Homotopic Path Planning on Manifolds for Cabled Mobile Robots

Takeo Igarashi and Mike Stilman

Abstract We present two path planning algorithms for mobile robots that are connected by cable to a fixed base. Our algorithms efficiently compute the shortest path and control strategy that lead the robot to the target location considering cable length and obstacle interactions. First, we focus on cable-obstacle collisions. We introduce and formally analyze algorithms that build and search an overlapped configuration space manifold. Next, we present an extension that considers cable-robot collisions. All algorithms are experimentally validated using a real robot.

1 Introduction

Mobile robots are typically untethered. This is not always desirable in household and high-power robotics. Wireless communication can be unreliable and batteries need to be charged regularly. These challenges can be solved by using cables for communication and power. Currently, cables are a viable option for robots that work in fixed environments such as homes and offices. The challenge addressed in this paper is that cables impose additional constraints on robot motion. First, robots cannot go further than the cable length. Second, they are blocked by the cable itself when the robots are not capable of crossing it. We present two practical planning algorithms that handle these constraints and validate them on a real robot system.

The first challenge is that a cabled robot's movement is constrained by the length of the cable. If there is no obstacle, the robot's motion is limited to stay within a circle around the fixed end-point of the cable. If there is an obstacle in the environment, the robot's movement is further constrained by the interaction between the cable and the obstacle as shown in Fig. 1(a) and (b). The locations of the robots are the same, but the shortest paths to the goal are different because the cable configuration in Fig. 1(b) cannot stretch to the goal. The second problem is that the robot's movement may be blocked by the cable when the robot is not capable of crossing it as shown in Fig. 1(c). The robot must make an auxiliary motion to move the blocking cable out of the way (Fig. 1(d)). This is difficult because the robot cannot directly control the cable. It must indirectly control it by pulling.

Takeo Igarashi
Department of Computer Science, The University of Tokyo
JST ERATO Igarashi Design Interface Project, e-mail: takeo@acm.org

Mike Stilman
Center for Robotics and Intelligent Machines, Georgia Institute of Technology
e-mail: mstilman@cc.gatech.edu

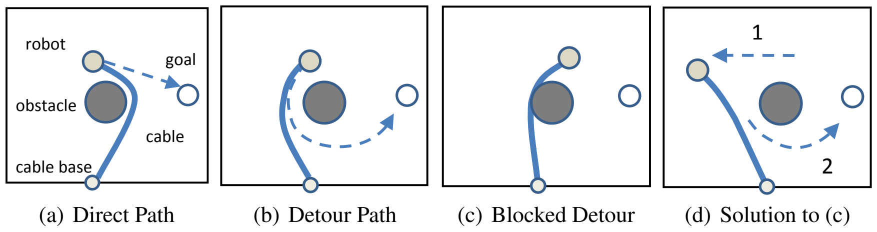


Fig. 1 The problem domain and a challenging example where the robot must untangle its cable.

In order to address these challenges, Section 3 introduces the *overlapped manifold* representation for the configuration space of cabled robots. We develop an efficient, resolution complete and optimal algorithm that constructs the manifold and solves practical planning problems. To handle collisions between the cable and the robot, Section 4 presents a second search method that applies physics-based simulation combined with heuristics to choose intermediate subgoals that maximize robot mobility. Section 5 experimentally demonstrates that both algorithms generate appropriate paths for a real robot that reaches targets in the presence of a cable.

2 Related Work

The topic presented in this paper is far more complex than general path planning [1, 2]. While the robot itself only operates in two dimensions, the cable is also part of the complete system or plant. By including the cable, the challenge is lifted to planning for an infinite-dimensional underactuated system. Previous work on tethered robots [3] treated the problem as multi-robot scheduling. Our approach focuses on a single agent and handles environment geometry.

Considerable research on high degree-of-freedom (DOF) robot systems such as [4, 5] has direct applications to domains with dozens of DOF and non-holonomic constraints. Our problem, however, requires handling even higher DOF and underactuation. Hence, the challenge is also distinct from deformable motion planning as presented in [6, 7]. Likewise, cable-routing [8] assumes that shape of the cable is directly controllable. However, a cabled robot cannot control all of its degrees of freedom and must rely on predictions of cable motion due to stretch.

Existing planning methods for underactuated deformable objects typically focus on local deformations [9, 10]. Studies on deformable needle steering also consider the path to a robot configuration [11, 12] with a focus on local environment deformation and curvature constraints. Our work complements these studies since our task is to determine globally optimal robot paths. Global constraints are imposed by the cable length, wrapping around obstacles and potentially colliding with the robot.

Typically, globally constrained underactuated planning and control is restricted to four DOF systems as shown in [13, 14, 15]. To handle the global problem complexity *our domain requires a different approach based on topological path homotopy.* Existing work in knot-tying [16] plans with distinct topological states. However it explicitly encodes and plans rope overlaps. Other planners that distinguish homotopic paths, [17, 18, 19], typically operate in a standard high-DOF configuration space. Instead, we build a configuration space manifold that implicitly encodes the homotopy of cable configuration and then search for shortest paths on the manifold.

In direct homotopic planning, [20] studies shortest paths but restricts the domain to a boundary-triangulated space. [21] requires semi-algebraic models of obstacles. [22] gives a configuration space representation that closely related to our work. Their complex-plane mapping of paths may increase the efficiency of our methods for single-query search. In contrast to our proposed manifold, existing techniques do not address global cable-length constraints or cable-robot interactions.

Existing methods for manifold construction tend to focus on relationships in recorded data [23, 24]. We present a novel, simple algorithm for global path planning with distance constraints on paths. The algorithm not only generates paths, but a complete vector field [25, 26] for robot motion on a manifold of homotopic paths. Our extensions to this algorithm also consider cable dynamics [27, 28] and evaluate strategies for robot motion when the cable itself is an obstacle in the space [29].

3 Distance Manifolds: Cable-Obstacle Interaction

We present a path planner for cabled mobile robots. The initial configuration, q_i, includes initial cable displacement. The goal is any configuration q_g that places the robot at p_g in a 2D environment. The robot is connected to a fixed base location, p_0, by a cable resting on the floor. The cable is a flexible, passive entity whose shape is determined solely by the previous motions of the robot. The environment contains fixed obstacles that restrict both cable and robot motion. For simplicity, we assume a disk-shaped robot with a given diameter and a cable attached by a freely rotating joint. Furthermore, we represent space by a grid where configuration space obstacles must occupy at least one grid vertex. This section introduces an algorithm that handles the constraint given by cable length.

First, we build the configuration space that represents the structure of the problem and then compute a vector field that guides the robot in the configuration space. Given a static environment the configuration space is generated off-line, reducing the cost of online planning. Sections 3.1,2 describe the space, its graph representation and formalize the problem statement. Sections 3.3,4 introduce the algorithms for graph construction and planning. Section 3.5 proves algorithm correctness.

3.1 Configuration Space

In order to build a complete planner for tethered robots, we consider the configuration space. Notice that the space must distinguish distinct homotopic paths. Some configurations that have identical robot locations have different cable configurations. If we ignore collisions between the cable and obstacles, the configuration space is a 2D circular region defined by the 2D environment (Fig. 2a). However, this representation cannot distinguish configurations with different cable positions (Fig. 3 A,E). The cable location determines the region of space that is immediately reachable by the robot. In order to differentiate between A and E, we define configuration space by an *overlapped manifold*. The manifold is planar, but it can be visualized with stitched or overlapped free space components (Fig. 2b,c).

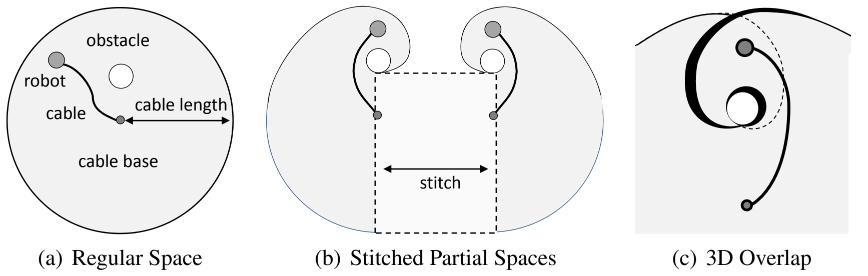


Fig. 2 Simple and overlapped manifold configuration space for tethered robots.

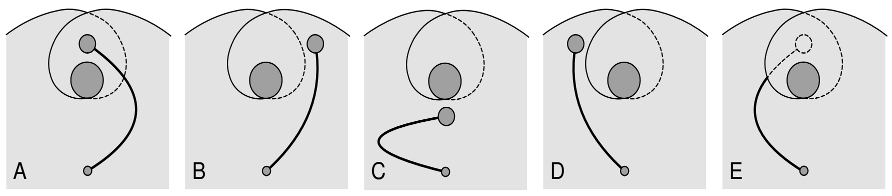


Fig. 3 Example traversal of configuration space.

Distinct configurations on the manifold with the same locations represent distinct cable configurations. A continuous region in the manifold corresponds to a set of configurations that can be reached by continuous robot motion. In Fig. 3, straight trajectories change the configuration from A to E via B, C, D. However, there is no straight path that can displace the robot from A to D or B to E.

Notice that the number of overlaps in the manifold increases exponentially with each additional obstacle. The robot has two options for circumnavigating each obstacle. It can go around to the left or to the right. Hence, for n reachable obstacles, there exist at least 2^n paths or cable routes that reach the same goal. This corresponds to at least 2^n possible overlaps in the configuration space manifold. We say "at least" because winding the cable around an obstacle also doubles the overlaps.

3.2 Graph Representation of Configuration Space

This section gives a formal representation of the problem domain and the problem statement. Our algorithm constructs a graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ that represents the configuration space manifold (Section 3.3) and then searches the graph (Section 3.4) for vector fields or paths. The key challenge is to construct a graph that completely encodes the configuration space. This section defines the graph properties that must be assured in graph construction. Section 3.5 verifies these properties.

First, consider the domain definitions: Manifold *vertices* are located at physical grid nodes. v_0 is the vertex that represents the cable base. D_{ij} is the manhattan distance on the grid between v_i and v_j. When $D_{ij} = 1$, the two vertices are referred to as *neighbors*. Notice that neighboring vertices are not necessarily connected by an edge since they may be on distinct overlapping folds of the manifold. P_{ij} represents paths between vertices on the manifold and $|P_{ij}|$ is path length.

The problem is to build a graph **G** *such that any shortest path from* v_i *to* v_g *corresponds to a shortest path from* q_i *to any* q_g *where* p_g *is the target and the cable does not cross an obstacle.*

Definition 1. Two paths from v_0 to any point are path *homotopic* if and only if there exists no obstacle in the area enclosed by the two paths. Otherwise they are not homotopic or *ahomotopic*.

While grid nodes are simply positions, p_i, manifold vertices, v_i are defined by the set of homotopic paths from v_0 to p_i. Each vertex is associated with a path of adjacent vertices of length less than D_{max}. In every set of homotopic paths, there exist minimal paths. Let us call them *m-paths* ($mP(v_0, v_i)$ or mP_i).

Definition 2. v_2 is an *m-child* of v_1 ($v_2 \succ v_1$) and v_1 is an *m-parent* of v_2 ($v_1 \prec v_2$) if and only if there exists a minimal path, mP_2, where v_1 is the last vertex before v_2.

Definition 3. v_1, v_2 are *m-adjacent* ($v_1 \sim v_2$) if and only if v_1 is an m-parent of v_2 or v_2 is an m-parent of v_1.

Definition 4. Collocated vertices v_1 and v_2 are manifold-equivalent, *m-equivalent* ($v_1 \equiv v_2$) if and only if every path to v_1 is homotopic to every path to v_2.

All m-equivalent vertices are collocated, $v_1 \equiv v_2 \Rightarrow v_1 \simeq v2$. However, not all collocated vertices are m-equivalent. This occurs when the paths to v_1 go around some obstacle while those to v_2 do not. In this case, the distance between vertices on the manifold can be greater than physical distance between their positions.

Definition 5. The *m-distance* between v_1 and v_2, mD_{12} is the length of any shortest path between v_1 and v_2 such that all consecutive nodes on the path are m-adjacent.

Lemma 1. *Let* v_1 *and* v_2 *be neighboring vertices such that* $D_{12} = 1$. *For any vertex,* v_3, *if* $D_{13} \geq D_{23}$ *then strictly* $D_{13} > D_{23}$. *Likewise,* $mD_{13} \geq mD_{23} \Rightarrow mD_{13} > mD_{23}$.

Proof. Any paths P_{13} and P_{23} must have distinct parity since they are separated by one edge [30]. One path has even length while the other is odd. Hence $|P_{13}| \neq |P_{23}|$.□

Proposition 1 (Adjacency). *For neighboring vertices:* $v_1 \sim v_2$ *(a) if and (b) only if there is no obstacle in the area enclosed by any* mP_1 *and* mP_2.

Proof. Since v_1 and v_2 are neighbors, $mD_{01} \neq mD_{02}$ by Lemma 1. Without loss of generality, consider $mD_{01} < mD_{02}$. (a) For any path mP_1, construct path $P_2 = \{mP_1, v_2\}$. This is a minimal path to v_2 with v_1 as the last vertex since $mD_{01} < mD_{02}$ and $|P_2| = |mP_1| + 1$. Hence, $v_1 \prec v_2$ and $v_1 \sim v_2$. (b) By contradiction: assume there exists an obstacle enclosed by some mP_1, mP_2. By the premise, $v_1 \sim v_2$ and therefore $v_1 \prec v_2$. Hence by Def. 2, there exists mP_2' with v_1 as the last vertex and mP_1 as a sub-path. mP_2' and mP_2 enclose an obstacle so they are not homotopic. Thus $v_2 \not\equiv v_2$. Contradiction. Likewise if $mD_{01} > mD_{02}$. □.

Consider again the problem statement: *Build a graph* **G** *such that any shortest path from* v_i *to* v_g *corresponds to a shortest path from* q_i *to any* q_g *s.t. the cable does not cross an obstacle.* Following Proposition 1, this graph must have the following property: *two neighboring vertices are connected by an edge if and only if they are m-adjacent.* Section 3.3 introduces our algorithm for constructing **G**.

```
BUILDMANIFOLD($v_0$, V, E)
 1  Q ← ENQUEUE($v_0$)
 2  while Q $\neq \emptyset$
 3  do $v_a$ ← DEQUEUE(Q)
 4      S ← SUCCESSORS($v_a$, V, E)
 5      for all $v_t \in$ S
 6      do V ← INSERT($v_t$)
 7          E ← INSERT($v_t, v_a$)
 8          if DIST[$v_t$] < DISTMAX
 9             then Q ← ENQUEUE($v_t$)
10          B ← ADJACENT(ADJACENT($v_a$))
11          for all $v_b \in$ B
12          do if POS[$v_b$] $\in$ NEIGHBORS($v_t$)
13                then E ← INSERT($v_t, v_b$)

SUCCESSORS($v_a$, V, E)
 1  N ← NEIGHBORS($v_a$)
 2  S ← $\emptyset$
 3  for all $p_i \in$ N
 4  do if COLLISIONFREE($p_i$) and
 5         $p_i \notin$ POS[ADJACENT($v_a$)]
 6        then $v_i$ ← NEWVERTEX
 7               POS[$v_i$] ← $p_i$
 8               DIST[$v_i$] ← DIST[$v_a$] + 1
 9               S ← INSERT($v_i$)
10   return S
```

Fig. 4 Manifold Construction Pseudo-code.

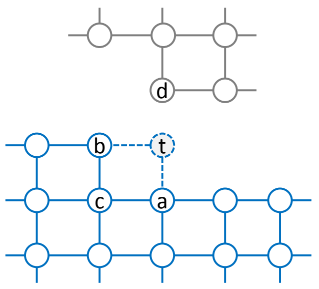


Fig. 5 BUILDMANIFOLD Line 11

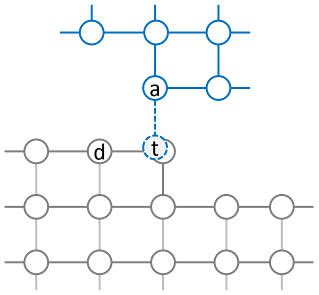


Fig. 6 Illustration of overlap

3.3 *Manifold Construction: Forward Search*

Our algorithm in Fig. 4 incrementally adds vertices and builds graph edges by connecting adjacent vertices to the north, south, east and west of each vertex. SUCCESSORS(v_a, **V**, **E**) returns the set of neighboring, collision-free vertices that are not yet in the graph. Since we assume that obstacles occupy at least one grid node, COLLISIONFREE(p) returns *true* when a node does not intersect an obstacle.

Multiple manifold vertices can share a single grid position as in Fig. 3 (A,E). Our algorithm, distinguishes grid positions, p_i, from manifold vertices, v_i. The position of vertex v_i is obtained by POS[v_i]. The function NEIGHBORS(v_i) returns the set of four neighboring positions of the vertex. The function ADJACENT(v_i) returns the set of all vertices in **V** that are adjacent/connected to v_i in **G** as follows: $\{v_j | \exists e(v_i, v_j) \in \mathbf{E}\}$. There are at most four such vertices. Likewise, ADJACENT(ADJACENT(v_i)) returns at most eight vertices $\neq v_i$ that are adjacent to the first four.

BUILDMANIFOLD is a variant of breadth-first search or wavefront expansion. Standard expansion adds all edges to existing neighbors when adding a new vertex. In contrast, we add an edge to a neighbor only when there is a common vertex that has edges to both the neighbor and the parent of the new vertex (Lines 10-13). This is illustrated by Fig. 5. An edge is added between v_t and v_b since both v_a and v_b have edges to a common vertex v_c. However, an edge is not added between v_t and v_d. Likewise, in Fig. 6, no edge is added between v_t and v_d, generating the manifold with overlaps as presented in Section 3.1.

3.4 Plan Generation: Backward Search

Given the graph representing the configuration space, we construct a vector field to guide the robot from any given starting location to any desired goal. This is required since the manifold is a roadmap that is created for all possible start and goal states. Basic dynamic programming or wavefront expansion is used compute a distance field over the configuration space starting from the target location. The gradient of the distance field is used to control the robot. Note that the target location can be associated with multiple vertices in the graph. Starting from all these vertices, we assign minimal distance values to the remaining vertices by breadth-first traversal. Consequently, the robot always follows the minimal path on the manifold.

3.5 Algorithm Analysis

This section analyzes the complexity, optimality and correctness of our algorithms. First of all, the computational complexity of manifold construction is $O(n)$ where n is the number of vertices in the configuration space. Likewise, the computational complexity of search is $O(n)$. Hence the entire algorithm is executed in $O(n)$. Furthermore, *manifold generation must only be computed once for static environments*, regardless of start state and goal. This yields efficient multi-query planning.

Given that $\mathbf{G}$ is correctly constructed and completely represents the manifold that the robot can traverse then dynamic programming is a complete and optimal method for finding a solution. Therefore plan generation is complete and optimal. The remaining task is to prove the correctness and completeness of manifold construction. We will use Proposition 2 in the validation of BUILDMANIFOLD in Proposition 3.

Proposition 2 (Equivalence). *If $v_1 \simeq v_2$ are both m-adjacent to v_a then $v_1 \equiv v_2$.*

Proof. Let P_1 and P_2 be any paths to v_1 and v_2. There exist shortest paths mP_1, mP_2 homotopic to P_1, P_2 respectively. Let mP_a be a shortest path to v_a. Since $v_1 \sim v_a$ and $v_2 \sim v_a$, by Prop. 1, there is no obstacle enclosed by $\{mP_1, mP_a\}$ and $\{mP_2, mP_a\}$. Hence, there is no obstacle enclosed by mP_1 and mP_2, so they are homotopic. Since $v_1 \simeq v_2$ and all paths to v_1 are homotopic to all paths to v_2 we have $v_1 \equiv v_2$. □

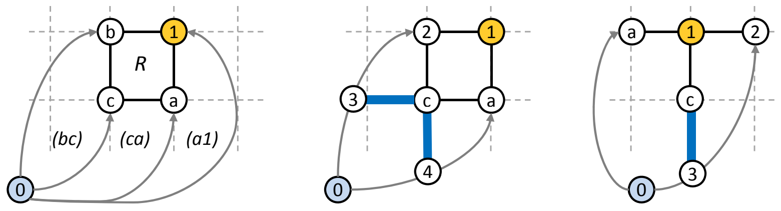
Proposition 3. *Prior to adding v_3 with $mD_{03} \geq D < D_{max}$,* BUILDMANIFOLD *maintains the following invariant. Let vertices v_1 and v_2 have $mD_{01} < D$, $mD_{02} < D$.*
(a) $v_1 \in \mathbf{V}$ if and only if v_1 is not m-equivalent to any other vertex, $v_2 \in \mathbf{V}$.
(b) $e(v_1, v_2) \in \mathbf{E}$ if and only if v_1 is m-adjacent to v_2 ($v_1 \sim v_2$).

Proof. We proceed by induction. Base case, $D = 1$, there are no edges and the only vertex is v_0, added in Line 1. The inductive step is split into the following Lemmas.

For Lemmas 2-5 assume Prop. 3. *Prior to adding any v_3 such that $mD_{03} \geq D+1$:* (BMY and SY refer to Line Y in BUILDMANIFOLD and SUCCESSORS respectively)

Lemma 2. *If $v_1 \in \mathbf{V}$ then v_1 is not m-equivalent to any other vertex, $v_2 \in \mathbf{V}$.*

Proof. BUILDMANIFOLD adds v_1 to $\mathbf{V}$ by expanding v_a only if v_a has no edge to any vertex at its position, p_1 (S5). By assumption, v_a is not m-adjacent to any vertex at p_1. Hence, v_1 is the only vertex at p_1 such that $v_a \prec v_1$. Therefore it is the only vertex with a minimal path mP_1 such that v_a is the last vertex. □



(a) (b) (c)

Fig. 7 Illustrations for the proof of Lemmas 4,5. Straight lines indicate precise grid displacements. Curves represent paths that preserve relative position but not necessarily distance.

Lemma 3. *If* $\exists v_1$ $(mD_{01} \le D)$ *not m-equivalent to any other vertex in* $\mathbf{V}$ *then* $v_1 \in \mathbf{V}$.

Proof. By contradiction: Suppose $v_1 \notin \mathbf{V}$. v_1 is an m-child of some v_a where $mD_{0a} = mD_{01} - 1$. By the assumption, $mD_{0a} < D$ so $v_a \in V$ and by BM9, $v_a \in Q$. Since there are finite vertices with $mD_{0a} < D$, v_a is dequeued and expanded in BM4. Since v_1 is a neighbor of v_a one of the following must hold: (1) By S4, v_1's position is not collision free. Contradiction. (2) By S5 and the inductive assumption, v_a has an m-adjacent vertex, $v_2 \simeq v_1$. By Prop. 2, $v_2 \equiv v_1$. Contradiction. □

Lemma 4. *If* $\mathbf{E}$ *contains edge* (v_1, v_2) *then* v_1, v_2 *are m-adjacent.*

Proof. Without loss of generality, suppose v_1 is added after v_2. An edge is added at (a)BM7 or (b)BM13. (a) $v_2 = v_a$ and v_1 is newly defined and implicitly associated with minimal paths homotopic to $mP_1 = \{mP_a, v_1\}$. Since v_2 is the last vertex on mP_1, $v_2 \prec v_1$. (b) $v_2 = v_b$. By BM7 there exists $v_a \prec v_1$. By BM10 and the inductive assumption there exists v_c such that $v_a \sim v_c \sim v_b$. Given that $v_b \sim v_c$ and $v_c \sim v_a$ and $v_a \sim v_1$, Prop. 1 states that there is no obstacle between any mP_b and mP_1 as shown by regions $(bc), (ca), (a1)$ and R in Fig. 7(a). Hence, by Prop. 1 $v_2 = v_b \sim v_1$. □

Lemma 5. *If* $v_1, v_2 (mD_{01}, mD_{02} \le D)$ *are m-adjacent,* $\mathbf{E}$ *contains edge* (v_1, v_2).

Proof. By the inductive assumption: $v_1, v_2 \in \mathbf{V}$. Without loss of generality, $v_2 \prec v_1$.

(a) $mD_{01}, mD_{02} < D$ then $(v_1, v_2) \in \mathbf{E}$ by the inductive assumption.

For the remaining cases $mD_{01} = D$ and $mD_{02} = D - 1$ by Lemma 1.

(b) $v_2 = v_a$ is the first m-parent of v_1 added to $\mathbf{V}$. Then $(v_1, v_2) \in \mathbf{E}$ by BM7.

For the remaining cases there exists $v_a \prec v_1$ $(v_a \ne v_2)$ that was added prior to v_2. Since $v_1 \equiv v_1$, there are two minimal paths $mP_1(a) = \{mP_a, v_1\}$ and $mP_1(2) = \{mP_2, v_1\}$ that enclose a region $\mathbf{R}$ with no obstacles. By Prop. 2 there are three relative positions for $v_2 \not\equiv v_a$. Due to symmetry of Fig. 7(b), that yields two cases.

(c) In the case of Fig. 7(b) there exists v_c, neighbor of v_a and v_2 contained in $\mathbf{R}$. Extend two straight paths P_{3c} and P_{4c} opposite v_a and v_2 respectively. Since $\mathbf{R}$ is closed, these paths must intersect mP_2 and mP_a at some vertices v_3, v_4

respectively. Since P_{3c} is straight, $|P_{3c}| < |P_{32}|$. Hence, the path $S_2 = \{P_{03}, v_c, v_2\}$ has length $|S_2| \leq |mP_2|$. Therefore $v_c \prec v_2$. Likewise, since P_{4c} is straight, $|P_{4c}| < |P_{4a}|$. Hence, $S_a = \{P_{04}, v_c, v_a\}$ has length $|S_a| \leq |mP_a|$. Therefore $v_c \prec v_2$. Thus there exists v_c such that $v_2 \sim v_c \sim v_a$. This satisfies BM10-13, thus $(v_1, v_2) \in \mathbf{E}$.

(d) In the case of Fig. 7(c) there exists v_c neighbor of v_1 that is contained in $\mathbf{R}$. Extend a straight path P_{3c} from v_3 opposite v_1. Since $\mathbf{R}$ is closed, P_{3c} must intersect either mP_2 or mP_a at some vertex v_3 respectively. Without loss of generality, assume it intersects mP_2. Since P_{3c} is straight, $|P_{3c}| < |P_{32}|$. Hence, the path $S = \{P_{03}, v_c, v_1\}$ has length $|S_2| \leq |mP_2|$. Since S is a path from v_0 to v_1 homotopic to $mP_1(2)$, we have $mD_1 < |mP_2| + 1$. Thus $mP_1(2)$ is not a minimal path. Contradiction. Likewise, if P_{3c} intersects mP_a, we find $mP_1(a)$ is not a minimal path. Contradiction.

In all valid cases where $v_1 \sim v_2$, $\mathbf{E}$ contains the edge (v_1, v_2). □

Lemma 6. *The algorithm terminates when all $v_i : mD_i < D_{max}$ are added to* $\mathbf{Q}$.

Proof. Every new vertex, v_i increments DIST$[v_i]$ by 1 from its parent (S8). For any D there are a finite number of vertices that are not m-equivalent with $mD_i < D$. Since no vertices are added to $\mathbf{Q}$ with DIST$[v_i] \geq$ DISTMAX (BM7) and each step dequeues, BUILDMANIFOLD terminates. □

By Lemmas 2-6, BUILDMANIFOLD is proven to add all the vertices on the manifold to $\mathbf{V}$ and all the edges between m-adjacent vertices to $\mathbf{E}$ prior to guaranteed termination. Therefore, the manifold generation algorithm is correct and complete.

3.6 Implementation Details

The presented algorithm computes Manhattan distance between the base and each vertex in the graph. This is a low-order approximation of physical distance. We therefore also allow diagonal moves when computing the distance value, creating an 8-connected lattice and obtaining a better approximation. The experiments in Section 5 demonstrate that it performs well in robot experiments.

In order to include diagonal moves, Line 7 of SUCCESSORS uses d instead of 1, where $d = \sqrt{2}$ for diagonally connected vertices. Furthermore, $\mathbf{Q}$ in BUILDMANIFOLD is a priority queue rather than a FIFO in order to always select vertices with the minimal distance from v_0. This approach increases computation time to $O(n \log n)$ due to priority queue operations.

4 Cable-Robot Interaction

Section 3 introduced a novel formulation of the configuration space for tethered robots and presented a complete solution to path planning for robots that are restricted by cable length. The proposed configuration space allows us to go further and consider additional constraints on robot motion. In this section, we examine *the case where the robot cannot cross the cable*. Cable-robot collisions present further algorithmic challenges that are not solved by existing methods. We evaluate two solutions and propose a novel algorithm in Section 4.3.

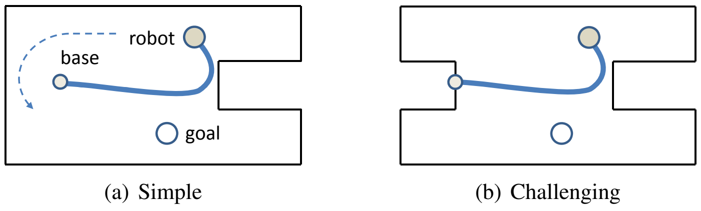


Fig. 8 Illustration of a cable blocking the path to the target.

4.1 Preliminary Algorithm

Simple domains such as Fig. 8(a) can be solved by adding the current cable shape as an obstacle to future motion[29]. We implemented an algorithm that incrementally removed vertices from the graph that were within the robot radius of the cable through wavefront expansion. The online controller continuously updated the vector field as the cable shape changed during motion.

In our experiments, the initial path typically remained valid during robot motion because the deformation of the cable occurred behind the robot. When the plan became inaccessible, the system replanned the path. This approach required continuous tracking of cable shape. Since this is difficult in practical environments we used physical simulation to predict the current shape of the cable based on the motion history of the robot. Section 5 shows that this approach works well in practice.

Notice, however, that *this simple method is not sufficient* when the cable completely blocks a path to the target as shown Fig. 8(b). Removing cable vertices from the configuration space blocks all path to the goal. We present two approaches that handle such cases. First, we consider a hardware solution in which the system retracts the cable. Second, we introduce a novel algorithm for *feasible* path planning that clear blocks through auxiliary robot motion.

4.2 Hardware Solution: Cable Retraction

First of all, the problem in Fig. 8(b) can be solved by continuously retracting the cable to make the cable as short as possible while allowing free robot motion. This approach requires additional hardware, but simplifies planning. Given cable retraction, the robot would simply need to follow the cable to the cable base until the path to the target is cleared. This can be accomplished by searching for a shortest path on the manifold to the base and directing the robot to move along that path.

The hardware implementation is not trivial because one must develop a special device than retracts the cable with appropriate force for the particular robot and cable type. The force must be simultaneously strong enough to pull a long cable and sufficiently weak to allow robot motion. Furthermore, it may be necessary to constantly adjust the force depending on the robot and cable status. We have not yet implemented this solution, however it remains an exciting topic for our future work.

4.3 Algorithmic Solution: Untangling

Given that the robot cannot retract the cable mechanically, it must perform auxiliary motions to clear the path blocked by the cable. We refer to this procedure as *untan-*

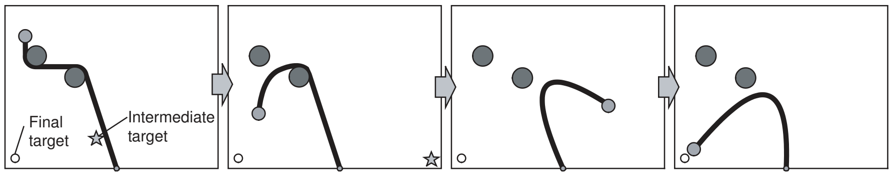


Fig. 9 Auxiliary motions open the path to the target. Each step is computed by our algorithm.

gling. Consider Fig. 1(c). The robot must first move to the left to clear the path to the target on the right. More complex domains, such as Fig. 9, have goals that are blocked by the cable multiple times along a single path. The algorithm is required to find a sequence of untangling motions. In contrast to Section 3, evaluating all possible motions was computationally infeasible. Instead, we developed a heuristic method that efficiently computes untangling motions and performs well in practice.

When our system identifies that there is no open path to the target from the current robot location, it selects an intermediate target and moves the robot towards it. Motion to the intermediate target is chosen to displace the blocking cable from the path of the robot to the goal (Fig. 9a). If the goal becomes accessible during travel to the target, the online algorithm discards the intermediate target and moves directly to the goal. If the intermediate target becomes inaccessible or if the robot reaches the target, the system computes the next target. This process repeats until the goal becomes accessible. Fig. 9 shows a complete untangling procedure.

For each step in the untangling process, we choose the intermediate target from several candidates. The most promising one is selected by internal physics-based simulation as in Section 4.1. First, we identify the region in the configuration space accessible from the current robot position (Fig. 10). We then relate each vertex in the configuration space graph to the minimum of the distance from the vertex to the region's graph center and that to the current robot position. Local maxima of the computed distances are chosen as candidates (Fig. 10 left). For each candidate, we compute a simulated robot motion where the robot moves towards the candidate pulling the cable, and test whether or not the motion clears the path. We use a simple spring-mass model to simulate the behavior of a cable. If a candidate clears the path in simulation, we select the candidate as the intermediate target. If no candidate clears the path to the goal, we choose the candidate that is expected to maximize the accessible region after the robot arrives at the target (Fig. 10 right).

The proposed algorithm is heuristic and is not guaranteed to find a solution if there is a solution. An alternative, systematic approach is to construct a search tree by recursively sampling candidates for the intermediate targets and search for a successful sequence of intermediate targets as in many path planning algorithms [1]. We did not implement such systematic approaches because our simple heuristic successfully found a path to the target via multiple intermediate targets in our experiments (Section 5) when there was a solution. When there is no solution, neither our heuristic method nor systematic approach can find one.

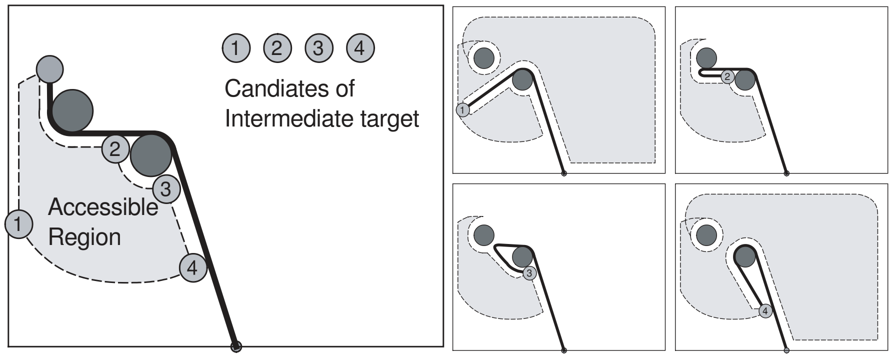


Fig. 10 A search for the most promising candidate. Candidates (left) and their accessible regions (right). Candidate 4 is selected in this case.

4.4 Deadlock Prevention

The algorithm described in the previous subsection cannot find a path when the robot is already trapped in a deadlock configuration as shown in Fig. 11 (left). The robot is trapped in the closed region and none of the auxiliary motions described above are able to open the way to the goal. To prevent this problem from occurring, we augmented the algorithm with a preprocessing step that removes configurations that can cause deadlocks from **G**. We then use the previously described runtime algorithms to find a deadlock free path to the target. This algorithm preforms well (Section 5) but does not guarantee deadlock avoidance. It is our future work to develop a complete run-time algorithm for deadlock prevention.

Starting from each graph node, the algorithm follows the path to the cable base by picking each adjacent vertex with minimum distance to the base. It identifies graph vertices that are in contact with an obstacle, yet their parents are not adjacent to an obstacle. These contact vertices are potential locations where a deadlock can occur (stars in Fig. 11 right).

Having identified contact vertices, the system examines whether or not the contacts are resolvable as follows. First, we compute a region in the configuration space separated by the path to the cable base and accessible from the contact vertex (gray area in Fig. 11). We then compare the maximum distance to any vertex in the region from the contact vertex and the *remaining cable length*. This is computed by subtracting the distance from the cable base to the contact node from the overall cable length. If the maximum distance is longer than the remaining cable length, then the contact is resolvable by moving the robot to the most distant position. Otherwise, the contact causes a deadlock. In this case, our system prevents the robot from causing the deadlock by removing all the configuration space vertices in the accessible area for which the distance longer than the one to the contact points (shaded in Fig. 11).

5 Experiments

In order to validate the practical effectiveness of the proposed algorithms, we conducted a series of experiments on a physical robot. We examined the basic case

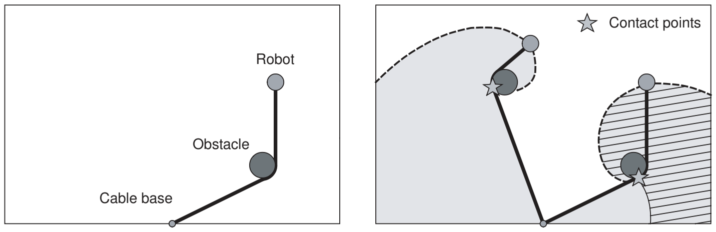


Fig. 11 Deadlock configuration (left) and the detection of potential deadlocks (right). Left contact (star) is resolvable, but the right contact (star) is not resolvable. When a potential deadlock is detected, we remove the affected area from the configuration space (shaded area).

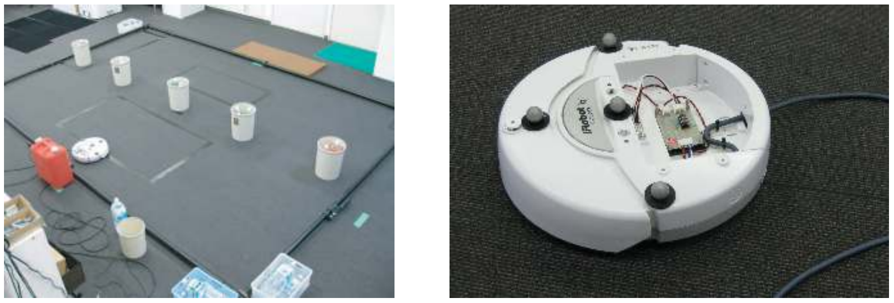


Fig. 12 The physical environment and the cabled robot used in our experiments.

involving overhead tracking for robot position and a robot with no hardware for cable retraction. The cable configuration was not tracked but predicted by means of internal physics-based simulations for the algorithms in Section 4. Hence, our robot was not guaranteed to avoid cable-robot collision 100%. However, the experiments demonstrate that our algorithm significantly reduced the occurrence of collisions.

We evaluated the proposed algorithm using a cabled robot on a flat floor. An iRobot CREATE robot was connected to a cable that provided power and control signal for a total of 5 bundled wires. The location of the robot was tracked by a vision-based motion capture system (Motion Analysis). The system consisted of 8 infra-read high speed cameras that observe the motion of retro-reflective markers attached to the robot. The control PC (Dell Latitude) received the robot location from motion capture and sent control commands to the robot via the cable.

Fig. 12 gives an overview of the physical environment. It is a standard office floor covered by carpet. The layout mimics an open office or home environment with obstacles such as columns and furniture. The size is $5m \times 3m$. Our algorithm represented this space with 50×30 grid. Fig. 13 shows the layouts used in the experiment. In each trial, the robot was placed near the cable base with a compactly assembled cable. It visited six given targets in a given order. The system judged that a target visit was complete when the distance between the robot center and the target center was less than the robot diameter. We ran 10 trials for every combination of a given algorithm and layout. We prepared a set of 10 random permutations of 6 targets and used the same set for all combinations.

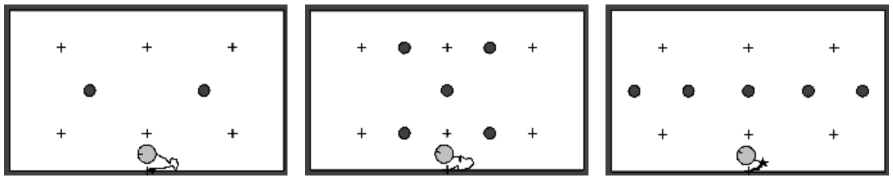


Fig. 13 Experimental layouts: dark gray circles are obstacles and plus marks are targets.

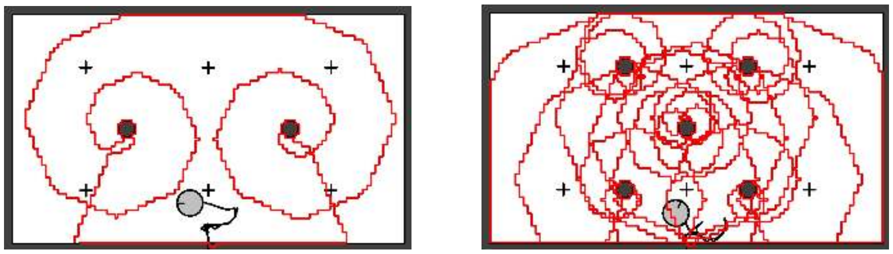


Fig. 14 Configuration space boundaries for two experimental layouts.

5.1 Experimental Results

Table 1 shows the statistics of our results. The basic algorithm completed the tasks with 100% success. The extended algorithm without deadlock prevention failed in some cases ($50-80\%$ success). However, adding deadlock prevention achieved 100% success. Collisions between the robot and the cable did occur even when we used the extended algorithm. However the number of collisions was significantly reduced compared with the basic one. Fig. 14 shows the configuration space for the first two layouts. Fig. 15 shows an example of untangling observed during the experiments. It demonstrates that our algorithm successfully identified an appropriate sequence of intermediate targets. Video of the experiments and demonstration software are available at: http://www.designinterface.jp/en/projects/cable.

	Basic Algorithm			**E**xtended Algorithm			**D**eadlock Prevention		
	Task1	Task2	Task3	Task1	Task2	Task3	Task1	Task2	Task3
Success Ratio	100%	100%	100%	80%	50%	80%	100%	100%	100%
Average Time (s)	83.9	100.4	88.5	100.9	103.2	93.0	89.4	110.4	92.5
Avg. Cable-Robot Collisions	1.8	1.7	2	0.25	0.4	0.125	0.4	0.3	0.2

Table 1 Results from the experiments. We ran 10 trials for each combination of algorithm $\times$ task.

6 Conclusion

Our work shows that path planning for cabled robots yields significant insight into homotopic path planning. We developed a configuration space formulation that distinguishes between robot positions with distinct cable configurations. We proposed complete algorithms that compute the configuration space manifold and plan optimal paths given cable length constraints. Furthermore, we studied a practical extension of our algorithm given that the robot is not permitted to cross its cable. These algorithms were validated on a real robot platform in a series of experiments.

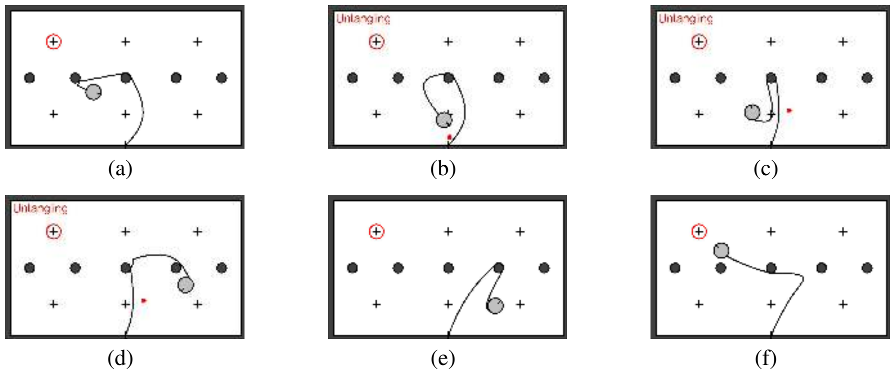


Fig. 15 Sample robot experiment with untangling: a) Initial configuration and goal (red circle) b) Approaching the first intermediate target (red dot) c) Approaching the second intermediate target. d) Arriving at the second intermediate target. e-f) Approaching and arriving at the goal.

This paper opens the door to numerous variations of planning homotopic paths and cabled robotics. Immediate future work is the development of runtime lookahead detection of deadlocks. An interesting variant is path planning for robots that grasp or push the cable [31]. Another interesting problem is optimal placement of the cable base for a given environment to minimize deadlocks. Similar analysis would identify problematic obstacles that can cause deadlocks and warn the user.

7 Acknowledgements

We thank Daisuke Sakamoto, Takanori Yoshikawa, and Yoichi Kamiyama for their help in the experiments. We are grateful to Prof John F. Hughes for valuable discussions and to the anonymous reviewers for their insightful feedback.

References

[1] J.C. Latombe. *Robot motion planning*. Springer Verlag, 1990.
[2] S.M. LaValle. *Planning algorithms*. Cambridge Univ Pr, 2006.
[3] S. Hert and V. Lumelsky. The ties that bind: Motion planning for multiple tethered robots. *Robotics and Autonomous Systems*, 17(3):187–215, 1996.
[4] LE Kavraki, P. Svestka, J.C. Latombe, and MH Overmars. Probabilistic roadmaps for path planning in high-dimensionalconfiguration spaces. *IEEE Transactions on Robotics and Automation*, 12(4):566–580, 1996.
[5] S. LaValle and J. Kuffner. Rapidly-exploring random trees: Progress and prospects. In *Algorithmic and computational robotics: new directions: Workshop on the Algorithmic Foundations of Robotics*, page 293. AK Peters, Ltd., 2001.
[6] L. Kavraki, F. Lamiraux, and C. Holleman. Towards planning for elastic objects. In *1998 Workshop on the Algorithmic Foundations of Robotics*, pages 313–325.
[7] O.B. Bayazit, J.M. Lien, and N.M. Amato. Probabilistic roadmap motion planning for deformable objects. In *IEEE Int. Conf. on Robotics and Automation*, volume 2, pages 2126–2133, 2002.
[8] I. Kabul, R. Gayle, and M.C. Lin. Cable route planning in complex environments using constrained sampling. In *Proceedings of the 2007 ACM symposium on Solid and physical modeling*, page 402. ACM, 2007.
[9] E. Anshelevich, S. Owens, F. Lamiraux, and L. Kavraki. Deformable volumes in path planning applications. In *Proc. IEEE Int. Conf. Robot. Autom.(ICRA)*, pages 2290–2295, 2000.

[10] F. Lamiraux and L.E. Kavraki. Planning paths for elastic objects under manipulation constraints. *The International Journal of Robotics Research*, 20(3):188, 2001.

[11] R. Alterovitz, J. Pouliot, R. Taschereau, I.C. Hsu, and K. Goldberg. Sensorless planning for medical needle insertion procedures. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, volume 4, pages 3337–3343. Citeseer, 2003.

[12] V. Duindam, R. Alterovitz, S. Sastry, and K. Goldberg. Screw-based motion planning for bevel-tip flexible needles in 3d environments with obstacles. In *IEEE Intl. Conf. on Robot. and Autom*, pages 2483–2488. Citeseer, 2008.

[13] HJ Sira-Ramirez. On the control of the underactuated ship: A trajectory planning approach. In *IEEE Conference on Decision and Control*, volume 3, pages 2192–2197, 1999.

[14] A. De Luca and G. Oriolo. Motion planning and trajectory control of an underactuated three-link robot via dynamic feedback linearization. In *IEEE Int. Conf. on Robotics and Automation*, volume 3, pages 2789–2795, 2000.

[15] M. Bergerman and Y. Xu. Planning collision-free motions for underactuated manipulators in constrained configuration space. In *IEEE Int. Conf. on Robotics and Automation*, pages 549–555, 1997.

[16] M. Saha, P. Isto, and J.C. Latombe. Motion planning for robotic manipulation of deformable linear objects. In *Experimental Robotics*, pages 23–32. Springer, 2008.

[17] O. Brock and O. Khatib. Real-time re-planning in high-dimensional configuration spacesusing sets of homotopic paths. In *IEEE Int. Conf. on Robotics and Automation, 2000. Proceedings. ICRA'00*, volume 1, 2000.

[18] E. Schmitzberger, JL Bouchet, M. Dufaut, D. Wolf, and R. Husson. Capture of homotopy classes with probabilistic road map. In *IEEE/RSJ Int. Conf. on Intelligent Robots and System, 2002*, volume 3, 2002.

[19] L. Jaillet and T. Siméon. Path deformation roadmaps. *Algorithmic Foundation of Robotics VII*, pages 19–34.

[20] J. Hershberger and J. Snoeyink. Computing minimum length paths of a given homotopy class. *Computational geometry*, 4(2):63–97, 1994.

[21] D. Grigoriev and A. Slissenko. Polytime algorithm for the shortest path in a homotopy class amidst semi-algebraic obstacles in the plane. In *ISSAC '98: Proceedings of the 1998 Int. Symp. on Symbolic and algebraic computation*, pages 17–24, 1998.

[22] S. Bhattacharya, V. Kumar, and M. Likhachev. Search-based Path Planning with Homotopy Class Constraints. In *Proceedings of the Conference on Artificial Intelligence (AAAI)*, 2010.

[23] J.B. Tenenbaum, V. Silva, and J.C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319, 2000.

[24] O.C. Jenkins and M.J. Matarić. A spatio-temporal extension to isomap nonlinear dimension reduction. In *Int. Conf. on Machine learning*, page 56. ACM, 2004.

[25] J. Barraquand. Automatic motion planning for complex articulated bodies. *Paris Research Laboratory, Tech. Rep*, 1991.

[26] C.I. Connolly and R.A. Grupen. Harmonic control. In *Proceedings of the IEEE International Symposium on Intelligent Control*, pages 503–506.

[27] J. Brown, J.C. Latombe, and K. Montgomery. Real-time knot-tying simulation. *The Visual Computer*, 20(2):165–179, 2004.

[28] M. Moll and LE Kavraki. Path planning for deformable linear objects. *IEEE Transactions on Robotics*, 22(4):625–636, 2006.

[29] T. Yoshioka, K. Goto, T. Sato, H. Oki, and H. Tsukune. A Path-Planning for Mobile Robot Leading Cable Around. *Nippon Kikai Gakkai Robotikusu, Mekatoronikusu Koenkai Koen Ronbunshu*, 2002(Pt 1):1A1, 2002.

[30] R.E. Korf and M. Reid. Complexity analysis admissible heuristic search. In *Proceedings of the fifteenth national/tenth conference on Artificial intelligence/Innovative applications of artificial intelligence*, page 310. American Association for Artificial Intelligence, 1998.

[31] M. Stilman and J. Kuffner. Navigation among movable obstacles: Real-time reasoning in complex environments. In *IEEE/RAS Int. Conf. on Humanoid Robots*, pages 322–341, 2004.