the threshold, `FMT`* [14] is AnO with respect to the path-length cost, while `BTT` [29] is AO with respect to the bottleneck cost. `RRG` behaves similarly for the two cost functions. Our results are also applicable to *multi-robot* motion planners such as the recently introduced `dRRT`* [6], and `M`* [35] when applied

This work has been supported in part by the Israel Science Foundation (grant no. 825/15), by the Blavatnik Computer Science Research Fund, and by the Blavatnik Interdisciplinary Cyber Research Center at Tel Aviv University. K.S. is supported by the Clore Israel Foundation. M.K. is supported in part by Yandex. Contact information: `kirilsol@post.tau.ac.il`; `michal.kleinbort@gmail.com`

[1] Quality can be measured in terms of energy, length of the plan, clearance from obstacles, etc.

[2] $0.4 \leqslant \gamma^* \leqslant 0.6$ for all $d \geqslant 2$.

[3] AnO means that the cost of the solution tends to at most a constant factor times the optimum, compared with AO in which this constant is equal to one.

to a continuous domain. See Figure 1 for additional PRM-based planners to which our analysis is applicable (see more information in the extended version of the paper [30]).

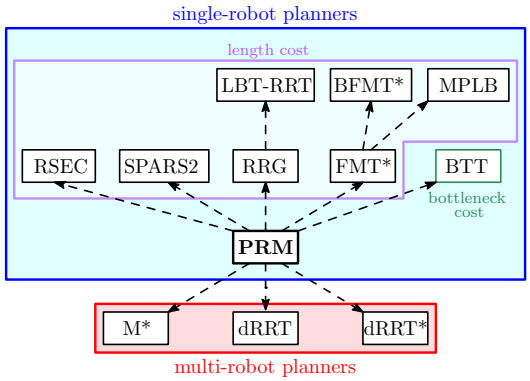


Fig. 1. The PRM dynasty. Single robot and multi-robot planners, are bounded into the blue and red frames, respectively. Planners inside the magenta frame aim to minimize the path-length cost, whereas BTT is designed for bottleneck cost, and RRG works for both costs (see more information below). Roughly speaking, arrow from "A" to "B" indicates that theoretical properties of "A" extend to "B", or that the latter maintains "A" as a substructure.

A practical implication of our work is that AO (or AnO), under the regime of uniform random sampling, can be achieved even when every sample is connected to $\Theta(1)$ neighbors. This is in stark contrast with previous work, e.g., [14, 15, 28, 31], which provided a rough estimate of this number, that is proportional to $O(\log n)$.

B. Organization

In Section II we discuss related work. Then we proceed to basic definitions and problem statement in Section III. In that section we also include a precise description of the robotic system our theory is developed for. Our central contribution (Theorem 1), which states the existence of a critical radius r_n^* with respect to PRM, is presented in Section IV. This section also contains an outline of the proof. In Section V we lay the foundations of the proof and discuss important aspects of continuum percolation, that would later be employed in the main proof, which appears in Section VI. In Section VII we present experimental work, which validates our theory. We conclude this paper with directions for future work (Section VIII).

An extended version of the paper [30] provides supplementary material, and includes proofs that were omitted from the main document, additional experimental results, and a table with values of γ^*. It also presents an extension of the analysis to FMT*, BTT, RRG and discusses the implications of our results to the multi-robot planners dRRT* and M*. Additionally, it presents a new theory for all the aforementioned planners when constructed with deterministic samples, which are then sparsified in a randomized fashion. We believe that this new model, and its analysis, is interesting in its own right.

II. Related Work

This section is devoted to a literature review of sampling-based planners with emphasis on their theoretical analysis.

The influential work [15] laid the theoretical foundations for analyzing quality in sampling-based planning. The authors introduced a set of techniques to prove AO. Using their framework, they showed that the following algorithms are AO: PRM*, which is a special case of PRM[4] with a specific value of the connection radius r_n; an AO variant of RRT [18] termed RRT*; RRG, which can be viewed as a combination between RRT and PRM. The analysis in [15] establishes that $r_n = \Theta\left((\log n/n)^{1/d}\right)$ guarantees AO, where the configuration space of the robot is assumed to be $[0,1]^d$. This indicates that the expected number of neighbors used per vertex should be $O(\log n)$. The authors also proved that for sufficiently-small radii of order $O(n^{-1/d})$ the planner is guaranteed to fail (asymptotically) in finding any solution.

Following the breakthrough in [15], other AO planners have emerged (see e.g., [1, 2, 10]). The paper [14] introduced FMT*, which is a single-query planner that traverses an implicitly-represented PRM graph, and is comparable in performance to RRT*. The authors refined the proof technique of [15], which allowed them to slightly reduce the connection radius r_n necessary to FMT* and PRM to achieve AO. We do mention that here again $r_n = \Theta\left((\log n/n)^{1/d}\right)$. BFMT*, which is a bidirectional version of FMT*, was introduced in [33]. In this paper the authors also proved that the success rate of PRM, FMT*, BFMT* can be lower bounded by $1 - O\left(n^{-\eta/d}\log^{-1/d} n\right)$, where $\eta > 0$ is a tuning parameter. In this context, we also mention the work of Dobson et al. [5], which bounds the success rate with an expression that depends on the amount of deviation from the optimum.

A recent work [31] developed a different method for analyzing sampling-based planners. It exploits a connection with *random geometric graphs* (RGGs), which have been extensively studied (see, e.g., [21]). Their work shows that one can slightly reduce the PRM and FMT* radius obtained in [14]. Furthermore, the connection with RGGs yields additional analyses of different extensions of PRM, which have not been analyzed before in a mathematically-rigorous setting.

We also mention that a number of methods have been developed to reduce the running time or space requirements of existing planners by relaxing AO to AnO, see, e.g., LBT-RRT [24], MPLB [23], SPARS2 [4], and RSEC [25].

A. Extensions

The aforementioned papers deal mainly with the cost function of *path length*. Two recent works [28, 29] considered the *bottleneck-pathfinding* problem in a sampling-based setting and introduced the BTT algorithm, which traverses an implicitly-represented PRM graph. The bottleneck-cost function, which arises for instance in high-clearance multi-robot motion [29] and Fréchet matching between curves [13], is defined as follows: Every robot configuration x is paired with a value $\mathcal{M}(x)$, and the cost of a path is the maximum value of $\mathcal{M}$ along any configuration on the path. It was shown

[4]Throughout this work we will refer to the more general algorithm PRM rather than PRM*.

([28, 29]) that BTT is AO, with respect to bottleneck cost, for the reduced connection radius that was obtained in [31].

The results reported until this point have dealt exclusively with holonomic robotic systems. Two recent papers [26, 27] develop the theoretical foundations of PRM and FMT*-flavored planners when applied to robots having differential constraints. Li et al. [19] develop an AO algorithm that does not require a steering function, as PRM for instance does.

III. Preliminaries

We provide several basic definitions that will be used throughout the paper. Given two points $x, y \in \mathbb{R}^d$, denote by $\|x-y\|$ the standard Euclidean distance. Denote by $\mathcal{B}_r(x)$ the d-dimensional ball of radius $r > 0$ centered at $x \in \mathbb{R}^d$ and $\mathcal{B}_r(\Gamma) = \bigcup_{x\in\Gamma} \mathcal{B}_r(x)$ for any $\Gamma \subseteq \mathbb{R}^d$. Similarly, given a curve $\pi : [0,1] \to \mathbb{R}^d$ define $\mathcal{B}_r(\pi) = \bigcup_{\tau\in[0,1]} \mathcal{B}_r(\pi(\tau))$. Given a subset $D \subset \mathbb{R}^d$ we denote by $|D|$ its Lebesgue measure. All logarithms are at base e.

A. Motion planning

Denote by $\mathcal{C}$ the configuration space of the robot, and by $\mathcal{F} \subseteq \mathcal{C}$ the free space, i.e., the set of all collision free configurations. Though our proofs may be extended to more complex robotic systems (see discussion in Section VIII), in this work we investigate the geometric (holonomic) setting of the problem in which no constraints are imposed on the motion of the robot. Additionally, we assume that $\mathcal{C}$ is some subset of the Euclidean space. In particular, $\mathcal{C} = [0,1]^d \subset \mathbb{R}^d$ for some fixed $d \geqslant 2$. We also assume that for any two configurations $x, x' \in \mathcal{C}$ the robot is capable of following precisely the straight-line path from x to x'.

Given start and target configurations $s, t \in \mathcal{F}$, the problem consists of finding a continuous path (curve) $\pi : [0,1] \to \mathcal{F}$ such that $\pi(0) = s, \pi(1) = t$. That is, the robot starts its motion along π on s, and ends in t, while remaining collision free. An instance of the problem is defined for a given $(\mathcal{F}, s, t)$, where $s, t \in \mathcal{F}$.

B. Cost function

It is usually desirable to obtain paths that minimize a given criterion. In this paper we consider the following two cost functions. Given a path π, its *length* is denoted by $c_\ell(\pi)$, and its *bottleneck cost* is defined to be $c_b(\sigma, \mathcal{M}) = \max_{\tau\in[0,1]} \mathcal{M}(\pi(\tau))$, where $\mathcal{M} : \mathcal{C} \to \mathbb{R}$ is a *cost map*.

We proceed to describe the notion of *robustness*, which is essential when discussing properties of sampling-based planners. Given a subset $\Gamma \subset \mathcal{C}$ and two configurations $x, y \in \Gamma$, denote by $\Pi^{\Gamma}_{x,y}$ the set of all continuous paths, whose image is in Γ, that start in x and end in y, i.e., if $\pi \in \Pi^{\Gamma}_{x,y}$ then $\pi : [0,1] \to \Gamma$ and $\pi(0) = x, \pi(1) = y$.

Definition 1. Let $(\mathcal{F}, s, t)$ be a motion-planning problem. A path $\pi \in \Pi^{\mathcal{F}}_{s,t}$ is *robust* if there exists $\delta > 0$ such that $\mathcal{B}_\delta(\pi) \subset \mathcal{F}$. We also say that $(\mathcal{F}, s, t)$ is *robustly feasible* if there exists such a robust path.

Definition 2. The *robust optimum* with respect to c_ℓ is defined as $c^*_\ell = \inf\{c_\ell(\pi) | \pi \in \Pi^{\mathcal{F}}_{s,t} \text{ is robust}\}$.

The corresponding definition for the bottleneck cost is slightly more involved.

Definition 3. Let $\mathcal{M}$ be a cost map. A path $\pi \in \Pi^{\mathcal{F}}_{s,t}$ is *$\mathcal{M}$-robust* if it is robust and for every $\varepsilon > 0$ there exists $\delta > 0$ such that for every $x \in \mathcal{B}_\delta(\pi)$, $\mathcal{M}(x) \leqslant (1+\varepsilon) c_b(\pi, \mathcal{M})$. We also say that $\mathcal{M}$ is *well behaved* if there exists at least one $\mathcal{M}$-robust path.

Definition 4. The *robust optimum* with respect to c_b is defined as $c^*_b = \inf\{c_b(\pi, \mathcal{M}) | \pi \in \Pi^{\mathcal{F}}_{s,t} \text{ is } \mathcal{M}\text{-robust}\}$.

C. Poisson point processes

We draw our main analysis techniques from the literature of continuum percolation, where point samples are generated with the following distribution. Thus we will use this point distribution in PRM, which would be formally defined in the following section.

Definition 5 ([7])**.** A random set of points $\mathcal{X} \subset \mathbb{R}^d$ is a *Poisson point process* (PPP) of density $\lambda > 0$ if it satisfies the conditions: (1) For mutually disjoint domains $D_1, \ldots, D_\ell \subset \mathbb{R}^d$, the random variables $|D_1 \cap \mathcal{X}|, \ldots, |D_\ell \cap \mathcal{X}|$ are mutually independent. (2) For any bounded domain $D \subset \mathbb{R}^d$ we have that for every $k \geqslant 0$, $\Pr[|\mathcal{X} \cap D| = k] = e^{-\lambda|D|} \frac{(\lambda|D|)^k}{k!}$.

It will be convenient to think about PRM as a subset of the following *random geometric graph* (RGG). We will describe various properties of this graph in later sections.

Definition 6. ([21]) Let $\mathcal{X} \subset \mathbb{R}^d$ be a PPP. Given $r > 0$, the random geometric graph $\mathcal{G}(\mathcal{X}; r)$ is an undirected graph with the vertex set $\mathcal{X}$. Given two vertices $x, y \in \mathcal{X}$, $(x, y) \in \mathcal{G}(\mathcal{X}; r)$ if $\|x - y\| \leqslant r$.

IV. Analysis of PRM

In this section we provide a mathematical description of PRM, which essentially maintains an underlying RGG with PPP samples. We proceed to describe our main contribution (Theorem 1) which is concerned with the conditions for which PRM converges to the (robust) optimum. We then provide an outline of the proof, in preparation for the following sections.

Recall that PRM accepts as parameters the number of samples $n \in \mathbb{N}_+$ and a connection radius r_n. Denote by $\mathcal{X}_n$ a PPP with mean density n. In relation to the definitions of the previous section, the graph data structure obtained by the *preprocessing stage* of PRM can be viewed as an RGG. For instance, when $\mathcal{F} = \mathcal{C}$, the graph obtained by PRM is precisely $\mathcal{G}(\mathcal{X}_n \cap [0,1]^d; r_n)$. In the more general case, when $\mathcal{F} \subset \mathcal{C}$, PRM produces the graph $\mathcal{G}(\mathcal{X}_n \cap \mathcal{F}; r_n)$. As $\mathcal{F}$ can be non-convex, we emphasize that the latter notation describes the maximal subgraph of $\mathcal{G}(\mathcal{X}_n; r_n)$ such that vertices and edges are contained in $\mathcal{F}$.

In the *query stage*, recall that PRM accepts two configurations $s, t \in \mathcal{F}$, which are then connected to the preprocessed graph. Here we slightly diverge from the standard definition of

PRM in the literature. In particular, instead of using the same radius r_n when connecting s, t, we use the (possibly larger) radius r_n^{st}. The graph obtained after query is formally defined below:

Definition 7. The PRM graph $\mathcal{P}_n$ is the union between $\mathcal{G}(\mathcal{X}_n \cap \mathcal{F}; r_n)$ and the supplementary edges

$$\bigcup_{y \in \{s,t\}} \left\{(x,y) \middle| x \in \mathcal{X}_n \cap \mathcal{B}_{r_n^{st}}(y) \text{ and } xy \subset \mathcal{F}\right\}.$$

We reach our main contribution:

Theorem 1. *Suppose that $(\mathcal{F}, s, t)$ is robustly feasible. Then there exists a critical radius $r_n^* = \gamma^* n^{-1/d}$, where γ^* is a constant[5], such that the following holds:*

i. If $r_n < r_n^*$ and $r_n^{st} = \infty$ then PRM fails (to find a solution) a.a.s.[6]

ii. Suppose that $r_n > r_n^*$. There exists $\beta_0 > 0$ such that for $r_n^{st} = \frac{\beta \log^{1/(d-1)} n}{n^{1/d}}$, where $\beta \geqslant \beta_0$, and any $\varepsilon > 0$ the following holds with probability $1 - O(n^{-1})$:

1. $\mathcal{P}_n$ contains a path $\pi_n \in \Pi_{s,t}^{\mathcal{F}}$ with $c_\ell(\pi_n) \leqslant (1+\varepsilon)\xi c_\ell^*$, where ξ is independent of n;

2. If $\mathcal{M}$ is well behaved then $\mathcal{P}_n$ contains a path $\pi_n' \in \Pi_{s,t}^{\mathcal{F}}$ with $c_b(\pi_n', \mathcal{M}) \leqslant (1+\varepsilon)c_b^*$.

A. Outline of proof

For the remainder of this section we briefly describe our technique for proving this theorem, in preparation for the full proof, which is given in Section VI. The critical radius r_n^* defined above, coincides with the *percolation threshold*, which determines the emergence of a connected component of $\mathcal{G}$ that is of infinite size. In particular, if $r_n < r_n^*$ then $\mathcal{G}(\mathcal{X}_n; r_n)$ breaks into tiny connected components of size $O(\log n)$ each. Thus, unless s, t are infinitesimally close, no connected component can have both s and t simultaneously.

More interestingly, the radius of $r_n > r_n^*$ leads to the emergence of a unique infinite component of $\mathcal{G}(\mathcal{X}_n; r_n)$. That is, in such a case one of the components of $\mathcal{G}(\mathcal{X}_n; r_n)$ must contain an infinite number of vertices (Section V-A). Denote this component by C_∞.

In contrast to $\mathcal{G}(\mathcal{X}_n; r_n)$, which is defined for the unbounded space $\mathbb{R}^d$, our motion-planning problem is bounded to $\mathcal{C} = [0,1]^d$. Thus, the next step is to investigate the properties of $\mathcal{G}(\mathcal{X}_n; r_n)$ when restricted to $[0,1]^d$ (Section V-B). Denote by C_n the largest connected component of $C_\infty \cap [0,1]^d$. This structure plays a key role in our proof (see Section VI): With high probability (to be defined), there exist vertices of C_n that are sufficiently close to s and t, respectively, so that a connection between the two vertices can be made through C_n.

Of course, this overlooks the fact that some portions of C_n lie in forbidden regions of $\mathcal{C}$. Thus, we also have to take the structure of $\mathcal{F}$ into consideration. To do so, we rely on [22] to prove that any small subset of $[0,1]^d$ must contain at least one point of C_n (see lemmata 2 and 3). This allows us to trace the robust optimum (and collision-free) path with points from C_n.

The final ingredient, which allows to bound the path length along $\mathcal{G}(\mathcal{X}_n; r_n) \cap [0,1]^d$, is Theorem 4. It states that the distance over this graph is proportional to the Euclidean distance between the end points. This also ensures that the trace points from C_n can be connected with collision-free paths over the graph.

To conclude, in Section V we provide background on continuum percolation in unbounded and bounded domains, and prove two key lemmata (Lemma 2 and Lemma 3). In Section VI we return to the setting of motion planning and utilize the aforementioned results in the proof of Theorem 1.

V. Elements of continuum percolation

In this section we describe some of the properties of the unbounded graph $\mathcal{G}(\mathcal{X}_n; r_n)$ that will be employed in our analysis in the following section.

A. The basics

A fundamental question is when $\mathcal{G}$ contains an infinite connected component around the origin.

Definition 8. The *percolation probability* $\theta(n, r)$ is the probability that the origin $o \in \mathbb{R}^d$ is contained in a connected component of $\mathcal{G}(\mathcal{X}_n \cup \{o\}; r)$ of an infinite number of vertices. That is, if C_o denotes the set of vertices connected to o in the graph, then $\theta(n, r) = \Pr(|C_o| = \infty)$.

We say that a graph *percolates* iff $\theta(n, r) > 0$. Note that the selection of the origin is arbitrary, and the following result can be obtained for any $x \in \mathbb{R}^d$ alternative to o.

Theorem 2. *([11, Theorem 12.35]) There exists a critical radius $r_n^* = \gamma^* n^{-1/d}$, where γ^* is a constant, such that $\theta(n, r_n) = 0$ when $r_n < r_n^*$, and $\theta(n, r_n) > 0$ when $r_n > r_n^*$.*

The following lemma states that the infinite connected component exists with probability strictly 0 or 1. See proof in [30].

Lemma 1. *Let $\psi(n, r)$ be the probability that $\mathcal{G}(\mathcal{X}_n; r)$ contains an infinite connected component, i.e., without conditioning on any specific additional vertex. Then $\psi(n, r) = 0$ when $\theta(n, r) = 0$ and $\psi(n, r) = 1$ when $\theta(n, r) > 0$.*

The following theorem establishes that the infinite connected component is unique.

Theorem 3. *([20, Theorem 2.3]) With probability 1, $\mathcal{G}(\mathcal{X}_n; r)$ contains at most one infinite connected component.*

B. Bounded domains

We study different properties of $\mathcal{G}(\mathcal{X}_n; r)$ when it is restricted to the domain $[0,1]^d$. In case that $\theta(n, r) > 0$, we use C_∞ to refer to the infinite connected component of the unbounded graph $\mathcal{G}(\mathcal{X}_n; r)$. Note that C_∞ exists (Lemma 1) and is unique (Theorem 3) with probability 1.

[5] In particular, for any $d \geqslant 2$, $\gamma^* \in (0.4, 0.6)$ (see [30]).

[6] Let $A_1, A_2, \ldots$ be random variables in some probability space and let B be an event depending on A_n. We say that B occurs *asymptotically almost surely* (a.a.s., in short) if $\lim_{n\to\infty} \Pr[B(A_n)] = 1$.

Denote by C_n the largest connected component of $C_\infty \cap [0,1]^d$. By definition, C_n is also a subgraph of $\mathcal{G}(\mathcal{X}_n \cap [0,1]^d; r_n)$. The following lemma shows that with high probability all the points from C_∞, that are sufficiently close to the center of $[0,1]^d$, are members of C_n.

Lemma 2. *Let $r_n > r_n^*$. Define*

$$H_n = [0,1]^d \setminus \mathcal{B}_{1/\log n}\left(\mathbb{R}^d \setminus [0,1]^d\right).$$

Denote by $\mathfrak{E}_n^1$ the event that $C_\infty \cap H_n \subset C_n$. Then there exist $n_0 \in \mathbb{N}$ and $\alpha > 0$ such that for any $n > n_0$ it holds that

$$\Pr[\mathfrak{E}_n^1] \geqslant 1 - \exp\left(-\alpha n^{1/d} \log^{-1} n\right).$$

Proof: This statement is an adaptation of Lemma 8 and Theorem 2 of [22][7]. Let $C_1', \ldots, C_k'$ denote the connected components of $\mathcal{C}_\infty \cap [0,1]^d$, and set C_n to be the component C_i' with the largest number of vertices. Without loss of generality, $C_n = C_1'$. See illustration in Figure 2.

Observe that if $C_i' \subset C_\infty$ then it must have at least one vertex $x_i' \in C_i'$ that lies closely to the boundary of $[0,1]^d$, i.e., $\|x - \partial([0,1]^d)\| \leqslant r_n$, as otherwise C_i' will not be able to connect to the rest of C_∞. Thus, $\operatorname{diam}(C_i') \leqslant \log^{-1} n - r_n$ is required for C_i' to be able to reach H_n, where $\operatorname{diam}(D) = \sup_{x,x' \in D} \|x - x'\|$ defines the diameter of a given $D \subset \mathbb{R}^d$. We shall show that this does not hold for $i > 1$.

Theorem 2 in [22] states that for any ϕ_n large enough there exist α', n_0 such that for any $n > n_0$ it holds with probability at least $1 - \exp\left(-\alpha' n^{1/d} \phi_n\right)$ that $\operatorname{diam}(C_i') < \phi_n$ for any $1 < i \leqslant k$. By setting $\alpha = \alpha'/2, \phi_n = 1/(2\log n)$ the conclusion immediately follows. ∎

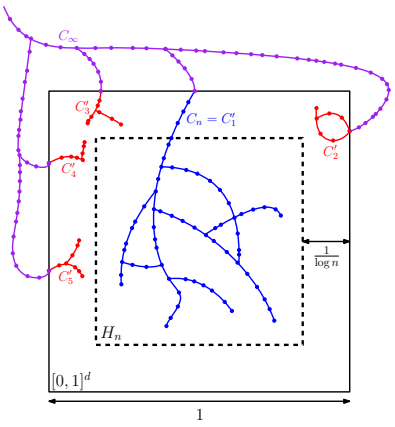


Fig. 2. Illustration for Lemma 2. The outer cube represents $[0,1]^d$, whereas the inner cube depicted with dashed boundary is H_n. Observe that H_n has side length of $1 - 2/\log n$. The purple, blue and red graphs combined describe C_∞, whereas C_n is depicted in blue, and C_2', C_3', C_4', C_5' are in red.

Observe that for any fixed point x internal to $[0,1]^d$ there exists $n_0 \in (0,\infty)$ such that for any $n > n_0$ it holds that $x \in H_n$. The following lemma bounds the probability of having at least one point from C_n in a (small) subregion of H_n. Note that the value β corresponds to the value used in Theorem 1.

Lemma 3. *Let $r_n > r_n^*$. Define $H_n' \subset H_n$ to be a hypercube of side length $h_n' = \frac{\beta \log^{1/(d-1)} n}{n^{1/d}}$. Denote by $\mathfrak{E}_n^2$ the event that $H_n' \cap C_n \neq \emptyset$. Then there exists $n_0 \in \mathbb{N}$ and $\beta_0 > 0$ such that for any $n > n_0, \beta > \beta_0$ it holds that $\Pr[\mathfrak{E}_n^2 | \mathfrak{E}_n^1] \geqslant 1 - n^{-1}$.*

Proof: Define $G_n = \mathcal{G}(\mathcal{X}_n, r_n) \cap H_n'$, $n' = E[|\mathcal{X}_n \cap H_n'|]$ and observe that $n' = n \cdot |H_n'| = \beta^d \log^{d/(d-1)} n$.

We treat G_n as a subset of $\mathbb{R}^d$ in order to apply a rescaling argument. Observe that (scalar) multiplication of every point of H_n' with $1/h_n'$ yields a translation of $[0,1]^d$. We will use the superscript $1/h_n'$ to describe this rescaling to a given object. For instance, applying the same transformation on G_n yields the graph $G_n^{1/h_n'}$, which has the same topology as G_n. Denote by $x_\ell \in H_n'$ the (lexicographically) smallest point of H_n', i.e., $x_\ell = (1/\log n, \ldots, 1/\log n)$. Notice that

$$G_n^{1/h_n'} - x_\ell^{1/h_n'} = \mathcal{G}(\mathcal{X}_n^{1/h_n'} \cap [0,1]^d, r_n/h_n') = \mathcal{G}(\mathcal{X}_{n'} \cap [0,1]^d, r_{n'}),$$

where the minus sign in the left-hand side represents a translation by a vector. This implies that G_n, which is defined over H_n' behaves as $\mathcal{G}(\mathcal{X}_{n'} \cap [0,1]^d; r_{n'})$. This allows to leverage Theorem 1 from [22], which bounds the number of vertices from the unbounded component. In particular, there exists $\beta_0 > 0$ such that

$$\begin{aligned}\Pr[C_\infty \cap H_n' = \Theta(n')] &\geqslant 1 - \exp\left(-\beta_0^{-(d-1)} n'^{\frac{d-1}{d}}\right) \\ &= 1 - \exp\left(-\beta_0^{-(d-1)} \beta^{d-1} \log n\right) \\ &\geqslant 1 - \exp(-\log n) = 1 - n^{-1}.\end{aligned}$$

While this is an overkill for our purpose, it does the job in proving that $\Pr[C_\infty \cap H_n' \neq \emptyset] \geqslant 1 - n^{-1}$. As we assume that $\mathfrak{E}_n^1$ holds (Lemma 2), it follows that $C_n \cap H_n' \neq \emptyset$ holds with probability at least $1 - n^{-1}$. ∎

The following statement allows to bound the graph distance between two connected vertices. We endow every edge of the graph with a length attribute that represents the Euclidean distance between the edges' endpoints. For every two vertices x, x' of $\mathcal{G}$, $\operatorname{dist}(\mathcal{G}, x, x')$ denotes the length of the shortest (weighted) path on $\mathcal{G}$ between the two vertices.

Theorem 4. *([9, Theorem 3]) Let $r_n r_n^*$. There exists a constant $\xi \geqslant 1$, independent of n, such that $\Pr[\mathfrak{E}_n^3] = 1 - O(n^{-1})$, where the event $\mathfrak{E}_n^3$ is defined as follows: For any two vertices x, x' in the same connected component of $\mathcal{G}(\mathcal{X}_n \cap [0,1]^d; r_n)$, with $\|x - x'\| = \omega(r_n)$, it holds that $\operatorname{dist}(\mathcal{G}_n, x, x') \leqslant \xi \|x - x'\|$.*

VI. Proof of Theorem 1

We provide a full proof for the positive setting with length cost. The proof for the bottleneck case is very similar to the length case, and it is therefore deferred to the extended version [30]. The incompleteness proof for $r_n < r_n^*$ is presented there as well.

Suppose that $r_n > r_n^*, r_n^{st} = \frac{\beta \log^{1/(d-1)} n}{n^{1/d}}$, $\beta > \beta_0$. For simplicity, we set $r_n = \gamma n^{-1/d}$, where $\gamma > \gamma^*$. By Lemma 1

[7] We mention that [22] uses a slightly different, but nevertheless equivalent model. While we consider an RGG that is bounded to $[0,1]^d$ and PPP of density n, they consider the domain $[0,n]^d$ with density 1. It is only a matter of rescaling and variable substitution to import their results to our domain.

and Theorem 3, $\mathcal{G}(\mathcal{X}_n; r_n)$ contains a unique infinite connected component C_∞. Recall that C_n denotes the largest connected component of $C_\infty \cap [0,1]^d$. Also note that $r_n^{st} = h'_n$, where h'_n is defined in Lemma 3.

Recall that c_ℓ^* denotes the robust optimum, with respect to path length (Definition 2). Fix $\varepsilon > 0$. By definition, there exists a robust path $\pi_\varepsilon \in \Pi_{s,t}^{\mathcal{F}}$ and $\delta > 0$ such that $c_\ell(\pi_\varepsilon) \leqslant (1+\varepsilon)c_\ell^*$ and $\mathcal{B}_\delta(\pi_\varepsilon) \subset \mathcal{F}$. See illustration in Figure 3.

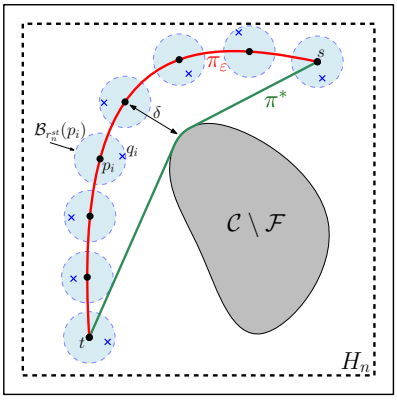


Fig. 3. Illustration for the proof of Theorem 1.ii. The outer cube represents the configuration space, whereas the dashed cube is H_n. The gray area represents the forbidden regions. The robust feasible path π^* and π_ε are depicted in green and red, respectively. Observe that every point along π_ε is at least δ away from $\mathcal{C} \setminus \mathcal{F}$. $p_1 = s, p_2, \ldots, p_{k-1}, p_k = t$ are depicted as black bullets, where $k = 8$. $\mathcal{B}_{r_n^{st}}(p_i)$ are depicted as blue circles, while the blue cross in each such circle represents q_i.

We now define a sequence of k points $p_1, \ldots, p_k$ along π_ε that are separated by exactly $\delta/2\xi$ units, where ξ is as defined in Theorem 4. In particular, define $k = \lceil c_\ell(\pi_\varepsilon) \cdot 2\xi/\delta \rceil$, set $p_1 = s, p_k = t$, and assign p_i along π_ε, such that $c_\ell\left(\pi_\varepsilon^{i-1,i}\right) = \delta/2\xi$, where $\pi_\varepsilon^{i-1,i}$ represents the subpath of π_ε starting at p_{i-1} and ending at p_i. Notice that k is finite.

Claim 1. *Denote by $\mathfrak{E}_n^4$ the event that for all $i \in [k]$ there exists $q_i \in C_n$ such that $q_i \in \mathcal{B}_{r_n^{st}}(p_i)$ and $q_i \in C_n$. Then $\Pr[\mathfrak{E}_n^4 | \mathfrak{E}_n^1] \geqslant 1 - k \Pr[\overline{\mathfrak{E}_n^2} | \mathfrak{E}_n^1]$ (see definition of $\mathfrak{E}_n^1, \mathfrak{E}_n^2$ in Lemma 2 and Lemma 3, respectively).*

Proof: Define $H'_n(x) \subset \mathbb{R}^d$ to represent a d-dimensional (axis-aligned) hypercube of side length h'_n that is centered in $x \in \mathbb{R}^d$. Formally, $H'_n(x) = x + h'_n \cdot \left[-\frac{1}{2}, \frac{1}{2}\right]^d$. Observe that $H'_n(p_i) \subset H_n$ for n large enough. Also note that $H'_n(p_i) \subset \mathcal{B}_{r_n^{st}}(p_i)$. Thus, the result follows from the union bound. ∎

Suppose that $\mathfrak{E}_n^1, \mathfrak{E}_n^4$ are satisfied. Let $q_1, \ldots, q_k \in C_n$ be the points obtained from Claim 1. These points reside in a single connected component of $\mathcal{G}_n$. Define the path

$$\pi_n := s \to q_1 \rightsquigarrow q_2 \rightsquigarrow \ldots \rightsquigarrow q_k \to t,$$

where $s \to q_1$ represents a straight-line path from s to q_1, and $q_i \rightsquigarrow q_{i+1}$ represents the shortest path from q_i to q_{i+1} in $\mathcal{G}_n$. The next claim states that if in addition $\mathfrak{E}_n^3$ is satisfied (Theorem 4), then π_n is also collision free.

Claim 2. *Suppose that $\mathfrak{E}_n^1, \mathfrak{E}_n^3, \mathfrak{E}_n^4$ are satisfied. Then $\pi_n \in \Pi_{s,t}^{\mathcal{F}}$ is a path in $\mathcal{P}_n$ (with probability 1).*

Proof: First, observe that the straight-line paths $s \to q_1, q_k \to t$ are contained in $\mathcal{P}_n$ due to the definition of PRM and the fact that $r_n^{st} < \delta$. Let us consider a specific subpath $q_i \rightsquigarrow q_{i+1}$. Recall from Claim 1 and by definition of p_i, p_{i+1} that $\text{dist}(\mathcal{G}_n, q_i, q_{i+1}) \leqslant 2r_n^{st} + \xi\|p_{i+1} - p_i\| = o(1) + \delta/2$, which is bounded by δ. Thus, for any point q'_i along $q_i \rightsquigarrow q_{i+1}$ it holds that $\|q'_i - p_i\| < \delta, \|q'_i - p_{i+1}\| < \delta$. Thus, $\text{Im}(\pi_n) \subset \mathcal{B}_\delta(\pi_\varepsilon) \subset \mathcal{F}$. ∎

The next claim states that the length of π_n is a constant factor from the optimum.

Claim 3. *Suppose that $\mathfrak{E}_n^1, \mathfrak{E}_n^3, \mathfrak{E}_n^4$ are satisfied. We have that $c_\ell(\pi_n) \leqslant (1+\varepsilon)\xi c_\ell^*$ (with probability 1).*

Proof: By Theorem 4 and the triangle inequality, it follows that

$$\begin{aligned}
c_\ell(\pi_n) &= \|s - q_1\| + \text{dist}(\mathcal{G}_n, q_1, q_k) + \|t - q_k\| \\
&\leqslant 2r_n^{st} + \sum_{i=2}^{k} \text{dist}(\mathcal{G}_n, q_{i-1}, q_i) \leqslant o(1) + \sum_{i=2}^{k} \xi\|q_{i-1} - q_i\| \\
&\leqslant o(1) + \xi \sum_{i=2}^{k} (\|q_{i-1} - p_{i-1}\| + \|p_i - p_{i-1}\| + \|q_i - p_i\|) \\
&\leqslant o(1) + \xi \sum_{i=2}^{k} c_\ell(\pi_\varepsilon^{i-1,i}) \leqslant o(1) + (1+\varepsilon)\xi c_\ell^*.
\end{aligned}$$

∎

To conclude, Claim 2 and Claim 3 show the existence of a (collision free) path π_n in $\mathcal{P}_n$, whose length is at most $(1+\varepsilon)\xi c_\ell^*$. It remains to bound the probability that $\mathfrak{E}_n^1, \mathfrak{E}_n^3, \mathfrak{E}_n^4$ hold simultaneously:

$$\begin{aligned}
\Pr[\mathfrak{E}_n^1 \wedge \mathfrak{E}_n^3 \wedge \mathfrak{E}_n^4] &= \Pr\left[\mathfrak{E}_n^3 \wedge \mathfrak{E}_n^4 | \mathfrak{E}_n^1\right] \cdot \Pr[\mathfrak{E}_n^1] \\
&= \left(1 - \Pr\left[\overline{\mathfrak{E}_n^3 \wedge \mathfrak{E}_n^4} | \mathfrak{E}_n^1\right]\right) \cdot \Pr[\mathfrak{E}_n^1] \\
&= \left(1 - \Pr\left[\overline{\mathfrak{E}_n^3} \vee \overline{\mathfrak{E}_n^4} | \mathfrak{E}_n^1\right]\right) \cdot \Pr[\mathfrak{E}_n^1] \\
&\geqslant \left(1 - \Pr\left[\overline{\mathfrak{E}_n^3} | \mathfrak{E}_n^1\right] - \Pr\left[\overline{\mathfrak{E}_n^4} | \mathfrak{E}_n^1\right]\right) \cdot \Pr[\mathfrak{E}_n^1]
\end{aligned}$$

By Claim 1 and Lemma 3, $\Pr[\overline{\mathfrak{E}_n^4} | \mathfrak{E}_n^1] \leqslant k \Pr\left[\overline{\mathfrak{E}_n^2} | \mathfrak{E}_n^1\right] \leqslant kn^{-1}$. Also note that $\Pr[\mathfrak{E}_n^3 | \mathfrak{E}_n^1] \geqslant \Pr[\mathfrak{E}_n^3]$, since $\Pr[\mathfrak{E}_n^3 | \overline{\mathfrak{E}_n^1}] = 0$. Therefore,

$$\begin{aligned}
\Pr[\mathfrak{E}_n^1 \wedge \mathfrak{E}_n^3 \wedge \mathfrak{E}_n^4] &\geqslant \left(1 - \Pr\left[\overline{\mathfrak{E}_n^3} | \mathfrak{E}_n^1\right] - \Pr\left[\overline{\mathfrak{E}_n^4} | \mathfrak{E}_n^1\right]\right) \cdot \Pr[\mathfrak{E}_n^1] \\
&\geqslant \left(1 - O(n^{-1}) - kn^{-1}\right)\left(1 - \exp\left(-\alpha n^{1/d} \log^{-1} n\right)\right) \\
&= 1 - O(n^{-1}).
\end{aligned}$$

VII. Experimental results

We present experiments demonstrating the effect of using different values of the connection radius r_n on the running time, cost of solution for the length cost c_ℓ, and success rate, when running the algorithms PRM and FMT* on problems of dimensions up to 12. In particular, r_n ranges between the critical radius (Theorem 1) and previously obtained upper bounds from [14, 15].

We validate our theory for PRM (Theorem1) and FMT* (See [30]) for the path-length cost c_ℓ. We observe that smaller connection radii, than previously-obtained bounds, still allow the planners to converge to high-quality, near-optimal paths. Furthermore, we identify situations in which using a radius that is close to r_n^* allows to obtain a high-quality solution more quickly. Moreover, although the resulting cost for the smaller radii can be slightly worse, we observe that postprocessing the paths using standard simplification methods yields solutions that are only marginally inferior to the best (postprocessed) solution.Specifically, in harder scenarios the advantage in using a smaller connection radius is more prominent; in some cases we obtain a reduction of 50% in running time, with an improved cost, and similar success rate when compared to the results obtained using the original FMT* connection radius.

A. Implementation details

In our experiments, we used the Open Motion Planning Library (OMPL 1.3) [34] on a 2.6GHz×2 Intel Core i5 processor with 16GB of memory. Result were averaged over 50 runs and computed for dimensions up to 12.

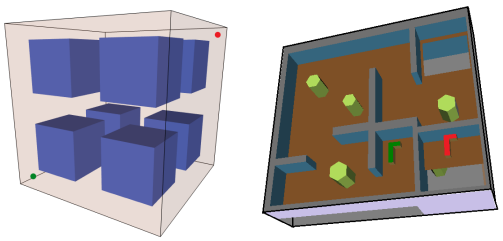
The planners that we used are PRM and the batch variant of FMT*, which were adapted for samples from a PPP, where n is the expected number of samples. The two planners use the connection radii r_n, r_n^{st}. Note that r_n^{st} should be at least $\frac{\beta \log^{1/(d-1)} n}{n^{1/d}}$, but the exact value of β is unknown. For simplicity, we set r_n^{st} to be identical to $r_{\texttt{PRM}^*}$, defined in [15]. We emphasize that although we use an asymptotically smaller value, it still yields (empirically) convergence in cost. This suggests that the bound on r_n^{st} can be further reduced.

Given a scenario and a value n, we define a set of $k+1$ increasing connection radii, $\{r_0, \ldots, r_k\}$, as follows. We set the minimal connection radius to be $r_0 = \gamma n^{-1/d}$, where $\gamma = 1$. Note that γ is larger than γ^* by a factor of roughly 2. The maximal connection radius, denoted by $r_k = r_{\texttt{FMT}^*}$, is as defined in [14]. For each $1 \leqslant i \leqslant k-1$ we define $r_i = r_0 + i \cdot \Delta$, where $\Delta = (r_k - r_0)/k$. Now, for every scenario and number of samples n we run our planning algorithm with r_n^{st}, and $r_n \in \{r_0, \ldots, r_k\}$. Note that all our plots are for $k = 10$, and that in some experiments an additional radius $r_{k+1} = r_{\texttt{PRM}^*} > r_{\texttt{FMT}^*}$ appears as well (see [15]). The figure to the right depicts the colors and labeling that will be used throughout this section.

B. Results

Euclidean space. The scenario we consider consists of a point robot moving in the obstacle-free unit d-dimensional hypercube. Therefore, $\mathcal{F} = \mathcal{C} = [0,1]^d$. We set the start and target positions of the robot to be $s = (0.1, \ldots, 0.1)$ and $t = (0.9, \ldots, 0.9)$, respectively.

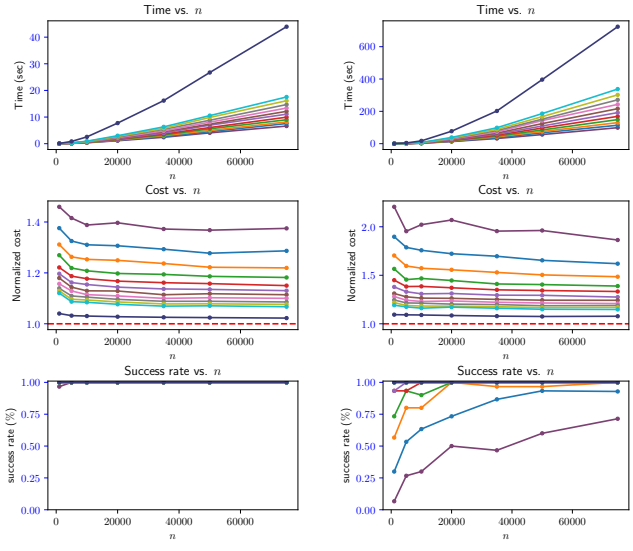
We run PRM and plot (i) the overall running time, (ii) the normalized cost (c_ℓ) of the obtained solution, where a value of 1 represents the best possible cost, and (iii) the portion of successful runs—all as a function of the expected number of samples n. The results for dimensions 4 and 8 are depicted in Figure 5a and Figure 5b, respectively. See [30] for plots for $d = 12$.



(a) (b)

Fig. 4. Two of the scenarios used in the experiments. (a) dD Hypercube with 2^d hypercubical obstacles, and (b) 3D Cubicles. Start and target configurations for a robot are depicted in green and red, respectively. Scenario (b) is provided with the OMPL distribution.

The plots demonstrate the following trend: for each radius r the cost obtained by PRM converges to some constant[8] times the optimal cost, which is marked with the dashed red curve in the "Cost vs. n" plot. Clearly, $r_{\texttt{PRM}^*}$ yields the best cost but at the price of increased running time. $r_{10} = r_{\texttt{FMT}^*}$ obtains the next best cost, with improved running time, and so on. Note that for $d = 4$, already for $n = 1$K a solution is found for all radii except r_0, whereas for $d = 8$ and $n = 5$K a solution is found for all radii above r_4. It is important to note that there is a clear speedup in the running times of PRM when using r_i for $i \leqslant 9$, over $r_{\texttt{FMT}^*}, r_{\texttt{PRM}^*}$, with a slight penalty (a factor of roughly 2) in the resulting costs. In [30] we also study how r_i affects the size of C_n.



(a) (b)

Fig. 5. PRM in the (a) 4D and (b) 8D Hypercube.

General Euclidean space. We consider the following d-

[8]We note that it is possible that for larger values of n the cost values for different radii will eventually converge to the same value.

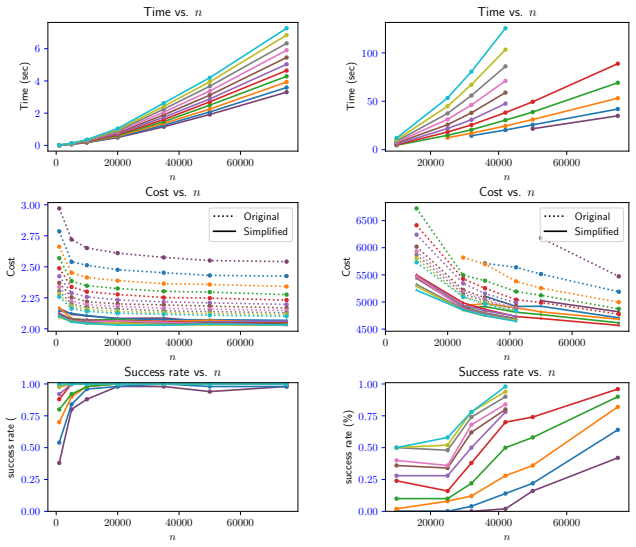
dimensional scenario (based on a scenario from [31]), for $d \in \{4, 8\}$, depicted in Figure 4a in 3D, and a point robot: $\mathcal{C} = [0, 1]^d$ is subdivided into 2^d sub-cubes by halving it along each axis. Each sub-cube contains a centered d-dimensional axis-aligned hypercubical obstacle that covers 25% of the sub-cube. The start position s of the robot is placed on the diagonal between $(0, \ldots, 0)$ and $(1, \ldots, 1)$, such that it is equidistant from the origin and from the closest hypercubical obstacle. t is selected similarly, with respect to $(1, \ldots, 1)$.

Here we use FMT* with radii ranging from r_0 to r_{10}. We also ran PRM which exhibited a similar behavior. Figure 6a presents the results for $d = 4$. We plot the average cost after simplifying the resulting paths in addition to the average original cost. We mention that there is a difference in cost between the various radii and that costs obtained using larger radii are often better. However, after applying OMPL's default path-simplification procedure we obtain paths with negligible differences in cost. This suggests that paths obtained using the smaller connection radii are of the same homotopy class as the path obtained using $r_{\texttt{FMT}^*}$. For $d = 8$ we obtain similar results. However, the success rates deteriorate as the dimension increases (see extended version [30]).

General non-Euclidean space. We use the Cubicles scenario (see Figure 4b) provided with the OMPL distribution [34]. Here the goal is to find a collision-free path for two L-shaped robots that need to exchange their positions (in green and red). Since the robots are allowed to translate and rotate, then $d = 12$ and $\mathcal{C}$ is non-Euclidean. Although our theory does directly apply here, we chose to test it empirically for such a setting. A similar experiment involving only one L-shaped robot is discussed in [30].

We ran our experiments with FMT* using a connection radii ranging from r_0 to $r_{10} = r_{\texttt{FMT}^*}$. Initially, no solution was found in all runs. Indeed, as was mentioned in [17], this is not surprising, since the theory from which the radius values are derived assumes that the configuration space is Euclidean. In the same paper the authors propose a heuristic for effectively using radius-based connections in motion planning. In our experiments (Figure 6b) we increased all radii by a multiplicative factor of 10 in order to increase the success rates. We mention that this yielded similar behavior to that when using the connection scheme suggested in [17].

Observe in Figure 6b, that the variance in cost after path simplification is again significantly smaller than the variance in the original cost. Clearly, smaller radii exhibit shorter running times, but with smaller success rates. However, since the success rate improves as the number of samples n increases, one could use the smaller radii with larger n and still obtain a solution with comparable cost and success rate in shorter running time. Indeed, using r_3 (green) with 75K samples we obtain a solution whose cost (after simplification) is slightly better than the cost obtained using r_{10} with 42K samples (after simplification). Moreover, the running time of the former is roughly 50% of the time taken for the latter, and both obtain similar success rates. This indicates that in such settings, one could benefit from using smaller radii in terms of favorable running times and obtain a comparable or even better cost with similar success rates.



(a) (b)

Fig. 6. (a) FMT* for a point robot in the 4D hypercube with obstacles scenario (Figure 4a) (b) FMT* for two rigid-body robots moving in OMPL's Cubicles scenario (Figure 4b). Both the average original cost and the average cost after simplification are presented for each radius. There is a difference in cost between the various radii (dashed lines), that diminishes after simplifying the resulting paths (solid lines).

VIII. Future Work

In this work we leveraged techniques from percolation theory to develop stronger analysis of PRM-based sampling-based motion planners. In the hope of providing mathematically rigorous presentation, while still being accessible, we chose to focus the discussion on simplified, possibly unrealistic, robotic systems. In particular, we assumed a holonomic system having a Euclidean configuration space.

Our immediate future goal is to extend our theory to non-Euclidean spaces, such as those arising from rigid bodies translating and rotating in space. The next challenge will be to extend the model to robots with differential constraints. While the latter task seems daunting, we should keep in mind that the state space of such systems can be modeled as a differential manifold. This may allow to locally apply our Euclidean-space techniques to analyze manifold spaces. Indeed, a similar approach has already been considered in previous work [26, 27].

The next algorithmic challenge is to consider robotic systems for which precise *steering*, i.e., solving the *two-point boundary value problem*, cannot be performed, at least not efficiently (see discussion in [19, Section 1.3]). In such cases, PRM-based planners (as we considered here), or RRT*-based techniques, cannot be applied. We pose the following question: Is it possible to extend existing planners to work in the absence of a steering function, while maintaining their theoretical properties? We plan to tackle this question in the near future.

References

[1] Ron Alterovitz, Sachin Patil, and Anna Derbakova. Rapidly-exploring roadmaps: Weighing exploration vs. refinement in optimal motion planning. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3706–3712, 2011.

[2] Oktay Arslan and Panagiotis Tsiotras. Use of relaxation methods in sampling-based algorithms for optimal motion planning. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2413–2420, 2013.

[3] Cenk Baykal and Ron Alterovitz. Asymptotically optimal design of piecewise cylindrical robots using motion planning. In *Robotics: Science and Systems*, 2017.

[4] Andrew Dobson and Kostas E. Bekris. Sparse roadmap spanners for asymptotically near-optimal motion planning. *International Journal of Robotics Research*, 33(1): 18–47, 2014.

[5] Andrew Dobson, George V. Moustakides, and Kostas E. Bekris. Geometric probability results for bounding path quality in sampling-based roadmaps after finite computation. In *IEEE International Conference on Robotics and Automation*, pages 4180–4186, 2015.

[6] Andrew Dobson, Kiril Solovey, Rahul Shome, Dan Halperin, and Kostas E. Bekris. Scalable asymptotically-optimal multi-robot motion planning. In *International Symposium on Multi-Robot and Multi-Agent Systems*, 2017.

[7] Massimo Franceschetti and Ronald Meester. *Random Networks for Communication: From Statistical Physics to Information Systems*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 2008.

[8] Emilio Frazzoli and Marco Pavone. Multi-vehicle routing. In *Encyclopedia of Systems and Control*. 2015.

[9] Tobias Friedrich, Thomas Sauerwald, and Alexandre Stauffer. Diameter and broadcast time of random geometric graphs in arbitrary dimensions. *Algorithmica*, 67(1):65–88, 2013.

[10] Jonathan D. Gammell, Siddhartha S. Srinivasa, and Timothy D. Barfoot. Batch informed trees (BIT*): Sampling-based optimal planning via the heuristically guided search of implicit random geometric graphs. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3067–3074, 2015.

[11] Geoffrey R. Grimmett. *Percolation*, volume 321 of *Comprehesive Studies in Mathematics*. Springer, 1999.

[12] Dan Halperin, Oren Salzman, and Micha Sharir. Algorithmic motion planning. In Jacob E. Goodman, Joseph O'Rourke, and Csaba D. Toth, editors, *Handbook of Discrete and Computational Geometry*, chapter 50. CRC Press LLC, 3rd edition, 2016. URL http://www.csun.edu/~ctoth/Handbook/HDCG3.html.

[13] Rachel M. Holladay, Oren Salzman, and Siddhartha Srinivasa. Minimizing task space frechet error via efficient incremental graph search. *CoRR*, abs/1710.06738, 2017. URL http://arxiv.org/abs/1710.06738.

[14] Lucas Janson, Edward Schmerling, Ashley A. Clark, and Marco Pavone. Fast marching tree: A fast marching sampling-based method for optimal motion planning in many dimensions. *International Journal of Robotics Research*, 34(7):883–921, 2015.

[15] Sertac Karaman and Emillio Frazzoli. Sampling-based algorithms for optimal motion planning. *International Journal of Robotics Research*, 30(7):846–894, 2011.

[16] Lydia E. Kavraki, Petr Svestka, Jean-Claude Latombe, and Mark Overmars. Probabilistic roadmaps for path planning in high dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12(4):566–580, 1996.

[17] Michal Kleinbort, Oren Salzman, and Dan Halperin. Collision detection or nearest-neighbor search? On the computational bottleneck in sampling-based motion planning. In *International Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2016.

[18] Steven M. LaValle and James J. Kuffner Jr. Randomized kinodynamic planning. *I. J. Robotics Res.*, 20(5):378–400, 2001.

[19] Yanbo Li, Zakary Littlefield, and Kostas E. Bekris. Asymptotically optimal sampling-based kinodynamic planning. *I. J. Robotics Res.*, 35(5):528–564, 2016.

[20] Ronald Meester and Rahul Roy. Uniqueness of unbounded occupied and vacant components in boolean models. *The Annals of Applied Probability*, 4(3):933–951, 1994.

[21] Mathew D. Penrose. *Random geometric graphs*. Oxford University Press, 2003.

[22] Mathew D. Penrose and Agoston Pisztora. Large deviations for discrete and continuous percolation. *Advances in Applied Probability*, 28(1):2952, 1996.

[23] Oren Salzman and Dan Halperin. Asymptotically-optimal motion planning using lower bounds on cost. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4167–4172, 2015.

[24] Oren Salzman and Dan Halperin. Asymptotically near-optimal RRT for fast, high-quality motion planning. *IEEE Trans. Robotics*, 32(3):473–483, 2016.

[25] Oren Salzman, Doron Shaharabani, Pankaj K. Agarwal, and Dan Halperin. Sparsification of motion-planning roadmaps by edge contraction. *I. J. Robotics Res.*, 33 (14):1711–1725, 2014.

[26] Edward Schmerling, Lucas Janson, and Marco Pavone. Optimal sampling-based motion planning under differential constraints: The drift case with linear affine dynamics. In *IEEE Conference on Decision and Control*, pages 2574–2581, 2015.

[27] Edward Schmerling, Lucas Janson, and Marco Pavone. Optimal sampling-based motion planning under differential constraints: The driftless case. In *IEEE International Conference on Robotics and Automation*, pages 2368–2375, 2015.

[28] Kiril Solovey and Dan Halperin. Sampling-based bottleneck pathfinding with applications to fréchet matching. In *European Symposium on Algorithms*, pages 76:1–76:16, 2016.

[29] Kiril Solovey and Dan Halperin. Efficient sampling-based bottleneck pathfinding over cost maps. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2017. to appear.

[30] Kiril Solovey and Michal Kleinbort. The critical radius in sampling-based motion planning. *CoRR*, abs/1709.06290, 2017. URL http://arxiv.org/abs/1709.06290.

[31] Kiril Solovey, Oren Salzman, and Dan Halperin. New perspective on sampling-based motion planning via random geometric graphs. In *Robotics: Science and Systems*, 2016.

[32] J. A. Starek, E. Schmerling, G. D. Maher, B. W. Barbee, and M. Pavone. Real-time, propellant-optimized spacecraft motion planning under Clohessy-Wiltshire-Hill dynamics. In *IEEE Aerospace Conference*, Big Sky, Montana, 2016.

[33] Joseph A. Starek, Javier V. Gómez, Edward Schmerling, Lucas Janson, Luis Moreno, and Marco Pavone. An asymptotically-optimal sampling-based algorithm for Bi-directional motion planning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2072–2078, 2015.

[34] Ioan A. Şucan, Mark Moll, and Lydia E. Kavraki. The Open Motion Planning Library. *IEEE Robotics & Automation Magazine*, 19(4):72–82, 2012. http://ompl.kavrakilab.org.

[35] Glenn Wagner and Howie Choset. Subdimensional expansion for multirobot path planning. *Artif. Intell.*, 219:1–24, 2015.

10

Conclusion and Future Work

This thesis has presented recent progress in motion planning, with an emphasis on multi-robot systems. We have studied theoretical aspects of the problem, including hardness results and efficient polynomial-time algorithms. We have also considered sampling-based techniques, which typically can be more easily adapted and applied to complex robotic systems, and studied the theoretical properties of such algorithms.

However, motion planning is far from being completely solved, and many real-life problems are still tackled today using heuristics that present no theoretical guarantees. This is not surprising considering that existing techniques, including the work presented in this thesis, typically make simplifying assumptions about the robot and the world it operates in. For instance, a geometric (and unrealistic) simplification of the robot system is often assumed, where the robot does not have to abide by physical constraints, such as velocity limits, friction, and bounded curvature of their trajectory. Additionally, in many situations a static and a priori known environment is often assumed, whereas in reality the robots have only a limited knowledge of their surrounding, which can also change with time. Furthermore, most techniques suffer from poor running times for higher-dimensional settings, e.g., when the number of DoFs is larger than ten.

The text below presents some ideas on how to overcome these limitations and identifies open problems for future research.

10.1 Multi-robot motion planning

We have mentioned three algorithms [1, 90, 98] for MRMP that have polynomial running time. Those approaches require some "clearance" around the start and target positions in order to guarantee their correctness. Perhaps the most limiting and unnatural aspect of such assumptions is the requirement to separate start positions from target positions, even though they need not be occupied simultaneously by the robots. Such assumptions limit the applicability of those techniques to scenarios that have plenty of free space for the robots to make their maneuvers. Thus, an immediate question is whether such assumptions can be relaxed to allow a broader range of problems to be solved efficiently. Additionally, the aforementioned techniques deal exclusively with the unlabeled setting. Would it be possible to come up with similar techniques for the labeled case?

From a theoretical side, a fundamental base case of MRMP has not be sufficiently researched and a deeper understanding of it may lead to insights for the more general case. Suppose that our problem consists of only two moving robots, and moreover, let us assume that they are equal-radius discs operating in a workspace with polygonal obstacles. The best known complete algorithm for this problem, which was developed by Sharir and Sifrony [80], runs in time $O(n^2)$. It should be noted that the algorithm is rather straightforward, in terms of the algorithmic tools that are employed. This begs the question of whether a more efficient algorithm can be developed, which would possibly exploit the algorithmic progress that has been made in the last 25 years since the publication of the aforementioned paper. Alternatively, would it be possible to come up with a lower bound affirming that the above result is tight? Here recent progress on conditional lower bounds (see, e.g., [10]) may come in handy.

Another interesting and underresearched aspect of this problem is optimal motion, e.g., in terms of the total length of the two robot paths. Is it possible to come up with a polynomial-time algorithm that returns an optimal or near-optimal solution for two discs amid polygonal obstacles? We note that there is some evidence for this problem being computationally hard. First, it requires reasoning about the optimal solution in a four-dimensional space, whereas finding the shortest path for a point robot in a three-dimensional Euclidean space is NP-hard [14]. Secondly, it was recently shown that optimal solutions for the two-disc problem, induce paths that have bounded curvature [43]. However, finding bounded-curvature paths for a single point robot in the plane amid obstacles is NP-hard [42] as well.

10.2 Sampling-based motion planning

Rigorous analysis of sampling-based motion planners is quite challenging, even for simple holonomic robots. Consequently, in order to make the analysis more manageable, most theoretical results assume a Euclidean configuration space. Our immediate future goal is to extend existing algorithms and their corresponding analyses to non-Euclidean spaces, such as those arising from rigid bodies translating and rotating in space. A possible line of attack can be embedding such spaces into Euclidean space, although this might require drastically increasing the dimension.

A related challenge is to incorporate in the analysis the differential constraints of the robot. While this task seems daunting, we should keep in mind that the state space of such systems can be modeled as

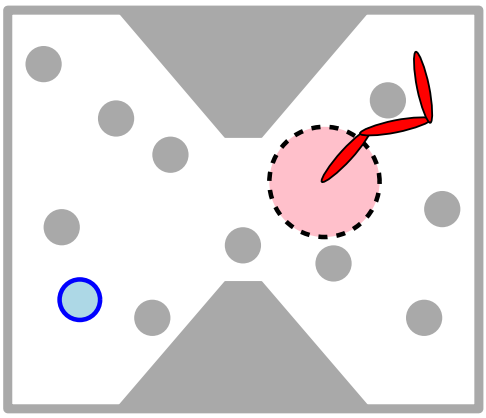


Figure 10.1: Motion planning with limited sensing.

a differential manifold. This may allow to locally apply our recently-developed theory for the Euclidean spaces [87] to analyze manifold spaces. Indeed, a similar approach has already been considered in the work of Schmerling et al. [76, 77]. We plan to similarly extend our result [87], which may yield a stronger analysis than the last two papers.

The next algorithmic challenge is to consider robotic systems for which precise *steering*, i.e., solving the *two-point boundary value problem*, cannot be performed, at least not efficiently (see discussion in Li et al. [56, Section 1.3]). In such cases, PRM-based planners, or RRT*-based techniques, cannot be applied as-is. We pose the following question: Is it possible to extend existing planners to work in the absence of a steering function, while maintaining their theoretical properties? We plan to tackle this question in the near future.

The next problem we pose is concerned with the underlying sampling technique used in current planners. Traditionally, configuration samples are generated in a randomized manner. Several works (see, e.g., [53, 109]) have considered the natural alternative of deterministic sampling, e.g., lattice points. They studied the theoretical properties of planners constructed using such sampling schemes, in terms of convergence to a feasible solution, and tested their performance in experiments. Interestingly, deterministic samples have yielded superior performance over the randomized case in some settings [53], by allowing to find an initial solution more quickly, in terms of the number of samples required. Janson et al. [36] have recently studied the optimality guarantees of such sampling techniques, which were shown to require fewer connections than randomized samples. Our recent paper [87] has also considered deterministic samples, which are then sparsified in a randomized fashion, which result in even smaller roadmaps. Those two papers suggest that deterministic samples can lead to a simpler optimality analysis of the underlying planner, and may even allow reduction in running time. However, a thorough experimental study of convergence to optimum for deterministic samples has not been conducted yet, and thus it is not clear whether randomized approaches are inferior when the cost of the solution is taken into consideration.

Perhaps the hardest task of all is planning in unknown environments. Consider the example depicted

in Figure 10.1, which consists of a snake-like robot that needs to reach the blue region while avoiding collisions with obstacles (depicted in gray). As it often occurs in practice, the robot starts its mission in an environment whose structure is unknown in advance. The robot is equipped with a limited-range sensor (depicted as the pink circle around the snake's head) using which it can gather information about its surroundings. The goal is to come up with an exploration strategy that minimizes the time necessary for the robot to arrive to the target. Note that in addition to the typical challenges encountered in motion planning, including the complex geometry of the free space, the planner also needs to cope with uncertainty in order to decide on the robot's next move. Some special cases for planar robots have been studied recently [24, 81], but to the best of our knowledge no general technique exists, at least not one that has theoretical guarantees. We thus plan to tackle this problem in a sampling-based setting, by leveraging techniques for exploration of discrete unknown graphs (see, e.g., [21]) over sampled roadmaps, which would serve as a discretization of the configuration space.

11 Bibliography

[1] A. Adler, M. de Berg, D. Halperin, and K. Solovey. Efficient multi-robot motion planning for unlabeled discs in simple polygons. *IEEE T. Automation Science and Engineering*, 12(4):1309–1317, 2015.

[2] H. Alt and M. Godau. Computing the Fréchet distance between two polygonal curves. *Int. J. Comput. Geometry Appl.*, 5:75–91, 1995.

[3] R. Alterovitz, S. Patil, and A. Derbakova. Rapidly-exploring roadmaps: Weighing exploration vs. refinement in optimal motion planning. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3706–3712, 2011.

[4] N. M. Amato, O. B. Bayazit, L. K. Dale, C. Jones, and D. Vallejo. OBPRM: A obstacle-based PRM for 3D workspaces. In *International Workshop on the Algorithmic Foundations of Robotics (WAFR)*, pages 155–168, 1998.

[5] B. Aronov, M. de Berg, A. F. van der Stappen, P. Švestka, and J. Vleugels. Motion planning for multiple robots. *Discrete & Computational Geometry*, 22(4):505–525, 1999.

[6] O. Arslan and P. Tsiotras. Use of relaxation methods in sampling-based algorithms for optimal motion planning. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2413–2420, 2013.

[7] A. Atias, K. Solovey, and D. Halperin. Effective metrics for multi-robot motion-planning. In *Proceedings of Robotics: Science and Systems*, Cambridge, Massachusetts, July 2017.

[8] V. Auletta, A. Monti, M. Parente, and P. Persiano. A linear time algorithm for the feasibility of pebble motion on trees. In *Scandinavian Symposium and Workshops on Algorithm Theory (SWAT)*, pages 259–270, 1996.

[9] C. Baykal and R. Alterovitz. Asymptotically optimal design of piecewise cylindrical robots using motion planning. In *Robotics: Science and Systems*, 2017.

[10] K. Bringmann. Why walking the dog takes time: Frechet distance has no strongly subquadratic algorithms unless SETH fails. In *IEEE Annual Symposium on Foundations of Computer Science*, pages 661–670, 2014.

[11] S. Cafieri and N. Durand. Aircraft deconfliction with speed regulation: new models from mixed-integer optimization. *Journal of Global Optimization*, 58(4):613–629, 2014.

[12] G. Calinescu, A. Dumitrescu, and J. Pach. Reconfigurations in graphs and grids. *SIAM Journal on Discrete*

Mathematics, 22(1):124–138, 2008.

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

[14] J. Canny and J. H. Reif. New lower bound techniques for robot motion planning problems. In *Foundations of Computer Science*, pages 49–60, 1987.

[15] P. Dasler and D. M. Mount. On the complexity of an unregulated traffic crossing. In *WADS*, pages 224–235, 2015.

[16] A. Dobson and K. E. Bekris. Sparse roadmap spanners for asymptotically near-optimal motion planning. *International Journal of Robotics Research*, 33(1):18–47, 2014.

[17] A. Dobson, G. V. Moustakides, and K. E. Bekris. Geometric probability results for bounding path quality in sampling-based roadmaps after finite computation. In *IEEE International Conference on Robotics and Automation*, pages 4180–4186, 2015.

[18] A. Dobson, K. Solovey, R. Shome, D. Halperin, and K. E. Bekris. Scalable asymptotically-optimal multi-robot motion planning. In *IEEE International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*, pages 120–127, 2017.

[19] M. A. Erdmann and T. Lozano-Pérez. On multiple moving objects. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1419–1424, 1986.

[20] G. W. Flake and E. B. Baum. Rush hour is PSPACE-complete, or "why you should generously tip parking lot attendants". *Theorertical Computer Science*, 270(1-2):895–911, 2002.

[21] R. Fleischer, T. Kamphans, R. Klein, E. Langetepe, and G. Trippen. Competitive online approximation of the optimal search ratio. *SIAM J. Comput.*, 38(3):881–898, 2008.

[22] M. Franceschetti and R. Meester. *Random Networks for Communication: From Statistical Physics to Information Systems*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 2008.

[23] E. Frazzoli and M. Pavone. Multi-vehicle routing. In J. Baillieul and T. Samad, editors, *Encyclopedia of Systems and Control*, pages 1–11. Springer London, London, 2013.

[24] Y. Gabriely and E. Rimon. Competitive complexity of mobile robot on-line motion planning problems. *Int. J. Comput. Geometry Appl.*, 20(3):255–283, 2010.

[25] J. D. Gammell, S. S. Srinivasa, and T. D. Barfoot. Batch informed trees (BIT*): Sampling-based optimal planning via the heuristically guided search of implicit random geometric graphs. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3067–3074, 2015.

[26] R. Geraerts and M. H. Overmars. Creating high-quality paths for motion planning. *International Journal of Robotics Research*, 26(8):845–863, 2007.

[27] R. Ghrist, J. M. O'Kane, and S. M. LaValle. Computing Pareto Optimal Coordinations on Roadmaps. *International Journal of Robotics Research*, 24(11):997–1010, 2005.

[28] G. Goraly and R. Hassin. Multi-color pebble motion on graphs. *Algorithmica*, 58(3):610–636, 2010.

[29] D. Halperin, O. Salzman, and M. Sharir. Algorithmic motion planning. In J. E. Goodman, J. O'Rourke, and C. D. Toth, editors, *Handbook of Discrete and Computational Geometry*, chapter 50, pages 1311–1342. CRC Press LLC, 3rd edition, 2016.

[30] S. Har-Peled and B. Raichel. The Fréchet distance revisited and extended. *ACM Transactions on Algorithms*, 10(1):3, 2014.

[31] R. A. Hearn and E. D. Demaine. PSPACE-completeness of sliding-block puzzles and other problems through the nondeterministic constraint logic model of computation. *Theoretical Computer Science*, 343(1-2):72–96, 2005.

[32] R. A. Hearn and E. D. Demaine. *Games, puzzles and computation*. A K Peters, 2009.

[33] S. Hirsch and D. Halperin. Hybrid motion planning: Coordinating two discs moving among polygonal obstacles in the plane. In *Workshop on the Algorithmic Foundations of Robotics (WAFR)*, pages 239–255. Springer, 2002.

[34] J. E. Hopcroft, J. T. Schwartz, and M. Sharir. On the complexity of motion planning for multiple independent objects; PSPACE-hardness of the "Warehouseman's problem". *International Journal of Robotics Research*, 3(4):76–88, 1984.

[35] D. Hsu, J.-C. Latombe, and R. Motwani. Path planning in expansive configuration spaces. *International Journal of Computational Geometry and Applications*, 9(4/5), 1999.

[36] L. Janson, B. Ichter, and M. Pavone. Deterministic sampling-based motion planning: Optimality, complexity, and performance. *International Journal of Robotics Research*, 37(1):46–61, 2018.

[37] L. Janson, E. Schmerling, A. A. Clark, and M. Pavone. Fast marching tree: A fast marching sampling-based method for optimal motion planning in many dimensions. *International Journal of Robotics Research*, 34(7):883–921, 2015.

[38] S. Karaman and E. Frazzoli. Sampling-based algorithms for optimal motion planning. *International Journal of Robotics Research*, 30(7):846–894, 2011.

[39] M. Katsev, J. Yu, and S. M. LaValle. Efficient formation path planning on large graphs. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3606–3611, 2013.

[40] L. E. Kavraki, M. N. Kolountzakis, and J. Latombe. Analysis of probabilistic roadmaps for path planning. *IEEE Trans. Robotics and Automation*, 14(1):166–171, 1998.

[41] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. Overmars. Probabilistic roadmaps for path planning in high dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12(4):566–580, 1996.

[42] D. G. Kirkpatrick, I. Kostitsyna, and V. Polishchuk. Hardness results for two-dimensional curvature-constrained motion planning. In *Canadian Conference on Computational Geometry*, 2011.

[43] D. G. Kirkpatrick and P. Liu. Characterizing minimum-length coordinated motions for two discs. In *Canadian Conference on Computational Geometry*, pages 252–259, 2016.

[44] M. Kleinbort, O. Salzman, and D. Halperin. Efficient high-quality motion planning by fast all-pairs r-nearest-neighbors. In *IEEE International Conference on Robotics and Automation*, pages 2985–2990, 2015.

[45] S. Kloder and S. Hutchinson. Path planning for permutation-invariant multi-robot formations. In *IEEE International Conference on Robotics and Automation*, pages 1797–1802, 2005.

[46] D. Kornhauser. Coordinating pebble motion on graphs, the diameter of permutation groups, and applications. M.Sc. thesis, Department of Electrical Engineering and Computer Scienec, Massachusetts Institute of Technology, 1984.

[47] D. Kornhauser, G. Miller, and P. Spirakis. Coordinating pebble motion on graphs, the diameter of permutation groups, and applications. In *Foundations of Computer Science (FOCS)*, pages 241–250. IEEE Computer Society, 1984.

[48] A. Krontiris, R. Luna, and K. E. Bekris. From feasibility tests to path planners for multi-agent pathfinding. In *Symposium on Combinatorial Search*, 2013.

[49] A. Krontiris, R. Shome, A. Dobson, A. Kimmel, and K. Bekris. Rearranging similar objects with a manipulator using pebble graphs. In *IEEE-RAS International Conference on Humanoid Robots*, pages 1081–1087, 2014.

[50] A. M. Ladd and L. E. Kavraki. Fast tree-based exploration of state space for robots with dynamics. In *Algorithmic Foundations of Robotics*, pages 297–312, 2004.

[51] A. M. Ladd and L. E. Kavraki. Measure theoretic analysis of probabilistic path planning. *IEEE Trans. Robotics and Automation*, 20(2):229–242, 2004.

[52] A. M. Ladd and L. E. Kavraki. Motion planning in the presence of drift, underactuation and discrete system changes. In *Robotics: Science and Systems*, Cambridge, USA, June 2005.

[53] S. M. LaValle, M. S. Branicky, and S. R. Lindemann. On the relationship between classical grid search and probabilistic roadmaps. *International Journal of Robotic Research*, 23(7-8):673–692, 2004.

[54] S. M. LaValle and S. A. Hutchinson. Optimal motion planning for multiple robots having independent goals. *IEEE Transactions on Robotics & Automation*, 14(6):912–925, 1998.

[55] S. M. LaValle and J. J. Kuffner. Randomized kinodynamic planning. *International Journal of Robotic Research*, 20(5):378–400, 2001.

[56] Y. Li, Z. Littlefield, and K. E. Bekris. Asymptotically optimal sampling-based kinodynamic planning. *International Journal of Robotic Research*, 35(5):528–564, 2016.

[57] J.-M. Lien, S. L. Thomas, and N. M. Amato. A general framework for sampling on the medial axis of the free space. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4439–4444, 2003.

[58] M. C. Lin, D. Manocha, and Y. J. Kim. Collision and proximity queries. In J. E. Goodman, J. O'Rourke, and C. D. Toth, editors, *Handbook of Discrete and Computational Geometry*, chapter 39, pages 1029–1056. CRC Press LLC, 3rd edition, 2016.

[59] R. Luna and K. E. Bekris. An efficient and complete approach for cooperative path-finding. In *Conference on Artificial Intelligence, San Francisco, California, USA*, pages 1804–1805, 2011.

[60] R. Luna, I. A. Şucan, M. Moll, and L. E. Kavraki. Anytime solution optimization for sampling-based motion

planning. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 5053–5059, 2013.

[61] K. M. Lynch and F. C. Park. *Modern Robotics: Mechanics, Planning, and Control*. Cambridge University Press, 2017.

[62] O. Nechushtan, B. Raveh, and D. Halperin. Sampling-diagram automata: A tool for analyzing path quality in tree planners. In *International Workshop on the Algorithmic Foundations of Robotics (WAFR)*, pages 285–301, 2010.

[63] P. A. O'Donnell and T. Lozano-Pérez. Deadlock-free and collision-free coordination of two robot manipulators. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 484–489, 1989.

[64] J. Peng and S. Akella. Coordinating multiple robots with kinodynamic constraints along specified paths. In J.-D. Boissonat, J. Burdick, K. Goldberg, and S. Hutchinson, editors, *Algorithmic Foundations of Robotics V*, pages 221–237. Springer-Verlag, Berlin, 2002.

[65] M. D. Penrose. *Random geometric graphs*. Oxford University Press, 2003.

[66] E. Plaku, L. E. Kavraki, and M. Y. Vardi. Motion planning with dynamics by a synergistic combination of layers of planning. *IEEE Transactions on Robotics*, 26(3):469–482, 2010.

[67] F. Pokorny, K. Goldberg, and D. Kragic. Topological trajectory clustering with relative persistent homology. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 16–23, 2016.

[68] B. Raveh, A. Enosh, and D. Halperin. A little more, a lot better: Improving path quality by a path-merging algorithm. *IEEE Transactions on Robotics*, 27(2):365–371, 2011.

[69] J. H. Reif. Complexity of the movers problem and generalization. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 421–427, 1979.

[70] O. Salzman and D. Halperin. Asymptotically-optimal motion planning using lower bounds on cost. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4167–4172, 2015.

[71] O. Salzman and D. Halperin. Asymptotically near-optimal RRT for fast, high-quality motion planning. *IEEE Trans. Robotics*, 32(3):473–483, 2016.

[72] O. Salzman, M. Hemmer, and D. Halperin. On the power of manifold samples in exploring configuration spaces and the dimensionality of narrow passages. *IEEE T. Automation Science and Engineering*, 12(2):529–538, 2015.

[73] O. Salzman, D. Shaharabani, P. K. Agarwal, and D. Halperin. Sparsification of motion-planning roadmaps by edge contraction. *International Journal of Robotic Research*, 33(14):1711–1725, 2014.

[74] O. Salzman, K. Solovey, and D. Halperin. Motion planning for multilink robots by implicit configuration-space tiling. *IEEE Robotics and Automation Letters*, 1(2):760–767, 2016.

[75] G. Sánchez-Ante and J. Latombe. Using a PRM planner to compare centralized and decoupled planning for multi-robot systems. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2112–2119, 2002.

[76] E. Schmerling, L. Janson, and M. Pavone. Optimal sampling-based motion planning under differential constraints: The drift case with linear affine dynamics. In *IEEE Conference on Decision and Control*, pages 2574–2581, 2015.

[77] E. Schmerling, L. Janson, and M. Pavone. Optimal sampling-based motion planning under differential constraints: The driftless case. In *IEEE International Conference on Robotics and Automation*, pages 2368–2375, 2015.

[78] J. T. Schwartz and M. Sharir. On the piano movers problem: II. General techniques for computing topological properties of real algebraic manifolds. *Advances in applied Mathematics*, 4(3):298–351, 1983.

[79] J. T. Schwartz and M. Sharir. On the piano movers problem: III. Coordinating the motion of several independent bodies. *International Journal of Robotics Research*, 2(3):46–75, 1983.

[80] M. Sharir and S. Sifrony. Coordinated motion planning for two independent robots. *Annals of Mathematics and Artificial Intelligence*, 3(1):107–130, 1991.

[81] I. Shnaps and E. Rimon. Online coverage by a tethered autonomous mobile robot in planar unknown environments. In *Robotics: Science and Systems*, Berlin, Germany, June 2013.

[82] T. Siméon, J. Laumond, and C. Nissoux. Visibility-based probabilistic roadmaps for motion planning. *Advanced Robotics*, 14(6):477–493, 2000.

[83] K. Solovey and D. Halperin. k-Color multi-robot motion planning. *International Journal of Robotic Research*, 33(1):82–97, 2014.

[84] K. Solovey and D. Halperin. On the hardness of unlabeled multi-robot motion planning. *International Journal of Robotic Research*, 35(14):1750–1759, 2016.

[85] K. Solovey and D. Halperin. Sampling-based bottleneck pathfinding with applications to fréchet matching. In

European Symposium on Algorithms, pages 76:1–76:16, 2016.

[86] K. Solovey and D. Halperin. Efficient sampling-based bottleneck pathfinding over cost maps. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2003–2009, 2017.

[87] K. Solovey and M. Kleinbort. The critical radius in sampling-based motion planning. In *Proceedings of Robotics: Science and Systems*, Pittsburgh, Pennsylvania, June 2018.

[88] K. Solovey, O. Salzman, and D. Halperin. Finding a needle in an exponential haystack: Discrete RRT for exploration of implicit roadmaps in multi-robot motion planning. *International Journal of Robotic Research*, 35(5):501–513, 2016.

[89] K. Solovey, O. Salzman, and D. Halperin. New perspective on sampling-based motion planning via random geometric graphs. In *Robotics: Science and Systems*, 2016.

[90] K. Solovey, J. Yu, O. Zamir, and D. Halperin. Motion planning for unlabeled discs with optimality guarantees. In *Robotics: Science and Systems*, 2015.

[91] D. Spensieri, R. Bohlin, and J. S. Carlson. Coordination of robot paths for cycle time minimization. In *CASE*, pages 522–527, 2013.

[92] P. G. Spirakis and C.-K. Yap. Strong NP-hardness of moving many discs. *Information Processing Letters*, 19(1):55–59, 1984.

[93] J. A. Starek, J. V. Gómez, E. Schmerling, L. Janson, L. Moreno, and M. Pavone. An asymptotically-optimal sampling-based algorithm for bi-directional motion planning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2072–2078, 2015.

[94] J. A. Starek, E. Schmerling, G. D. Maher, B. W. Barbee, and M. Pavone. Real-time, propellant-optimized spacecraft motion planning under Clohessy-Wiltshire-Hill dynamics. In *IEEE Aerospace Conference*, pages 1–16, Big Sky, Montana, 2016.

[95] I. Şucan and L. Kavraki. A sampling-based tree planner for systems with complex dynamics. *IEEE Transactions on Robotics*, 28(1):116–131, 2012.

[96] I. A. Şucan, M. Moll, and L. E. Kavraki. The open motion planning library. *IEEE Robotics & Automation Magazine*, 19(4):72–82, 2012.

[97] J. Tromp and R. Cilibrasi. Limits of rush hour logic complexity. *CoRR*, abs/cs/0502068, 2005.

[98] M. Turpin, N. Michael, and V. Kumar. Concurrent assignment and planning of trajectories for large teams of interchangeable robots. In *IEEE International Conference on Robotics and Automation*, pages 842–848, 2013.

[99] C. Urmson and R. G. Simmons. Approaches for heuristically biasing RRT growth. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1178–1183, 2003.

[100] J. van den Berg and M. H. Overmars. Prioritized motion planning for multiple robots. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 430 – 435, 2005.

[101] J. van den Berg, J. Snoeyink, M. C. Lin, and D. Manocha. Centralized path planning for multiple robots: Optimal decoupling into sequential plans. In *Robotics: Science and Systems*, 2009.

[102] C. Voss, M. Moll, and L. E. Kavraki. A heuristic approach to finding diverse short paths. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4173–4179, 2015.

[103] P. Švestka and M. H. Overmars. Coordinated path planning for multiple robots. *Robotics and Autonomous Systems*, 23:125–152, 1998.

[104] G. Wagner and H. Choset. Subdimensional expansion for multirobot path planning. *Artif. Intell.*, 219:1–24, 2015.

[105] R. Wein, J. P. van den Berg, and D. Halperin. Planning high-quality paths and corridors amidst obstacles. *International Journal of Robotic Research*, 27(11-12):1213–1231, 2008.

[106] P. R. Wurman, R. D'Andrea, and M. Mountz. Coordinating hundreds of cooperative, autonomous vehicles in warehouses. *AI Magazine*, 29(1):9–19, 2008.

[107] Y. Yang, V. Ivan, W. Merkt, and S. Vijayakumar. Scaling sampling-based motion planning to humanoid robots. In *IEEE International Conference on Robotics and Biomimetics*, pages 1448–1454, 2016.

[108] C.-K. Yap. Coordinating the motion of several discs. Technical report, Courant Institute of Mathematical Sciences, New York, 1984.

[109] A. Yershova and S. M. LaValle. Deterministic sampling methods for spheres and SO(3). In *IEEE International Conference on Robotics and Automation*, pages 3974–3980, 2004.

[110] J. Yu. A linear time algorithm for the feasibility of pebble motion on graphs. *CoRR*, abs/1301.2342, 2013.

[111] J. Yu and S. M. LaValle. Distance optimal formation control on graphs with a tight convergence time guarantee. In *Proceedings IEEE Conference on Decision & Control*, pages 4023–4028, 2012.

[112] J. Yu and S. M. LaValle. Multi-agent path planning and network flow. In *Workshop on the Algorithmic Foundations of Robotics (WAFR), MIT, Cambridge, Massachusetts, USA*, pages 157–173, 2012.

[113] J. Yu and S. M. LaValle. Planning optimal paths for multiple robots on graphs. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3612–3617, 2013.