Computing Cell-Based Decompositions Dynamically for Planning Motions of Tethered Robots *

Reza H. Teshnizi and Dylan A. Shell †

February 13, 2014

Abstract

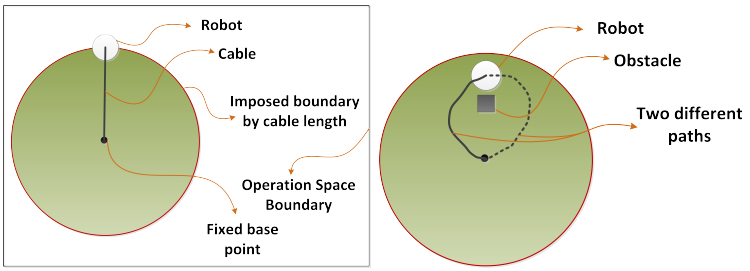
Recently researchers have approached the problem of motion planning with topological constraints. In such problems, the inputs to the planner are source and destination points and the output is expected to be a valid path that respects the topological constraints. A concrete example of such problems —and the topic of this paper— is planning for a robot which is connected with a cable of limited length to a fixed point in the operation space. This paper presents a planning method for such problems by examining how the configuration space manifold can be represented efficiently. We introduce a convenient method for generating either parts or the complete atlas for the manifold based on special "cable events". Generating parts of the configuration space on-the-fly enables improvements over the state of the art: (a) we avoid discretizing the manifold, obtaining competitive time and space complexity for our planner, (b) we are able to exploit topological structure to represent robot-cable configurations concisely, (c) we generalize the representation in order to examine cable-to-cable contacts, which have been widely ignored in the literature until now. 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.

1 Introduction

The classic problem of motion planning can be defined as moving an object in a space while avoiding obstacles [2]. Many practical scenarios require more complicated variations of the problem. Consider a robot which uses a cable as a source of power and/or communication. Robots which perform high power

*This paper is an extended version of [1].

†R. H. Teshnizi and D. A. Shell are with the Distributed AI and Robotics Lab, Dept. of Computer Science and Engineering , Texas A&M University, College Station, TX 77843, USA. Emails: {r.teshnizi, dshell}@cse.tamu.edu



(a) Length Constraint (b) Topological Constraint

Figure 1: The limits caused by fixing the robot to a fixed point.

tasks (*e.g.,* street cleaning machines) need to use a cable as a source of power as their required power cannot be provided by batteries. Also, in several cases a cable is necessary for a robot to be able to transfer data (*e.g.,* underground and underwater robots). Two important constraints are imposed on the motion of the robot because of the cable: the cable length which limits the radius of the movement (Fig. 1a), and the topological constraints imposed by the cable and obstacles in the environment (Fig. 1b).

Recent research has focused on different aspects and variations of this problem [3–6]. The present work examines the structure of the configuration space induced by a robot tethered by a taut (possibly retracting) cable and proposes to represent the manifold of configurations by a special atlas [7]. 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 straight forward using this representation.

The method we propose creates a set of locally continuous charts based on the set of *cable events* that we will define. Initially, we examine events that occur when a cable touches or wraps around an obstacle, and show that they are intimately connected with a subgraph of the visibility graph of the environment. We will show that for a tethered robot this subgraph is in fact a tree and thus the topological structure of the atlas representing the configuration space forms a tree structure. This tree 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—generalizes to this case despite the fact that the involved topology becomes rather more complex: events no longer occur at a countable set of locations, the configuration space's structure is no longer separated easily. Nevertheless, because only parts

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

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 Related Work and Contributions

Much prior work on motion planning for tethered robots has the common underlying idea of creating a graph approximation of the configuration space and using an efficient method to search through the space of possible paths to choose the optimal one among those in the homotopy class.

The work of Hert and Lumelsky [8] first considered the problem of ordering the motions of multiple tethered robots by sorting a graph representing their paths. While influential, the paper never addressed finding those orderings. The work of Grigoriev and Slissenko [9] and Narayanan, Vernaza, Likhachev, and LaValle [10] addressed the problem by representing paths in a given homotopy class by defining an alphabet used to describe the event of crossing a ray. Determining whether two paths are in the same homotopy class then requires comparison of their related strings. The latter paper also addresses homologically equivalent paths. That research group also explored the topic of winding constraints for motion planning [5]. A radically different approach is taken by Bhattaharya, Kumar, and Likhachev [3], who provide an alternative way of defining homotopy classes of paths in 2D based on the *Cauchy Integral Theorem*.

Igarashi and Stilman [4] designed an algorithm for creating a graph of the configuration space manifolds based on cable length constraint. Their idea of representing the configuration space by multiple overlapping manifolds influenced the present work. Recent work has also examined problems beyond path planning in tethered robots [6].

Based on the body of prior work, we have identified several distinguishing contributions of the current paper. Firstly, the approach we propose avoids discretizing the configuration space, instead using a subset of the visibility graph to induce a natural cell decomposition of the space. Secondly, it connects 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. Thirdly, 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 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. And finally, the visibility graph has been previously used in motion planning for non-tethered robots [11, 12], but our construction of an atlas of charts leads to new insight with regards to the cable's dependency on the environment topology and its connection with

the visibility graph.

3 The Preliminaries

This section provides the fundamental definitions used throughout the paper. 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.

Figs. 2a–2c 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 [13]. Let $a \twoheadrightarrow b$ denote the line segment connecting point a to b. In moving from $p_s \twoheadrightarrow p_d$, an obstacle may affect the robots motion directly (Fig. 2b) or indirectly via a cable-obstacle contact which will bend the cable. Fig. 2c. In the latter case, the radius of movement of the robot after that bend is affected.

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. 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. 3). 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.

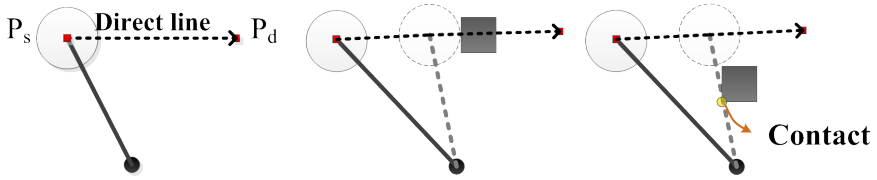
Definition 1. *In the context of the cabled robot, there are two kinds of events:*

1. *Cable to obstacle contact which we call wrapping event*
2. *Cable to cable contact which we call cable crossing event*

In the following subsections, we are not concerned about cable-cable interactions. We return to cable to cable interactions in Section 5 where the method is extended.

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



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

Figure 2: Different scenarios of a tethered robot with respect to the presence of obstacles.

graph [14]. 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} [9].

Since planning the motion for each cell of a decomposed configuration space is straight forward, 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. *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.*

As shown graphically in Fig. 3, we identify a cell uniquely by the following fields:

- *Base Point*: as discussed in Section 3.1, each bending 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. 4 illustrates a 3D model of how the visibility cells are connected together.

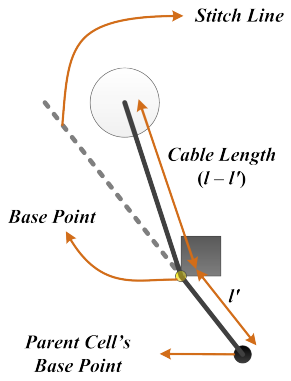


Figure 3: The defining elements of a visibility cell.

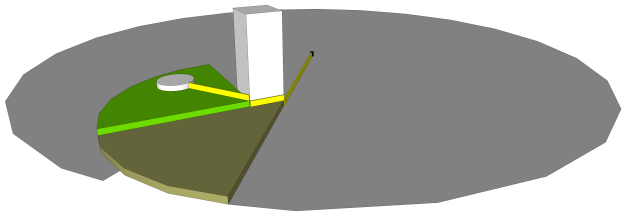


Figure 4: A 3D model of the visibility cells and the way they are connected together. Each cell is in a different color. The robot is white and the cable is colored yellow.

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 (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 with it characterizing its topological structure, which we call Maximum Ray Length Visibility Atlas Graph (MRLVAG) denoted by G_{A_l} (see Fig. 5).*

It is important to notice that the number of child charts is always less than or equal to the actual visibility graph's vertices. In order to have a detailed procedure for creating a MRLVA we encourage the reader to see the appendix, where a detailed algorithm is provided, and which constitutes a formal constructive definition of MRLVAs and MRLVAGs.

Lemma 1. *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*

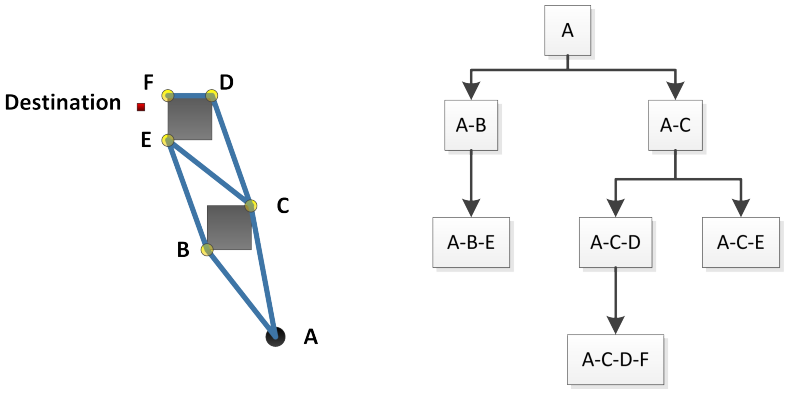


Figure 5: An example of a MRLVAG. The environment producing this MRLVAG is given on the left.

end points. If one of the end points meets an obstacle it is locally supporting to P_{obs} *at the contact point [13].*

Theorem 1. *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 the 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. □

4 The Basic Planning Algorithm

The previous section is a foundation leading to a basic planning algorithm for the robot with cable wrapping events. 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 relies on the direct path $p_s \twoheadrightarrow p_d$ as it is always the locally shortest path [9]. 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, then there is no path to p_d and 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. The next step is to ensure that no shorter path from the parent chart to the destination point exists, otherwise the final path will go through the parent chart instead.

Algorithm 1: The shortest path greedy algorithm

FindPath$(p_s, p_d, path)$
mark this chart as visited
if p_d is directly reachable **then**
  if $p_s \twoheadrightarrow p_d$ will not cause an event **then**
    return $path.Push(p_d)$
  else
    m_c = Create a chart for the upcoming event
    Set p_m as the point where $p_s \twoheadrightarrow p_d$ crosses the stitch line of the new chart
    $path.Push(p_m)$
    $minPath = m_c.FindPath(p_m, p_d, path)$
  end if
else if p_d might fall into one of this chart's children **then**
  Create child charts for each of the visible vertices of the obstacles if the chart is needed
  for all c $\in$ children **do**
    $temp = path.Push(c.Base)$
    $temp = c.FindPath(c.Base, p_d, temp)$
    if *temp* is the shortest path to this point **then**
      store it as $minPath$
    end if
  end for
end if
if $Parent \neq \varnothing$ and is not visited yet **then**
  $temp = Parent.FindPath(Base, p_d, path)$
  if *temp* is the shortest path to this point **then**
    store it as $minPath$
  end if
else
  $minPath = \varnothing$
end if
return $minPath$

5 Handling Cable-Cable Interaction

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

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 [4] wherein the authors deliberately plan movements to avoid motions that will cause cable-to-cable interactions. The main reason that available methods widely chose to ignore 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. 6).

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. 7).

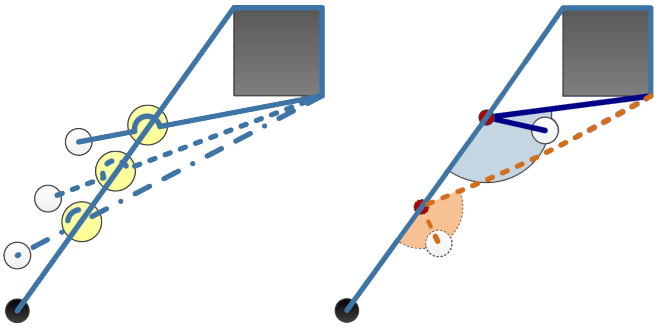
5.2 Violation of the tree structure of MRLVAG

For these cable interactions, two more events are added to the event set. 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 7 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.



(a) Uncountable contact points.

(b) Changing movement radius. Two different contact points and their accessible areas are shown.

Figure 6: The complexities of the cable-to-cable contacts

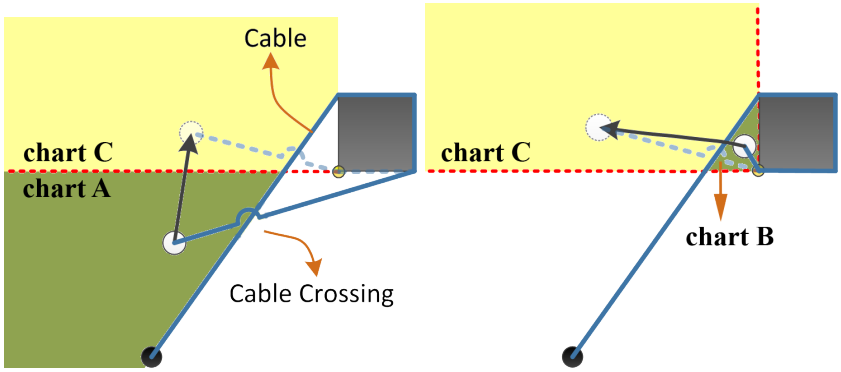
Reexamining Fig. 7, we see that if the goal is reaching chart C, then Fig. 7b is a shorter distance than Fig. 7a. In fact, if the movement to cross the cable is arbitrarily small, then a robot going from chart B to chart C moves the same distance as if cable crossing were ignored entirely.

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. 8). Thus, the algorithm is extended to swap events that are out of order. In Fig. 7a, 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.

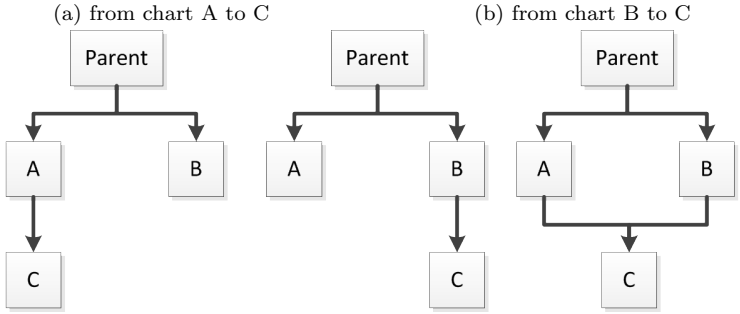
5.4 Sequential Cable Events

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. 9). 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



(a) from chart A to C

(b) from chart B to C



(c) From left to right: the tree made by movement in Fig. 7a, in Fig. 7b, and the inherent loop.

Figure 7: An example of an MRLVAG growing toward a destination.

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 online.

6 Experimental Results

To demonstrate the method and evaluate its performance, we developed a simulation environment and implemented the algorithm in C#. Fig. 10 is a screenshot of the simulation environment depicting an operation space with polygonal obstacles. The results we present next are from experiments using the two test environments Figs. 11a and 11b.

Our aim is to understand the relationship between the proposed approach and existing methods. Although picking appropriate criteria for comparison is obviously important, fundamental differences between our method and existing work make this challenging. In particular, running time is problematic because other state-of-the-art methods employ a discretization of the configura-

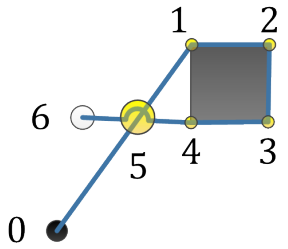


Figure 8: The best ordering of events for reaching from the base point to the destination point. In other words, for having the shortest path, the points with smaller number should be reached earlier.

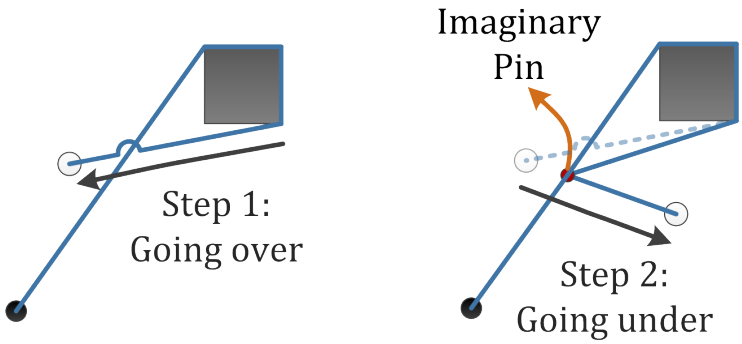


Figure 9: An example of sequential cable crossing events.

tion 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.

	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 1: Results of the Experiments

6.1 On-line generation versus off-line generation

In order to demonstrate the savings enabled by on-line generation of the configuration space, we have calculated the total volume of the configuration space. We then compared it to the summed volume of the robot's current visibility cell

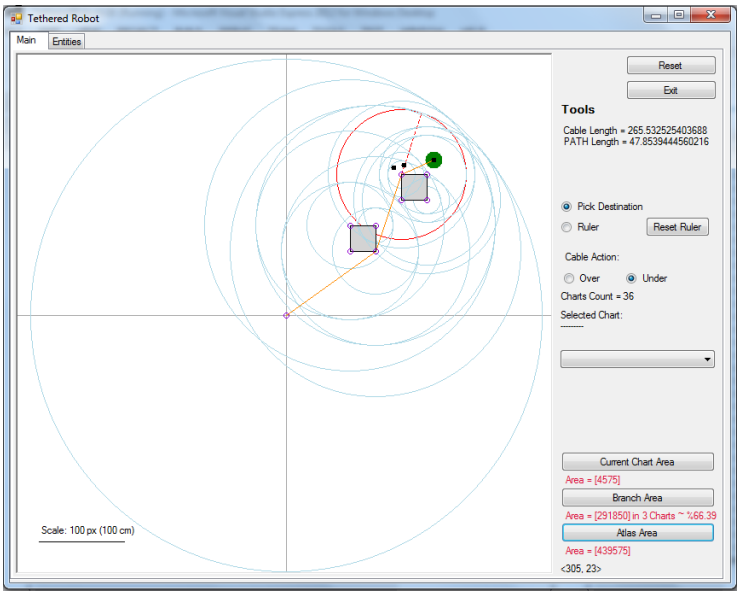


Figure 10: Screen-shot of the simulation environment.

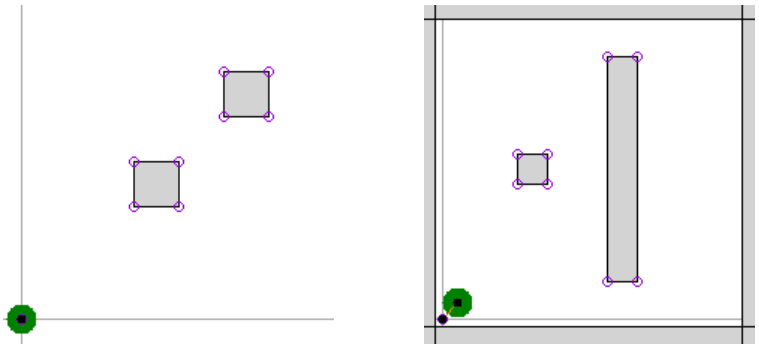
and its parents up to the root since these cells suffice to specify the robot and cable's configuration.

The results in Table 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 report values for the cheapest and most costly to represent, respectively. 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.

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 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.

In such cases, the number of charts used is likely to be a more important factor than area. We computed the number of nodes in the MRALVAG 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;



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

Figure 11: The two test environments used for presenting the results. Gray is obstacle and green is robot.

consequently these numbers are reported too. Results for the two test scenarios, shown in the Table 1, illustrate that the savings are substantial in both simple and complex manifolds.

6.3 Cable Induced Manifold Structure

The presented data also lead to additional observations. Different environments (*e.g.*, with differing numbers and complexity of obstacles) result in 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 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.

7 Conclusion

This paper approaches 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.

First, we formalize the concept of cable events: actions that would add a new boundary to the robot's movement. These events are then used to break the configuration space into visibility cells in which the robot could move without causing additional events. The concept of cable events is quite general and is applicable to many problems. As shown in Section 5, we are able to define new types of event to handle the cable-cable interactions. Other contexts may be represented with further event types — for example, a mountain climbing robot may affix its cable to an anchor point: an action that can be regarded as a specialized cable event. Many other examples exist but are outside the scope of the current paper.

Next, we define an Maximum Ray Length Visibility Atlas (MRLVA) of the configuration space along with a graph to record MRLVA's topological structure (MRLVAG). In path planning, we used the MRLVA as the search space. The definition of cable events in this case relates the MRLVAG to the visibility graph of the environment. As data structures these allow parts of configuration space to be generated so that a computation on the space (*e.g.*, a search) may exploit local structure in order to avoid construction and representation of unnecessary parts of the configuration space.

In this paper assumptions about the physical properties of the cable (tautness and friction) allow for well-defined cable-cable interactions. Our immediate future work is to generalize this concept so that it can be applied to frictionless modeling of a cable as well. 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. We are also interested in incorporating the MRLVA while connecting multiple robots together with a cable.

[Maximum Ray Length Visibility Atlas Generation Algorithm]

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 Defi-

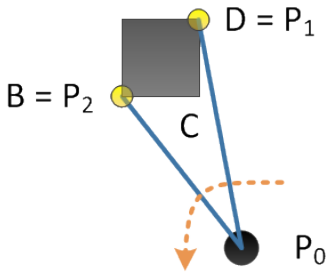


Figure 12: B, C and D are visible vertices from P_0. Only B and D are enqueued in the algorithm.

nition 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 $\varphi_2(x, y) = \varphi_1(x, y) = (x, y)$ 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 [14]. However, not all of the apexes are enqueued because the line segment will not bend around them (see Fig. 12). As a result the number of child charts created is never more than the number of vertices in visibility graph.

References

[1] R. H. Teshnizi and D. A. Shell, "Computing cell-based decompositions dynamically for planning motions of tethered robots," in *accepted for presentation at Proc. of IEEE Int. Conf. on Robotics and Automation*, 2014.

[2] S. Lavalle, *Planning algorithms*. Cambridge University Press, 2006.

[3] S. Bhattacharya, V. Kumar, and M. Likhachev, "Search-based path planning with homotopy class constraints," in *Annual Symposium on Combinatorial Search*, 2010.

[4] T. Igarashi and M. Stilman, "Homotopic path planning on manifolds for cabled mobile robots," in *Algorithmic Foundations of Robotics IX*. Springer, 2011, pp. 1–18.

[5] D. S. Yershov, P. Vernaza, and S. M. LaValle, "Continuous planning with winding constraints using optimal heuristic-driven front propagation," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 5551–5556.

[6] I. Shnaps and E. Rimon, "Online coverage by a tethered autonomous mobile robot in planar unknown environments," in *Proceedings of Robotics: Science and Systems*, Berlin, Germany, June 2013.

[7] J. M. Lee, *Introduction to smooth manifolds*. Springer, 2012, vol. 218.

[8] S. Hert and V. Lumelsky, "The ties that bind: Motion planning for multiple tethered robots," *Robotics and autonomous systems*, vol. 17, no. 3, pp. 187–215, 1996.

[9] D. Grigoriev and A. 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*. ACM, 1998, pp. 17–24.

[10] V. Narayanan, P. Vernaza, M. Likhachev, and S. M. LaValle, "Planning under topological constraints using beam-graphs," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 431–437.

[11] T. Lozano-Pérez and M. A. Wesley, "An algorithm for planning collision-free paths among polyhedral obstacles," *Communications of the ACM*, vol. 22, no. 10, pp. 560–570, 1979.

[12] T. Lozano-Pérez, "Spatial planning: A configuration space approach," *Computers, IEEE Transactions on*, vol. 100, no. 2, pp. 108–120, 1983.

[13] T. Krick, A. Slisenko, P. Solernó, and J. Heintz, "Search for shortest path around semialgebraic obstacles in the plane," *Journal of Mathematical Sciences*, vol. 70, no. 4, pp. 1944–1949, 1994.

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