Manipulation Planning for Deformable Linear Objects

Mitul Saha and Pekka Isto, *Member, IEEE*

***Abstract*—Research on robotic manipulation has mainly focused on manipulating rigid objects so far. However, many important application domains require manipulating deformable objects, especially deformable linear objects (DLOs), such as ropes, cables, and sutures. Such objects are far more challenging to handle, as they can exhibit a much greater diversity of behaviors, and their manipulation almost inevitably requires two robotic arms, or more, performing well-coordinated motions. This paper describes a new motion planner for manipulating DLOs and tying knots (both self-knots and knots around simple static objects) using two cooperating robotic arms. This planner blends new ideas with preexisting concepts and techniques from knot theory, robot motion planning, and computational modeling. Unlike in traditional motion planning problems, the goal to be achieved by the planner is a topological state of the world, rather than a geometric one. To search for a manipulation path, the planner constructs a topologically biased probabilistic roadmap in the configuration space of the DLO. During roadmap construction, it uses inverse kinematics to determine the successive robot configurations implied by the DLO configurations and tests their feasibility. Also, inspired by the real life, the planner uses static "needles" (by analogy to the needles used in knitting) for maintaining the stability of the DLO during manipulation and to make the resulting manipulation plan robust to imperfections in the physical model of the DLO. The implemented planner has been tested both in graphic simulation and on a dual-PUMA-560 hardware platform to achieve various knots, like bowline, neck-tie, bow (shoe-lace), and stun-sail.**

***Index Terms*—Deformable objects, knot tying, manipulation planning, probabilistic roadmaps.**

I. Introduction

ROBOTIC manipulation of rigid objects is a rather well-studied problem. Here, we focus on manipulating deformable linear objects (DLOs), such as ropes, cables, and sutures. Progress in robotic manipulation of DLOs can benefit several application domains, like manufacturing (e.g., loading cable harnesses) and medical surgery (especially suturing). It can also help humanoid robots perform various household services. However, DLOs add a number of difficulties to the manipulation task. They exhibit a much greater diversity of behaviors than rigid objects by continuously changing shape when submitted to external forces. As a result, both self-collisions and collisions with other objects are possible. Furthermore, the manipulation of DLOs almost inevitably requires two robotic arms performing well-coordinated motions and re-grasp operations, as well as additional static tools to maintain the integrity of critical portions of the DLO (e.g., loops). Finally, the topology of the goal state of a DLO is usually far more important than its exact shape.

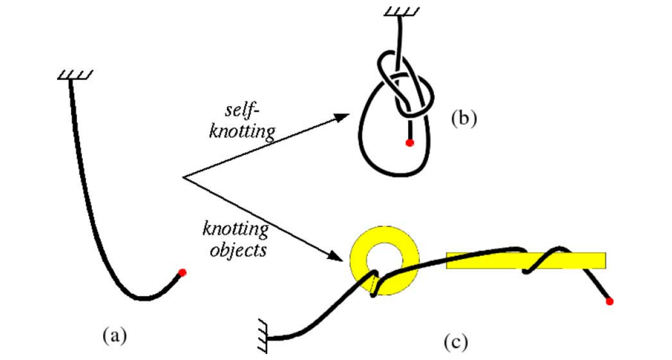


Fig. 1. (a) Initial state of a DLO manipulation problem. (b) and (c) Two types of goal states.

In this paper, we describe a new motion planner for manipulating DLOs with two cooperating robotic arms. Fig. 1 illustrates two typical problems (without the arms). Fig. 1(a) shows a segment of rope in its initial unwound state. Fig. 1(b) and (c) depicts two types of goal states, where the rope forms a self-knot (bowline) and winds around some static rigid objects, respectively. In either case, the topological description of the goal consists of a set of "crossings" of the rope with itself and with the rigid object (Section III-C). Our planner does not rely on any specific physical model of the DLO. Instead, it takes a computational model of the DLO as input, in the form of a state-transition function (Section III-B). Using this function and the kinematic model of the robot arms, the planner constructs a probabilistic roadmap [4], [17] in the configuration space of the DLO (Section V). The sampling of the roadmap nodes is biased toward achieving the topology of the goal state of the DLO. During roadmap construction, the planner tests that the grasp points on the rope are accessible by the arms without collision. The planner assumes that simple static sliding supports (independent of the robot arms), which we call needles (by analogy to the needles used in knitting), are available, and can be used when needed, to maintain the integrity of certain portions of the DLO during manipulation. A novel method is proposed (Section VI) to account for the interaction of the DLO with simple static

Manuscript received August 2, 2006; revised January 13, 2007. This paper was recommended for publication by Associate Editor S. Hirai and Editor K. Lynch upon evaluation of the reviewers' comments. This work was supported in part by the National Science Foundation (NSF) under Grant ACI-02-5671 and in part by the Academy of Finland.

M. Saha is with the NIH Center for Biomedical Computation, Stanford University, Stanford, CA 94305 USA (e-mail:mitul@ai.stanford.edu).

P. Isto is with the Research Institute for Technology, University of Vaasa, FI-65101 Vaasa, Finland (e-mail: isto@uwasa.fi).

This paper has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the authors. This includes five simulation videos corresponding to Fig. 15(a)–(e), which are, respectively, 17.1, 15.7, 20.7, 30.7, and 7MB in size. The sixth real-life video corresponds to Fig. 16 and is 24.6 MB in size.

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

Digital Object Identifier 10.1109/TRO.2007.907486

objects. Curve representations of these objects, obtained from their skeletonization, are "chained" with the DLO to produce a semi-deformable linear object (sDLO), in which some pieces are rigid and others are deformable. Thereafter, the problem reduces to that of tying self-knots with the composite sDLO. We implemented the planner and tested it both in simulation and on a two-arm hardware platform (Section VII). We demonstrated its effectiveness by tying commonly used knots like bowline, neck-tie, bow (shoe-lace), and stun-sail.

II. Relation to Previous Work

A. Manipulation Planning

Early works on robot manipulation planning with rigid objects include [10] and [28]. The randomized potential field method proposed later in [18] is closer to our research. It generates motion paths for cooperating arms manipulating a movable rigid object among static obstacles. Like ours, it computes a path of the movable object and uses inverse kinematics to get the motion of the arms. The method assumes that a discrete set of grasps on the movable object is given as input. The algorithm proposed in [26] works with continuous sets of grasps of rigid objects, but plans the motion of a single manipulator, with the regrasps being done by releasing the object on a horizontal surface. In general, releasing a DLO on a surface would cause the DLO to conform to this surface under gravitational forces, and therefore, to drastically change its shape. An algorithm for planning paths for elastic objects, especially for flexible plates and cables, is presented in [12]. It attempts to minimize an energy function based on the object elasticity. This algorithm plans only the motion of the deformable object, not that of the arms. The problem of path planning for DLOs in the presence of obstacles is addressed as a variational problem in [29], but the proposed formulation does not consider the manipulator either. All these works focus on achieving a single given geometric pose of an object. We, instead, focus on achieving an input topological state of a DLO. This state implicitly defines a continuous set of acceptable geometric shapes of the DLO.

B. Application of Knot Theory in Robotics

Knot theory provides tools to represent and analyze the topological states of a DLO [2], [15]. Its applications in robotics include the work presented in [20], where the topological state of a DLO is represented by a sequence of signed crossings. State transitions are caused by the so-called Reidemeister moves and a crossing operation that moves the end of the DLO over another part of the DLO to make a new crossing. Similar ideas are being used in [30] to plan knotting and unknotting of DLOs, and further developed for tightening of the knots. The planner uses an efficient procedure to generate qualitative plans. However, collision constraints and the physical behavior of the DLO are not considered during the implemented planning phase. Nevertheless, the plans are robust enough for vision-guided execution on a robot system for one-arm manipulation of a DLO with the aid of the floor. Motion planning techniques from robotics are used in [11] to untangle mathematical knots, but without taking the robotic manipulation system into account.

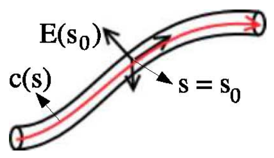


Fig. 2. Representation of a DLO.

C. Sensing-Based DLO Manipulation

The difficulty of accurately modeling deformable objects has motivated sensing-based (mostly visual) approaches to DLO manipulation. In an early work [9], a robot system, guided by stereo vision, is being developed that is capable of inserting a rope into a ring and tying a rope. The authors of this work were severely constrained by the limited computational resources available at their time. One of our manipulation problems is reminiscent of theirs. Methods are presented in [19] for DLO modeling, recognition, and parameter identification. These methods have been embedded in a two-arm robotic system capable of tying a rope around a cylinder. In [24], a sensing-based method is proposed for picking up hanging DLOs. Building on the approach of enumerating contact states between rigid objects, a contact state model is proposed in [1] for describing contacts between polyhedral objects and a DLO. A discrete contact state model abstracts away instable and semi-stable contacts as they are difficult to maintain in manipulation. State transitions are grouped into classes that can be detected with one sensing algorithm. The model can be used to support automatic sensor-based robot program generation from demonstration in virtual reality [16]. In an earlier work [8], demonstration was performed with a dedicated data acquisition hardware. The collected data were used to develop a controller for sensor-guided hose insertion task. In our research, we do not rely on the availability of any sensing system to guide the manipulation in real time. Instead, we focus on precomputing manipulation plans that are robust to uncertainties in the physical model of the DLO.

III. Modeling a Deformable Linear Object

A. Geometric Model

We describe the geometry of a DLO by a curved cylinder of curvilinear length L and constant circular cross section of non-zero radius (Fig. 2). The *axis* of this cylinder is a smooth curve c parameterized by its curvilinear length s, i.e.,

$$c : s \in [0, L] \rightarrow c(s) \in \mathbf{R}^3$$

where $c(0)$ is referred to as the *tail* of the DLO and $c(L)$ as its *head*. Whenever the physical model of the DLO includes torsion or twist, a Cartesian coordinate frame $E(s)$ is associated with each point $c(s)$. We call $q = (c, E)$ a *configuration* of the DLO.

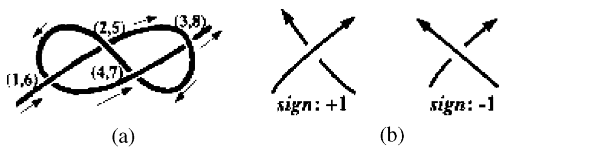


Fig. 3. (a) Crossings in the figure-8 knot (the arrows indicate the orientation of c from its tail to its head). (b) Sign convention for the crossings [in both cases, the section of c pointing diagonally from left to right contains $c(s_1)$ and the section pointing from right to left contains $c(s_2)$].

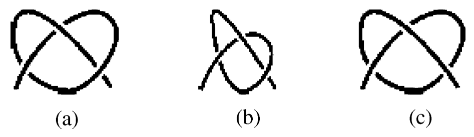


Fig. 4. (a) and (b) Two configurations of a DLO in $C(\{(1,4,+1),(2,5,+1),(3,6,+1)\})$. (a) [or (b)] and (c) Two geometrically similar configurations of a DLO that achieve distict topological states.

B. Physical Model

Designated points located at $s_1, \ldots, s_k$ on the DLO are designated as *grasp* points. They are the only points whose positions $c(s_i)$ and orientations $E(s_i), i = 1, \ldots, k$, can be directly controlled by the robotic manipulation system.

The physical model of a DLO is given in the form of a *state transition* function f that maps a DLO configuration q_{cur} and a control input defined by a $3k$-vector u of small displacements of the grasp points to a new DLO configuration q_{new}. We assume that both q_{cur} and q_{new} are statically stable configurations, but the DLO may exhibit dynamic behavior (e.g., snapping) during the transition from q_{cur} to q_{new}. We do not consider the amount of time spent by the transition between q_{cur} and q_{new}, although it could be easily added to the model.

We assume that the model embedded in f handles collisions with obstacles, as well as self-collisions, so that the DLO does not penetrate itself or obstacles when it moves between q_{cur} and q_{new}. In addition, if u causes violations of physical constraints associated with the DLO, e.g., over-stretching, then the transition function f returns that the execution of u is impossible, so that the DLO configuration is not modified. As we do not allow the robot arms to touch the DLO at points other than the grasp points, f also reports failure if it detects a collision between the DLO and an arm.

Several physical models of DLOs proposed in the literature, e.g., [3], [6], [14], [21], [22] [27] [31], are relevant to the construction of f. In our system, we use the model presented in [27].

C. Topological Model

We describe the topology of the DLO at a configuration $q = (c, E)$ by a set of typed crossings on a projective plane P defined by a unit normal vector $\vec{v}$ [2]. P is said to be in general orientation if the following are satisfied:

1) No more than two points in c project to the same point in P.
2) For any two points $c(s_1)$ and $c(s_2)$ in c that project to the same point in P, the tangents to c at these two points project to two distinct vectors in P.

A *crossing* in P is any point X that is the projection of two distinct points $c(s_1)$ and $c(s_2)$ [see Fig. 3(a)]. Let $s_1 < s_2$. The crossing is said to be *over* if the vector $\vec{v}$ points from $c(s_2)$ to $c(s_1)$; otherwise, it is said to be *under*. Moreover, X is assigned the sign $+1$ iff:

1) either X is over and the counterclockwise angle between the projected tangents is less than π;
2) or X is under and the clockwise angle between the projected tangents is less than π.

Otherwise the sign of X is -1. In both the conditions for the positive sign, the angle is measured from the projected tangent at s_1 to that at s_2. This sign convention is depicted in Fig. 3(b).

Suppose that c has n crossings $X_1, \ldots, X_n$ in P. Let us "walk" along c from $s = 0$ to $s = L$, and assign the integral labels $1, \ldots, 2n$ in increasing order to the crossings as they are encountered. Since each crossing X_j is encountered exactly twice, it receives two distinct labels, which we denote by X_j^1 and X_j^2. Fig. 3(a) illustrates the labeling of crossings in the figure-8 knot. We define the *topological state* of c with respect to P, in general orientation, by the set of triplets

$$\{(X_j^1, X_j^2, \epsilon_j)\}_{j=1,\ldots,n}$$

where ϵ_j is the sign of the crossing X_j.

Note that the same topological state x of c can be achieved by various configurations q of the DLO. We let $C(x)$ denote the set of all these configurations. To illustrate, Fig. 4(a) and (b) show two configurations of a DLO in $C(\{(1,4,+1),(2,5,+1),(3,6,+1)\})$. In contrast, Fig. 4(a) [or (b)] and (c) shows two configurations that achieve distinct topological states despite being geometrically similar.

For conciseness, in the rest of the paper, we will list crossings without their respective signs.

IV. DLO Manipulation Planning Problem

We define a DLO manipulation planning problem by the following inputs: the radius and length L of the DLO, the curvilinear abscissae $s_1, \ldots, s_k$ of the grasp points, the state transition function f, an initial configuration q_{init} of the DLO, a goal topological state x_{goal} of the DLO relative to a projective plane P, a geometric and kinematic model of the robotic arms used to manipulate the DLO, and the geometry and positions of a set of fixed obstacles.

The solution to this problem is a sequence of collision-free paths of the arms that achieve the goal state x_{goal}, that is, a configuration $q \in C(x_{\mathrm{goal}})$. Any two consecutive paths in this sequence are separated by (re-)grasp operations. During the manipulation, the robots are not allowed to touch the DLO, except at the grasp points. But the DLO is allowed to touch obstacles. The transition function f models the interaction between the DLO and the obstacles, and rejects attempted moves that cause the DLO to touch an arm.

In the rest of this paper, we simplify this problem by making the following assumptions:

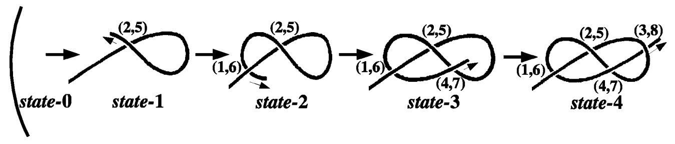


Fig. 5. This figure-8 knot is tied crossing-by-crossing in the order implied by its forming sequence $[(2,5), (1,6), (4,7), (3,8)]$.

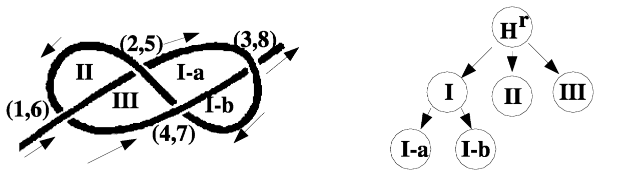


Fig. 6. Loop structure of the figure-8 knot.

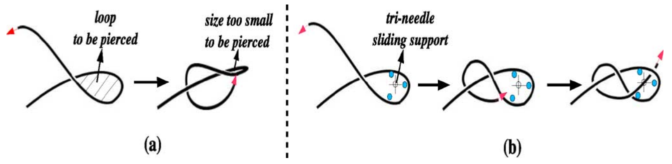


Fig. 7. (a) After a split loop has been created, it must remain sufficiently wide open to allow the head of the DLO to be passed through it. (b) Static sliding support, called a tri-needle, is used to maintain the size of the loop.

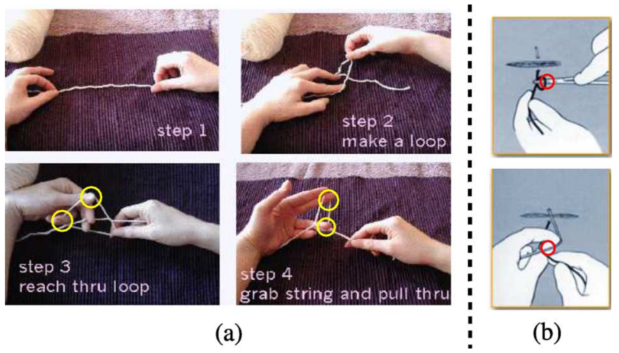


Fig. 8. In real life, sliding supports [fingers in (a) and scissors in (b)] are commonly used while tying knots.

1) The DLO admits only two-point grasp, located at $s = 0$ (tail) and $s = L$ (head). The tail of the DLO is fixed at some given position and orientation.
2) The robotic system consists of two arms, both of which can grasp the DLO's head. At any point of time, a single arm moves the head, but both arms can simultaneously grasp the head to eventually switch grasp.
3) In its initial configuration q_{init}, the DLO has no crossing in P (we say that it is *unwound*).
4) Simple static sliding supports, called needles, are available to maintain the integrity of certain portions of the DLO during manipulation. But the planner must decide when and where to use them.

In Section VI, we will extend the definition of a topological state of a DLO to take obstacles into account, in order to tie knots around obstacles or, instead, to avoid undesired loops of the DLO around obstacles.

V. Planning Approach

A. Forming Sequence

The first step of our planning approach ignores the manipulating arms and determines the order in which the crossings in the goal state x_{goal} of the DLO will be achieved. We call this abstract plan the *forming sequence* of the goal state. It will be used in a second stage to bias the sampling of a probabilistic roadmap in the DLO's configuration space.

Suppose we walk along the DLO, in a given configuration q, from its tail to its head. We say that a crossing is *formed* when it is encountered for the second time. The forming sequence of the DLO at configuration q is the sequence in which crossings are formed during the walk. So, if the goal state is $x_{\text{goal}} = \{(X_j^1, X_j^2)\}_{j=1,\ldots,n}$, then the forming sequence of the DLO at any configuration in $C(x_{\text{goal}})$ is $((X_j^1, X_j^2))_{j=j_1,\ldots,j_n}$, where $\{j_1, j_2, \ldots, j_n\}$ is a permutation of $\{1, 2, \ldots, n\}$ such that $X_{j_k}^2 < X_{j_l}^2$ if $k < l$. Knots can be created crossing-by-crossing in the order defined by the forming sequence of the goal state (see Fig. 5). However, this is not a unique way of tying knots.

B. Loop Structure and Needles

Let $q = (c, E)$ be any configuration of the DLO in $C(x_{\text{goal}})$ and c' be the projection of c in the projective plane P. Let us walk along the DLO from the tail to the head and concurrently draw c'. Each time a crossing is formed, either a new *loop* of c' is created or an existing loop is split into two loops. For example, in Fig. 6, the crossing (2, 5) is formed first. When it is formed, loop I is created. Then, crossing (1, 6) creates loop II and crossing (4, 7) creates loop III. Finally, crossing (3, 8) splits loop I into two loops denoted by I-a and I-b. The *loop structure* is the hierarchy of all the loops thus formed. The root of this structure points to the newly created loops (I–III in the example of Fig. 6). Each loop in the structure that has been split (only loop I in Fig. 6) points toward the two loops resulting from that split. The structure may have arbitrarily many levels.

During manipulation, it is critical to maintain each split loop (loop I in Fig. 6) sufficiently wide open, so that the robot arms can move the head of the DLO through it. Our planner achieves this condition by means of a *tri-needle*, as illustrated in Fig. 7. This tri-needle consists of three thin straight parallel bars inserted through the loop perpendicular to P, but it could be designed in other ways. It is inspired from the way people use their extra fingers and tools during manipulation. While two fingers in one hand (usually, the thumb and the index) are used to grasp a DLO, other fingers are often used to maintain the integrity of loops [see Fig. 8(a)]. Tools such as scissors and needles may also be used as sliding supports [Fig. 8(b)]. The size of a tri-needle used to maintain the integrity of a split loop O depends on whether any of the two loops resulting from the split of O will be split in turn. Our experiments have shown that the tri-needles critically contribute to making the manipulation

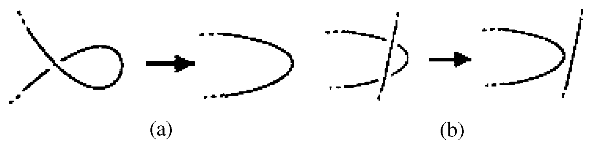


Fig. 9. (a) Reidmeister move I. (b) Reidmeister move II.

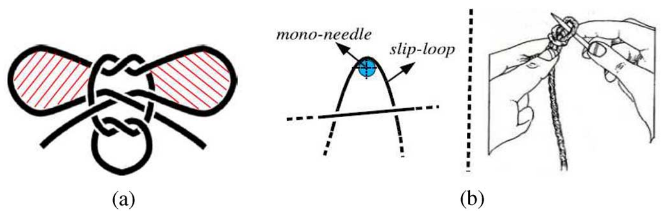


Fig. 10. (a) Semi-tight topological state. (b) Needle, inspired from real life (right), is used to preserve a slip loop.

plans generated by planner robust to inaccuracies in the physical model of the DLO.

Simpler needles are also used to maintain the integrity of another type of loops, which we call slip loops. We describe this type of loop next.

C. Slip Loops

Intuitively, a slip loop is one that could be undone by pulling the head of the DLO. Their definition and recognition by the planner requires introducing some additional definitions.

In knot theory [2], a knot is defined as a homeomorphic embedding of the unit circle in R^3, and two knots are said to be the same if there is an ambient isotopy between them. More simply stated, if a DLO is manipulated (excluding cutting), and the two ends are glued together, the result is a knot. Any further manipulation of the DLO (again excluding cutting) is an ambient isotopy that preserves that knot. The unknot is ambiently isotopic to the unit circle. Reidmeister moves are used in knot theory to simplify crossing configurations without changing a knot. Fig. 9 shows the Reidmeister moves I and II.

In our paper, we do not use such moves to simplify the goal topological state x_{goal} of the DLO. Instead, we assume that this state has been input so that all the loops it implies are desired loops, including loops that could be eliminated using Reidmeister moves. However, since common knots do not contain arbitrary loops, we make some additional assumptions about the loops that x_{goal} may imply.

We say that x_{goal} is *tight* if no crossing can be removed by any Reidmeister move, and its forming sequence is alternating, i.e., the successive crossings in the forming sequence are alternately over and under. The topological state depicted in Fig. 3(a) is tight. If x_{goal} is tight, then only the integrity of split loops need to be maintained using tri-needles.

However, we also allow x_{goal} to be semitight. We define a *semitight* state to be the one in which the forming sequence is such that:

1) no more than two consecutive crossings are all over or all under;
2) any two consecutive crossings (X_j^1, X_j^2) and (X_k^1, X_k^2) that are both over or both under verify $|X_j^1 - X_k^1| = 1$ and $|X_j^2 - X_k^2| = 1$.

Fig. 10(a) shows a semitight crossing configuration. Most practical knots that rely on frictional contact along the DLO for their integrity (e.g., shoe-lace knot) yield semi-tight states. In such a knot, a loop bounded by a curve segment joining two consecutive crossings that are both over or under is called a *slip* loop. In the shoe-lace knot of Fig. 10(a), there are two slip loops shown with striped interiors. To prevent a slip loop from being undone during manipulation, a mono-needle perpendicular to P is used, as shown in Fig. 10(b).

```
Algorithm TWO-ARM-KNOTTER ($q_{\mathrm{init}}$, $x_{\mathrm{goal}}$, $P$)
1. $\varphi \leftarrow$ Forming-Seq($x_{\mathrm{goal}}$), $\sigma \leftarrow$ Loop-Struct($\varphi$)
2. $R$.Insert(NULL, $q_{\mathrm{init}}$)
3. Loop $Max$ times
    a. Pick $q$ from $R$ with a probability measure biased
       towards $x_{\mathrm{goal}}$
    b. Pick a control vector $u$ at random
    c. $q_{\mathrm{new}} \leftarrow f(q, u)$
    d. Return to Step 3 if:
       - $f$ reports that the move was impossible, or
       - the forming sequence of $q_{\mathrm{new}}$ is not a
         sub-sequence of $\varphi$
    f. If a slip or split loop is about to be formed
       then add a (tri-)needle
    g. Return to Step 3 if the arms cannot perform $u$
    h. $R$.Insert($q$, $q_{\mathrm{new}}$)
    i. If $q_{\mathrm{new}} \in C(x_{\mathrm{goal}})$ then exit with manipulation path
4. Exit with *failure*
```

Fig. 11. DLO manipulation planning algorithm.

D. Motion Planning Algorithm

The algorithm TWO-ARM-KNOTTER is shown in Fig. 11 (also see Fig. 12). Its inputs are the initial configuration of the DLO (q_{init}), the goal topological state of the DLO (x_{goal}), and the projective plane P in which this state is defined. P is assumed to be vertical. One of the robotic arms is assumed to be holding the head of the DLO (at q_{init}) in the beginning.

The algorithm first computes the forming sequence φ and the loop structure σ of x_{goal} (step 1). Then, it constructs a tree-shaped probabilistic roadmap R in the DLO's configuration space. Each node of R is a sampled configuration of the DLO and each edge is a transition computed by the state-transition function f (see Section III-B). At step 2, the algorithm installs the initial configuration q_{init} of the DLO as the root of R. At step 3, it iteratively adds new nodes (and edges) to R, until either R contains a path connecting q_{init} to a configuration in $C(x_{\mathrm{goal}})$, or *Max* iterations have been performed unsuccessfully. At each iteration of step 3, the algorithm picks a node q from R (step 3-a), a small move u of the DLO's head (step 3-b), and evaluates $f(q, u)$ to obtain a new configuration q_{new} (step 3-c), which will be added to R as a child of q (step 3-h). This part of the algorithm

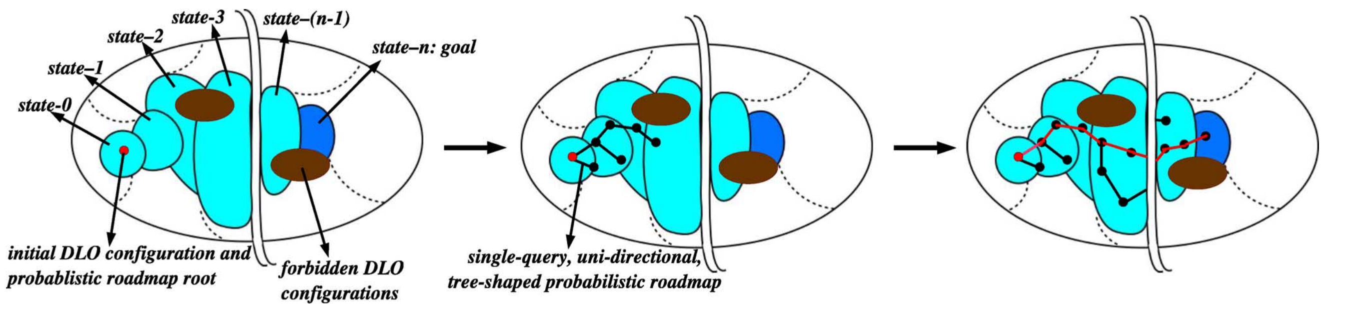

state-0
state-1
state-2
state-3
state-(n-1)
state-n: goal
initial DLO configuration and probablistic roadmap root
forbidden DLO configurations
single-query, uni-directional, tree-shaped probabilistic roadmap

Fig. 12. Our manipulation planning algorithm works by growing a tree-shaped probabilistic roadmap in the topologically partitioned configuration space of the DLO. The tree is grown until it discovers the goal subspace where all configurations have the same topology characterized by the crossing configuration $x_{\rm goal}$. The subspace state-0, corresponding to the initial DLO configuration, and the goal subspace state-n, is separated by a sequence of subspaces ($\mathtt{state}$-1, ..., $\mathtt{state}$-n-1). The subspace state-i consists of all DLO configurations whose crossing configuration is a subsequence of FS_g (forming sequence of $x_{\rm goal}$) formed by the first i crossings of FS_g.

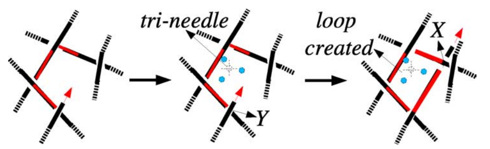


Fig. 13. Suppose a split loop is created when the crossings Y followed by X are formed. The planner places a tri-needle just after the crossing Y is formed, along the DLO in the region where the loop is likely to be formed (surrounded by the red curves), which is determined by matching the crossings in the current configuration of the DLO with the crossings that define the loop in $x_{\rm goal}$.

is similar to the approach previously used in [7] and [13] to plan trajectories under kinodynamic constraints.

However, prior to adding $q_{\rm new}$ to R, the algorithm checks several conditions. First, $f(q, u)$ must have reported that the motion induced by the control vector u is feasible (see Section III-B), and the forming sequence at $q_{\rm new}$ must be φ itself or a subsequence of φ beginning at the same crossing as φ. If any of these two conditions is not satisfied, the algorithm ends the step 3 current iteration and starts a new one (step 3-d). Otherwise, the algorithm uses the loop structure σ to determine whether a new needle is needed. Needles are needed whenever the first crossing of a slip or a split loop has been achieved in $q_{\rm new}$. The configuration $q_{\rm new}$ is then used to select the placement of the new needle, as illustrated in Fig. 13. After the needles have been placed, the head of the DLO is constrained to move around them till the respective loop has been formed.

Next, at step 3-g, the planner checks that the robotic arm currently holding the head of the DLO can track the motion of the DLO's head defined by u without colliding. If not, it checks whether the other arm can perform the motion instead, after a grasp switch between the two arms at q. In our implementation, the collision-free motion of the arms performing a grasp switch is computed using the probabilistic roadmap planner described in [25]. If no feasible motion for the arms is found, then TWO-ARM-KNOTTER ends the current iteration at step 3 and starts a new one.

If the aforementioned conditions are satisfied, then the new configuration $q_{\rm new}$ is added to R as a child of q (step 3-h). The motion of the arms is stored with $q_{\rm new}$ in R, along with the positions of all the needles, if any, that have been placed between $q_{\rm init}$ and $q_{\rm new}$.

The planner terminates with success (step 3-i) when it achieves a configuration $q_{\rm new} \in C(x_{\rm goal})$. It then returns a manipulation path, which is a series of collision-free paths of the arms, separated by (re-)grasp operations, and the description of needle placements. The planner terminates with failure (step 4) if it has not achieved a configuration in $C(x_{\rm goal})$ after a prespecified number (*Max*) of iterations at step 3.

At step 3-a, the configuration q is selected at random among the nodes currently in R, with a probability measure that favors the nodes with more crossings, since they are topologically closer to $C(x_{\rm goal})$. At step 3-b, the control vector u is a small move of the DLO's head selected uniformly at random. One can also bias this choice in order to favor the creation of the next crossing in FS_g.

Currently, our planner does not plan for the manipulation of the needles (a separate rigid body manipulation planner could be used to plan the needle manipulations). So, in our experimental setup, the needles are hand-placed at the positions determined by the planner (see Section VII). However, once it has planned for the addition of a new needle, the planner treats this needle as an additional obstacle to the future motion of the robotic arms. The position of the needle is also passed to the transition function, so that this function appropriately computes the new motions of the DLO.

VI. TYING KNOTS AROUND STATIC RIGID OBJECTS

So far, the topological states of a DLO have not taken other objects into account. With such states, there is no way to state that TWO-ARM-KNOTTER must or must not wind the DLO around some static objects. Here, we propose a simple technique to describe topological states taking static objects in the environment into account. This technique is restricted to a class of objects defined as follows: an object B is in this class if there exists a curve $r(B)$ such that any point in B is within distance δ from $r(B)$, where δ is small compared to the length of the curve. Simple examples of such objects include torus and long cylinders. In general, the curve representations of such objects can

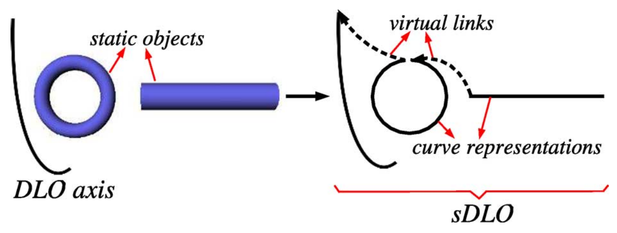


Fig. 14. sDLO is constructed by chaining the curve representations of the static objects with the DLO axis.

be obtained by skeletonizing the objects [5] and pruning less prominent branches of the skeleton.

To describe the topological state of a DLO among static objects, we define an sDLO combining the DLO and curve representation of the static objects, and then, we describe the topological state of the sDLO. Let S be the set of curve representations of the static objects. The sDLO is obtained by chaining the curves in S and the DLO's axis c sequentially, with c being the last, as illustrated in Fig. 14. Virtual links connect consecutive curves in the sequence.

The topological state of an sDLO is defined in the same way as for a DLO, except for the following difference. A crossing in the projective plane P is any point X that is the projection of two distinct points of the sDLO, one of which at least being a point of the DLO's axis c. Said otherwise, we simply ignore the crossings that only involve curves in S and virtual links, since these crossings do not contribute to the topological state of the DLO. The TWO-ARM-KNOTTER algorithm remains unchanged.

Here, we choose a projective plane such that the curves in S are minimally distorted after projection. This reduces the possibilities of crossings, between the projections of a curve in S and the DLO axis c, lumping into a small region. Any plane parallel to the one that minimizes the sum of squares of distances between the points of the curves in S and the plane could be used. However, it is difficult to choose a good plane in situations when a plane suited for one object is bad for another object, or when an object is not quasi-planar.

VII. Experimental Results

We implemented the DLO manipulation planner TWO-ARM-KNOTTER in C++ and ran knot-tying experiments on a 1.5 GHz Intel Xeon PC with 1 GB RAM. In this implementation, we use the physical model of a DLO described in [27], which takes into account the essential mechanical properties of a typical DLO such as stretching, compressing, bending, and twisting, as well as the effect of gravity. It manages self-collisions efficiently and also accounts for the interaction of the DLO with other static objects in the environment. We tuned the parameters of the rope model such that the simulated rope "visually" behaved like a typical rope, as one of its ends was being manipulated and the other end was kept fixed. Two robot arms, each with 6 DOF, are used to perform the manipulation. The videos of the result reported in this paper are available at http://ai.stanford.edu/∼mitul/dlo. They have also been submitted as multimedia material along with this paper.

A. Results in Simulation

Fig. 15 shows snapshots from the manipulation motions generated by the planner for five manipulation problems. In the first four problems, the goal is to tie common knots, respectively, bowline, neck-tie, bow (shoe-lace), and stun-sail. In the fifth problem, the goal requires winding the DLO around static objects. Bowline and bow require one tri-needle each, while neck-tie and stun-sail require two tri-needles each. Bow also requires a mono-needle to maintain its slip loop. The fifth problem does not require any needle. The planner took between 10 and 15 min of CPU time to generate each of these motions.

B. Test on Hardware Platform

We tested manipulation plans generated by TWO-ARM-KNOTTER on a hardware platform made up of two PUMA-560 arms. Fig. 16(a) shows several snapshots of the two arms tying a bowline knot. For this plan, a single tri-needle is needed as soon as the execution of the plan starts [see snapshot from the first example in Fig. 15(a)]. This needle, which is made of three thin aluminum rods, was hand-placed at the position determined by the planner. We executed the same plan computed by TWO-ARM-KNOTTER using six types of ropes with different materials and diameters [Fig. 17(a)]. The rope model used by the planner did not exactly represent the physics of any of these ropes. However, the execution of the plan succeeded consistently on the first five ropes, but failed with the plastic rope, which turned out to be too stiff for the motion plan. Fig. 17(b) shows a final bowline shape obtained with each of the first five ropes.

The ability of our planner to generate plans that are robust to modeling inexactness relies critically on the needles. We consider them as key tools for manipulating DLOs. However, both sensory feedback and re-planning could still be used to further improve the robustness.

C. Sensitivity Analysis

In the aforementioned experiment, we were actually surprised that the same manipulation plan worked well with quite different ropes. This led us to quantify the sensitivity of plans generated by TWO-ARM-KNOTTER to inaccuracies in the rope model. Since rope manufacturers do not usually provide numerical values for the mechanical properties of ropes, we proceeded in the following manner. After generating a manipulation plan for tying a bowline using a particular rope model, we tested in simulation if the same manipulation plan still achieves a bowline after corrupting the rope model parameters with Gaussian noises. Table I lists the means and standard deviations (estimated numerically) for the Gaussian distributions from which three main parameters α, β, and γ of the model were independently chosen, such that a bowline was achieved more than 90% of the time. The variables α, β, and γ in [27] correspond to stretching, bending, and twisting constants of the spring primitives, respectively. The mean values in Table I correspond to the original model parameter values, used to generate the manipulation plan. The high values of the standard deviations indicate high degree of robustness. As suggested earlier, the robustness is mainly derived from the

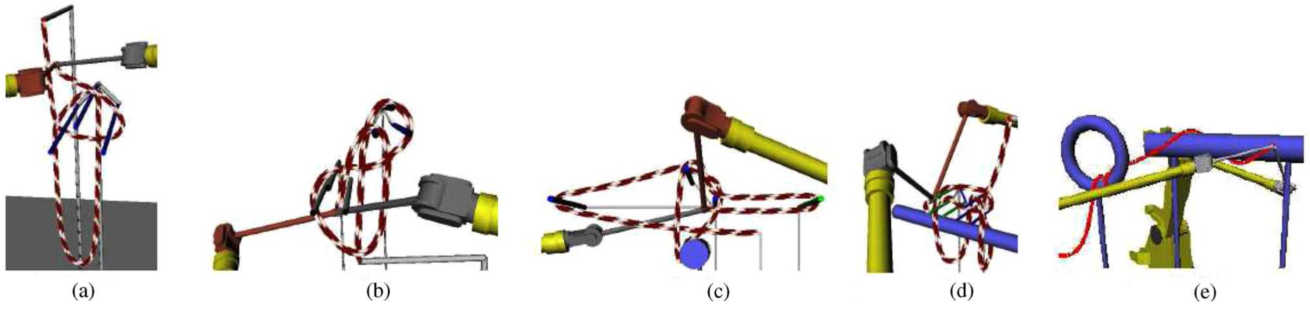


Fig. 15. Snapshots from manipulation plans generated by TWO-ARM-KNOTTER for five manipulation problems.

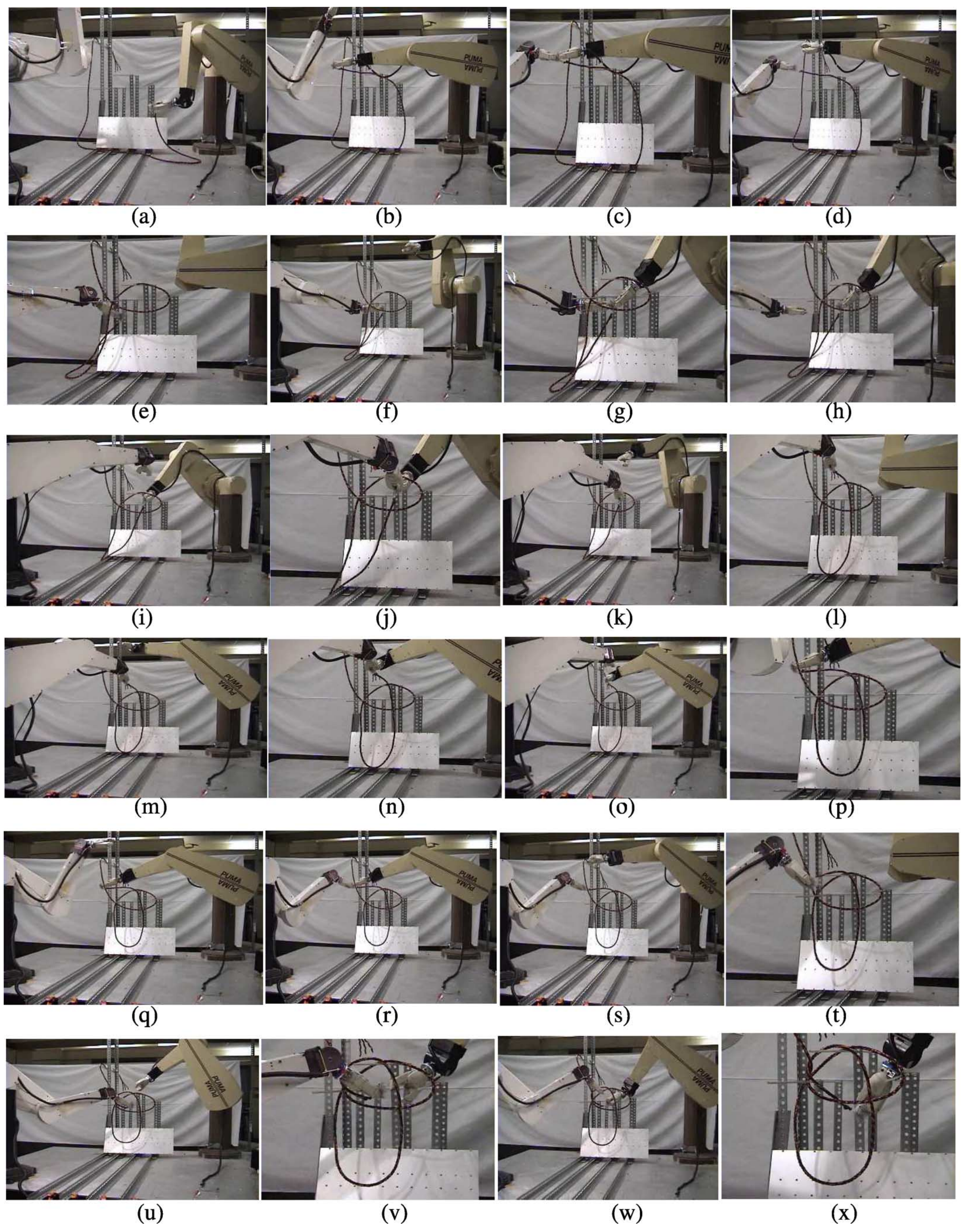
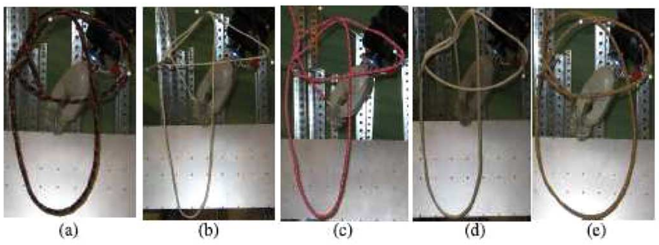


Fig. 16. Two PUMA-560 arms tying a bowline knot.

	Rope	Material	Thickness(mm)
(a)		polypropolene	9.66
(b)		cotton	3.44
(c)		polyester	6.48
(d)		nylon	5.66
(e)		jute	5.94
(f)		plastic	4.55

(a)



(b)

Fig. 17. (a) Different types of ropes used. (b) Final shapes of bowline knots achieved with different ropes.

TABLE I
MEANS AND STANDARD DEVIATIONS OF GAUSSIAN DISTRIBUTIONS FROM WHICH THE THREE MAIN ROPE MODEL PARAMETERS WERE CHOSEN FOR THE ROBUSTNESS ANALYSIS

model parameter	mean	standard deviation
α	13500	3950
β	1250	455
γ	25	15

needles that provide structural support to unstable portions of the DLO configuration.

VIII. CONCLUSION AND FUTURE DIRECTIONS

This paper described a motion planner for manipulating DLOs using two cooperating robotic arms to tie self-knots and knots around simple static objects. The planner does not assume a specific physical model of the DLO. Instead, it takes a model encoded in the form of a state transition function as input. This model may, thus, be adjusted to the type of DLO manipulated. We have demonstrated the effectiveness of our planner by tying some commonly used knots, both in simulation and on a real robotic platform. To our knowledge, our planner is a first of its kind, i.e., we are not aware of any previous planner that generates the motions of robotic arms to manipulate a DLO in 3-D environments with obstacles.

For the future, we suggest the following:

1) Currently, our planner assumes that the DLO is being manipulated using a single grasp point located at the head of the DLO. But, in practice, many knots, for example, the shoelace knot, are more conveniently tied by grasping the DLO at locations other than the endpoints. One can try to extend our planner so that it can also choose intermediate grasp points. Also, one can allow the robot arms to contact the DLO at any point along the DLO and on the arms. The design of the state transition function f would not be significantly different, but the planner would be much more complex, since it could potentially use these additional contacts in very creative ways.
2) It would be interesting to study the extent to which a given DLO is controllable using our needle-dependent two-robot manipulation system. We believe that one would have to take into account the physical model of the DLO while doing any controllability analysis.
3) The technique that we use to measure the topology of a DLO configuration (Section III-C) is a conservative approach. That is, in many cases, the technique will conclude that two DLO configurations are topologically different even though they might be equivalent [2], [15]. In the future, one should investigate if a more accurate technique, based on more recent developments in knot theory, can be used instead. In fact, in our planner implementation, we are able to resolve some of the ambiguities by simply perturbing the projective plane P.
4) One could also investigate if the topological metrics used in [23] could also be used by our topological planner to bias its search.
5) Since it is difficult to choose a good projective plane when there are several static objects or when an object is not quasi-planar, one could try to extend our method to tie knots around objects with the help of multiple projective planes. Input objects could be broken into a number of quasi-planar objects, and a separate plane could be used for each of them.
6) The new concepts of forming sequence and loop structure could seed future research in knot theory and its application domains.

Finally, we hope that our contribution in this paper will eventually lead to the opening of crucial application domains for robotics in future, automated surgical suturing being one of them. It could be of particular interest to humanoid robotics—aiming to assist humans in their daily life activities, knot-tying being one of them.

ACKNOWLEDGMENT

The authors thank Prof. J.-C. Latombe for his overall advisory role, Prof. E. Burdet and F. Wang for giving the access to the rope model developed in their research group at the National University of Singapore, Prof. O. Khatib for allowing to use the PUMA 560 robots in his laboratory, and J. Park, J. S. Kwon, I. Pashchenko, and A. Dhanik for helping with the experimental setup.

REFERENCES

[1] J. Acker and D. Henrich, "Manipulation of deformable linear objects: From geometric model towards program generation," in *Proc. IEEE Int. Conf. Robot. Autom.*, Barcelona, Spain, 2005, pp. 1553–1559.
[2] C. C. Adams, *The Knot Book*. New York: Freeman, 1994.
[3] J. Brown, J. C. Latombe, and K. Montgomery, "Real-time knot tying simulation," *Vis. Comput. J.*, vol. 20, no. 2/3, pp. 165–179, 2004.

[4] H. Choset, K. M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun, *Principles of Robot Motion*. Cambridge, MA: MIT Press, 2005.

[5] N. Gagvani and D. Silver, "Parametric controlled volume thinning," *Graph. Models Image Process.*, vol. 61, no. 3, pp. 149–164, May 1999.

[6] A. Dhanik, "Development of a palpable virtual nylon thread and handling of bifurcations" Master's thesis, Nat. Univ Singapore (NUS), Singapore, 2005.

[7] D. Hsu, R. Kindel, J. C. Latombe, and S. Rock, "Randomized kinodynamic motion planning with moving obstacles," *Int. J. Robot. Res.*, vol. 21, no. 3, pp. 233–255, 2002.

[8] K. Hirana, T. Suzuki, S. Okuma, K. Itabashi, and F. Fujiwara, "Realization of skill controllers for manipulation of deformable objects based on hybrid automata," presented at the Int. Conf. Robot. Autom., Seoul, Korea, May 2001.

[9] H. Inoue and M. Inaba, "Hand–eye coordination in rope handling," in *Proc. Robot. Res.: First Int. Symp.*. Cambridge, MA: MIT Press, 1984, pp. 163–174.

[10] R. Alami, T. Simeon, and J. P. Laumond, "A geometrical approach to planning manipulation tasks," in *Proc. Int. Symp. Robot. Res.*, 1989, pp. 113–119.

[11] A. M. Ladd and L. E. Kavraki, "Using motion planning for knot untangling," *Int. J. Robot. Res.*, vol. 23, no. 7/8, pp. 797–808, 2004.

[12] F. Lamiraux and L. E. Kavraki, "Planning paths for elastic objects under manipulation constraints," *Int. J. Robot. Res.*, vol. 20, no. 3, pp. 188–208, 2001.

[13] S. M. LaValle and J. J. Kuffner, "Randomized kinodynamic planning," *Int. J. Robot. Res.*, vol. 20, no. 5, pp. 378–400, May 2001.

[14] M. Moll and L. E. Kavraki, "Path planning for deformable linear objects," *IEEE Trans. Robot.*, vol. 22, no. 4, pp. 625–636, Aug. 2006.

[15] K. Murasugi, *Knot Theory and Its Applications*. Boston, MA: Birkhäuser, 1996.

[16] B. Kahl and D. Henrich, "Virtual robot programming for deformable linear objects: System concept and prototype implementation," presented at the Int. Symp. Meas. Control Robot., Bourges, France, 2002.

[17] L. E. Kavraki, P. Svestka, J. C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Trans. Robot. Autom.*, vol. 12, no. 4, pp. 566–580, Aug. 1996.

[18] Y. Koga and J. C. Latombe, "On multi-arm manipulation planning," in *Proc. IEEE Int. Conf. Robot. Autom.*, San Diego, CA, 1994, pp. 945–952.

[19] T. Matsuno, T. Fukuda, and F. Arai, "Flexible rope manipulation by dual manipulator system using vision sensor," in *Proc. IEEE/ASME Int. Conf. Adv. Intell. Mechatron.*, Como, Italy, 2001, pp. 677–682.

[20] T. Morita, J. Takamatsu, K. Ogawara, H. Kimura, and K. Ikeuchi, "Knot planning from observation," in *Proc. IEEE Int. Conf. Robot. Autom.*, Taipei, Taiwan, 2003, pp. 3887–3892.

[21] D. K. Pai, "Strands: Interactive simulation of thin solids using cosserat models," presented at the Eurographics, Saarbruken, Germany, 2002.

[22] J. Phillips, A. Ladd, and L. E. Kavraki, "Simulated knot tying," in *Proc. IEEE Int. Conf. Robot. Autom.*, Washington, DC, 2002, pp. 841–846.

[23] P. Rogen and H. Bohr, "A new family of global protein shape descriptors," *Math. Biosci.*, vol. 182, no. 2, pp. 167–181, Apr. 2003.

[24] A. Remde, D. Henrich, and H. Worn, "Picking-up deformable linear objects with industrial robots," presented at the Int. Symp. Robot., Tokyo, Japan, 1999.

[25] G. Sanchez and J. C. Latombe, "A single-query bi-directional probabilistic roadmap planner with lazy collision checking," presented at the Int. Symp. Robot. Res., Lorne, Vic., Australia, 2001.

[26] T. Simeon, J. P. Laumond, J. Cortes, and A. Sahbani, "Manipulation planning with probabilistic roadmaps," *Int. J. Robot. Res.*, vol. 23, no. 7/8, pp. 729–746, 2004.

[27] F. Wang, E. Burdet, V. Ronald, and H. Bleuler, "Knot-tying with visual and force feedback for VR laparoscopic training," in *Proc. 27th IEEE EMBS Annu. Int. Conf.*, Shanghai, China, 2005, pp. 5778–5781.

[28] G. Wilfong, "Motion planning in the presence of movable obstacles," presented at the ACM Symp. Comput. Geom., Urbana-Champaign, IL, 1988.

[29] H. Wakamatsu and S. Hirai, "Static modeling of linear object deformable based on differential geometry," *Int. J. Robot. Res.*, vol. 23, no. 3, pp. 293–311, 2004.

[30] H. Wakamatsu, A. Tsumaya, E. Arai, and S. Hirai, "Knotting/unknotting manipulation of deformable linear objects," *Int. J. Robot. Res.*, vol. 25, no. 4, pp. 371–395, 2006.

[31] H. Wakamatsu, K. Takahashi, and S. Hirai, "Dynamic modeling of linear object deformation based on differential geometry coordinates," in *Proc. IEEE Int. Conf. Robot. Autom.*, Barcelona, Spain, 2005, pp. 1028–1033.

Mitul Saha received the M.S. degree in computer science and the Ph.D. degree in mechanical engineering from Stanford University, Stanford, CA, in 2005 and 2006, respectively.

He is currently a Postdoctoral Fellow at the NIH Center for Biomedical Computation, Stanford University. His current research interests include motion planning, computer vision, graphics, and structural biology.

Pekka Isto (M'06) received the M.Sc. degree from the University of Helsinki, Helsinki, Finland, in 1998, and the Ph.D. degree from Helsinki University of Technology, Espoo, Finland, in 2003, both in computer science.

During 2004, he was a Postdoctoral Fellow in the Department of Computer Science, Stanford University, Stanford, CA. He is currently a Senior Researcher with the Research Institute in Technology, University of Vaasa, Vaasa, Finland. His current research interests include robot motion planning and control, agent systems, parallel computing, and design knowledge representation and management.