MOTION PLANNING FOR A TETHERED MOBILE ROBOT

A Thesis

by

REZA HOSSEINITESHNIZI

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	Dylan A. Shell
Committee Members,	Dezhen Song
	Suman Chakravorty
Head of Department,	Dilma DaSilva

August 2015

Major Subject: Computer Science

ABSTRACT

Recently there has been surge of research in motion planning for tethered robots. In this problem a planar robot is connected via a cable of limited length to a fixed point in $\mathbb{R}^2$. The configuration space in this problem is more complicated than the one of a classic motion planning problem as existence of the cable causes additional constraints on the motion of the robot. In this thesis we are interested in finding a concise representation of the configuration space that results in a straightforward planning algorithm. To achieve such a representation we observe that configuration space manifold has a discrete structure that conveniently can be separated from its continuous aspect when it is represented as an atlas of charts. We provide a method for generating either the complete atlas or a subset of its charts based on special *cable events*. Generating parts of the configuration space on-the-fly enables the following improvements over the state-of-the-art. a) We decompose the environment into cells as needed rather than an off-line global discretization, obtaining competitive time and space complexity for our planner. b) We are able to exploit topological structure to represent robot-cable configurations concisely leading us towards solutions to the more complex problems of interest.

To underscore the potential of this representation, we take further steps to generalize it to two more complicated instances of the tethered robot planning problem that has been widely disregarded in the literature. We will first consider a simplified model of cable-to-cable contacts, giving the robot the option to perform knot-like tying motions. Next, we will address the planning problem for a tethered robot whose cable has a constraint on its curvature. This adds to the realism of the model since most practical cables have some degree of stiffness which limits curvature. In this

case we provide a novel technique to relate Dubins' theory of curves with work on planning with topological constraints. Our results show the efficiency of the method and indicate further promise for procedures that represent manifolds via an amalgamation of implicit discrete topological structure and explicit Euclidean cells.

DEDICATION

To my lovely grandmother,

To my mom and my dad,

To Hedy and Ali.

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to Dr. Dylan Shell who has been more than a great advisor by believing in me. Without his constant support, enthusiasm, and knowledge this thesis would have not been completed.

I also would like to thank Dr. Andrew Birt, Dr. Maria Tchakerian, and – soon to be Dr. – Elvis Takow for their kind and unsparing help and guidance throughout my journey in Texas A&M University.

Last but not least, I thank Brad Noffsker, Sahil Dilwali, Brian Alg, and Tiffany Truong who made me feel like home, here in College Station.

TABLE OF CONTENTS

Page

ABSTRACT ii

DEDICATION iv

ACKNOWLEDGEMENTS v

TABLE OF CONTENTS vi

LIST OF FIGURES viii

LIST OF TABLES xii

1. INTRODUCTION 1

1.1 The Problem's Complexity 3
1.1.1 The Robot's Characteristics 3
1.1.2 The Cable's Characteristics 4
1.1.3 Planning Metric's Characteristics 5
1.2 Thesis Outline 5

2. CONSIDERING A TETHER 7

2.1 Introduction 7
2.2 Related Work 8
2.3 The Preliminaries 11
2.3.1 Events 12
2.3.2 Visibility Cells 13
2.3.3 The Maximum Ray Length Visibility Atlas 14
2.4 The Basic Planning Algorithm 17
2.4.1 A Note on a Tethered Robot with Extent 17
2.5 Handling Cable-Cable Interaction 19
2.5.1 Cable-Cable Interaction Events 19
2.5.2 Violation of the tree structure of MRLVAG 20
2.5.3 Maintaining the Preferred Tree 21
2.5.4 Sequential Cable Events 22
2.6 Experimental Results 23
2.6.1 On-line Generation Versus Off-line Generation 25

2.6.2 Cell Decomposition Technique . . . 25
2.6.3 Cable Induced Manifold Structure . . . 26
2.7 Conclusion . . . 27

3. THE ROLE OF CURVATURE . . . 29

3.1 Introduction . . . 29
3.2 Related Work . . . 30
3.3 The Preliminaries . . . 32
3.3.1 Curvature Constraint and Stiff Cables . . . 32
3.3.2 Problem Statement . . . 33
3.3.3 Shortest Curvature Constrained Paths . . . 34
3.3.4 C-Space Skeleton . . . 36
3.3.5 Decomposition Method and Dubins Cells . . . 36
3.3.6 The Boundaries of a Cell . . . 40
3.3.7 The Atlas . . . 42
3.4 Planning With the Representation . . . 44
3.5 Discussion of the Method . . . 47
3.5.1 Memory Consumption . . . 47
3.5.2 Discussion of the Algorithm . . . 48
3.5.3 Covering Complex Spaces . . . 49
3.5.4 Parameterized Curvature Constraint . . . 50
3.6 Conclusion . . . 51

4. OUTLOOK: CONNECTED ROBOTS . . . 53

4.1 Problem Description . . . 53
4.1.1 Inputs . . . 53
4.1.2 Outputs . . . 54
4.2 Our Approach to the Problem . . . 54
4.3 Approaching Category 1 Solution . . . 56
4.3.1 Initial Configuration . . . 57
4.3.2 Compromising for the Shared Cable Length . . . 58
4.3.3 A Dynamic Programming Approach for finding the Optimal Pair of Paths . . . 59
4.4 Conclusion . . . 62

5. CONCLUSION . . . 65

REFERENCES . . . 67

APPENDIX A. MAXIMUM RAY LENGTH VISIBILITY ATLAS . . . 71

LIST OF FIGURES

FIGURE Page

1.1 The two constraints caused by a cable. 2

2.1 Different scenarios of a tethered robot with respect to the presence of obstacles. 11

2.2 The defining elements of a visibility cell. 12

2.3 A 3D model of the visibility cells and the way they are connected together. Each cell is in a different color. The robot is white and the cable is colored yellow. 15

2.4 An example of a MRLVAG. The environment producing this MRLVAG is given on the left. 15

2.5 The complexities of the cable-to-cable contacts: (a) Uncountable contact points and (b) changing movement radius. 19

2.6 An example of an MRLVAG growing toward a destination. 20

2.7 The best ordering of events for reaching from the base point to the destination point. In other words, for having the shortest path, the points with smaller number should be reached earlier. 22

2.8 An example of sequential cable crossing events. 23

2.9 Screen-shot of the simulation environment. 24

2.10 The two test environments used for presenting the results. Fig. 2.10a and Fig. 2.10b are representative of simple and complex environments, respectively. Gray is obstacle and green is robot. The length of the cable is 300 units in both test cases. 25

2.11 The two configurations associated to min and max rows in Table 2.1 in Test Environment 2. 26

2.12 The two configurations associated to min and max rows in Table 2.1 in Test Environment 2. 27

3.1 Curvature Constraint: the yellow dotted circle represents the cable's curvature constraint. Due to the stiffness of the cable, it cannot be bent with sharp angles. The gray arrow shows a planned motion for the robot. The figure on the left depicts an invalid configuration for the cable after execution of the motion. The valid configuration is shown on the right. 30

3.2 The two types of Dubins Path. The path on the left is of the type CLC, and the path on the right is of the type CCC. The three segments are colored in blue, magenta, and yellow, respectively. The arrows at the beginning and at the end of the paths shows the initial and goal poses, P_i and P_g, respectively. 33

3.3 Non-uniqueness of shortest curvature constrained path 34

3.4 The defining attributes of a Dubins Cell. The yellow dotted circles are the boundaries of maximal curvature. The green arrows are the orientations at the base poses of each cell. 38

3.5 A single Dubins Cell in an environment without any obstacle. The boundary of maximal curvature is shown with red dotted circles. The magenta spirals show the boundary of farthest reachable points (without considering a goal pose). The green arrow shows the orientation of the robot at the base pose. Cable's length is 60 units with curvature constraint $\kappa_0 = 1/20$ 39

3.6 The Ω region divides the set of all path going from P_i to P_g into two disjoint sets. One contains all the paths that are completely contained within this region and the other set contains the rest of the paths. . . 41

3.7 Examples of compatible and incompatible tangents. The red and the green tangents in each of the above figures are incompatible and compatible tangents, respectively. The yellow arrow shows the direction of the motion on the circle. The vector of direction at the tangent point should have the same orientation as the same the yellow arrow. 44

3.8 The simulation environment. The robot is blue, the dotted red circles show the boundary of maximal curvature, the green arrows show orientation vector in each base pose. The cable is colored in magenta. 48

3.9 The reachability all the points in the environment when they are assigned a goal orientation, v_g. The reachability has been tested from the initial pose $P_i = ((0,0),(0,1))$ single Dubins Path. Reachable points and unreachable point are marked by a green crosses and red dots, respectively. The black point and the blue arrow show the initial location and orientation, respectively. The cable's length is 200 units with curvature constraint $\kappa_0 = 1/20$. 49

3.10 $v_g = (1,-1)$. Notice the unreachable region near the initial pose that is caused by the incompatibility of v_i and v_g. Although it seems unintuitive, some points behind the obstacles are still reachable via a Dubins path despite not being physically visible from p_{base}. The properties of the cable and the color code in the figure is the same as Fig. 3.9. 50

3.11 The effect of parameterized curvature. In both figures, the cable's length is 200 units. 51

4.1 An example of how the initial configuration may appear. The dotted line is the cable and black dots are the obstacles. 55

4.2 The obstacles after partitioning. Black, red, and green obstacles belong to s_1, s_2, and s_3 respectively. 57

4.3 An example of cable configuration prior to finding a C1 solution. $P^s_{r^+}$ and $P^d_{r^+}$ indicate the source and destination of robot r^+'s motion, respectively. The same is true for $P^s_{r^-}$ and $P^d_{r^-}$ and robot r^-. The dotted orange curve illustrates the cable's configuration. 58

4.4 Robot r^+ goes directly from $P^s_{r^+}$ to $P^d_{r^+}$. The green circle sector shows the radius that is reachable to r^- with the remaining cable length. . . 60

4.5 Shortest path for r^+ that untangles obstacle number 1 while going from $P^s_{r^+}$ to $P^d_{r^+}$. The green circle sector shows the radius that is reachable to r^- with the remaining cable length. 61

4.6 Shortest path for r^+ that untangles obstacle number 1 and 2 while going from $P^s_{r^+}$ to $P^d_{r^+}$. The green circle sector shows the radius that is reachable to r^- with the remaining cable length. 62

4.7 Shortest path for r^+ that untangles obstacle number 1, 2, and 3 while going from $P^s_{r^+}$ to $P^d_{r^+}$. The green circle sector shows the radius that is reachable to r^- with the remaining cable length. 64

A.1 B, C and D are visible vertices from P_0. Only B and D are enqueued in the algorithm. 72

LIST OF TABLES

TABLE Page

2.1 Results of the Experiments 24

4.1 the distances between the obstacle in touch with the cable in the initial configuration (Figure 4.3). The distances are in the same arbitrary unit. 59

4.2 The effect of the chosen motion by r^+ on r^- in Figures 4.4 – 4.7. The distances and cable lengths in the table are expressed in the same arbitrary unit (the same as Table 4.1). The maximum cable length is 100 units. This table only considers the shortest paths in each homotopy class from $P^s_{r^+}$ to $P^d_{r^+}$ that the path does not make a new cable-obstacle contact. 60

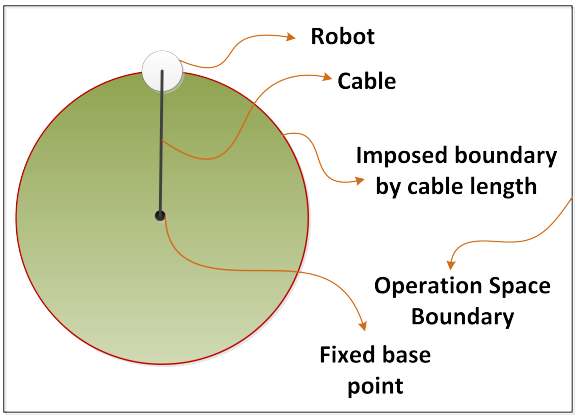
1. INTRODUCTION

Motion Planning, the problem of moving an object in a space while avoiding obstacles, has been addressed by more than three decades of research [28].

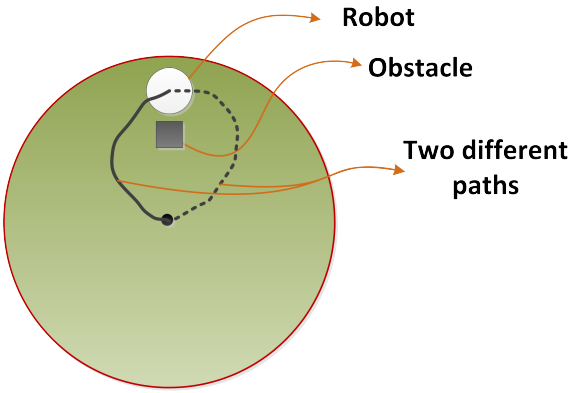
Many practical scenarios require a robot to use a tether. Consider a robot which uses a cable as a source of power and/or communication. Robots which perform high power tasks (*e.g.,* industrial robots) need to use a cable as a source of electrical power as their required energy cannot be provided by batteries. In several cases, a cable is necessary for a robot to be able to transfer data (*e.g.,* underground and underwater robots). Moreover, in some cases, the cable is not required but it can improve the performance of the given task. For example, a robot can use a cable to collect [10, 5] or separate objects from each other [19]. These examples highlight the importance of efficient methods that solve the motion planning problems for tethered robots. Recent research has focused on different aspects and variations of this problem [6, 17, 33, 29].

In motion planning for tethered robots, along with the usual constraints that are considered in any motion planning problem (*e.g.,* collision avoidance, distance, etc.), two important additional constraints are imposed on the motion of the robot due to the existence of the cable:

1. The radius of the robot's movement is limited by the cable's length (Figure 1.1a). For example, in a single connected environment a free robot (*i.e.,* without a tether) can reach any point in that environment, regardless of the distance that is traveled. However, in the case of a tethered robot, the reachable distance is determined by the length of its tether. In an obstacle-free environment, for example, the reachable area is a disk centered at the robot's



(a) Length Constraint. The green disk represents the reachable area for the tethered robot.



(b) Topological Constraint. This figure shows one example of the this class of constraints, that is, the placement of the cable in the environment with respect to obstacles.

Figure 1.1: The two constraints caused by a cable.

tether point with radius equal to the length of the cable.

2. Topological constraints are imposed by the cable and obstacles in the environment [7]. For instance, in Figure 1.1b, one difference between the two shown paths is the trajectory that the robot has to take to retract its cable and return to its initial position. If the robot takes the path on the left of the obstacle, it must return on the left side to get to its initial position. The same is true for the path on the right side of the obstacle.

1.1 The Problem's Complexity

Motion planning for a tethered robot can be modeled by considering a variety of elements, each of which contributes to the complexity of this problem. We have identified the following three main classes of characteristics and the different possible values for each them.

1.1.1 The Robot's Characteristics

The behavior of a mobile robot and its capability in performing different tasks changes the complexity of the planning problem. This behavior can be defined with three different parameters of interest in the context of this problem.

Extent of the Robot The shape of the robot in practice or simulations can vary. This characteristic of the robot affects the map of the configuration space. The common technique for considering the dimensions of the robot is to extrude the boundaries of the environment (including the obstacles) so that the points that are to close the boundaries are not considered in the planning process. This procedure will reduce the motion planning problem for a robot with extent to a case of a point robot [9].

Curvature Bounded Motion In the simplest instance of this problem, the robot's motion is free of its previous movement. That is, there are no bounds the how curved the trajectory of the robot's movement can feasibly be. In the literature, this type of motion is called *holonomic*. On the other hand, the curves in the robot's trajectory can be less than, greater than, or equal to some parameter due to design aspects of a particular robot. In such cases, the motion is said to be *nonholonomic* [13]. One instance of this problem in the literature is known as *Dubins Car Problem* [22].

Reverse Motion The robot's ability to move in both forward and reverse directions adds to the problem's degrees of freedom. One instance of such problems is the Reeds-Shepp Car [27]. The problem becomes more interesting if each movement direction has a different cost for the robot or there is cost associated with changing the movement direction, since the state space becomes nonuniform.

1.1.2 The Cable's Characteristics

Planning motions of the tethered mobile robot is greatly affected by the characteristics of its cable. Many of the configurations of the non-tethered robot become infeasible with the existence of a tether. The tether itself can be modeled with many physical features, some of which are considered in this thesis.

Length A theoretical cable can be of infinite length so that it does not impose limits on the distance between the robot and the cable's base point in the environment. However, the configuration space for such a robot and its cable still has a different topology compared to a robot without a cable. For example, the trajectory that the robot needs to take in order to retract/untangle its cable is dependent on its previous movements. However, all practical cables have a limited length and hence define a boundary on how far the robot can go from the base point of the cable.

Retracting vs. loose In a general case, a cable can be loose and have slack. However, one can consider a scenario in which the robot has a retracted/taut cable. The retraction itself can be done in different ways. The robot can keep its cable retracted at all times while moving or, alternatively, it can retract the cable once it has reached its destination. Depending on how the robot performs the retraction different methods can be applied to plan the motions of the robot.

Curvature Bounded Similar to the robot's motion, the cable can also be free of limits on how much it can be bent. Although, this might be the case in many practical scenarios, a more general model is one that considers an upper curvature bound for the cable. This can be a representation of the stiffness of the cable. It is possible that the robot does not have a curvature constraint on its motion while the tether has bounded curvature.

1.1.3 Planning Metric's Characteristics

Below we will provide two of the main criteria considered in this thesis for assessing aspects of feasible solutions. Based on the importance of each criterion, an evaluation function can be defined to choose the most suitable answer to the given instance of the problem. The planning metric can greatly change the complexity of the search for the solution.

Distance Traveled by the Robot One common metric is the distance that robot has to travel to get to its goal. In most cases, the minimum distance is the preferred over other solutions. Nevertheless, some problems require the maximum distance traveled (*e.g.,* surveillance).

Cable Length While the traveled distance can be an important factor, the length of the consumed cable can also be used as a criterion to evaluate solutions. One could choose a solution with the longest/shortest amount of cable consumed over other solutions. This becomes more important specifically when we consider the cable as a resource to the robot.

1.2 Thesis Outline

Based on the above characteristics of the robot, the cable, and the planning metric, different instances of this problem can be generated. In this study, we will look

for efficient methods of motion planning for tethered robots with a new perspective. In Chapter 2, we examine the structure of the configuration space induced by a robot tethered by a taut (possibly retracting) cable and propose to represent the manifold of configurations by an atlas [23]. In Chapter 3, we will extend this method and consider a more general model of cables. In the extended method, we model the robot's cable as a stiff tether with curvature constraint. Finally, Chapter 4 provides an outlook on how to approach the motion planning problem for two robots that are connected via a cable.

2. CONSIDERING A TETHER

2.1 Introduction

This chapter provides the theoretical foundation for considering a tethered robot with a new perspective. We examine the structure of the configuration space for a robot tethered by a taut (possibly retracting) cable and propose to represent the manifold of configurations by a special atlas [23]. In so doing, the topological regularity, captured as a graph, is naturally separated from the continuous aspects, captured as charts. The graph provides an understanding of the complexity induced by the cable and nodes within the graph provide sufficient topological context for points in $\mathbb{R}^2$ to represent configurations. The planning method we present is quite straightforward using this representation.

This method creates a set of locally continuous charts based on a set of *cable events*. Initially, we consider events that occur when a cable touches or wraps around an obstacle, and we show that they are intimately connected with a subgraph of the visibility graph of the environment. We will show that in this model this subgraph is in fact a tree, and thus the topological structure of the atlas representing the configuration space forms a tree structure. This structure provides context that allows path planning to proceed by moving locally (either up to parent charts, or downward to children charts) in the configuration space.

Next, we extend the set of cable events to consider cable-to-cable interactions, by considering two different ways to cross the cable: the robot can move *over* or *under* the cable[1]. The method—we believe, uniquely—is applicable to this case despite the fact that the involved topology becomes rather more complex: events no longer

[1]We do assume in this case, that the cable has infinite friction in contacting itself.

occur at a countable set of locations, the configuration space's structure is no longer separated easily. Nevertheless, because only parts of the configuration manifold need be generated and the path planning operations proceed exploring locally, the approach succeeds. We believe that this underscores the strength of dynamically computing relevant parts of the atlas on-the-fly and treating the configuration space as if it were a just-in-time computed data structure.

2.2 Related Work

A common underlying idea of previous works on this subject, is to create a graph based on a discretized approximation of the configuration space and provide an efficient method to search through the space of possible paths and choose the optimal one among all of which fall into the same homotopy class [15]. Here we will provide a list of the closely related works.

- *Hert and Lumelsky [16]:* As one of the oldest and influential papers on this subject, it discusses the problem of planning for multiple tethered robots. The authors designed an algorithm that gives as output a sequence of motions for robots (*i.e.,* an ordering of robots) so that the paths do not entangle the tethers. In doing so, the method receives a Directed Acyclic Graph (DAG) representing the required motions for reaching the final configuration. In this DAG, each node represents a robot and there is a directed edge from node v to node u if movement of robot v before u will not entangle the cables. The DAG is then used to perform a topological sort [8] on the motions. One possible drawback of this work is that the authors do not provide an algorithmic way of creating such a DAG.

- *Xavier [32]:* The author utilized visibility graph to trace back the cable and store changes in the visibility of the vertices in the environment. He constructs

a path from the collected vertices and modifies it to its shortest equivalent in the same homotopy class. Although the use of visibility graph in his method is similar to our technique, there are fundamental differences between the algorithms and data structures used in the two methods, which we will explain below.

- *Grigoriev and Slissenko [14]:* This work presents a polynomial-time algorithm for finding the shortest path among all the paths in a given homotopy class. It is presumed that the obstacles are semi-algebraic. The authors defined cuts of the plane and then attributed a letter to each cut. By creating a word for each path, they showed that two paths belong to a same homotopy class if and only if reducing their words results in a same irreducible word.

- *Bhattaharya, Kumar, and Likhachev [6]:* In this work, homotopy classes of paths are defined based on the *Cauchy Integral Theorem*. This method provides a way for defining homology classes of paths.

- *Igarashi and Stilman [17]:* The authors designed an algorithm for creating a graph of the configuration space manifolds based on the cable length constraint. In this graph, one point in the operation space could be represented by many vertices in the graph due to the fact that two identical robot locations may have different cable configurations. An extended version of this basic algorithm is also provided which avoids the robot-cable interactions. Their idea of representing the configuration space by multiple overlapping manifolds influenced the present work.

- *Yershov, Vernaza, and LaValle [33]:* In a less related topic, this work focuses on motion planing with winding constraints (*i.e.,* homology constraints). The

authors defined an optimization problem and using the properties of underactuated systems and dynamic programming they achieved a feedback policy to find the solution.

- *Narayanan, Vernaza, Likhachev, and LaValle [26]:* In their work, a similar concept to [14], called *virtual sensor beams*, is used to represent the topological information of a path by constructing words of letters belonging to a free group. These words are then reduced to identify homotopy and homology class of the given path.

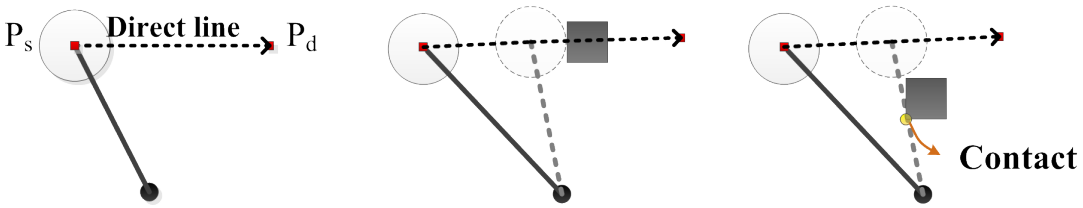
- *Shnaps and Rimon [29]:* This paper approaches the problem of coverage of a planar environment by a tethered robot. They propose an algorithm for performing an online coverage. It is assumed that the robot has no prior information about the obstacles in the environment and has a detection sensor.

Based on the body of the prior related studies, the method presented in this chapter has several distinguishing features:

- The approach we propose avoids regularly discretizing the configuration space. It instead uses a subset of the visibility graph to induce a natural cell decomposition of the space.

- We connect motions of the robot to discrete events that occur with the cable, each representing qualitative changes in the robot and cable configuration. The idea of identifying events of this form allows the method to be generalized so as to model cable-cable interactions appropriately.

- We avoid off-line creation of the entire configuration space, keeping data structures that allow for dynamic generation of necessary parts of the space. We



(a) No Obstacles (b) Obstacle blocking the direct line (c) Cable-obstacle contact

Figure 2.1: Different scenarios of a tethered robot with respect to the presence of obstacles.

represent topological context and local metric information, where planning in the local chart is trivial, and new charts are only created on-the-fly as needed.

- Finally, the visibility graph has been previously used in motion planning for non-tethered robots [25, 24]. A few methods also take advantage of visibility graph for the tethered case [32, 1]. We are interested in construction of an atlas of charts of the configuration space. This atlas leads to new insight with regards to the cable's dependency on the environment topology and its connection with the visibility graph.

2.3 The Preliminaries

This section provides the fundamental definitions used throughout the Chapter. We consider the problem of planning for a non-oriented robot situated in a planar environment with a cable tethered to a fixed point. It is assumed that the cable of maximum length l is always taut (*e.g.*, through the use of a retracting or spooling mechanism). And also that the obstacles are known and polygonal.

Fig. 2.1a–2.1c illustrate aspects of the problem. In an obstacle free environment, motion planning for a tethered robot is identical to an untethered robot with circular boundary of radius l. A path from p_s to p_d is a straight line segment [20]. Let $a \twoheadrightarrow b$ denote the line segment connecting point a to b. In moving from $p_s \twoheadrightarrow p_d$, an

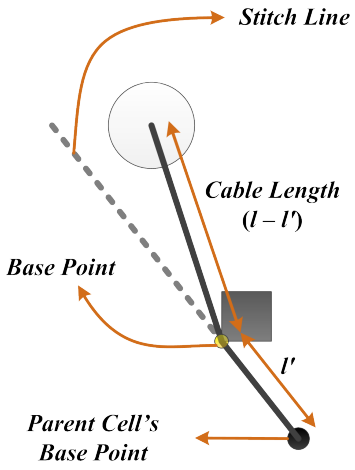


Figure 2.2: The defining elements of a visibility cell.

obstacle may affect the robot's motion directly (Fig. 2.1b) or indirectly via a cable-obstacle contact which will bend the cable. Fig. 2.1c. In the latter case, the radius of movement of the robot after that bend is affected.

2.3.1 Events

Consider the fixed base point of the cable. Due to the limited length of the cable, the distance between the robot and this point is never greater than l. Therefore, the accessible area for the robot will be a circle with radius l centered at that point (Fig. 1.1a). Interestingly, this is true for any contact point as well. If the robot consumes l' of its cable length to reach the contact point, since the cable is always taut, its distance from the contact point can never become greater than $l - l'$ unless it untangles itself (Fig. 2.2). Since the configuration of a taut does not change beyond the contact point unless it is detached from the contact point, then we can safely store the state of the cable up to the contact point and be sure that it will remain the same regardless of the configuration of the cable after the contact point. This is the basis for cell decomposition of the configuration space.

The contact points also define a homotopy class of trajectories that the robot can follow in order to untangle the cable and get back to the initial configuration (*i.e.,* the cable is completely retracted and the robot is at the origin of the cable). Incidents that change the homotopy class of returning trajectory and/or the boundaries of accessible area for the tethered robot are referred to as *events*.

Definition 1 (Cable Events)**.** *In the context of the cabled robot, there are two kinds of events:*

1. *Cable-to-obstacle contact: wrapping event*
2. *Cable-to-cable contact: cable crossing event*

In the following subsections, we are not concerned about cable-cable interactions. We return to cable-cable interactions in Section 2.5 where the method is extended.

2.3.2 Visibility Cells

A wrapping event will only occur when the cable touches an apex of one of the polygonal obstacles. These apexes are the same as the vertices of the visibility graph [9]. We can construct a connected component of the free space of the environment by partitioning the planar map into a semi-algebraic set P_{obs} consisting of all the obstacles and the free space P_{free} that is the complement of the set P_{obs}[14].

Since planning the motion for each cell of a decomposed configuration space is straightforward, we are going to define these cells in a way that the union of them will cover the configuration space. That is, each cell should contain all the points so that no movement of the tethered robot from any point in the cell to any other point in the same cell causes an event. Therefore we can reach the following definition for the cells of the configuration space in our context.

Definition 2 (Visibility Cell)**.** *A visibility cell is a chart,* (U, φ)*, where* $U \subseteq P_{free}$ *and homeomorphism* φ *is* $\varphi(x, y) = (x, y)$*, and the collision-free path between any two points in* U *is a straight line segment inside* U *connecting the two points.*

We identify cells uniquely by the following fields (Fig. 2.2):

Base Point as discussed in Section 2.3.1, each wrapping point is used as a base point.

Cable Length determines the maximum distance between the robot and the base point of a chart.

Parent Cell is the cell describing the robot and its cable configuration directly before occurrence of the event. This information is crucial when the planner is searching for paths and/or untangling the cable.

Stitch Line this is the line where one chart is connected to another and can be considered as an interface between them. Formally this line is the domain for transition map between the two charts. Once the robot have crossed the stitch line, a contact is made/released and thus the robot will be transfered from a chart (cell) to the other.

Fig. 2.3 illustrates a 3D model of how the visibility cells are connected together.

2.3.3 The Maximum Ray Length Visibility Atlas

Since each cell is basically a chart, we next define an atlas as a model of the decomposition.

Definition 3 (MRLVA (informal))**.** *A Maximum Ray Length Visibility Atlas (MRLVA) denoted by* A_l *is an atlas which contains visibility cells whose stitch lines are edges of the visibility graph of the environment. Each MRLVA has a graph associated*

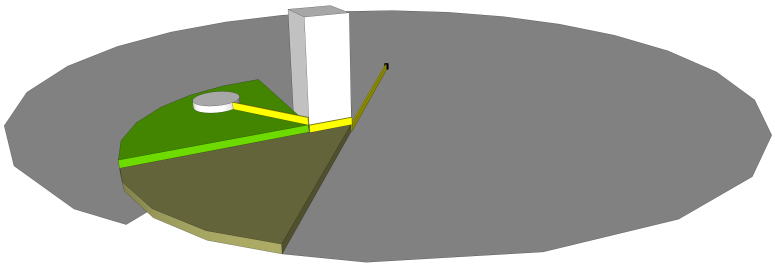


Figure 2.3: A 3D model of the visibility cells and the way they are connected together. Each cell is in a different color. The robot is white and the cable is colored yellow.

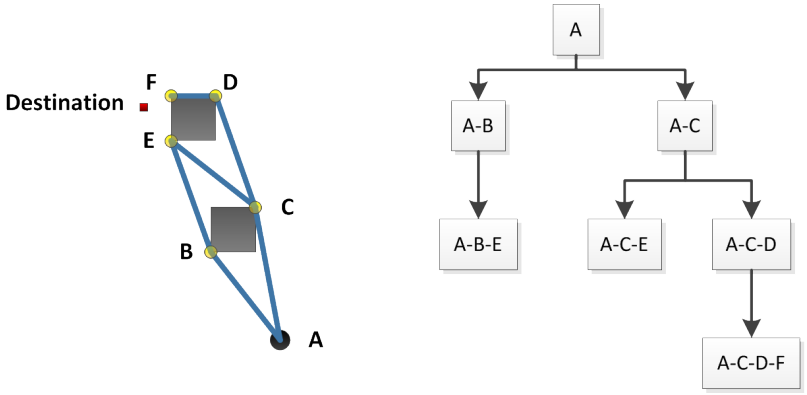


Figure 2.4: An example of a MRLVAG. The environment producing this MRLVAG is given on the left.

with it characterizing its topological structure, which we call Maximum Ray Length Visibility Atlas Graph (MRLVAG) denoted by G_{A_l} *(see Fig. 2.4).*

A detailed procedure for creating a MRLVA and MRLVAG is provided in Appendix A.

It is important to note that number of child charts is always less than or equal to the actual visibility graph's vertices.

Lemma 1 (Canonical Motion)**.** *In an environment with polygonal obstacles, a locally*

shortest path has a canonical form: in P_{free}, it is a straight line segment connecting the two end points. If one of the end points meets an obstacle it is locally supporting to P_{obs} at the contact point[20].

Theorem 1 (Completeness of MRLVA)**.** *The MRLVA A_l generated on P_{free} describes exactly the space of all the possible configurations of a robot (non-oriented) and its taut cable with maximum length l in P_{free}.*

Proof. The tautness of the cable implies each segment of it is a locally shortest path. So by Lemma 1, it is always in form of a line segment in P_{free}, and these line segments connect apexes of the obstacles which are a subset of edges of visibility graph, hence constituting all the configurations of the taut cable, except the final segment connecting the robot to the last apex. This last piece of information is provided by the position of the robot represented as a point in a visibility cell (that is a chart). Therefore, by connecting this point to the last apex, we will have all the robot and its taut tether's configurations. □

Corollary 1. *Given a MRLVA, by having a path from the root to a chart $(U_i, \varphi_i) \in A_l$ and a point $p \in U_i$ can identify a unique configuration of a non-oriented robot and its taut tether in P_{free}.*

Proof. Follows from Theorem 1 and Lemma 1. □

Theorem 2. *The MRLVAG, G_{A_l}, generated on P_{free}, is a tree.*

Proof. Suppose, to the contrary, that G_{A_l} has a loop. This implies two distinct sequences of cells and their transition maps to reach same cell. Each sequence implies an ordering of apexes, that is two distinct configurations of the cable for reaching the same cell. This contradicts Corollary 1. □

2.4 The Basic Planning Algorithm

Algorithm 1 shows the pseudo code for finding a path from source point p_s to destination point p_d. This algorithm is greedy since it values the direct line connecting the source point to the destination point more than other paths.

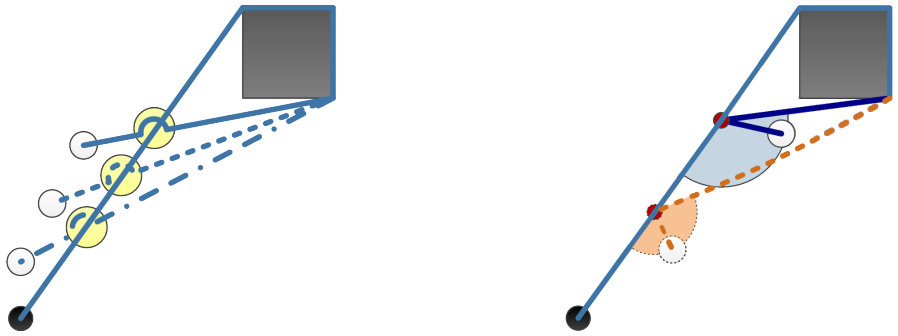
In order to initiate the algorithm the robot calls *FindPath* for *current* chart given the source and destination points. This function first considers $p_s \twoheadrightarrow p_d$ as it is always the locally shortest path [14]. If it cannot move directly toward p_d, it then checks whether a path can be obtained either from its children or its parent. If there is no child or the cable is not long enough, search in this branch of atlas tree will be terminated. Otherwise, the planner searches through all the children and stores the shortest path found in them. Next it ensures no shorter path from the parent chart to the destination point exists, otherwise the final path will go through the parent chart instead.

2.4.1 A Note on a Tethered Robot with Extent

Throughout this work, we have considered motion planning problem for a tethered point robot. This is assumption about the robot being a point is made in order to ease the understanding of the representation introduced in this work. However, it is possible to plan motions of a tethered robot with extent using this representation. A common technique, when planning for a robot with extent is to reduce the problem to a planning problem for a point robot using Minkowski sums [9]. In the context of planning for a tethered robot it is important to notice that the map produced by such procedures cannot be applied to the cable as the cable always makes contact with the — actual — apices of the obstacles (as opposed to the apices in the reduced problem). To overcome this we just have to plan the path for the robot using the map of reduced problem, however the length of the cable and the boundaries of the

1: **FindPath**$(p_s, p_d, path)$
2: mark this chart as visited
3: **if** p_d is directly reachable **then**
4: **if** $p_s \twoheadrightarrow p_d$ will not cause an event **then**
5: **return** $path.Push(p_d)$
6: **else**
7: m_c = Create a chart for the upcoming event
8: Set p_m as the point where $p_s \twoheadrightarrow p_d$ crosses the stitch line of the new chart
9: $path.Push(p_m)$
10: $minPath = m_c.FindPath(p_m, p_d, path)$
11: **end if**
12: **else if** p_d might fall into one of this chart's children **then**
13: Create child charts for each of the visible vertices of the obstacles if the chart is needed
14: **for all** c ∈ children **do**
15: $temp = path.Push(c.Base)$
16: $temp = c.FindPath(c.Base, p_d, temp)$
17: **if** *temp* is the shortest path to this point **then**
18: store it as $minPath$
19: **end if**
20: **end for**
21: **end if**
22: **if** $Parent \neq \varnothing$ and is not visited yet **then**
23: $temp = Parent.FindPath(Base, p_d, path)$
24: **if** *temp* is the shortest path to this point **then**
25: store it as $minPath$
26: **end if**
27: **else**
28: $minPath = \varnothing$
29: **end if**
30: **return** $minPath$

Algorithm 1: The shortest path greedy algorithm



(a) (b)

Figure 2.5: The complexities of the cable-to-cable contacts: (a) Uncountable contact points and (b) changing movement radius.

cells should be calculated using the actual location of the apices of the obstacles.

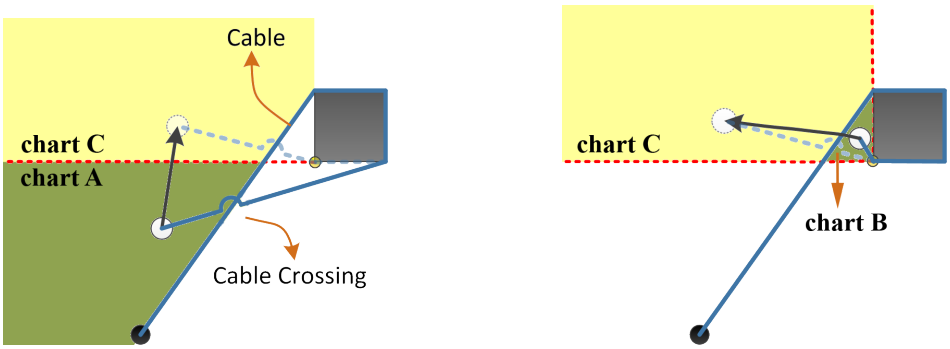
2.5 Handling Cable-Cable Interaction

This section discusses how to handle a robot crossing its own cable and the importance of these events.

2.5.1 Cable-Cable Interaction Events

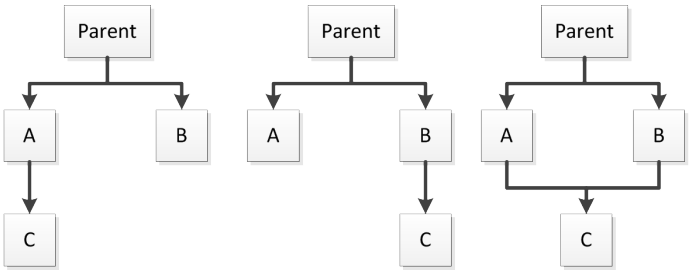
Previous work generally opts to ignore instances in which a robot's path crosses the cable. An exception is [17] wherein the authors deliberately plan movements to avoid such paths. The main reason of ignoring the cable-to-cable contacts is that it is either impossible to model the cable contact points, or it is computationally inefficient to do so (*e.g.,* discretizing the cable.) In contrast, we show that the MRLVAG can be extended to consider events of this type so that the cable configuration is stored dynamically.

Mainly, modeling this situation is complicated for two reasons: (a) the cable can make contact to another segment of the cable in uncountable number of points, (b) the radius of feasible subsequent movements depends on the location of the contact point (see Fig. 2.5).



(a) from chart A to C

(b) from chart B to C



(c) From left to right: the tree made by movement in Fig. 2.6a, in Fig. 2.6b, and the inherent loop.

Figure 2.6: An example of an MRLVAG growing toward a destination.

When encountering the cable, the robot may either go over the cable or under it. For these cases, the algorithm is modified by checking whether the last point added to the path will cause a cable crossing and making the binary choice of over or under. Each time the robot crosses the cable a new visibility cell is created whose set of stitch lines contains the cable that the robot has recently crossed in addition to the set of stitch lines of its parent (see Fig. 2.6).

2.5.2 Violation of the tree structure of MRLVAG

For cable interactions, in addition to *wrapping* events, *over* and *under* events are needed, yielding the set $\{o, u, w\}$.

Theorem 3. *With cable crossing events the configuration space no longer has genus 0, i.e., the topological structure is changed so that the MRLVAG is no longer a tree.*

Proof. Figure 2.6 provides an example by construction. It is possible to arrive at the same chart from two different charts, implying that there are loops in the atlas structure (in this case we can arrive at chart C either from chart A or B). □

The practical ramifications of this issue are resolved easily by finding any spanning tree of the MRLVAG. Since the MRLVA is itself complete, any of its spanning trees is complete as well. Nevertheless, the question remains: *which spanning tree do we prefer?* To answer this, we consider the effect of the topology on the optimality of the robot's movement. The order in which the events occur is related to how suboptimal the path taken by the robot can be.

Reexamining Fig. 2.6, we see that if the goal is reaching chart C, then Fig. 2.6b is a shorter distance than Fig. 2.6a. In fact, if the movement to cross the cable is arbitrarily small, then a robot going from chart B to chart C moves the same distance as if cable crossing were ignored entirely.

2.5.3 Maintaining the Preferred Tree

With a preferable atlas tree, we must generate that tree. If the robot knew which events were going to occur along a trajectory to its destination, the shortest route would start by connecting the current location of the robot to the first event, the first event to the second event, and so on until it connects the last event location to the destination. This event ordering is the order in terms of their location on the cable (see Fig. 2.7). Thus, the algorithm is extended to swap events that are out of order. In Fig. 2.6a, for example, although the cable event happens before the wrapping event, since the position of the wrapping event is before the cable event on the cable, the algorithm will reorder them by replacing the wrapping event behind the cable events.

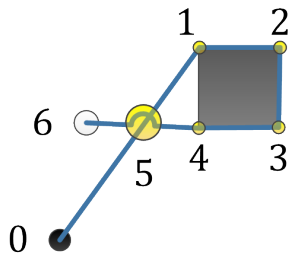


Figure 2.7: The best ordering of events for reaching from the base point to the destination point. In other words, for having the shortest path, the points with smaller number should be reached earlier.

2.5.4 Sequential Cable Events

The precise configurations that result from cable-cable interactions depend on the physical properties of the cable. For our theoretical treatment, we simplify these complexities by assuming that whenever the cable wraps around itself there is an *imaginary pin* that prevents the displacement of the wrapping point as the robot moves. This models a cable with infinite friction with itself. This allows one to define a chart wherever the robot crosses a cable followed by crossing the same cable except with the other type of action (*i.e.,* o followed by u or vice versa, see Fig. 2.8). The *Base* of this chart is the imaginary pin point and the cable is the stitch line. Such charts need not be stored permanently in the atlas tree, as there can be infinitely many different charts on a single cable (each depending on the location of the imaginary pin). A chart that is created because of a sequence of cable crossing events and all its children will be removed from the atlas whenever the robot's configuration leaves that chart. This illustrates the power of dynamically generating parts of the configuration space on-line.

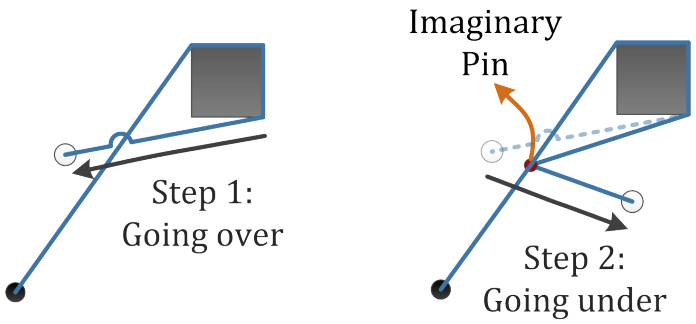


Figure 2.8: An example of sequential cable crossing events.

2.6 Experimental Results

To demonstrate the method and evaluate its performance, we developed a simulation environment and implemented the algorithm in C# (see Fig. 2.9).

Fundamental differences between our method and existing work make picking appropriate criteria for comparison challenging. In particular, running time is problematic because other state-of-the-art methods employ a discretization of the configuration space. The running time is directly affected by the discretization resolution, and memory utilization suffers from the same illegitimacy. After careful consideration, it was determined that the most equitable means for evaluation was to give a measure of the proportion of configuration space expanded by the algorithm, and the number of cells (*i.e.,* an indication of memory footprint) used in doing so.

For each test scenario we calculated the total volume of the configuration space. We then compared it to the summed volume of the cells that are sufficient for specifying the robot and cable's configuration: current visibility cell and its parents up to the root. The results in Table 2.1 were generated by examining all points of the configuration space for the two test scenarios, and looking at the proportion of the configuration space volume stored in memory. The minimum and maximum rows re-

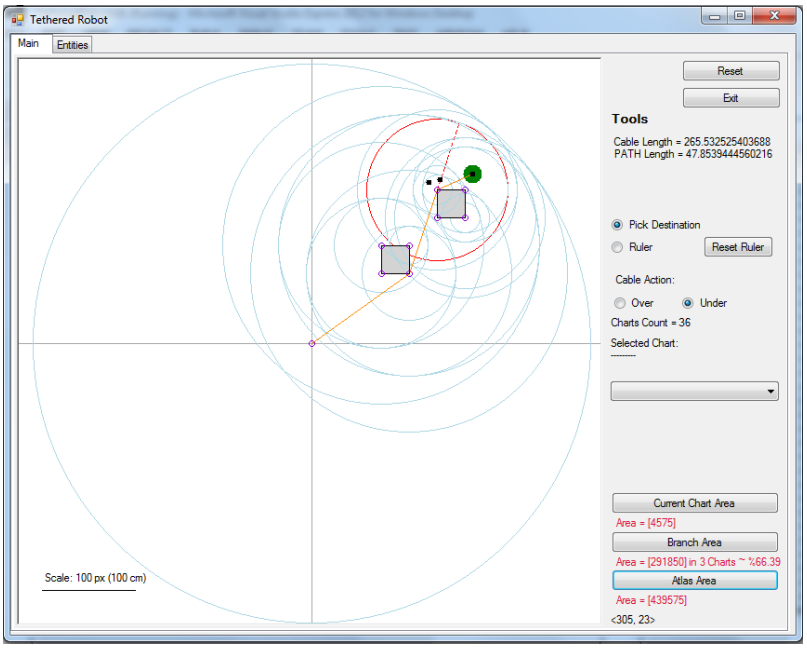


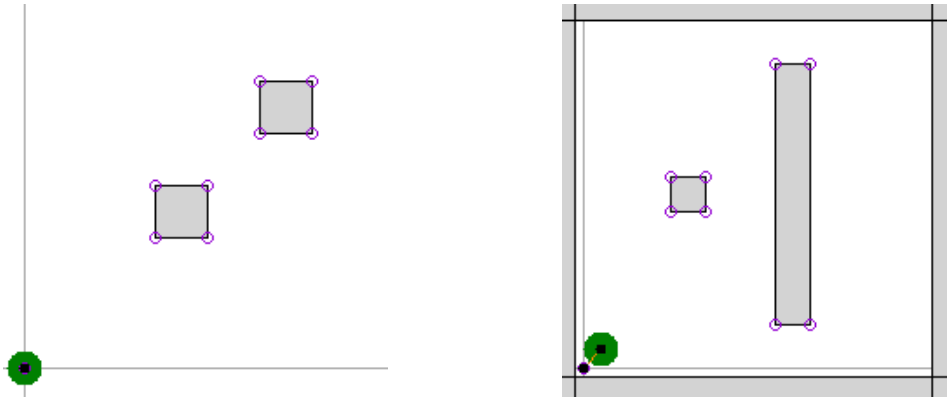
Figure 2.9: Screen-shot of the simulation environment.

port values for the cheapest and most costly configurations to represent, respectively (see Figures 2.11 and 2.12).

	Environment 1		Environment 2	
Memory Usage	% of MRLVA volume covered	Cell Count	% of MRLVA volume covered	Cell Count
Min	62.52	1 of 36	7.12	1 of 115
Max	74.8	9 of 36	21.47	13 of 115

Table 2.1: Results of the Experiments

The cable induces different degrees of topological complexity in environments with differing numbers and complexity of obstacles. To demonstrate this effect, we have chosen the two test scenarios in Fig. 2.10.



(a) Test Environment 1 (simple) (b) Test Environment 2 (complex)

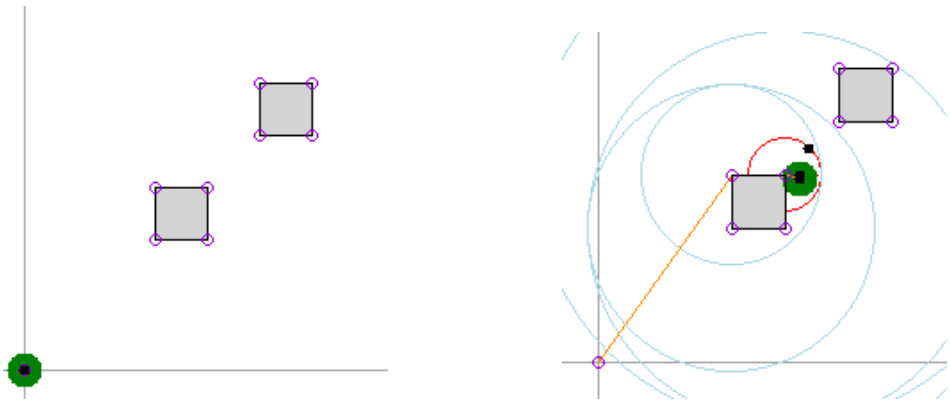
Figure 2.10: The two test environments used for presenting the results. Fig. 2.10a and Fig. 2.10b are representative of simple and complex environments, respectively. Gray is obstacle and green is robot. The length of the cable is 300 units in both test cases.

2.6.1 On-line Generation Versus Off-line Generation

Table 2.1 demonstrates the savings enabled by on-line generation of the configuration space. They show that if we were going to use a discretization of the configuration space represented as an MRLVA, storing the current chart and its ancestors would reduce the volume, the number of vertices in the graph, and consequently the memory use and searching time needed over the off-line methods used in the state-of-the-art.

2.6.2 Cell Decomposition Technique

Although using the on-line atlas-based method reduces the number graph nodes, it is not the only advantage of the method. As explained in Section 2.3.2, when the atlas is comprised of charts encoding the visibility properties of the environment, a special data structure can be used to represent the chart as a continuous space. Doing so requires only a constant amount of memory no matter how big or small (in volume) the chart is.



(a) The configuration associated with the minimum costly representation. In this configuration the robot is residing in the root cell of the tree (no cable events has happened yet).

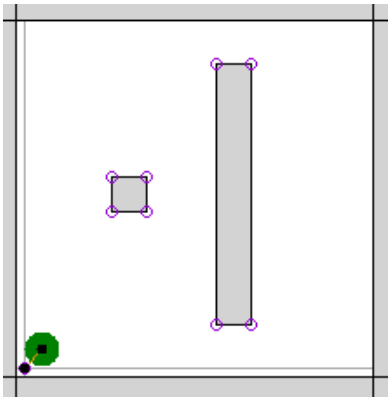
(b) The configuration associated with the maximum costly representation. This configuration requires the most number of nodes in the atlas graph of c-space representation. The robot has crossed its own cable once and wrapped it around the square obstacle 1.5 times (touched the obstacle's vertices 6 times).

Figure 2.11: The two configurations associated to min and max rows in Table 2.1 in Test Environment 2.

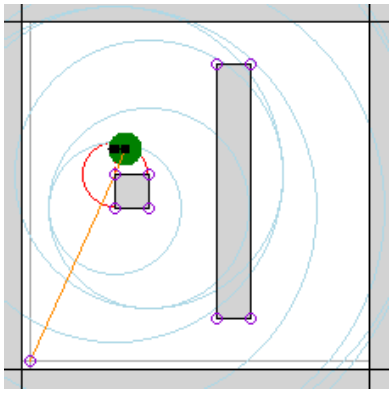
Therefore, the number of charts used is a more important factor than area. We computed the number of nodes in the MRLVAG generated by the on-line method. The total number of charts needed for the configuration space depends on the complexity of the environment as it affects the manifold; consequently these numbers are reported too. Results for the test scenarios, shown in Table 2.1, illustrate that savings are substantial in both simple and complex manifolds.

2.6.3 Cable Induced Manifold Structure

The presented data also lead to an additional observation about memory saving in environments with different degrees of topological complexity. At one end of the scale, simple environments result in a configuration space that is mostly planar and has a small MRLVAG. In these cases, large volumes of the configuration space are captured with single cells. The environment in test scenario 1 is an exemplar: more



(a) The configuration associated with the minimum costly representation. In this configuration the robot is residing in the root cell of the tree (no cable events has happened yet).



(b) The configuration associated with the maximum costly representation. This configuration requires the most number of nodes in the atlas graph of c-space representation. The robot has crossed its own cable twice and wrapped it around the square obstacle twice.

Figure 2.12: The two configurations associated to min and max rows in Table 2.1 in Test Environment 2.

than 60% of the configuration space is represented by a single chart. At the other end of the scale, complicated environments increase the topological complexity (reflected in large MRLVAGs) and have many visibility cells and charts. The resulting graphs are large, but have the form of wide and short trees. The environment in test scenario 2 illustrates this, with a total of 115 cells, but at most 13 ever need to be kept in memory. Thus, the cell decomposition approach results in significant memory saving across environment types.

2.7 Conclusion

This chapter approached planning for tethered robots with a new perspective: previous work employs a discretization of the configuration space along with an efficient search method. In addition to the need to perform substantial off-line precomputation, existing approaches are unable to represent cable-cable interactions, and either ignore this problem or avert configurations which lead to them. The proposed

method solves the basic tethered robot planning problem in a time and memory efficient way. Moreover, it is sufficiently general to form a consolidated representation for several other problems of interest, for example, winding constraints, some knot-like tying motions, *etc.* The method is, thus, more convenient and powerful than available approaches.

In this paper assumptions about the physical properties of the cable (tautness and friction) allow for well-defined cable-cable interactions. Another powerful property of the charts is their independence. One may use this in selecting distinct coordinate system representations for each chart to further improve efficiency.

3. THE ROLE OF CURVATURE

3.1 Introduction

In the previous Chapter, we introduced a method for planning the motions of a tethered mobile robot. To establish the theoretical foundation for the method, we considered a simplified model of a cable. Namely, the cable was taut at all times, it was frictionless except at the self intersection points, and it could be bent freely. Though some of these simplifications are employed herein, this Chapter is a step toward a more realistic model as we consider potential stiffness in the cable. We treat stiffness with a parameterized curvature constraint on the cable.

As we saw in the previous chapter, even the simplest case involving a frictionless, taut, and unconstrained flexible cable imposes two constraints on the motion of the robot: (1.) the radius of the robot's movement is limited by the cable's length (Fig. 1.1a), and (2.) topological constraints are imposed by the cable and the obstacles in the environment (Fig. 1.1b). For a stiff cable, a third constraint is added, not to the robot's motions, but to feasible cable configurations: (3.) the cable cannot be bent with more than some sharpness (Fig. 3.1). One naïve approach is to consider a discretization of the cable which is then forced to abide by the curvature constraint. This technique, however, leads to a high dimensional space and the concomitant computation is costly.

We tackle the problem by taking advantage of two complementary ideas. We generalize the decomposition method proposed in chapter 2, which separates topological regularities from the continuous aspects of the c-space. In doing so, we employ Dubins' theory of optimal trajectories under curvature constraints [11] to exclude infeasible cable configurations from the robot-cable c-space. Building a concise rep-

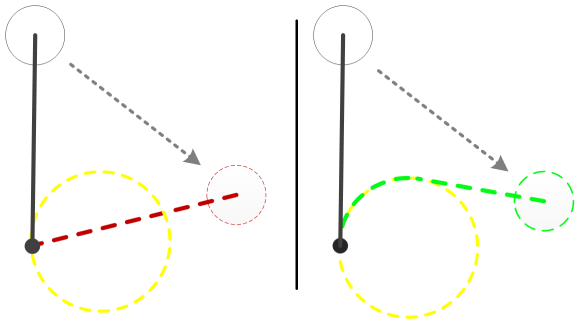


Figure 3.1: Curvature Constraint: the yellow dotted circle represents the cable's curvature constraint. Due to the stiffness of the cable, it cannot be bent with sharp angles. The gray arrow shows a planned motion for the robot. The figure on the left depicts an invalid configuration for the cable after execution of the motion. The valid configuration is shown on the right.

resentation of this space, enables an efficient, complete search of the manifold. In Section 3.4, we present an algorithm that explores the c-space while searching it, lazily computing parts of the manifold as and when they are needed. This form of on-line search and exploration minimizes the memory use of the method. We demonstrate that parameterizing curvature makes the method a general enough technique to solve several related problems.

3.2 Related Work

This work is related to two distinct areas of motion planing research: planning for tethered mobile robots, and nonholonomic motion planing. It is important to note that the nonholonomic constraint in this problem is not on the robot's motion, but it is on the feasible poses of the cable. The robot is controllable in all of its three degrees of freedom (x, y, and θ). The cable, however, cannot be bent beyond a certain angle when anchored to a point.

The reader is encouraged to see Section 2.2 where we discussed closely related works to the area of motion planning for tethered robots in details. In short, the common approach for planning motion of tethered robots is to create a graph ap-

proximation of the configuration space and then search through the graph to find the solution among paths in the same homotopy class.

There has been considerable work on the general topic of *nonholonomic motion planning*. Dubins [11] proved that any optimal path in an obstacle-free unbounded environment consists of 3 segments which are either straight line segments or circle sectors. Jacobs and Canny [18] provided an $O(n^4 \log(n) + (n)^2/\delta^2)$ algorithm that finds a δ-*robust* approximate shortest path, where n is the total number of vertices of the obstacles. They employed a plane sweep technique and quadtrees to achieve the aforementioned running time. Their method describes a graph that divides the c-space into simple trajectories. Similar to Dubins's approach, they showed that these trajectories are made up of subpaths that pass between the points on the boundaries of the obstacles. Agarwal *et al.* [3] extended Dubins's theory to deal with environments with so-called moderate obstacles. By improving on Jacobs and Canny's work, the paper provided an algorithm that builds a graph of straight lines and circles. This graph was then searched to find an optimal path. The total time of this method is $O(n^2 \log(n))$, where n is the total number of edges. Their later work [2] took a similar approach to find the shortest path in an obstacle free environment but with polygonal boundaries with n edges in time $O(n^2 \log(n))$. There are also several works addressing the problem from a control perspective (see [21, 30] and the references therein.)

We know of no prior work that considers both the curvature constraint and the topological constraints, while modeling the space of robot-cable configurations. One may choose to use available motion planning techniques designed for tethered robots and then to double check the validity of cable configurations after simulated execution of the motion. A second alternative is to consider a discretized cable. If the discretization is to be accurate, this technique results in a high dimensional space

and costly computations thence. We believe the method introduced in this work describes the robot-cable configuration space accurately and elegantly, capturing the necessary constraints before the search phase.

3.3 The Preliminaries

3.3.1 Curvature Constraint and Stiff Cables

Definition 4 (Nonholonomic Constraint)**.** *If r is the radius vector of an object in $\mathbb{R}^2$, a constraint f of the form $f(r, \dot{r}, t) = 0$ is nonholonomic if it cannot be expressed as $f(r, t) = 0$ [13].*

Definition 5 (Average Curvature)**.** *For a path $P : I \to \mathbb{R}^2$, the Average Curvature is*

$$\kappa(s) = \frac{d^2 P(s)}{d(s^2)}; \forall s \in I \tag{3.1}$$

and therefore is a nonholonomic condition.

Definition 6 (Curvature Constraint)**.** *If there exists an upper-bound $\kappa_0 \in [0, \infty)$ on the average curvature of a path, we say the path has a curvature constraint of κ_0. That is*

$$\kappa(s) \leq \kappa_0 = \frac{1}{r_0}; \forall s \in I \tag{3.2}$$

where r_0 is the radius of the smallest circle that the path can turn around.

Since a stiff cable cannot be bent more than a certain amount defined by some curvature constraint, κ_0, any robot-cable configuration in which the cable is bent with a radius less than r_0 is infeasible and is excluded from c-space (see Fig. 3.1). Notice that equation 3.2 provides a general model since a cable without curvature constraint can be parametrized as $\lim_{r_0 \to 0} \kappa_0 = \infty$.

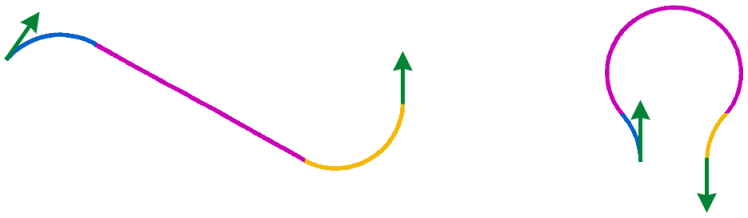
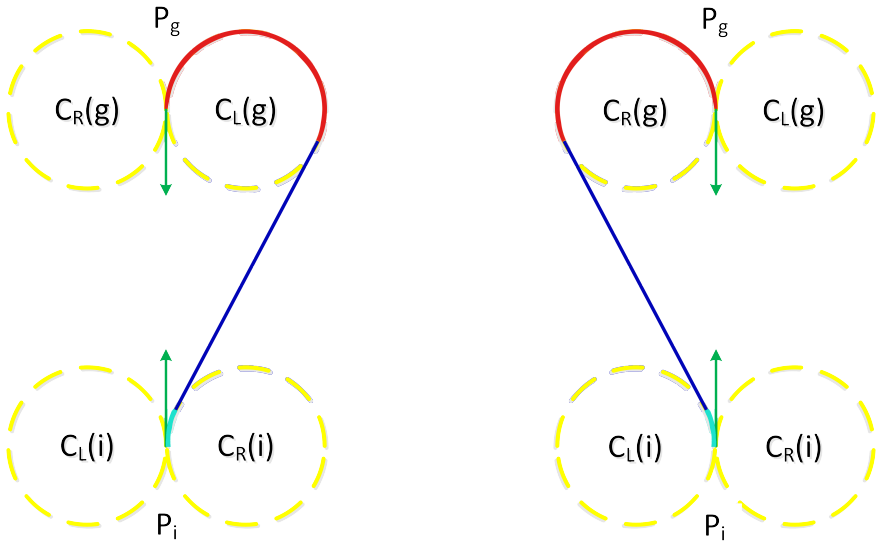


Figure 3.2: The two types of Dubins Path. The path on the left is of the type CLC, and the path on the right is of the type CCC. The three segments are colored in blue, magenta, and yellow, respectively. The arrows at the beginning and at the end of the paths shows the initial and goal poses, P_i and P_g, respectively.

3.3.2 Problem Statement

Definition 7 (Pose)**.** *A pose, P_k, is a pair $P_k = (p_k, \theta_k)$ where $p_k \in \mathbb{R}^2$ is a point and $\theta_k \in S^1$ representing the location and orientation of P_k, respectively. Alternatively, the orientation of a pose can be represented by a velocity vector $v_k = (v_{kx}, v_{ky}) \in \mathbb{R}^2$ of unit length where $\theta_k = arctan\frac{v_{ky}}{v_{kx}}$.*

This work considers the problem of planning from an initial pose, P_i, to a goal pose, P_g, for an oriented point robot[1] situated in $\mathbb{R}^2$ in existence of polygonal obstacles whose vertices are known to the robot. The robot is tethered to a fixed base with pose P_{base}, via a cable with curvature constraint $\kappa_0 = 1/r_0$. It is assumed that the cable has a maximum length l and is always retracted to its tightest form. By *tight* we mean there is no perturbation that can be applied to the path such that it would make the length of the given path shorter without violating either the length or curvature constraints.



(a) The path starting by turning right. (b) The path starting by turning left.

Figure 3.3: In the case of a symmetry between $C_L(i)$ and $C_L(g)$ with $C_R(i)$ and $C_R(g)$ finding a shortest curvature constraint path makes it possible to find the second shortest curvature constraint path by doing a reflection.

3.3.3 Shortest Curvature Constrained Paths

Let M be a 1-connected unbounded manifold in $\mathbb{R}^2$ and τ be a path contained in M. Following the terminology in [3], a *C-segment* of τ is a non-empty maximal subpath of τ that has the form of a circular arc with radius r_0; an *L-segment* is a non-empty maximal subpath of τ that has the form of a straight line segment. We say, for example, a path is of type CLC if it is made up of only three segments of types C, L, and C in this order.

Definition 8 (Shortest Curvature Constrained Path)**.** *Denoted by $P_a \rightsquigarrow P_b$, a shortest curvature constrained path is a minimal length path in M from initial pose P_a to goal pose P_b, with curvature constraint κ_0.*

[1]We made a note on how to apply the representation technique in this work to a tethered robot with extent in Section 2.4.1.

Theorem 4 (Dubins' Theorem [11]). *Every shortest curvature constrained path in M is necessarily a path of type CLC or CCC or a substring of either of these (see Fig. 3.2).*

Let $P_x = (p_x, v_x)$ be a pose in the environment. Let L_x be the line parallel to v_x which passes through p_x. We use the notation $C_L(x)$ and $C_R(x)$ for the circles to the left and right (according to v_x) of L_x which are tangent to L_x at p_x. We will call $C_L(x)$ and $C_R(x)$ the tangent circles to P_x.

Let P_i and P_g are the initial pose and the goal pose respectively. It is important to notice that the shortest curvature constrained path in M, as defined in Definition 8, is not unique in a set of special configurations of P_i and P_g. When p_g is on L_i, There is a symmetry between $C_L(i)$ and $C_L(g)$ with $C_R(i)$ and $C_R(g)$. Therefore, if there is one shortest curvature constrained path $\tau = P_i \rightsquigarrow P_g$, a second path can be created just by doing a reflection on τ about L_i (see Fig. 3.3).

Now let M be an unbounded[2] subspace of $\mathbb{R}^2$ with polygonal obstacles and τ be a curvature constrained path in M. C and L-segments are defined as before. An O-segment of τ is a maximal segment of τ that lies on the boundary of an obstacle. A C-segment is called *terminal* if it is the first or last segment of a path.

Lemma 2 (Non-terminal C-segment [3]). *A non-terminal C-segment of a locally shortest path[3] is either tangent to at least one obstacle or it is adjacent (on the path) to a terminal C-segment.*

Corollary 2. *If any subpath of a locally shortest path contains O-segments, they are either at the beginning or at the end of the subpath.*

[2]If M is bounded then we require that the boundary of the environment be define via boundary obstacles.

[3]A path is locally shortest if any perturbation in the path is physically impossible or makes the length of the path longer.

Proof. Follows from Lemma 2. □

Lemma 3 (Dubins Subpaths [12, 18, 3])**.** *Any subpath of a locally shortest curvature constrained path is itself a shortest curvature constrained path if the subpath does not touch any obstacle.*

3.3.4 C-Space Skeleton

Throughout this work, we use the term "*decomposition*" versus "*discretization*" to convey a different meaning. A decomposition is the act of finding a skeleton in c-space as a graph whose vertices represent the largest subspaces of c-space in which the solution to the motion planning problem is canonical. Whereas, a discretization method provides an approximation of c-space. A cable induces a structure into c-space manifold which can be exploited in an elegant way: an appropriate decomposition of c-space yields a concise discrete topological skeleton and a set of continuous subspaces. This observation was made in Chapter 2 where the cable in each subspace is always in the form of a straight line.

Similarly, we construct solutions to this problem from simpler subpaths found in subspaces. In the context of this problem, we require the cable to be in the form a shortest path in each subspace as defined by Definition 8. This in turn means, we are able to use the exact same data structures and path finding algorithms as before, with modifications made to only adapt the method for finding shortest curvature constrained paths. Once this method is determined, we are able to define the appropriate cell decomposition. This method is discussed in the following subsection.

3.3.5 Decomposition Method and Dubins Cells

Since the cable is always taut, it lies on a path from P_{base} to P_g, where each of its segment is a locally shortest path.

Corollary 3 (Taut Cable Decomposition)**.** *Every given taut configuration of a curvature-bounded cable that lies on the path τ_c can be decomposed into subpaths, each of which is a Dubins Path.*

Proof. If the cable does not touch any obstacle along its path, τ_c from P_{base} to P_g, then the locally shortest path is also the globally shortest path, and hence following Theorem 4 it is a Dubins path. In this proof, we will follow regular expression notations to show the string representing the C, L, and O segments in a path. Let $[C, L]^*$ be a path constituted of zero or more C or L-segments. Moreover, let $O[C, L]^*$ denote a path beginning with an O-segment and followed by a path of type $[C, L]^*$. $[O[C, L]^*]^+$ denotes one or more consecutive paths of type $O[C, L]^*$. Finally, $C[O[C, L]^*]^+C$ is a path beginning with a C-segment, followed by a path of type $[O[C, L]^*]^+$, and ending in a C-segment. By Corollary 2, if the cable does touch obstacles, then τ_c is always of type $C[O[C, L]^*]^+C$ since the O-segments only occur at the beginning or at the end of each subpath. Then each such subpath is in the form of $O[C, L]^*$. If $[C, L]^*$ is an empty string then it is representing a path of zero length which is trivially a shortest curvature constrained path, otherwise since it is a subpath of a locally shortest path and it does not touch any obstacle, by Lemma 3, it is a shortest curvature constrained path. Thus, breaking a locally shortest path into subpaths at the points that it touches the obstacles will yield a set of subpaths that each of them is a shortest curvature constrained path. □

This cable decomposition corollary is the basis for finding the appropriate skeleton of the c-space. Since the environment contains semi-algebraic obstacles, it can be partitioned into a semi-algebraic set M_{obs} consisting of all the obstacles and the free space M_{free} that is the complement of the set M_{obs}[14].

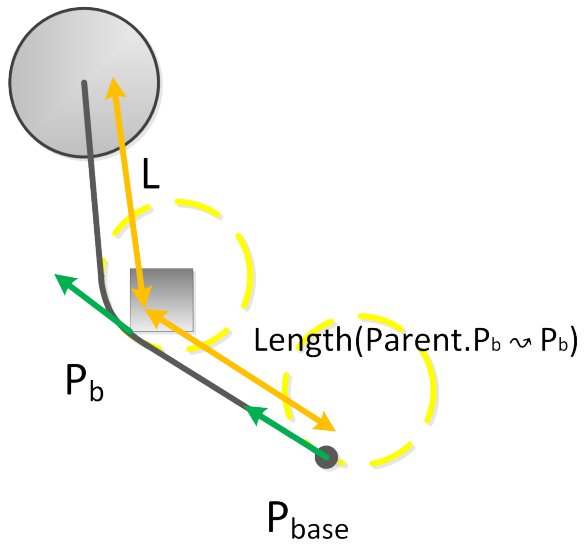


Figure 3.4: The defining attributes of a Dubins Cell. The yellow dotted circles are the boundaries of maximal curvature. The green arrows are the orientations at the base poses of each cell.

Definition 9 (Dubins Cell)**.** *Given a base pose P_b, a Dubins Cell is a chart, (U, φ), where $U \subseteq M_{free}$; the homeomorphism φ is $\varphi(x, y) = (x, y)$; and every point inside U can be reached from P_b via a shortest curvature constrained path.*

A Dubins Cell can be represented with the following four attributes (see Fig. 3.4):

- ***Base Pose***: $P_b = (p_b, v_b)$. p_b is the location of the base and v_b is the orientation of the cable at p_b.
- ***Parent Cell***: a reference or pointer to the cell describing the robot-cable configuration directly before occurrence of the O-segment.
- ***Cable Length***: L, determines the maximum allowed distance between the robot and the base pose. The exact value is $L = Parent.L - Length(Parent.P_b \rightsquigarrow P_b)$

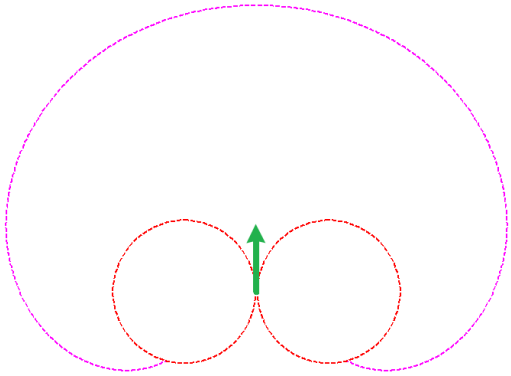


Figure 3.5: A single Dubins Cell in an environment without any obstacle. The boundary of maximal curvature is shown with red dotted circles. The magenta spirals show the boundary of farthest reachable points (without considering a goal pose). The green arrow shows the orientation of the robot at the base pose. Cable's length is 60 units with curvature constraint $\kappa_0 = 1/20$

- ***Stitch Line***: this is the line where one chart is connected to another and can be considered as an interface between them. Formally, it is the domain for the transition map, φ, between the two charts. Once the robot crosses this line, a contact is made or released and the robot will be transfered from a chart (cell) to the other.

Fig. 3.5 shows a single Dubins Cell in an environment without any obstacle. Notice in the figure how the boundary of farthest reachable points is in the shape of an Archimedean spiral, arising from the curvature constraint. To understand why the boundary is this shape, imagine the cable is anchored with pose $P_0 = (p_0, v_0)$ where $p_0 = [r_0, 0]$ and $v_0 = [0, 1]$. If the robot moves along the boundary of maximal curvature for θ radians, then $(r_0\theta)$ of the cable's length will be consumed. Therefore the polar equation $r = l - r_0\theta$ will describe the farthest boundary of the cell. The general equation can easily be obtained.

3.3.6 The Boundaries of a Cell

Due to the curvature constraint of the cable, the boundaries of a cell are dependent on the base pose and goal pose. In this subsection, we would like to provide a better understanding of the boundaries.

Let $c_l(x)$ and $c_r(x)$ be the centers of $C_L(x)$ and $C_R(x)$ respectively. Let $d(p_x, p_y)$ denote the Euclidean distance in $\mathbb{R}^2$ between the points p_x and p_y. There are four possible conditions regarding the Euclidean distance between the centers of the tangent circles of P_i and P_g.

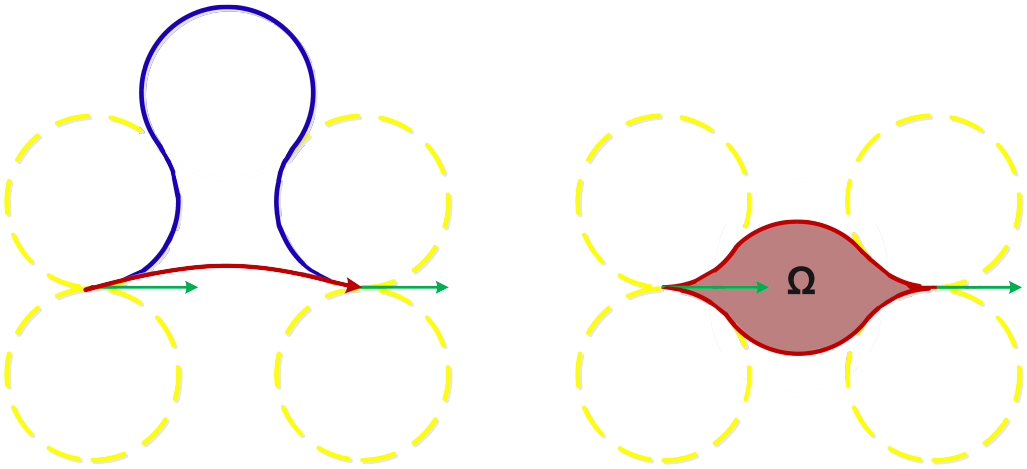
1. $d(c_l(i), c_l(g)) \geq 4$ and $d(c_r(i), c_r(g)) \geq 4$
2. $d(c_l(i), c_l(g)) < 4$ and $d(c_r(i), c_r(g)) \geq 4$
3. $d(c_l(i), c_l(g)) \geq 4$ and $d(c_r(i), c_r(g)) < 4$
4. $d(c_l(i), c_l(g)) < 4$ and $d(c_r(i), c_r(g)) < 4$

Notice that condition 2 and condition 3 give the same planar configuration via reflection. Also, condition 4 raises three possible scenarios [4]:

4.1 Shortest curvature constrained path from P_i to P_g is a single C-segment with length less than πr_0 or the concatenation of two C-segments with a total length less than πr_0.

4.2 The four tangent circles enclose a region Ω. The boundary of Ω is concatenation of six circles with radius r_0 (see Fig. 3.6).

4.3 Shortest curvature constrained path contains either a C-segment with length greater than or equal to πr_0 or a L-segment with length greater than or equal to $4r_0$.



(a) There is no continuous transformation that can convert the blue path to the red path that can maintain the curvature constraint throughout the transformation. The boundaries of maximum curvature (yellow) and the blue path form a bight.

(b) Any path that leaves the Ω region is not homotopic to the paths that are entirely inside the Ω region.

Figure 3.6: The Ω region divides the set of all path going from P_i to P_g into two disjoint sets. One contains all the paths that are completely contained within this region and the other set contains the rest of the paths.

We adopt the following four proximity conditions from Ayala and Hyam in [4].

- **Proximity Condition A.** If P_i and P_g satisfy condition 1.
- **Proximity Condition B.** If P_i and P_g satisfy condition 2 or 3.
- **Proximity Condition C.** If P_i and P_g satisfy condition 4.1 or 4.2.
- **Proximity Condition D.** If P_i and P_g satisfy condition 4.3.

A path $\tau : [0, 1] \to \mathbb{R}^2$ is not in Ω if $\exists s \in [0, 1]$ such that $\tau(s) \notin \Omega$. Therefore, Ω divides all paths from P_i to P_g into two disjoint sets of paths: $\Delta(\Omega)$ set of all paths in Ω, and $\Delta'(\Omega)$ set of all paths not in Ω. The importance of the existence of the Ω region is that the paths in $\Delta(\Omega)$ and $\Delta'(\Omega)$ belong to different homotopy classes [4] and hence they should be uniquely identified. To better understand this see Fig. 3.6.

The blue path in 3.6a is of type CCC. This path cannot be transformed into the red path as we have $d(c_l(i), c_l(g)) < 4$ and $d(c_r(i), c_r(g)) < 4$. Therefore it is not possible to push the middle C-segment of the blue path to go through $C_L(i)$ and $C_L(g)$. In other words, the blue path and $C_L(i)$ and $C_L(g)$ form a bight.

In the next subsection we explain how we are able to uniquely identify all the possible conflagrations of a robot and its taut curvature-bounded cable.

3.3.7 The Atlas

We next define an atlas as a model of the decomposition containing all the Dubins Cells of the c-space. To build the complete atlas, all the possible locations for O, C, and L-segments are required. An algorithm is provided by Agarwal *et al.* [3] that, given an initial pose and a goal pose, outputs a graph which encodes the locations and connections between the segments. However, pre-computing the graph is not required as its parts can be generated on-the-fly. Since the obstacles are polygonal, all the tangent lines and circles will touch the obstacles at their apexes. Therefore, at each step finding the next child cell can be done directly by enumerating collision-free tangents (see Algorithm 4). An atlas, A_l, is associated with a skeleton graph, $G_{A_l} = (V_{A_l}, E_{A_l})$, and represents the topological structure of the atlas. For each Dubins Cell, c_i, in the atlas there is a vertex, v_i, in V_{A_l}. There is an edge, $e_{ij} \in E_{A_l}$ between v_i and v_j, if v_i is the parent cell of n_j. Here, e_{ij} represents the shortest curvature constrained path between the base of c_i and the base of c_j.

Theorem 5 (Completeness of the atlas)**.** *The atlas of Dubins Cells generated on M_{free} describes exactly the space of all the possible configurations of a robot and its taut cable with maximum length l and curvature constraint κ_0 in M_{free}.*

Proof. The tautness of the cable implies each segment of the cable is a locally shortest path. So by Lemma 3, each segment is always in the form of a Dubins path in M_{free}

connecting one cell's base to the next. Starting from P_{base} and connecting all the Dubins paths up to the last base pose, we can construct all the configurations of the taut cable, except the last segment connecting the robot to the final base pose. This last piece of information is provided by the robot's pose, P_r, inside the final Dubins cell. By Definition 9, the path connecting the last base pose to the robot's pose is also a feasible Dubins path. Therefore, by concatenating this tail-end path to the rest, we have all robot and taut tether configurations. □

In order to identify the cable configurations uniquely, we need to store information for the special cases that we have discussed earlier. Firstly, in the case of symmetry between left and right tangent circles, we need to make a distinction between the chosen shortest curvature constrained path as it is not unique. Secondly, some information should be stored to distinguish paths in $\Delta(\Omega)$ from paths in $\Delta'(\Omega)$. Therefore, for each subpath of a locally shortest path we need to store the following tuple:

$$\tau =< P_i, P_g, o_\tau, r_\tau > \tag{3.3}$$

where P_i is the initial pose and P_g is the goal pose; $o_\tau \in \{C_L(i), C_R(i)\}$ indicates whether the path τ starts by turning left or right; $r_\tau \in \{\varnothing, \Delta, \Delta'\}$ indicates whether Ω region exists and if so whether the path is in $\Delta(\Omega)$. Hence, the completeness of the Atlas means that we can obtain every possible configuration of the robot and its cable uniquely.

Corollary 4. *Given an atlas, A_l, its associated skeleton graph, $G_{A_l} = (V_{A_l}, E_{A_l})$, a path in G_{A_l}, $s = [v_{root}, v_1, v_2, ..., v_n]$ where v_n is associated with Dubins Cell (U_n, φ_n), and a point $p \in U_n$, the pair (s, p) can identify a unique configuration of the robot and its taut tether in M_{free}.*

(a) Tangents when they leave a circle. (b) Tangents when they arrive at a circle.

Figure 3.7: Examples of compatible and incompatible tangents. The red and the green tangents in each of the above figures are incompatible and compatible tangents, respectively. The yellow arrow shows the direction of the motion on the circle. The vector of direction at the tangent point should have the same orientation as the same the yellow arrow.

Proof. Follows from Theorem 5 and Lemma 3. □

Theorem 6. *The graph G_{A_l} associated with the atlas A_l generated on M_{free}, is a tree.*

Proof. Suppose, to the contrary, that G_{A_l} has a loop. This implies two distinct sequences of cells and their transition maps reach the same Dubins Cell. Each sequence implies an ordering of poses, that is two distinct configurations of the cable for reaching the same cell. This contradicts Corollary 4. □

3.4 Planning With the Representation

Let L_i be the line tangent to pose $P_i = (p_i, v_i)$, *i.e.,* it passes through p_i and is parallel to v_i; and let $C_L(i)$ and $C_R(i)$ be the circles tangent to L_i at point p_i on the left and on the right side of L_i, respectively, when looking in the direction v_i from p_i.

Algorithm 2 shows pseudo code for finding a Dubins path from P_i to P_g. Line 4, produces four tangents for each pair of circles, of which only one is compatible with direction of both v_i and v_g (see Fig 3.7). Line 5 will remove the tangents that

are incompatible, yielding in only one Dubins path. In line 9, length of each path is calculated and then is compared to the current minimum in line 10. Finding length of a Dubins path is done by accumulating the length of the L-segment and $r_0 \times \theta$ for C-segment(s) (arcs), where θ is the angle traversed on C-segment in radians.

1: **FindShortestCurvatureConstrainedPath**(P_a, P_b)
2: Find $C_L(a)$, $C_R(a)$, $C_L(b)$, and $C_R(b)$
3: $minPath = \varnothing$
4: $t = \{$ all common tangents between $\big(C_i(a), C_j(b)\big); i, j \in \{L, R\}\}$
5: Remove *incompatible* tangents from t
6: **for all** $t_k \in t$ **do**
7: $p_{a,k}$ = point at which t_k touches $C_i(a)$ for $i = R$ or L
8: $p_{k,b}$ = point at which t_k touches $C_i(b)$ for $i = R$ or L
9: $d = Length\big(arc(p_a, p_{a,k}) + (p_{a,k} - p_{k,b}) + arc(p_{k,b}, p_b)\big)$
10: **if** $d < Length(minPath)$ **then**
11: $minPath = arc(p_a, p_{a,k}) + (p_{a,k} - p_{k,b}) + arc(p_{k,b}, p_b)$
12: **end if**
13: **end for**
14: **return** $minPath$

Algorithm 2: Find Shortest Curvature Constrained Path

Algorithm 3 presents pseudo code for finding the solution to the problem of planning from a initial pose, P_i, to a goal pose, P_g, for a planar robot situated in $\mathbb{R}^2$ in the presence of polygonal obstacles. Note that the information regarding the cable's maximum length and its base pose is encoded within *cell* passed as an input argument. The algorithm is initiated by passing the following arguments: current cell in which the robot is situated, P_i, P_g, and an empty path. The if statement in Line 4, checks whether there exists a shortest curvature constrained path which is completely contained inside *cell*. Finding the shortest curvature constrained path is done by Algorithm 2. The condition in line 4 ensures that the robot will not leave *cell*. If so, a *robot path* is created and is appended to the end of the path taken up to this point (see section 3.5.2). Otherwise the algorithm will proceed searching

down the tree in a depth-first fashion and storing the minimum length path that is found (or $\varnothing$ if no such path exists). It is worth mentioning that any arbitrary graph search method can be used in place of the Depth-First Search. The next step is to look up in tree (ancestors) for a solution and compare the result of this search to the minimum length path found in child nodes. It is worth mentioning that the root cell of the atlas does not have parent (the value is set to $\varnothing$). Line 7 is a critical part of Algorithm 3; it allows the method to search for child cells and expand the tree on-the-fly. Algorithm 4 shows the details on how the child cells are found.

1: **FindPath**$(cell, P_i, P_g, path)$
2: mark *cell* as visited
3: $minPath = \varnothing$
4: **if** $cell.P_b \rightsquigarrow P_g$ is inside Dubins Cell **then**
5: $minPath = path.Append$(Find robot's path inside *cell* to P_g)
6: **else**
7: $children =$ **GetChildCells**$(cell)$
8: **for all** $c \in children$ **do**
9: $\tau = path.Append$(Find robot's path inside *cell* to $c.StitchLine$)
10: $\tau =$ **FindPath**$(c, robot.P_r, P_g, temp)$
11: **if** τ is shorter than $minPath$ **then**
12: $minPath = \tau$
13: **end if**
14: **end for**
15: **if** $Parent \neq \varnothing$ and is not visited yet **then**
16: $\tau = path.Append$(Find robot's path inside *cell* to $cell.P_b$)
17: $\tau =$ **FindPath**$(Parent, cell.P_b, P_g, temp)$
18: **if** τ is shorter than $minPath$ **then**
19: $minPath = \tau$
20: **end if**
21: **end if**
22: **end if**
23: **return** $minPath$

Algorithm 3: The Shortest Path Algorithm

1: **GetChildCells**(*cell*)
2: $children = \varnothing$
3: $C = \{C_L(cell.P_b), C_R(cell.P_b)\}$
4: keep *valid* circles of C
5: **for all** $c \in C$ **do**
6: **for all** $p_o \in$ APEXES such that p_o is inside *cell* **do**
7: $L = \{$ tangents to c passing from $p_o\}$
8: remove *incompatible* tangents from L
9: **for all** $l \in L$ **do**
10: $v_o =$ direction of l
11: $P_o = \text{Pose}(p_o, v_o)$
12: $child =$ cell with $Parent = cell$ and $P_b = P_o$
13: $children = children \cup \{child\}$
14: **end for**
15: **end for**
16: **end for**
17: **return** *children*

Algorithm 4: Algorithm for Getting Child Cells

3.5 Discussion of the Method

To demonstrate the method and evaluate its performance, we developed a simulation environment and implemented the algorithm in MATLAB (see Fig. 3.8). In the following sections we will provide our observations.

3.5.1 Memory Consumption

Benefiting from the on-line exploration, this method reduces the number of graph vertices stored in memory during the search. The atlas also uses a special data structure for all of the Dubins Cells stored in the graph. Using the data structure requires only a constant amount of memory regardless of the cell's volume. These memory saving aspects of the method has been discussed in greater details in chapter 2.

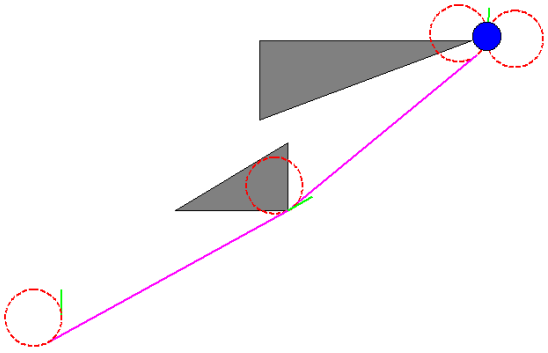
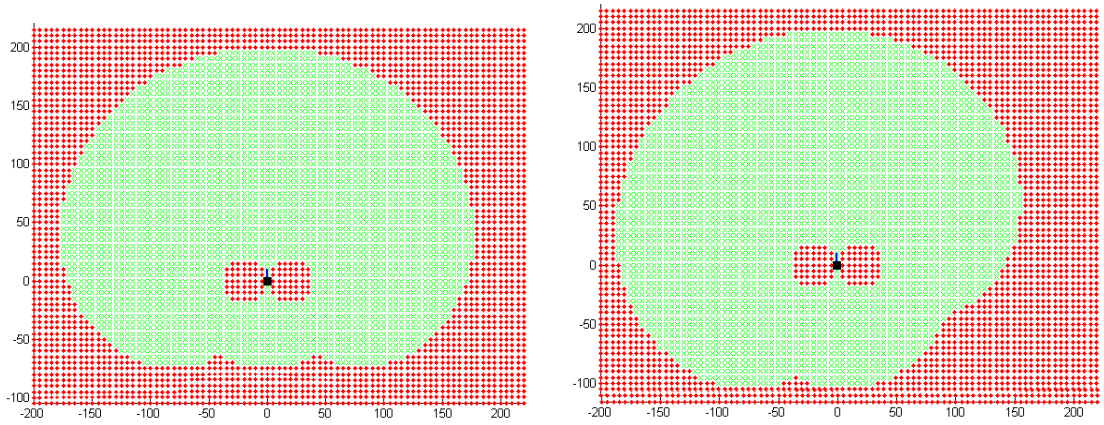


Figure 3.8: The simulation environment. The robot is blue, the dotted red circles show the boundary of maximal curvature, the green arrows show orientation vector in each base pose. The cable is colored in magenta.

3.5.2 Discussion of the Algorithm

The main algorithm is a graph search problem which lazily expands the unexplored parts of the c-space. In this problem, the combinatorial nature of the homotopy classes of cable configurations will result in a space which is exponential in the number of vertices.

However, the fundamental function calls in Algorithm 3 are all polynomial in the number of vertices in the environment, n. Algorithm 2 for finding a Dubins Path is a constant time operation, resulting in $O(n)$ time in line 4. Finding a path for the robot in line 5 is also done in $O(n)$ since all the boundaries of a Dubins Cell are known. The robot is controllable in all three dimensions. Therefore its path inside a cell is a straight line if it is not obstructed by the boundary of maximal curvature or any obstacle. Otherwise, it is a line segment tangent to the boundary followed by a segment supporting to the boundary, followed by another tangent from the boundary



(a) $v_g = (0, 1)$ (b) $v_g = (-1, 1)$

Figure 3.9: The reachability all the points in the environment when they are assigned a goal orientation, v_g. The reachability has been tested from the initial pose $P_i = ((0, 0), (0, 1))$ single Dubins Path. Reachable points and unreachable point are marked by a green crosses and red dots, respectively. The black point and the blue arrow show the initial location and orientation, respectively. The cable's length is 200 units with curvature constraint $\kappa_0 = 1/20$.

to the goal point. Finally, Algorithm 4 returns a set with $O(n)$ elements.

3.5.3 Covering Complex Spaces

One key feature of this method is its capability in modeling the complex configuration space of the robot and its cable efficiently. A naïve representation results in a high dimensional c-space. The use of poses as defined in Definition 7 provides a concise way of encoding the complex Dubins Cells in an atlas. The orientation of the robot at the goal pose, v_g, affects the length of the Dubins paths $P_i \rightsquigarrow P_g$ and consequently the volume and shape of a cell. Using the four attributes of each Dubins Cell (see sections 3.3.5) and the constant time algorithm for finding a Dubins path (see Algorithm 2), this method can evaluate reachability of each pose efficiently. Fig. 3.9 and Fig. 3.10 demonstrates this feature, by comparing the reachable area of three cells from the same initial point, $P_i = ((0, 0), (0, 1))$.

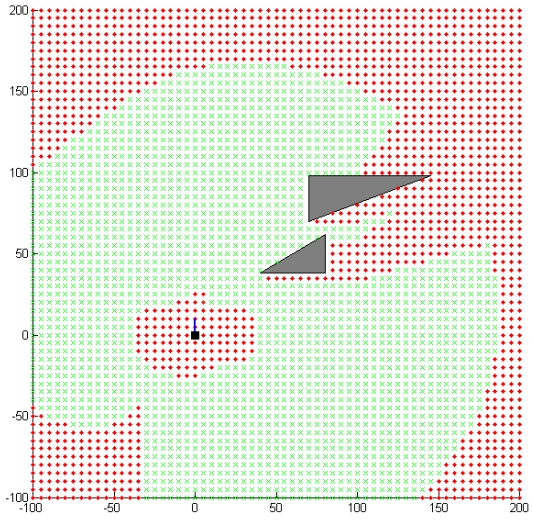
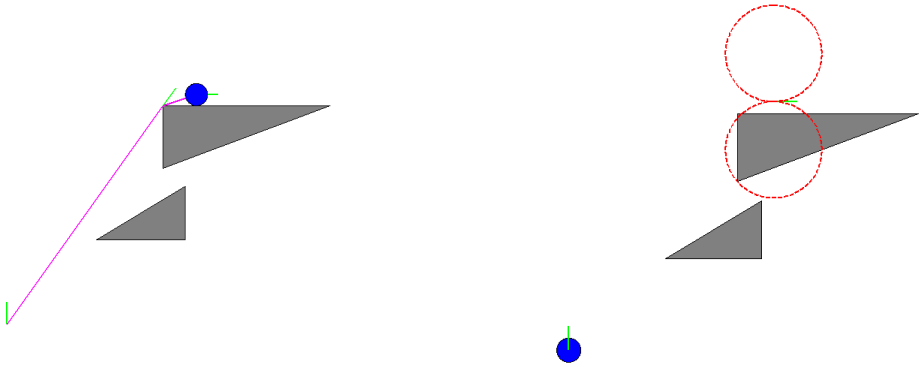


Figure 3.10: $v_g = (1, -1)$. Notice the unreachable region near the initial pose that is caused by the incompatibility of v_i and v_g. Although it seems unintuitive, some points behind the obstacles are still reachable via a Dubins path despite not being physically visible from p_{base}. The properties of the cable and the color code in the figure is the same as Fig. 3.9.

3.5.4 Parameterized Curvature Constraint

Another main advantage of this method is its flexibility in modeling different cable specifications. To demonstrate this capability, we tested the algorithm with different curvature upper-bounds. Fig. 3.11a illustrates a cable without curvature constraint (*i.e.,* r_0 is a positive infinitesimal). As a result, the cable has bent around the corner of the obstacle on the path $P_b \rightsquigarrow P_g$ with sharp angles. In contrast, Fig. 3.11b r_0 shows a non zero positive value ($r_0 = 20$ units) which does not allow the cable to bend freely. The algorithm was not able to find a feasible configuration to solve the problem. It is worth mentioning that going from the right side of the obstacles, in this figure, is not an option due to the short length of the cable (200 units).



(a) A cable whose curvature is unbounded (*i.e.,* $\kappa_0 = 1/\varepsilon$, where ε is a positive infinitesimal). Notice the sharp angles in the cable's configuration.

(b) A cable whose curvature is bounded by $\kappa_0 = 1/20$. In this case a feasible cable configuration has not been found.

Figure 3.11: The effect of parameterized curvature. In both figures, the cable's length is 200 units.

3.6 Conclusion

This chapter addresses the problem of planning motions for tethered robots when the tether has a constraint on its curvature, motivated by the fact that in practice cables have some degree of stiffness which limits curvature. None of the available methods consider this constraint for a tethered robot while constructing the configuration space and, thus, may plan motions that pass through unreachable configurations. Moreover, even in the environments with few obstacles, the topology of the configuration space for a robot with a curvature constrained tether is complex. Therefore, naïve approaches to this problem for getting sound and complete models lead to costly computations.

By relating two distinct areas of motion planing research, *planning for tethered mobile robots* and *nonholonomic motion planing*, we tackle this problem in the modeling phase. We use a more general version of the decomposition method introduced in Chapter 2, to separate topological regularities from the continuous aspects of the

c-space by representing it with an atlas and its accompniying graph. We also employ Dubins's theory [11] to exclude infeasible cable configurations from the robot-cable c-space. We present an algorithm to explore the c-space lazily while searching it. Our method benefits from the on-line exploration to minimize the memory consumption. The use of the parameterized curvature constraint, we believe, makes this method suitable for solving several related problems.

There are other properties of a cable which can be considered during modeling phase to add to the realism the c-space, namely, thickness of a tether. The decomposition technique, itself, is applicable to many other similar scenarios.

4. OUTLOOK: CONNECTED ROBOTS

As a future topic, we are interested in developing a general method for planning motions of two robots, r^+ and r^-, that are connected to each other via a cable. We believe this problem can be addressed using the methods provided in this thesis for finding partial solutions. These solutions are primarily intended to be used as guidance for finding the globally optimal solution for future research.

4.1 Problem Description

We will refer to this problem as the *two tethered robot motion planning (2TRMP)*. In order to develop the foundation of the method, we assume the obstacles and robots are points in $\mathbb{R}^2$ for simplicity. Once we have the method for the point obstacles and robots, it is relatively straightforward to generalize it to other polygonal shapes. We also assume, similar to our preliminary work, that the cable is always taut (*e.g.,* retracting cable); therefore it is always compromised of a set of straight line segments. The inputs and outputs of the method are described below.

4.1.1 Inputs

The following is the list of information that we believe the algorithm requires to solve the problem.

- two points, $P^s_{r^+} \in \mathbb{R}^2$ and $P^s_{r^-} \in \mathbb{R}^2$, indicating the current (source) locations of the (non-oriented) robots r^+ and r^- respectively,
- two points, $P^d_{r^+} \in \mathbb{R}^2$ and $P^d_{r^-} \in \mathbb{R}^2$, indicating the destination of the (non-oriented) robots r^+ and r^- respectively,
- a path π in $\mathbb{R}^2$ indicating the current configuration of the cable where in $\pi(0) =$

P_{r^+} and $\pi(1) = P_{r^-}$, and

- a set of point obstacles, O, in the environment where $\forall o_i \in O : o_i \in \mathbb{R}^2$.

4.1.2 Outputs

Given this input, the method is expected to provide as output two separate paths, τ_{r^+} and τ_{r^-}, for robots r^+ and r^- to reach their respective destinations, such that, if the two paths are executed simultaneously, the length constraint of the cable is never violated. In general the execution phase of the robot may involve one robot waiting for the other if the cable is taut. We believe this does not need to be provided as output, since the robots can compute it locally. Otherwise, if no such pair of paths exist, the method will correctly indicate so. The optimality of the solution, however, is dependent on the specific cost function and the associated notion of optimality in the problem at hand.

4.2 Our Approach to the Problem

Figure 4.1 shows an example of initial configuration described in the previous subsection. We begin solving this problem by checking the distance between $P^d_{r^+}$ and $P^d_{r^-}$. If this distance is more than the cable length connecting the two robots, obviously there does not exist a solution to the problem. Otherwise we continue by partitioning the set of obstacles into three subsets (s_1, s_2, and s_3) by Algorithm 5.

Figure 4.2 shows how the obstacles are partitioned after using Algorithm 5. From now on, we will refer to the obstacles in s_1, s_2, and s_3 as black, red, and green obstacles respectively. Algorithm 5 will mark all the obstacles currently lying on the cable as black. The green obstacles are the ones that fall inside an ellipse whose foci are $P^d_{r^+}$ and $P^d_{r^-}$ and its semimajor axis is $L/2$ (see Figure 4.2). That is, if during the execution of a given solution the cable contacts one (or possibly more)

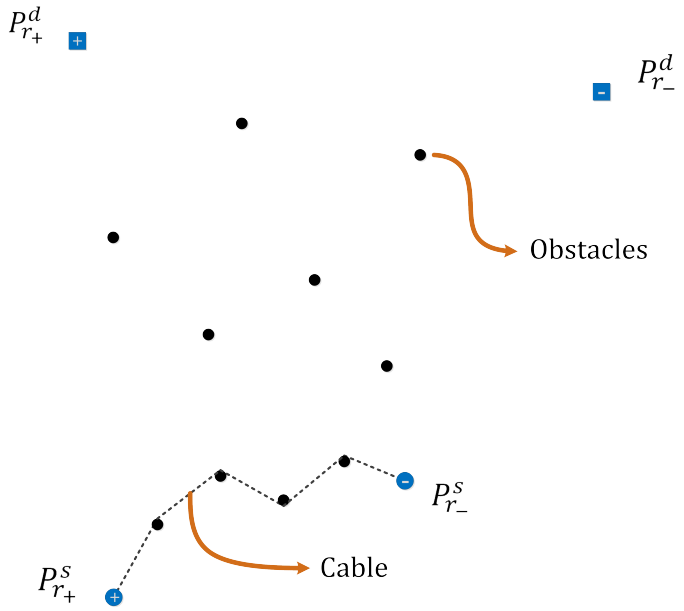


Figure 4.1: An example of how the initial configuration may appear. The dotted line is the cable and black dots are the obstacles.

of the green obstacles, the robots are still able to reach their destinations. The rest of the obstacles are marked red. After execution of any feasible solution the final configuration of the cable never has a contact with any red obstacle. We will then use these subsets in order to break the problem into subproblems that will lead us to the optimal solution.

Definition 10 (Category 1 (C1) solution)**.** *a solution to the 2TRMP problem that keeps at least one black obstacle in contact with the cable after its execution.*

Definition 11 (Category 2 (C2) solution)**.** *any non-C1 solution to the 2TRMP problem.*

Proposition 1. *The optimal solution always starts with detaching the cable from zero or more obstacles and continues with attaching the cable to zero or more obstacles.*

Proposition 2. *The shortest C1 solution is always shorter than all the C2 solutions.*

1: **Partition**(O)
2: **for all** $o_i \in O$ **do**
3: define $s_1, s_2, s_3 = \varnothing$
4: **if** o_i is currently on the cable **then**
5: $s_1 = s_1 \cup \{o_i\}$
6: **else if** $distance(o_i, P^d_{r^+}) + distance(o_i, P^d_{r^-}) \leq$ Cable's length **then**
7: $s_2 = s_2 \cup \{o_i\}$
8: **else**
9: $s_3 = s_3 \cup \{o_i\}$
10: **end if**
11: **end for**
12: **return** s_1, s_2, s_3

Algorithm 5: Algorithm for partitioning the obstacles.

The above propositions provide a starting point for developing a method to solve the described problem. We will begin our search by looking for C1 solutions. If no C1 solutions are found, we will look into C2 solutions and pick the optimal, if such a solution exists. Due to the combinatorial nature of C2 solutions, the process of finding the optimal C2 solution is much more computationally expensive than finding a C1 solution. We will incorporate our earlier work in the process of finding the optimal C1 solution. We are looking to find ways to eliminate the classes of C2 solution to reduce the complexity of the problem. In the next Section we will provide the current state of our research on this problem.

4.3 Approaching Category 1 Solution

In this section we will discuss how we find a C1 solution in more detail. Let us first look at the initial configuration of the two robots and their cable. We will then describe how we can determine existence of a C1 solution by solving the given configuration as a Dynamic Programming using any planner for tethered robots. This, of course, is an in-progress research and the details of this method can be further refined.

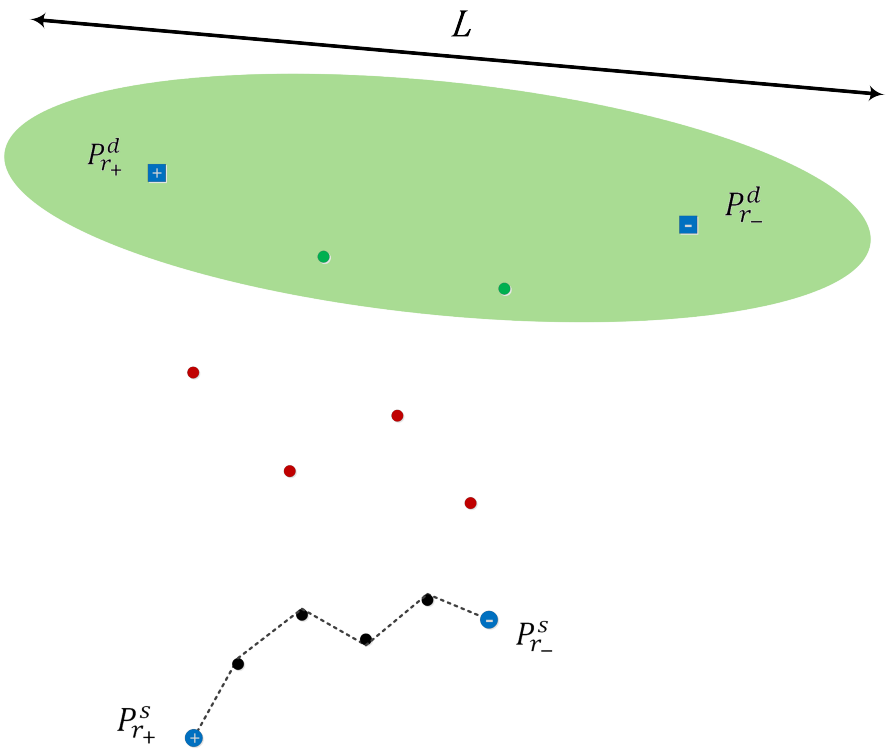


Figure 4.2: The obstacles after partitioning. Black, red, and green obstacles belong to s_1, s_2, and s_3 respectively.

4.3.1 Initial Configuration

Figure 4.3 shows an example of the initial configuration of the 2TRMP problem. Finding a C1 solution requires the cable connecting the two robots to be in touch with at least one point obstacle in the environment. This means that the two robots are residing in two different visibility cells. It also means the atlas graph (MRLVAG) representing the structure of their c-space is different[1]. The key to finding a proper method to address this problem is that the two robots have a shared amount of cable to consume for reaching their respective destinations. For example, let us imagine that the shortest motion that takes robot r^+ to its destination consumes L units of

[1]The readers are encouraged to familiarize themselves with the notion of atlas representation of the c-space and its associated graph by referring to Chapter 2.

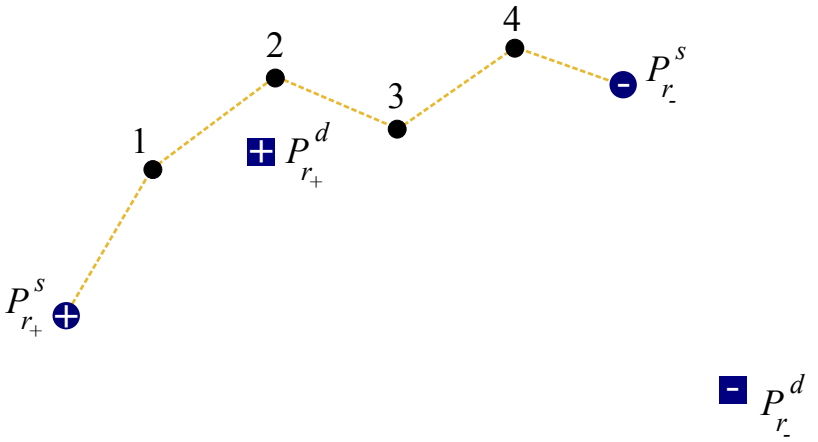


Figure 4.3: An example of cable configuration prior to finding a C1 solution. $P^s_{r^+}$ and $P^d_{r^+}$ indicate the source and destination of robot r^+'s motion, respectively. The same is true for $P^s_{r^-}$ and $P^d_{r^-}$ and robot r^-. The dotted orange curve illustrates the cable's configuration.

cable. This means there is no cable left for r^- to reach its destination. Therefore, r^+ should — probably — compromise on a longer motion to free some cable for r^-. We will elaborate on this in the next subsection.

4.3.2 Compromising for the Shared Cable Length

As outlined above, the motion taken by one robot changes the accessible area for the other robot. Let us take Figure 4.3 as an example. To understand how the selected motion by r^+ affects the accessible radius to r^-, we will keep r^- static and see how the motion chosen by r^+ affects that radius. Figures 4.4 – 4.7 and Table 4.2 demonstrate this intuition supporting the fact that the planning process for robot r^- can be eased significantly by robot r^+ taking a longer trajectory for reaching its destination. Figure 4.4 shows the shortest path available to r^+ to reach its destination. However, if r^+ chooses to take this trajectory, the remaining cable is not long enough for r^- to take the direct line segment to $P^d_{r^-}$. To overcome this, robot r^+ can choose to free some cable for r^- by untangling the cable from obstacle number 1, as shown in Figure 4.5. Nevertheless, this longer trajectory still does not

provide enough cable to put $P^d_{r^-}$ inside the reachable radius of r^-. If the goal for r^+ is to free enough cable for r^- so that it can the direct line segment from $P^s_{r^-}$ to $P^d_{r^-}$, then r^+ can take an even longer path by untangling obstacles 1 and 2 and then going to $P^d_{r^+}$. At this point, as shown in Figure 4.6, r^- has enough cable to move directly towards $P^d_{r^-}$. Finally, Figure 4.7 shows the configuration in which maximum possible cable is free by r^+ while still keeping one obstacle in touch with the cable. Table 4.2 shows the traveled distances by r^+ associated to each of these figure.

Unlike the example above, it is not entirely the responsibility of one the robots to free the cable. We could have assumed that r^+ will stand still while r^- frees the necessary cable. Cooperation between the robots can lead to an optimal distance traveled in total by both robots for which enough cable is available for the two of them. The question then becomes, which pair of trajectories would provide the minimum traveled distance.

Obstacles No.	Distance
1, 2	17.4
2, 3	15.9
3, 4	16.9

Table 4.1: the distances between the obstacle in touch with the cable in the initial configuration (Figure 4.3). The distances are in the same arbitrary unit.

4.3.3 A Dynamic Programming Approach for finding the Optimal Pair of Paths

To find the optimal solution we have introduced a Dynamic Programming approach. We will consider all the solutions that fit into the definition of a C1 solution. For example, in Figure 4.3 it is infeasible to consider a solution in which r^+ untangles up to obstacle number 3 and r^- untangles up to obstacle number 2. That is because

Figure No.	Motion Steps by r^+	Traveled Distance	Cable Config. after r^+'s Motion	Cable Consumed by r^+	Cable Left for r^-
4.4	$P^s_{r^+}, P^d_{r^+}$	28.3	$P^d_{r^+}, 1, 2, 3, 4, P^s_{r^-}$	75.9	24.1
4.5	$P^s_{r^+}, 1, P^d_{r^+}$	31.9	$P^d_{r^+}, 2, 3, 4, P^s_{r^-}$	55.4	44.6
4.6	$P^s_{r^+}, 1, 2, P^d_{r^+}$	46.1	$P^d_{r^+}, 3, 4, P^s_{r^-}$	45.3	54.7
4.7	$P^s_{r^+}, 1, 2, 3, P^d_{r^+}$	67.8	$P^d_{r^+}, 4, P^s_{r^-}$	43.6	56.4

Table 4.2: The effect of the chosen motion by r^+ on r^- in Figures 4.4 – 4.7. The distances and cable lengths in the table are expressed in the same arbitrary unit (the same as Table 4.1). The maximum cable length is 100 units. This table only considers the shortest paths in each homotopy class from $P^s_{r^+}$ to $P^d_{r^+}$ that the path does not make a new cable-obstacle contact.

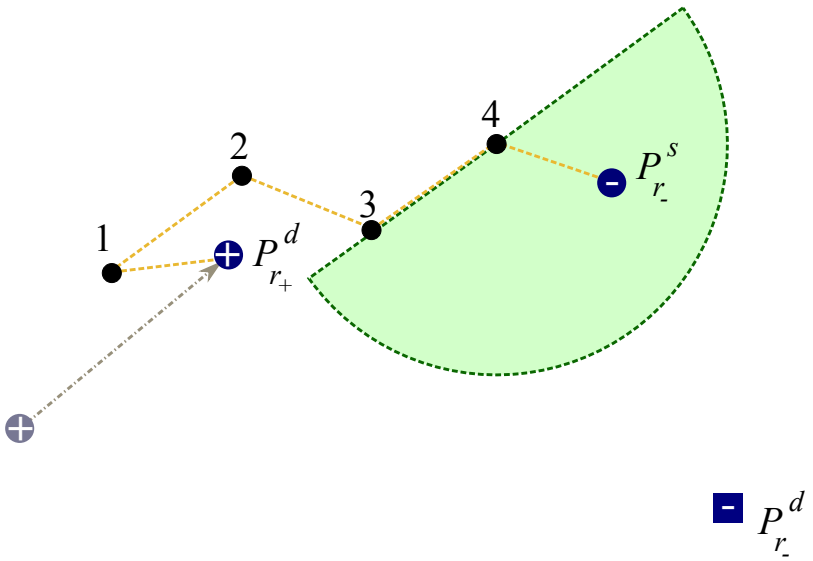


Figure 4.4: Robot r^+ goes directly from $P^s_{r^+}$ to $P^d_{r^+}$. The green circle sector shows the radius that is reachable to r^- with the remaining cable length.

in the process of executing such pair of paths the cable will be released from all the obstacles at some point.

To find the optimal solution to the problem we construct several n-element arrays. The first array, denoted by d_{r^+}, where the i^{th} element holds the shortest path for r^+ from $P^s_{r^+}$ to $P^d_{r^+}$ such that after the execution of the motion the cable is still in touch with obstacle i. A similar array, denoted by d_{r^-}, is constructed for r^-. We also store

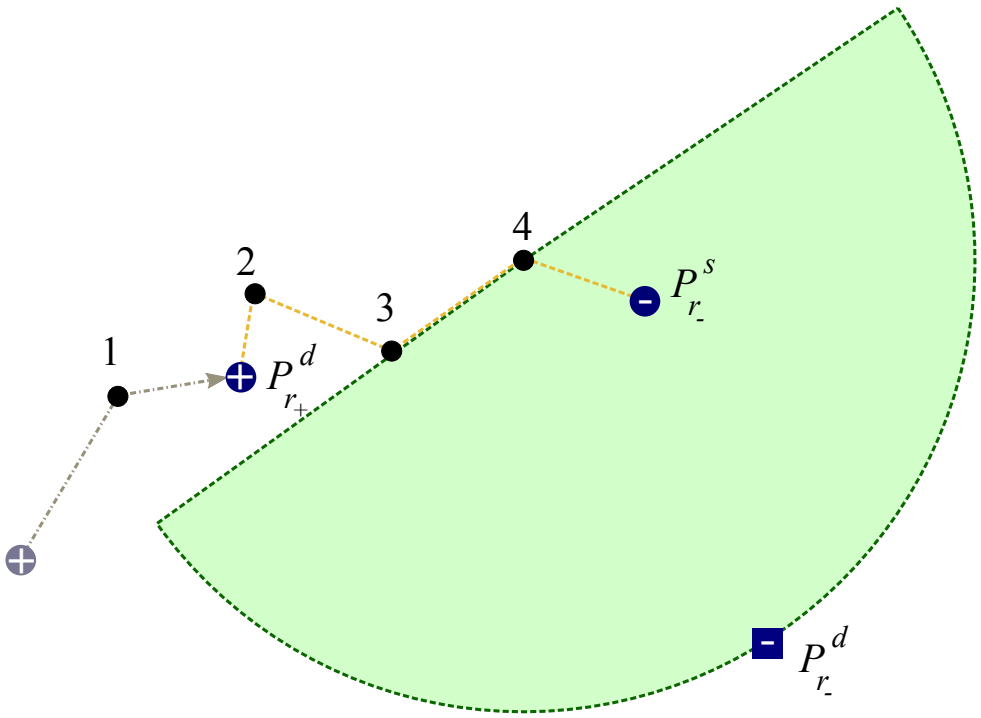


Figure 4.5: Shortest path for r^+ that untangles obstacle number 1 while going from $P^s_{r^+}$ to $P^d_{r^+}$. The green circle sector shows the radius that is reachable to r^- with the remaining cable length.

the consumed cable associated to the paths in d_{r^+} and d_{r^-} in two n-element arrays, c_{r^+} and c_{r^-}. The i^{th} element in c_{r^+} holds the length of the cable consumed from obstacle i to the destination.

For instance, in the configuration shown in Figure 4.3, $d_{r^+} = [28.3, 31.9, 46.1, 67.8]$. These values are the same as the distances in third column of Table 4.2 and their associated path can be seen in the second column of that table. We also have $c_{r^+} = [12.5, 9.3, 15.2, 30.4]$. The distances in c_{r^+} are assumed to be an output of the planner.

Once the values for d_{r^+}, d_{r^-}, c_{r^+}, and c_{r^-} have been determined finding the optimal solution is straightforward. Algorithm 6 shows the procedure for filling the arrays (lines 3 through 10) and finding the shortest path that does not violate the

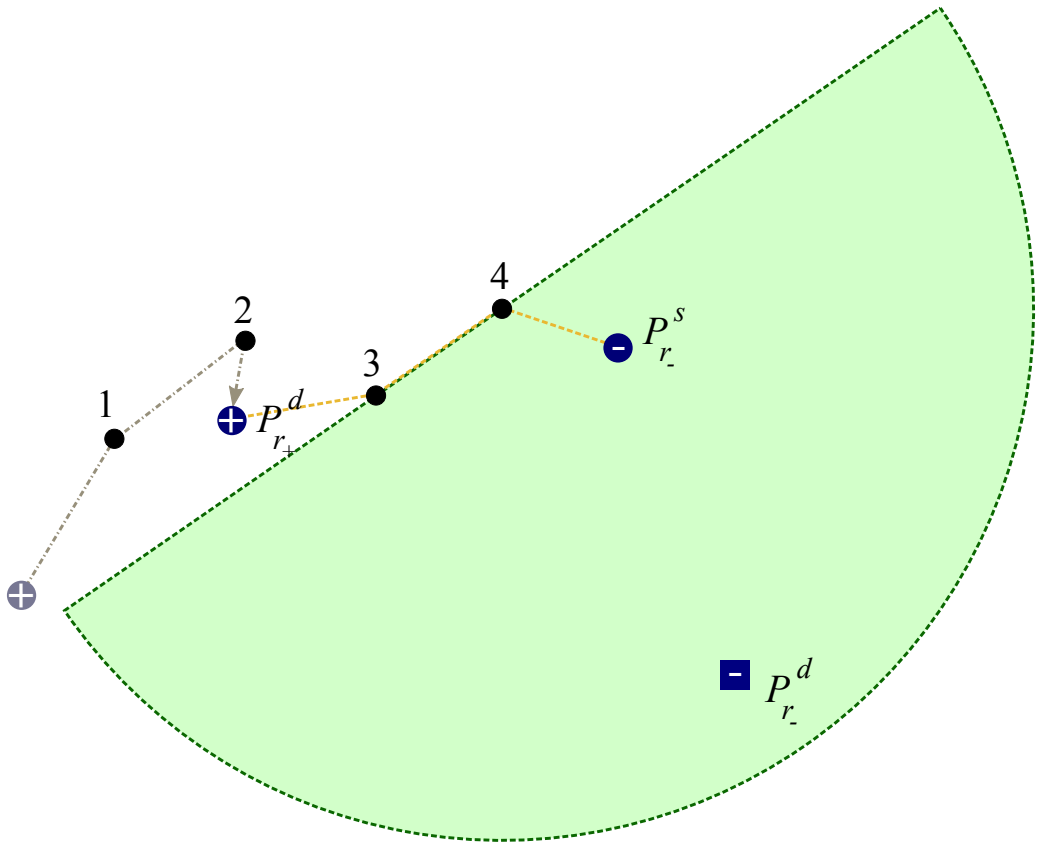


Figure 4.6: Shortest path for r^+ that untangles obstacle number 1 and 2 while going from P^s_{r+} to P^d_{r+}. The green circle sector shows the radius that is reachable to r^- with the remaining cable length.

cable length. The shortest path here is the path that requires minimum sum of traveled distance by the two robots. Something to note here is that this procedure does not rely on any specific planner (lines 4 and 7), as long as it is returns the path length and the length of the cable that is required for that path.

4.4 Conclusion

This Chapter provided a brief description of current state of our research on the motion planning problem for two robots that are connected to each other with a cable. The examples in this Chapter were simplistic to provide an easier understanding to

```
1: index_+ = −1; index_− = −1; minPath = ∞
2: d_{r+} = ∅; d_{r−} = ∅; c_{r+} = ∅; c_{r−} = ∅;
3: for all o_i ∈ s_1 do
4:     [d_tmp, c_tmp] = Plan From o_i to P^d_{r+}
5:     Add d_tmp to d_{r+}
6:     Add c_tmp to c_{r+}
7:     [d_tmp, c_tmp] = Plan From o_i to P^d_{r−}
8:     Add d_tmp to d_{r−}
9:     Add c_tmp to c_{r−}
10: end for
11: for i = 1 to s_1.Length do
12:     for j = i to s_1.Length do
13:         c = c_{r+}[i] + c_{r−}[j]
14:         d = d_{r+}[i] + d_{r−}[j]
15:         end = min(j, s_1.Length − 1)
16:         for k = i to end do
17:             c = c+ distance between o_k and o_{k+1}
18:         end for
19:         if c <= Cable Length then
20:             if d <= minPath then
21:                 minPath = d
22:                 index_+ = i
23:                 index_− = j
24:             end if
25:         end if
26:     end for
27: end for
```

Algorithm 6: Algorithm for finding C1 solution.

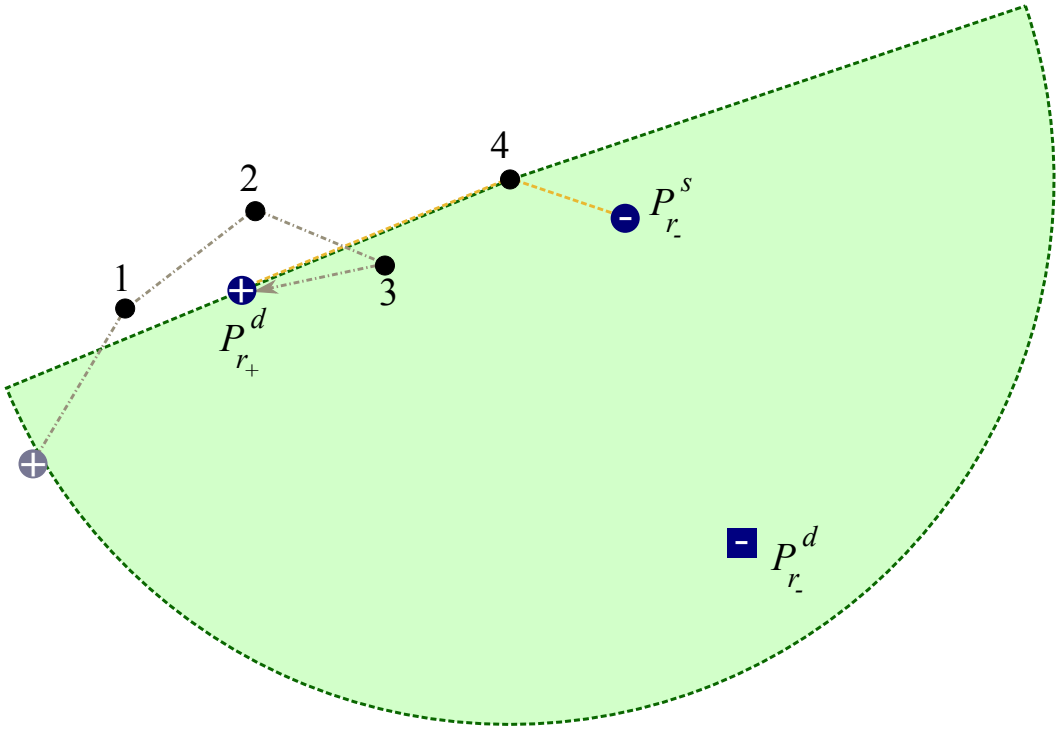


Figure 4.7: Shortest path for r^+ that untangles obstacle number 1, 2, and 3 while going from $P^s_{r^+}$ to $P^d_{r^+}$. The green circle sector shows the radius that is reachable to r^- with the remaining cable length.

the readers. However we are working to gather more insight into its behavior in more complicated cases and the general performance of the method.

5. CONCLUSION

In this work we approached the motion planning problem for a point tethered robot with a new perspective. Throughout this thesis we made three assumptions: a) the robot is a point robot, b) the cable connected to the robot is always taut (possibly retracting), and c) the obstacles are polygonal. Although the work is motivated by motion planning, the most important take away is the decomposition technique which makes representation of the configuration space (c-space) intuitive and concise.

We began with a simplified model of a point tethered robot in which the cable was flexible. We considered the two constraints imposed by the cable on the motions of a tethered robot: 1) limited radius of movement and 2) the topological constraints. Using the notion of *cable events*, we were able to store the configurations of the cable via a simple structure, which we called *visibility cell*, that represents a continuous subspace of $\mathbb{R}^2$. We observed that in an environment with polygonal obstacles, the set of all the visibility cells constructs an atlas, MRLVA, that covers the c-space. This naturally separates the discrete structure of the c-space as a skeleton graph, MRLVAG, from its continuous aspect. Each node in the MRLVAG represents a visibility cell in which finding the shortest path from one point in the cell to another point in that same cell is trivial (a constant time procedure). We pointed out that using this representation of the c-space enables us to build the necessary parts of the c-space as needed rather than generating the complete c-space by performing an offline — and potentially costly — pre-computation.

We next showed how using the notion of discrete events can help us capture cable-cable contacts. In this case we assumed the cable has infinite friction when it

is wrapped around itself. In doing so, we added two new types of events. *over* event represents a change in the configuration of the cable when the robot crosses its own cable by going over the cable. Similarly, *under* events were used to store the event of crossing the cable by going under the cable. The addition of these two types of events underscores the flexibility of this technique in addressing problems of interest.

Finally, We added to the realism of our approach by considering a parameterized curvature constraint. A curvature constraint is one that limits how much a cable can be bent at every point from the beginning to the end of the cable. The parameterized model allows this method to be applicable to both stiff and flexible cables. In this case we also considered an oriented point robot as the orientation of the goal changes the amount of cable consumed by the motion that reaches the goal. This modification significantly adds to the complexity and dimensionality of the c-space. To overcome this complexity the notion of *reachability* was used instead of visibility. Thus we were still able to use the exact same cell structure as before for constructing the atlas of c-space. To asses reachability, we utilized Dubins' theory to be able to find the shortest path inside a cell in a constant time.

This work emphasizes on the fact that one key technique for overcoming complex and high dimensional spaces is finding the appropriate cell decomposition. As we saw in the previous sections, the main difference between a flexible cable and a stiff one is in the constant time procedure of finding a shortest path in a continuous space that contains no holes. Once the basis for decomposition has been determined, the configurations of the cable becomes implicit in the c-space structure. In other words, the cable that can take many different configurations is flattened into the skeleton (graph structure) of the c-space. This view point is not limited to the problem of a tethered robot as the cable can be a virtual one modeling communication network.

REFERENCES

[1] Pablo Abad-Manterola, Issa AD Nesnas, and Joel W Burdick. Motion planning on steep terrain for the tethered axel rover. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 4188–4195. IEEE, 2011.

[2] Pankaj K Agarwal, Therese Biedl, Sylvain Lazard, Steve Robbins, Subhash Suri, and Sue Whitesides. Curvature-constrained shortest paths in a convex polygon. *SIAM Journal on Computing*, 31(6):1814–1851, 2002.

[3] Pankaj K Agarwal, Prabhakar Raghavan, and Hisao Tamaki. Motion planning for a steering-constrained robot through moderate obstacles. In *Proceedings of the twenty-seventh annual ACM symposium on Theory of computing*, pages 343–352. ACM, 1995.

[4] José Ayala and Hyam Rubinstein. Non-uniqueness of the homotopy class of bounded curvature paths. *arXiv preprint arXiv:1403.4911*, 2014.

[5] Subhrajit Bhattacharya, Hordur Heidarsson, G Sukhatme, and Vijay Kumar. Cooperative control of autonomous surface vehicles for oil skimming and cleanup. In *Robotics and automation (ICRA), 2011 IEEE international conference on*, pages 2374–2379. IEEE, 2011.

[6] Subhrajit Bhattacharya, Vijay Kumar, and Maxim Likhachev. Search-based path planning with homotopy class constraints. In *Annual Symposium on Combinatorial Search*, 2010.

[7] Subhrajit Bhattacharya, Maxim Likhachev, and Vijay Kumar. Topological constraints in search-based robot path planning. *Autonomous Robots*, 33(3):273–290, 2012.

[8] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms, Third Edition*. MIT Press, 2009.

[9] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry, Algorithms and Applications*. Springer, 1991.

[10] Bruce Donald, Larry Gariepy, and Daniela Rus. Distributed manipulation of multiple objects using ropes. In *IEEE International Conference on Robotics and Automation (ICRA)*, volume 1, pages 450–457. IEEE, 2000.

[11] Lester E Dubins. On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. *American Journal of mathematics*, pages 497–516, 1957.

[12] Steven Fortune and Gordon Wilfong. Planning constrained motion. *Annals of Mathematics and Artificial Intelligence*, 3(1):21–82, 1991.

[13] Herbert Goldstein. *Classical Mechanics*. Addison-Weseley, 2nd edition, 1980.

[14] Dima Grigoriev and Anatol Slissenko. Polytime algorithm for the shortest path in a homotopy class amidst semi-algebraic obstacles in the plane. In *Proceedings of the 1998 international symposium on Symbolic and algebraic computation*, pages 17–24. ACM, 1998.

[15] Allen Hatcher. *Algebraic Topology*. Cambridge University Press, 2002.

[16] Susan Hert and Vladimir Lumelsky. The ties that bind: Motion planning for multiple tethered robots. *Robotics and autonomous systems*, 17(3):187–215, 1996.

[17] Takeo Igarashi and Mike Stilman. Homotopic path planning on manifolds for cabled mobile robots. In *Algorithmic Foundations of Robotics IX*, pages 1–18. Springer, 2011.

[18] Paul Jacobs and John Canny. Planning smooth paths for mobile robots. In *Nonholonomic Motion Planning*, pages 271–342. Springer, 1993.

[19] Soonkyum Kim, Subhrajit Bhattacharya, Hordur Kristinn Heidarsson, Gaurav Sukhatme, and Vijay Kumar. A topological approach to using cables to separate and manipulate sets of objects. In *Robotics: Science and Systems*, 2013.

[20] T Krick, AO Slisenko, Pablo Solernó, and J Heintz. Search for shortest path around semialgebraic obstacles in the plane. *Journal of Mathematical Sciences*, 70(4):1944–1949, 1994.

[21] Florent Lamiraux and J-P Lammond. Smooth motion planning for car-like vehicles. *Robotics and Automation, IEEE Transactions on*, 17(4):498–501, 2001.

[22] Steve Lavalle. *Planning algorithms*. Cambridge University Press, 2006.

[23] John M Lee. *Introduction to smooth manifolds*, volume 218. Springer, 2012.

[24] Tomás Lozano-Pérez. Spatial planning: A configuration space approach. *Computers, IEEE Transactions on*, 100(2):108–120, 1983.

[25] Tomás Lozano-Pérez and Michael A Wesley. An algorithm for planning collision-free paths among polyhedral obstacles. *Communications of the ACM*, 22(10):560–570, 1979.

[26] Venkatraman Narayanan, Paul Vernaza, Maxim Likhachev, and Steven M LaValle. Planning under topological constraints using beam-graphs. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 431–437. IEEE, 2013.

[27] James Reeds and Lawrence Shepp. Optimal paths for a car that goes both forwards and backwards. *Pacific journal of mathematics*, 145(2):367–393, 1990.

[28] Stuart Jonathan Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2009.

[29] Iddo Shnaps and Elon Rimon. Online coverage by a tethered autonomous mobile robot in planar unknown environments. In *Proceedings of Robotics: Science and Systems*, Berlin, Germany, June 2013.

[30] Ryo Takei, Richard Tsai, Haochong Shen, and Yanina Landa. A practical path-planning algorithm for a simple car: a hamilton-jacobi approach. In *American Control Conference (ACC), 2010*, pages 6175–6180. IEEE, 2010.

[31] Reza H. Teshnizi and Dylan A. Shell. Computing cell-based decompositions dynamically for planning motions of tethered robots. In *Robotics and automation (ICRA), 2014 IEEE international conference on*, 2014.

[32] Patrick G Xavier. Shortest path planning for a tethered robot or an anchored cable. In *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, volume 2, pages 1011–1017. IEEE, 1999.

[33] Dmitry S Yershov, Paul Vernaza, and Steven M LaValle. Continuous planning with winding constraints using optimal heuristic-driven front propagation. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 5551–5556. IEEE, 2013.

APPENDIX A

MAXIMUM RAY LENGTH VISIBILITY ATLAS

Given P_{free}, a fixed point $p_0 \in P_{free}$ as the origin, and a maximum length l, the following algorithm will produce an atlas of charts, which we will call Maximum Ray Length Visibility Atlas (MRLVA) denoted by A_l, and an associated graph representing its topological structure, which we call Maximum Ray Length Visibility Atlas Graph (MRLVAG).

We define a chart (U_0, φ_0), where $U_0 = P_{free}$ and homeomorphism φ_0 is defined as $\varphi_0(x, y) = (x, y)$. Then atlas A_l is initially $\{(U_0, \varphi_0)\}$. Also we create the graph $G = (V, E)$ with a single vertex u_0 in V representing U_0 and $E = \varnothing$. We also need queues Q_1 and Q_2 for storing the contact points.

We then create a straight line segment with one of its endpoints at p_0 and length l. All the points that are within line of sight from p_0 can be reached without the cable being bent around an obstacle. Therefore, according to Definition 2 all these points belong to the same visibility cell. To find these points, we sweep the line for 2π radians around p_0 and remove all the points from U_0 which fall into the shadow of an obstacle. That is, we rotate this line segment around p_0 counterclockwise starting from angle 0 until it touches an obstacle. Let p_1 be the point around which the line segment should bend. If α_1 is the associated angle, we enqueue a pair (p_1, α_1) to Q_1 and continue rotating the line in the same direction until it is again tangent to this obstacle with angle α_2. Let p_2 be the tangent point. We will enqueue the pair (p_2, α_2) to Q_2. All the points between α_1 and α_2 that are farther from p_0 than the obstacle's boundary are removed from U_0 and are added to two new sets U_1 and U_2. We create two new charts (U_1, φ_1) and (U_2, φ_2) where again we have $\varphi_2(x, y) = \varphi_1(x, y) = (x, y)$

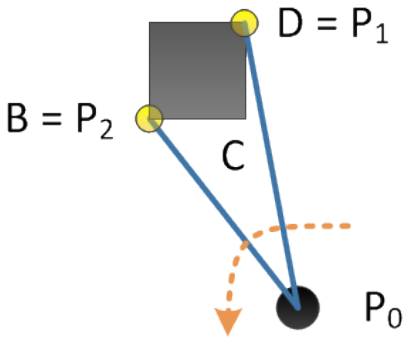


Figure A.1: B, C and D are visible vertices from P_0. Only B and D are enqueued in the algorithm.

and add them to the atlas A_l. Now we add two new vertices u_1 and u_2 to V that represent the two charts, adding edges (u_0, u_1) and (u_0, u_2) to E, which show that U_0 is the parent of U_1 and U_2.

We continue rotating the line segment and collecting the bending points as described in the above paragraph until it has rotated the whole unit circle (2π).

Next we will dequeue a pair (p_i, α_i) from Q_1 and execute the following procedure. It is similar to above, but with a few differences to remove unreachable points from chart (U_i, φ_i). Let u_j be the parent node of u_i in G. We create the straight line that one of its endpoints is p_i. The length of the line is $l' = l - distance(p_j, p_i)$ that is less than l because p_i is distinct from p_j. Thus, the volume covered by U_i is smaller than U_j. Again we want to remove the points from U_i that are not in line of sight from p_i. To do so, we rotate this line segment counterclockwise starting from α_i. While rotating we collect the bending points and angles in queue Q_1. But, this time we do not rotate 2π radians. Instead we rotate until the line is tangent to the obstacle that p_i belongs to, because sweeping the line more than that will cover the points that are previously seen from p_j. We do this until Q_1 is empty. The same procedure is performed for all the pairs in Q_2, except with clockwise rotation.

Since we have removed points from U_j (parent chart) and have added them to U_i (the child chart), it is clear that $U_m \bigcap U_n = \varnothing$ iff $(u_m, u_n) \notin E$ and $(u_n, u_m) \notin E$. If there is an edge between u_m and u_n the intersection will be exactly the line that stitches these two charts together. Therefore, a transition map between chart (U_m, φ_m) and chart (U_n, φ_n) is defined only if there is an edge between u_m and u_n in G.

It is important to notice that all the points that are enqueued in Q_1 and Q_2 are apexes of the obstacles, which are the vertices of the visibility graph [9]. However, not all of the apexes are enqueued because the line segment will not bend around them (see Fig. A.1). As a result the number of child charts created is never more than the number of vertices in visibility graph.