Motion Planning for Robotic Manipulation of Deformable Linear Objects

Mitul Saha[(1)], Pekka Isto[(2)], and Jean-Claude Latombe[1]
(1) Artificial Intelligence Lab, Stanford University, Stanford, CA, USA. (email: {mitul,latombe}@cs.stanford.edu)
(2) Computer Science Department, University of Vassa, Finland. (email: isto@uwasa.fi)

***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. This paper describes a new motion planner for manipulating DLOs and tying knots (self-knots and knots around simple static objects) using cooperating robot arms. The planner constructs a topologically-biased probabilistic roadmap in the DLO's configuration space. Unlike in traditional motion planning problems, the goal is a topological state of the world, rather than a geometric one. The implemented planner was tested in simulation to achieve various knots like bowline, neck-tie, bow (shoe-lace), and stun-sail. In real-life, the planner was used to tie bowline knots with various household ropes on a hardware platform with two PUMA 560 robots.**

Index Terms—Manipulation planning, deformable objects, knot tying, 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 many application domains, like manufacturing (loading cable harnesses and robot dresses), medical surgery (especially suturing), and agriculture, where DLOs are ubiquitous. It can also benefit service-based humanoid robots, since tying knots is a common activity in daily life. However, DLOs add a number of difficulties to the manipulation task. They exhibit a much greater diversity of behaviors than rigid objects, by taking many different shapes when submitted to external forces. In particular, self-collisions are possible and must be considered. Furthermore, the manipulation of DLOs almost inevitably requires two, or more, arms performing well-coordinated motions and re-grasp operations. Finally, the topology of the goal state of a DLO is usually far more important than its exact shape.

In this paper we describe a new motion planner for manipulating DLOs with two cooperating robot arms. Figure 1 shows two typical problems. In Figure 1(a), a segment of rope is initially unwound. Figures 1(b) & (c) depict two types of goal states in which the rope forms a self-knot (bowline) and winds around some static objects, respectively. Our planner does not depend on any particular physical model of the DLO. Instead, it takes a model as input, in the form of a state-transition function. Using this function and the model of the robot arms, the planner constructs a probabilistic roadmap in the configuration space of the DLO. The sampling of this roadmap 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 used to account for the interaction of the DLO with simple rigid objects. Curve representations of the objects, obtained from their skeletonization, are "chained" with the DLO to produce a *composite* semi-deformable linear object (sDLO). Thereafter, the problem reduces to that of tying self-knots with the composite sDLO. We implemented the planner and tested it in simulation and real-life. We demonstrated its effectiveness by tying commonly used knots like bowline, neck-tie, bow (shoe-lace), and stun-sail.

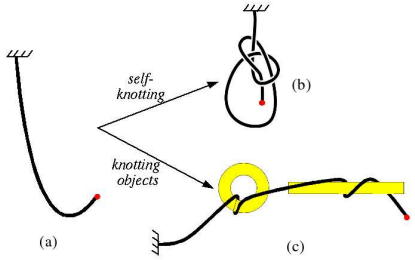


Fig. 1. Initial (a) and goal (b)-(c) states in a manipulation problem

II. Relation To Previous Work

A. Manipulation Planning

Robot manipulation planning with rigid objects was first addressed in [22]. The randomized algorithm proposed in [10] generates motion paths for multiple cooperating manipulators to manipulate a movable rigid object. The algorithm assumes that a set of discrete grasps is given as input. The algorithm proposed in [19] works with continuous sets of grasps of rigid objects but plans for a single manipulator. The re-grasp is done by releasing the object. In the case of a DLO, releasing the grasp makes the DLO very unstable due to its deforming nature. An algorithm for planning paths for elastic objects, especially for flexible plates and cables, is presented in [8].

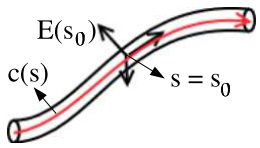
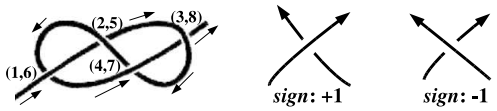


Fig. 2. Representation of a DLO



(a) (b)

Fig. 3. (a): The crossings in the "figure-8" knot. (b): The sign convention for the crossings

This algoritm plans motions only for the object, not for the manipulator. The problem of path planning for DLOs in presence of obstacles is addressed as a variational problem in [20]. Their formulation does not consider the manipulator either. All these works focus on geometric planning while we focus on topological planning, because while manipulating DLOs, especially to tie knots, it is more important to achieve an acceptable topology than a specific geometry.

B. Application of Knot Theory in Robotics

Knot theory provides means to capture and analyse the topological states and state transitions of a DLO [1]. Its applications in robotics include work presented in [12]. Like us, they present a data structure for describing the state of a DLO as a sequence of signed crossings. State transitions are caused by Reidemeister moves and a crossing operation that moves the end of the DLO over another part to make a new crossing. Similar ideas are being used in [21] to build a vision guided robot system for one-handed manipulation of a DLO with the aid of the floor. However, collision constraints and the physical behavior of the DLO are not considered during the planning phase. In [7], motion planning techniques from robotics are used to untangle mathematical knots.

C. Vision-Based DLO Manipulation

The difficulty of accurately modeling deformable objects has motivated vision-based approaches to DLO manipulation. Among other examples, in [11], methods for DLO modeling, recognition, and parameter identification are presented, which have been embedded in a system capable of tying a rope around a cylinder with two manipulators. In [15], a sensing-based method is proposed for picking up hanging DLOs. In our work, we do not rely on the availablity of any sensing system to guide the manipulation in real-time. Instead, we focus on precomputing manipulation plans which 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 length L and circular cross-section of constant non-zero radius (see Figure 2). The $axis$ of this cylinder is a smooth curve c parameterized by the Euclidean length s along the DLO, i.e.:

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

where $c(0)$ is the $tail$ of the DLO and $c(L)$ its $head$. Whenever the physical model of the DLO includes torsion or twist, we also attach a Cartesian coordinate frame $E(s)$ with each point $c(s)$ as shown in Figure 2. We call $q = (c, E)$ a $configuration$ of the DLO.

B. Physical Model

Designated points located at $s_1, ..., s_k$ on the DLO are $grasp$ points. These are the only points whose positions $c(s_i)$ and orientations $E(s_i), i = 1, ..., k$, 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 both a configuration q_{old} of the DLO and a k-vector u of small displacements of the grasp points (the control input) to a new configuration q_{new}. We assume that both q_{old} and q_{new} are stable configurations, although the DLO may exhibit dynamic behavior (e.g., snapping) during the transition from q_{old} to q_{new}. We do not consider elapsed time between q_{old} and q_{new}, although it could be easily added to the model.

We assume that the model in f handles collisions with obstacles, as well as self-collisions, so that the DLO does not "jump" over itself or obstacles. In addition, if u would cause violations of physical constraints associated with the DLO, e.g., over-stretching, then the function f indicates that the execution of u is impossible. As we do not allow the robot arms to touch the DLO at points other than the grasp points, f reports failure if it detects a collision between the DLO and an arm.

Several physical models of DLOs proposed in the literature (e.g., [2], [5], [13], [14], [23]) are relevant to the construction of f.

C. Topological Model

We characterize the topology of the DLO at some configuration $q = (c, E)$ by means of its crossing configuration [1]. A crossing configuration is defined with respect to a reference plane P. Let c' be the projection of c into P. The crossing configuration of c with respect to P is defined only when no more than two points in c project to the same point in P. Morever, for any two points a and b in c that projects to the same point $a' = b'$ in P, c' must admit two distinct tangents at $a' = b'$, which are the respective projections of the tangents to c at a and b. Then a $crossing\ X$ in P is any point on c' that is the projection of two distinct points of c.

Let $c(s_1)$ and $c(s_2)$, with $s_1 < s_2$, be the two points on c that projects to the crossing X in P. Let τ_1 and τ_2 be the projections of the tangent to c at $c(s_1)$ and $c(s_2)$, respectively. The status of X is said to be $over$ if $c(s_1)$ lies above $c(s_2)$ with respect to P, otherwise it is $under$. Moreover, X is assigned a sign. This sign is + if (1) X is over and the counter-clockwise angle between τ_1 and τ_2 is less that π, or (2) X is under and the clockwise angle between τ_1 and τ_2 is less than π. The sign of X is - in all other cases. The sign convention is also depicted in Figure 3(b).

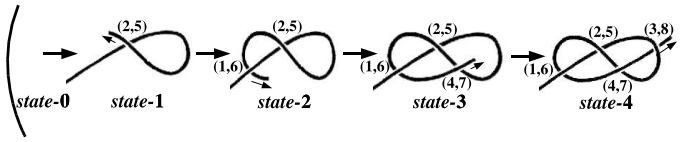


Fig. 4. A knot can be tied crossing-by-crossing in the order implied by its forming sequence. The above figure illustrates this fact for the "figure-8" knot, whose crossing configuration is $\{(1,6),(2,5),(3,8),(4,7)\}$ and the associated forming sequence is $((2,5),(1,6),(4,7),(3,8))$.

Suppose that c' has n crossings $X_1, ..., X_n$. Let us "walk" along c' from $s = 0$ to $s = L$ and assign the integral labels $1, ..., 2n$ in increasing order to the crossings as they are encountered. Since each crossing X_j is encountered exactly twice, it receives two distinct labels X_j^1 and X_j^2. Figure 3(a) illustrates the labeling of crossings in the "figure-8" knot. The $crossing\ configuration$ of c with respect to P is defined by the set of triplets:

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

where ϵ_j denotes the sign of the crossing X_j. Morever, the over/under status associated with X_j can be encoded by assigning opposite signs to X_j^1 and X_j^2. In the rest of the paper we will ignore the signs, when listing the crossings, for the sake of compactness.

Note that many configurations q of a DLO can achieve the same crossing configuration with respect to P. We denote by $C(x)$ the set of all configurations q of the DLO that achieve a crossing configuration x.

IV. DLO Manipulation Planning Problem

A DLO manipulation planning problem is defined by the following inputs: the radius and length L of the DLO, the coordinates $s_1, ..., s_k$ of the grasp points, the state transition function f, an initial configuration q_{init} of the DLO, a reference projection plane P, a goal crossing configuration x_{goal} of the DLO, a model of the robot arms forming the manipulation system, and a set of fixed obstacles.

The solution to this problem is a sequence of collision-free paths of the robots that achieve the goal crossing configuration x_{goal} with respect to P, that is, a configuration $q \in C(x_{goal})$. Any two consecutive paths are separated by (re-)grasp operations. During the manipulation, the robots are not allowed to touch the DLO, except at the grasp points. 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 make the following assumptions: (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, which can both grasp the DLO's head. At any point of time, a single arm moves the head, but both arms 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 passive/static sliding supports, which we call needles (see Section V-C), are available and can be used to maintain the integrity of certain portions of the DLO during manipulation.

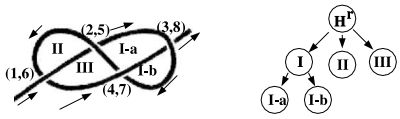


Fig. 5. The $loop\ structure$ for the "figure-8" knot

In Section VI we will extend the definition of a crossing configuration 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 is to ignore the manipulating arms and derive a "qualitative" plan, which we call the $forming\ sequence$ of the crossings in the goal topology x_{goal} of the DLO. It will be used later to bias the sampling of a probabilistic roadmap in the DLO's configuration space.

Suppose we walk along the DLO, in a given configuration, from its tail to its head. We say that a crossing is $formed$ when it is encountered for the second time. $Forming\ sequence$ is the sequence in which crossings are formed during the walk. Alternatively, if the goal crossing configuration is $x_{goal} = \{(X_j^1, X_j^2)\}_{j=1,...,n}$, then its forming sequence is $((X_j^1, X_j^2))_{j=j_1,...,j_n}$, where $\{j_1, j_2, ..., j_n\}$ is a permutation of $\{1, 2, ..., n\}$ such that $X_{j_k}^2 < X_{j_l}^2$ if $k < l$. Qualitatively, knots can be tied crossing-by-crossing in the order implied by the forming sequence of the goal crossing configuration (see Figure 4).

B. Loop structure

The loop structure is built to later identify portions of the DLO whose integrity must be maintained during manipulation by means of needles (see Section V-C). Let c' be any curve in the reference projection plane P that forms the crossings defined in x_{goal}. Let us draw c' from the tail to the head. Each time a crossing is formed, either a new $loop$ is created, or an existing loop is split into two loops. For example, in Figure 5, the crossing (2,5) is first formed, which creates the loop denoted I; 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, II, and III in Figure 5). Each loop in the structure that has been split (only loop I in Figure 5) points toward the two loops resulting from that split. The structure may have arbitrary many levels.

During manipulation, it is critical to maintain some loops sufficiently wide open, so that they can later be split. In addition, some loops could be undone by pulling the head of the DLO. The planner guarantees the integrity of all such loops by introducing needles through them, as described below.

Fig. 6. (a): Reidmeister move I (b): Reidmeister move II

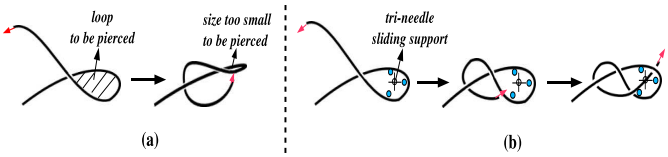


Fig. 7. (a): After a loop has been created, it is not guranteed that its size will be maintained during further manipulation (b): Passive/static sliding supports (tri-needle) is used to maintain the size of the loops to be pierced in future.

C. Pierced and slip loops

Reidmeister moves are a classical technique used in knot theory to simplify crossing configurations without changing the topology of a knot [1]. Figure 6 shows the Reidmeister moves I and II. We do not use such moves to simplify the input goal crossing configuration x_{goal}. Instead, we assume that the planner's user has appropriately described x_{goal} so that all the loops it implies are desired loops. However, since common knots do not contain arbitrary loops, we make some additional assumptions about the loops that x_{goal} may imply.

Let us say that a goal crossing configuration of a DLO in P is $tight$ if it cannot be simplified (i.e., no crossing can be removed) by any Reidmeister move and its forming sequence is $alternating$, i.e., the successive crossings in the sequence are alternately over and under. The crossing configuration depicted in Figure 3(a) is tight.

In a tight goal crossing configuration, each split loop O (loop I in Figure 5) is eventually pierced, meaning that split occurs just after two consecutive crossings of different over/under status are formed. O must be wide enough to make it possible for the robot arms to move the head of the DLO through it. The planner achieves this condition by using a tri-needle. The role of the tri-needle is illustrated in Figure 7. Its size depends on whether any of the two loops resulting from the split of O will be split in turn. The tri-needle could be defined in many ways. Here, it consists of three thin straight bars inserted through the loop perpendicular to P.

We also allow x_{goal} to be semi-tight. A $semi\text{-}tight$ crossing configuration is one in which (1) at most two consecutive crossings in the forming sequence have the same over/under status and (2) any two consecutive crossings (X_j^1, X_j^2) and (X_k^1, X_k^2) in the forming sequence that have the same over/under status are such that $|X_j^1 - X_k^1| = 1$ and $|X_j^2 - X_k^2| = 1$. Figure 8 shows a semi-tight crossing configuration. Most practical knots that rely on friction along the DLO for their integrity (e.g., shoe-lace knot) yield semi-tight crossing configurations. In such a configuration, a loop bounded by a

Fig. 8. A $semi\text{-}tight$ configuration

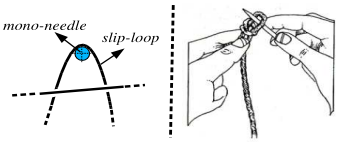


Fig. 9. A needle, inspired from real-life (right figure), is used to preserve a slip loop.

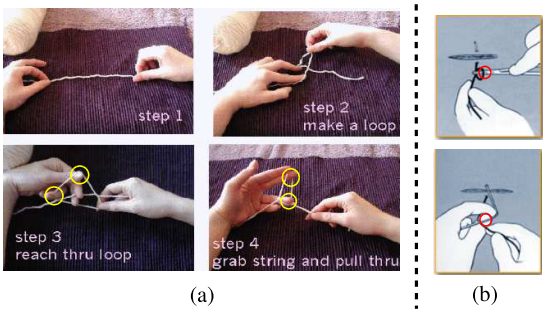


Fig. 10. In real life, sliding supports (fingers in (a) and scissors in (b)) are commonly used while tying knots.

curve segment joining two consecutive crossings with the same over/under status is called a $slip$ loop. In Figure 8 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 Figure 9. Pierced loops in semi-tight crossing configurations are handled with tri-needles as previously described.

We assume that the goal crossing configuration x_{goal} is either tight or semi-tight. Once x_{goal} is achieved, all needles can be removed by translating them perpendicular to P.

The needles are structural supports along which the DLO can slide during manipulation. They are 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 Figure 10(a)). Tools such as scissors and needles may also be used as sliding supports (Figure 10(b)).

D. Motion Planning Algorithm

The algorithm is shown in Figure 11. At Step 1 it computes the forming sequence FS_g and the loop structure H_g of x_{goal}. Then it constructs a single-query probabilistic roadmap T in the DLO's configuration space. Our approach is similar to those proposed in [6] and [9] to plan trajectories under kinodynamic constraints. The roadmap T is a tree rooted at the initial configuration q_{init}. Each node of T is a sampled configuration of the DLO and each edge is a transition computed by the state-transition function f (see Section III-B). T is built iteratively. At each iteration (Step 3), the algorithm selects a node q from T (Step 3.a) and a small move of the DLO's head u (Step 3.b), and evaluates $f(q, u)$ to obtain a new configuration q_{new}, which is then inserted in T as a successor of q (Step 3.h).

However, before q_{new} is added to T, the algorithm checks that a number of conditions are satisfied. Any violation of

Algorithm TWO-ARM-KNOTTER (q_{init}, x_{goal}, P)
1. $FS_g \leftarrow$ Forming-Seq(x_{goal}), $H_g \leftarrow$ Loop-Struct(FS_g)
2. T.Insert(NULL, q_{init})
3. Loop Max times
   a. Pick q from T with a probability measure biased towards x_{goal}
   b. Pick control vector u at random
   c. $q_{new} \leftarrow f(q, u)$
   d. Return to Step 3 if:
      - f returned that the move was impossible, or
      - the forming sequence of q_{new} is not a sub-sequence of FS_g
   f. Inspect H_g and add needles if needed
   g. Return to Step 3 if the arms cannot perform u
   h. T.Insert(q, q_{new})
   i. If $q_{new} \in C(x_{goal})$, then exit with manipulation path
4. Exit with $failure$

Fig. 11. The DLO manipulation planning algorithm

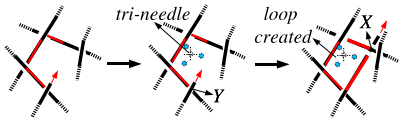


Fig. 12. Suppose a split loop is created when the crossings Y followed by X are formed. A tri-needle is placed, just after Y is formed, along the DLO in the region infered by matching the crossings associated with the current configuration of the DLO and the crossings in x_{goal} which define the loop.

these conditions leads to start a new iteration at Step 3. First, $f(q, u)$ must have indicated that the control vector u is possible (see Section III-B). If this is the case, the planning algorithm verifies that the forming sequence at q_{new} is a subsequence of FS_g beginning at the same crossing as FS_g (this causes the topological biasing of T). If yes, it checks whether new needles are needed and adds them (they then become obstacles). Needles are placed when a slip or split loop is about to be formed, as indicated by H_g. They are placed along the DLO where the loop is expected to be formed (see Figure 12). However, the planner does not plan for the manipulation of the needles. It can be done with the help of additional manipulators and some movable fixtures for the needles. Throughout the planning, the crossing configuration of x_{new} is determined with respect to P. Next, the planner checks that the robot arm currently grasping the head of the DLO can track the motion of the DLO's head without colliding (using the arm's IK). If not, it checks whether the other arm can perform the motion instead, after a grasp switch between the two arms at q. The collision free motion of the arms for performing a grasp switch can be computed using any single-query probabilistic-roadmap planner [3]. In our implementation, we use the SBL planner [17]. The motion of the arms and the needle placements are stored along with q_{new} in T.

The planner succeeds when it achieves a configuration $q_{new} \in C(x_{goal})$, which occurs when the number of crossings in q_{new} is equal to the number of crossings in x_{goal}. It returns a manipulation path, retrieved by backtracking from q_{new} to q_{init} in T, which consists of sequence of collision-free paths of the robots, separated by (re-)grasp operations, and the description of needle placements. The planner fails if it has not achieved a desired configuration after a specified number of iterations at Step 3.

At Step 3.a, the configuration q is selected at random among the nodes currently in T, with a probability measure that favors the nodes with more crossings, since they are topologically closer to $C(x_{goal})$. At Step 3.b, the control vector u is a small move of the DLO's head selected uniformly at random. Here, one can also bias this choice to favor the creation of the next crossing in FS_g.

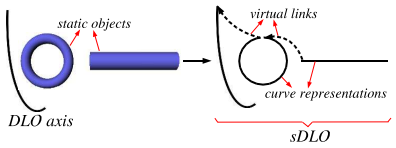


Fig. 13. A composite semi-deformable linear object sDLO is constructed by chaining the curve representations of the rigid objects with the DLO axis. Thereafter, sDLO is used to account for the topological interactions of the DLO with rigid objects in the environment.

VI. TYING KNOTS AROUND STATIC RIGID OBJECTS

So far, we have focussed on achieving topological states of a DLO defined with respect to itself, i.e., tying self-knots. But in many applications, DLOs also knot around static objects. Here, we provide a simple technique to account for the topological interaction of a DLO with static objects for which curve representations exist. We say that for a given object o, its curve representation exists if there exists a curve $r(o)$ such that any point in o is within δ distance from $r(o)$, where δ is small compared to the length of the curve. Examples of such objects are torus, cylinders, and long bars. In the case of a torus, the curve swept by the center of its generator conic and in the case of cylinder its central axis could be chosen as their curve representations. In general, curve representations can be obtained by skeletonizing the objects by the methods presented in [4] and pruning less prominent branches of the skeletal tree to obtain a curve.

We will account for the interaction of the DLO with rigid objects by defining a *composite* semi-deformable linear object (sDLO), and work with the crossing configuration of the sDLO thereafter. Let R be the set of curve representations of the concerned static objects. sDLO is created by "chaining" the curves in R and c sequentially, c being the last (see Figure 13). There will be virtual links connecting consecutive curves in the sequence. We will ignore the crossings between the curves in R, and also any associated with the virtual links.

Here we choose a reference projection plane such that the curves in R are minimally distorted after projection. This reduces the possibilities of crossings, between the projections of a curve in R 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 R

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 our proposed DLO manipulation planner in C++ and ran knot-tying experiments on a 1.5GHz Intel Xeon PC with 1GB RAM. We used the physical model described in [23] to account for the physics of the DLO. This physical model 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 and rigid objects in the environment. Two robot arms, each with 6 degrees of freedom and capable of providing point grasps, were used for the manipulation. The planner took 10 to 15 minutes of CPU time to generate knot tying motions for the dual arms.

Figure 15 shows sequences of snapshots from the manipulation motion generated by our planner for the five manipulation problems that we considered. The first four sequences tie common knots: bowline, neck-tie, bow (shoe-lace), and stun-sail, respectively. The last sequence corresponds to a typical manipulation problem of winding the DLO around static objects.

Bowline and bow required one tri-needle, and neck-tie and stun-sail required two tri-needles each. Bow needed an additional mono-needle to maintain its slip loop. The last problem did not require any needle.

Our real-life experimental set-up consists of two PUMA-560 robots at the Stanford Artificial Intelligence Lab. We tuned the parameters of the rope model such that the simulated rope "visually" behaved like a typical common-life rope, as one of its ends was being manipulated and the other end was kept fixed. Then we let our planner generate a manipulation plan for tying a bowline knot. Thin aluminium rods were used to simulate the needles and hand-placed, prior to the execution of the manipulation plan, at the positions determined by the planner. Using the generated plan, the two PUMA robots were able to co-operatively tie a bowline knot with a real rope (Fig. 15(i)), successfully, even though the tuned rope model did not exactly represent the physics of the real rope. The robustness of our planner to inaccuracies in the physical model of the rope was further ascertained when the robots were able to tie bowline knots with the same plan, but using four other ropes of different nature, i.e. ropes with different materials and thicknesses. However, the same plan failed to tie the knot for a plastic rope, because it was too stiff and the plan was originally generated for significantly less stiff ropes. Fig. 15(ii) lists the materials and thicknesses of the ropes used. Fig. 15(iii) shows the final bowline shape attained by the ropes.

The ability of our planner to generate robust plans minimizes the need for online sensing and re-planning of robot motions. However, online sensing and re-planning could be required if the model and the actual physics of the rope are significantly out of tune.

We have also been interested in quantifying the robustness of generated plans to inaccuracies in the rope model. But it is not feasible to do so from real experiments since rope manufacturers do not provide numerical values for the mechanical properties of ropes. So, we quantified the robustness of generated plans from computer simulations 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. The table in Fig. 15(iv) lists the means and standard deviations (estimated numerically) for the Gaussian distributions from which 3 main parameters of the model were independently chosen, such that a bowline was achieved more than 90% of the time. The mean values correspond to the original model parameter values, used to generate the manipulation plan. The standard deviations and the corresponding mean values have comparable values, indicating a high degree of robustness.

The videos of the results are available at: http://ai.stanford.edu/∼mitul/dlo

VIII. Conclusion

In this paper we have proposed a topological motion planner for manipulating deformable linear objects (DLOs) using cooperating dual robot arms to tie self-knots and knots around simple static objects. The planner does not assume a specific physical model of the DLO. The user has the flexibility of providing an appropriate model (e.g., rope, suture, strand etc.) depending upon the application. To our knowledge, our planner is a first of its kind, i.e., we are not aware of any other planner which can generate collision-free motions for robot arms which lead to DLO manipulation in environments with obstacles.

We have demonstrated the effectiveness of our planner by tying some commonly used knots. Currently we are analyzing the probabilistic completeness our planner, i.e., if it can tie any type of (semi-)tight knot given unbounded time and computational resources.

From this point, there are many interesting research directions to head for. For example, one could consider the topological metrics used in [16] for biasing our topological planner. Since it is difficult to choose a good reference projection plane when there are several static objects or when an object is not quasi-planer, one could try to extend our method to tie knots around objects with the help of multiple planes. Input objects could be broken into a number of quasi-planar objects and a separate plane could be used for each of them. Also, the new concepts of forming sequence and loop structure could seed future research in knot theory and its application domains.

Finally we believe that our contribution in this paper could potentially lead to opening of crucial application domains for robotics in future, 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.

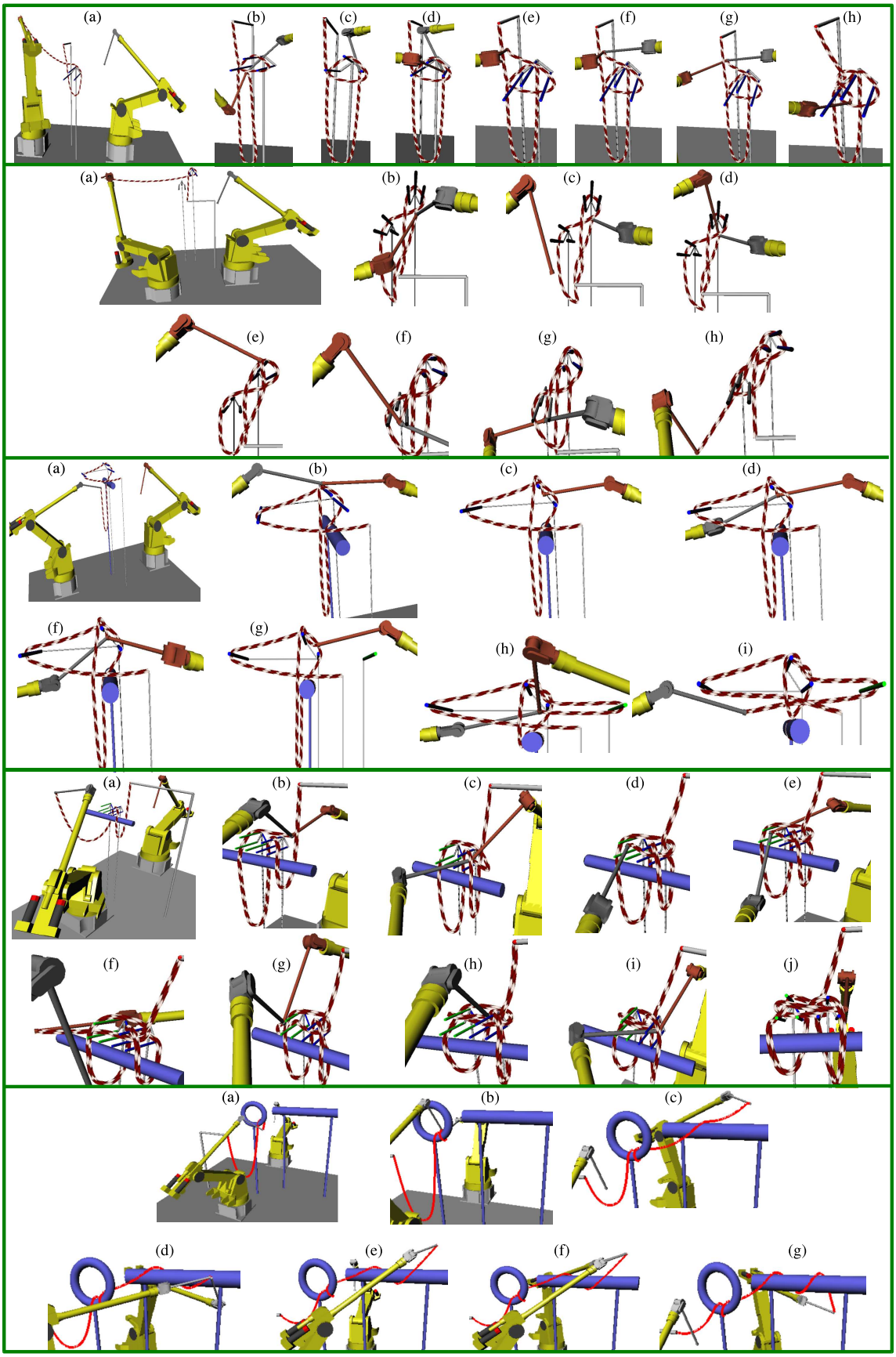
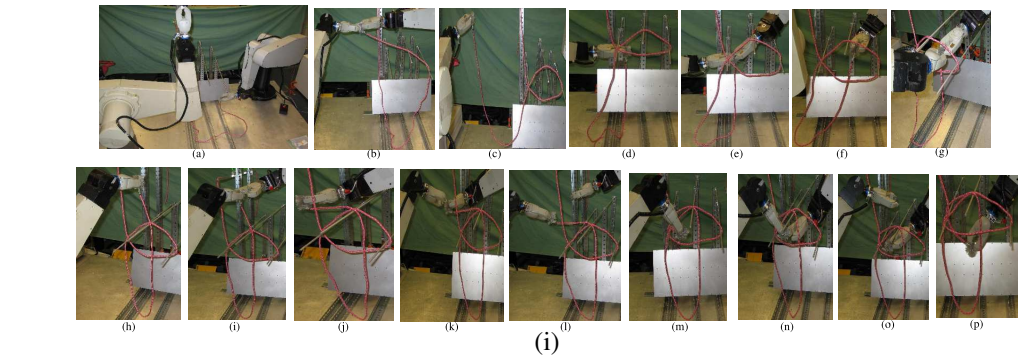


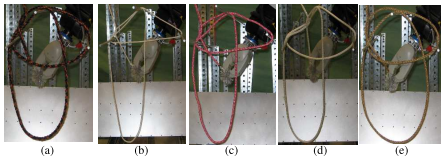
Fig. 14. Sequences of snapshots generated by our planner for five manipulation problems.



(i)

	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

(ii)



(iii)

	mean	standard deviation
Parameter 1	13500	3950
Parameter 2	1250	455
Parameter 2	25	15

(iv)

Fig. 15. (i): Snapshots of two PUMAs in the process of tying a bowline knot, (ii): Different types of ropes used for tying knots. (iii): Final shapes of bowline knots achieved with different ropes. (iv): Means and standard deviations of Gaussian distributions from which the three main rope model parameters were chosen for the robustness analysis.

IX. Acknowledgement

This work has been partially funded by NSF grant ACI-02-5671. We are immensely thankful to Prof. Oussama Khatib for allowing us to use the PUMA 560 robots in his Lab. We are also thankful to Jaeheung Park, Jin Sung Kwon, and Ankur Dhanik for helping with the experimental setup.

References

[1] C. C. Adams, *The Knot Book*, W.H. Freeman and Company, New York, NY, 1994.

[2] J. Brown, J.C. Latombe, and K. Montgomery, "Real-time knot tying simulation", *The Visual Computer J.*, 20(2-3):165-179, 2004.

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

[4] N. Gagvani and D. Silver, "Parametric controlled volume thinning", *Graphics Models and Image Processing*, 61(3):149-164, May 1999.

[5] A. Dhanik, "Development of a palpable virtual nylon thread and handling of bifurcations", Masters Thesis, NUS, Singapore, 2005.

[6] D. Hsu, R. Kindel, J.C. Latombe, and S. Rock, "Randomized kinodynamic motion planning with moving obstacles", *Int. J. Robotics Research*, 21(3):233-255, 2002.

[7] A. M. Ladd, and L. E. Kavraki, "Using motion planning for knot untangling", *Int. J. Robotics Research*, 23(7-8):797-808, 2004.

[8] F. Lamiraux and L. E. Kavraki, "Planning paths for elastic objects under manipulation constraints", *Int. J. Robotics Research*, 20(3):188-208, 2001.

[9] S. M. LaValle and J. J. Kuffner, "Randomized kinodynamic planning", *Int. J. Robotics Research*, 20(5):378-400, May 2001.

[10] Y. Koga and J. C. Latombe, "On multi-arm manipulation planning", *Proc. IEEE Int. Conf. Robotics and Automation*, San Diego, CA, 1994.

[11] T. M., T. Fukuda, and F. Arai, "Flexible rope manipulation by dual manipulator system using vision sensor", *Proc. IEEE/ASME Int. Conf. Advanced Intelligent Mechatronics*, Como, Italy, 2001.

[12] T. Morita, J. Takamatsu, K. Ogawara, H. Kimura, and K. Ikeuchi, "Knot planning from observation", *Proc. IEEE Int. Conf. Robotics and Automation*, Taipei, Taiwan, 2003.

[13] D. K. Pai, "Strands: interactive simulation of thin solids using cosserat models", *Proc. Eurographics*, Saarbruken, Germany, 2002.

[14] J. Phillips, A. Ladd, L. E. Kavraki, "Simulated knot tying", *Proc. IEEE Int. Conf. Robotics and Automation*, Washington, DC, 2002.

[15] A. Remde, D. Henrich, and H. Worn, "Pick-up deformable linear objects with industrial robots", *Proc. Int. Symp. on Robotics*, Japan, 1999.

[16] P. Rogen and H. Bohr, "A new family of global protein shape descriptors", *Mathematical Biosciences*, 182(2):167-181, April 2003.

[17] G. Sanchez and J. C. Latombe, "A single-query bi-directional probabilistic roadmap planner with lazy collision checking",. *Proc. Int. Symp. on Robotics Research*, Australia, 2001.

[18] T. W. Schmidt and D. Henrich, "Manipulating deformable linear objects: robot motions in single and multiple contact points", *Proc. Int. Symp. on Assembly and Task Planning*, Japan, 2001.

[19] T. Simeon, J. P. Laumond, J. Cortes, and A. Sahbani, "Manipulation planning with probabilistic roadmaps", *Int. J. Robotics Research*, 23(7-8):729-746, 2004.

[20] H. Wakamatsu and S. Hirai, "Static modeling of linear object deformable based on differential geometry", *Int. J. Robotics Research*, 23(3):293-311, 2004.

[21] H. Wakamatsu, A. Tsumaya, Eiji Arai and Shinichi Hirai, "Planning of one-handed knotting/raveling manipulation of linear objects", *Proc. IEEE Int. Conf. Robotics and Automation*, LA, 2004.

[22] G. Wilfong, "Motion planning in the presence of movable obstacles", *Proc. ACM Sym. on Computational Geometry*, Urbana-Champaign, IL, 1988.

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