Algorithmic Foundations of Robotics V, Springer Tracks in Advanced Robotics, Springer Verlag, STAR 7, p. 7–23 2004

Using Motion Planning for Knot Untangling

Andrew M. Ladd and Lydia E. Kavraki
Department of Computer Science,
Rice University, Houston, TX 77005, USA

Abstract

This paper investigates the application of motion planning techniques to the untangling of mathematical knots. Knot untangling can be viewed as a high-dimensional planning problem in reparameterizable configuration spaces. In the past, simulated annealing and other energy minimization methods have been used to find knot untangling paths. We developed a probabilistic planner that is capable of untangling knots described by over four hundred variables. We tested on known difficult benchmarks in this area and untangled them more quickly than has been achieved with minimization in the literature. In this work, the use of motion planning techniques was critical for the untangling. Our planner defines local goals and makes combined use of energy minimization and randomized tree-based planning. We also show how to produce candidates with minimal number of segments for a given knot. The planner developed in this work is novel in that studies issues arising in practical motion planning for high dimensional and reparametrizable geometry. The use of energy methods, local goals and tree-based expansion is also novel and may suggest solutions in other planning applications. Finally, we discuss some possible applications of our untangling planner in computational topology, in the study of DNA rings and protein folding and for planning with flexible robots.

1 Introduction

Some of the current challenges at the frontiers of modern motion planning applications stem from high dimensional or even non-parametric con-

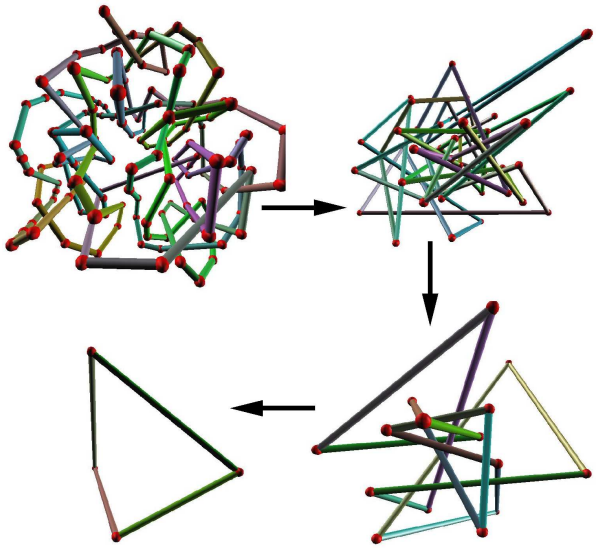

Figure 1: Knot untangling of a tangled 133 segment unknot.

figuration spaces, complex but 'soft' constraints on allowable motions imposed by the energetics of an underlying physical system and the possibility of innovative uses of planners for discovery. Recent problems of interest are daunting but may have a great deal of structure that can be exploited to sidestep the inherent complexity of high dimensional planning in constrained spaces. Motivating examples can be found in recent work on planning schemes for modular and reconfigurable robots [BKRT01, VSY02, WTA02], planning for deformable robots [LK01, BLA02], abstractions of random planning analysis to non-parametric, non-manifold spaces [LK02], approximate dimensionality reduction of configuration space [TPK02], determining protein folding pathways [SA01], planning in the energy landscapes of molecules [ASBL01], and computing stationary distributions of Markov chains defined over roadmaps for analysis of Monte Carlo simulations of protein folding phenomena [ABG+02].

This paper deals with the knot untangling problem. A knot is a simple[1], closed piecewise linear curve and is called an unknot if there exists an am-

[1]non self-intersecting

bient isotopy[2] transforming the knot to a triangle. Otherwise the knot is a non-trivial [Lic97]. The induced equivalence relation separate the knots into different classes that are called knot types. Our approach applies techniques from motion planning to find a sequence of motions and transformations that reduces the number of segments in a knot to the minimal number possible for that knot type. During these motions and transformations the topological type of the knot is preserved. When cast as a robotics problem, untangling is about manipulation of a high-dimensional, reparametrizable, deformable object possibly requiring many parameters to describe important configurations. Figure 1 illustrates some intermediate steps in the untangling of a tricky 133 segment knot.

We describe the design and implementation of a novel motion planner for knot untangling. We take the novel approach of decomposing the global untangling into sequence of local goals and reparametrization operations that, when taken together, determine a solution. Planning towards the local goals is achieved by combining energy minimization and randomized tree-based planning in the spirit of those employed by the RRT framework [KL99, KL00, KL01b, KL01a] and expansive spaces framework [HLM97, Hsu00, HKLR01, SL01]. Our approach is minimalistic and we show that a simple planner suffices for solving this problem. We demonstrate the effectiveness of this planner by applying it to some standard benchmarks used in other implementations, some random knots and by finding minimal segment candidate for a sixteen crossing knot. Our experimental results are significantly better than comparable implementations in the literature. We also discuss how the techniques employed might be applied to other high dimensional motion planning problems such as those found in robotics approaches to molecular biology or in problems with deformable robots.

2 Background

This section is summary of some of literature specific to knot untangling. We begin by describing some the efforts in constructing exact algorithms and complexity results for knot classification. We continue by describing energy functions for knots and previous work on heuristic untangling. The section ends with a history of knotting in biology. This develops the relation of knot untangling by computer to applications in bioinformatics.

[2]Piecewise linear homeomorphism of the space around the knot.

2.1 Exact Classification

Classification of knots is often attempted by the computation of knot invariants [Lic97, Ane99, HT98]. Invariants are typically computable constants or polynomials that are invariant over all embeddings of a knot type. If two knots differ over some particular invariant then they are different knots, however the converse does not necessarily hold.

We first describe two classic problems in computational knot theory [BEa99].

Problem 2.1 (UNKNOT). *Given a knot K, determine if K is the same knot type as the unknot.*

Problem 2.2 (KNOT EQUIVALENCE). *Given two knots K_1 and K_2, determine if K_1 is the same type as K_2.*

One possible proof that a given knot is the unknot is to exhibit an explicit isotopy transforming it to the triangle, in other words, to find a path transforming the given knot into a triangle. In the remainder of the background section, we will discuss in more detail some of the previous work on the untangling problem.

There are many interesting invariant quantities defined for knots. The crossing number is the minimum number of crossings over all regular drawings of the knot on the plane. The unknotting number of a knot is the minimum number of the crossing flips required to transform the knot to the unknot. The stick number is the minimum number of segments required to draw a given knot in 3-space. The genus is the minimum genus over all surfaces that the knot can be drawn on. For a knot K, we write $c(K)$ for crossing number, $u(K)$ for unknotting number, $s(K)$ for stick number and $g(K)$ for the genus. An unknot K has $c(K) = u(K) = g(K) = 0$ and $s(K) = 3$. The simplest non-trivial knot, the trefoil, has $c(K) = 3$, $u(K) = g(K) = 1$ and $s(K) = 6$.

The UNKNOT problem is known to be in **NP** and KNOT EQUIVALENCE is in **PSPACE** [HLP97]. More recently the MAXIMUM GENUS problem, $g(K) \leq k$, has been shown to be **NP**-complete [AHT]. Another recent unknot recognition algorithm works by enumeration of unknot diagrams and is polynomial in some cases [BBRV00]. Hass notes in [AHT] that if MINIMUM GENUS, $g(K) \geq k$, is in **NP** then UNKNOT is in both **NP** and $\mathbf{co-NP}$.

There are several polynomial invariants that are computed on knots. Each has various distinguishing powers. The Alexander-Conway invariant can be

computed in **PTIME** but has limited distinguishing power [Lic97]. Computing the Jones, Kauffmann and HOMFLY polynomials are #**P**-hard [JVW90]. These polynomials can be computed in practice for small knots although is not known if there are multiple knots with the same polynomial as the unknot. Jenkins gives an efficient implementation of a HOMFLY calculator, although the size of knots it can handle is limited by bit arithmetic in the code. If c is the number of crossings in a link, then the HOMFLY polynomial can be computed in time $O(n!2^n c^3)$ and in space $O(n!c^2)$, where n is bounded above by $\sqrt{c}+1$ [Jen89]. By using combinations of invariants [Ane99], attempts to tabulate all distinct knots have been made. A recent list consists of all prime knots with at most 16 crossings and consists of 1701936 distinct knot types [HT98].

A key tool used in polynomial invariant calculation is a regular plane projection of a knot or knot diagram. These graph-like objects can be manipulated by three simple moves, called the Reidemeister moves, to obtain all knot diagrams for that knot type [Lic97]. Hass and Lagarias derive an upper bound on the number of Reidemeister moves to transform a tangled unknot into a triangle. If k is the number of crossings in a given unknot diagram, it takes at most $O(2^{ck})$ where $c = 10^{11}$ [HL] to transform this diagram to an unknot. Some intuition for the difficulty here comes from the fact that for some diagrams a non-minimizing Reidemeister move must be made, that is to say a crossing must be added before others can be removed.

2.2 Energy Functions

Knot energy was originally defined as a way of arriving at "ideal" knot configurations. Existing untanglers are essentially energy minimizers so this issue is of critical importance. A good survey of this area can be found in Scharein's thesis [Sch88].

There are several different varieties of energy functions in the literature. Many of the early equations were for the case where the function was a smooth curve and hence the energy was infinite for piecewise linear knots. For further details on energy for smooth curves refer to [DD00]. Due to the computational nature of our work, we restrict ourselves to energy defined over piecewise linear knots and give only two important examples from the literature. Other examples of energy functions can be found in [LS94, DER97, GHK97, Sch88] and include geometric energy, Möbius energy, spring energy, electrostatic energy and others. In our work, we used a modified version of

minimum distance energy.

Minimum Distance Energy [Sim94] Minimum distance (MD) energy is an approximation of the distance integral over all pairs of points. The approximation is for the case of a piecewise linear knot and uses the minimum distance between pairs of segments (d_{MD}). Given a knot K with segments $s_1, ..., s_n$, we define MD energy of the knot as follows

$$E_{MD}(K) = \sum_{s_i, s_j \text{ disjoint}} \frac{|s_i||s_j|}{d_{MD}(s_i, s_j)^2}.$$

This energy function can be computed in $O(n^2)$ time. The energy is bounded from below by $2\pi \cdot c(K)$ [Sim94]. To our knowledge, no convergence guarantees have been proved for a gradient descent energy minimization and experimental evidence suggests the presence of local minima, large plateaus and nearly non-unimodal behaviour [Wu96, GHK97].

A key problem with MD energy is that there is a tendency for tangled portions of the knot to become small. This is a result of the length normalizer term in the equation. In particular, the global minimum for non-trivial knots do not conform well to intuitive notions of "ideal" configurations. It has been suggested that this causes convergence problems during energy minimization [LS94].

Symmetric Energy [Sch88] Another approach is to view the knot as a radiating tube of small radius. Symmetric energy measures the amount of self-illumination of the knot. This energy can be calculated for a knot K parametrized by $x(t)$, $y(t)$,

$$E_S(K) = \int \int \frac{|dx \times r||dy \times r|}{|x - y|^2},$$

where $dx = \dot{x}(t)dt$ and $r = (x - y)/|x - y|$. This energy function does not shrink tangled portions of the knot and has been used to make beautiful drawings of knots with the software KnotPlot [Sch88].

2.3 Previous Work on Untangling Knots

Given an energy function defined over the space of knots, various techniques have been explored to find its global minimum. The objective is to optimize

the rate of convergence to the global minimum or at least to a good local minimum. In knotting, the methods employed have been random perturbation, simulated annealing, gradient descent, and the method of stochastic energy functions.

Random perturbation has been used by Simon's untangler [Sim94], KED and KnotPlot [Sch88]. Random perturbation methods are slowed by convergence problems and many self-intersection checks.

The annealing schedule for knots proposed by Ligocki and Sethian generates new configurations with a standard deviation proportional to an inverse logarithmic function of time [LS94]. The high-dimensionality of the problem seems to necessitate slow cooling and this method suffers from convergence problems and many self-intersection checks.

An important advantage of simulated annealing is that, under reasonable assumptions, it is guaranteed to find the global minimum. Unfortunately, there seems to be a very poor rate of convergence to the global minimum for several reasons: rejection of self-intersecting perturbations, minima effects in the energy function and the high-dimensionality of the space necessitates slow cooling. An additional concern is that a great deal of computation time is spent on collision detection in this method.

An important criticism of perturbation and annealing methods is that much of the time is spent checking for self-intersection. It is difficult to avoid making $\frac{n^2-3n}{2}$ intersection checks when the vertices are moved and usually many perturbations are required to escape a local minimum or difficult regions. Experimental evidence suggests that there are many such regions.

Another standard energy minimization approach is to use gradient descent which will quickly find a local minimum. Few self-intersection checks are needed assuming an energy function that goes to infinity as the knot approaches a singular embedding. Since most energy functions go to infinity quickly as the knot approaches self-intersection little in the way of intersection detection is needed and effort wasted choosing bad pertubations is avoided as well. The difficulty is that convergence to a local minima is the only guarantee that such an algorithm can provide. To escape a local minimum, heuristic perturbation methods can be used after convergence occurs. This technique has been successfully implemented in the MING knot evolver [Wu96]. This evolver uses MD energy and was used to find a 22 edge unknot which was a MD energy local minimum as well as other difficulties with MD energy. Although effective, this method is fairly slow.

The stochastic energy untangler [GHK96, GHK97] uses spring energy,

electrostatic energy, randomly positioned point charges and reparametrization of the knot to create a random family of energy functions. The magnitude of the perturbation is determined by an annealing schedule. These energy functions are then minimized by gradient descent. The authors report their method to be roughly twice as fast as MING on a difficult benchmark.

We would like to point out the relation between efforts in knot energy minimization and potential field efforts in motion planning. Many of the same difficulties arise. In motion planning, potential field methods have largely been replaced by global probabilistic planners [KSLO96, HLM97, HKLR01, Hsu00, KL99, KL00, KL01b, KL01a, LK01, BLA02]. Scharein, in his thesis, remarks that by manually dragging vertices around, minimizations of some very tangled knots could be achieved considerably more quickly than without intervention [Sch88].

2.4 Knots in Nature

The physical results for knotting come in to play in our work in two ways. The behaviour and importance of knots in nature was significant in the design of energy techniques. It motivates the design of physical algorithms to solve such problems. Also, in structural bioinformatics problems, particularily those involving DNA, some robotics techniques are employed. Our method may provide some insight in solving such problems. For example, by adding crossing operations to the planner, a knot untangler could be used for sampling unknots far faster than the approach of rejection sampling from a uniform distribution. The algorithm of randomly deforming an unknot until it is complicated is also extremely slow and produces knots that can be instantly untangled.

The definition of energy functions on knots has had several purposes: to guide the simplification of tangled embeddings, to find aesthetic, symmetric drawings, to search for ideal energy minimal configurations and to explore connections with the energetics of knotting phenomena in DNA.

This final purpose is motivated by the abundance of long ring-shaped molecules in living organisms. The discovery of knotted DNA led to the application of knot theory to the study of long rings. Since then a number of DNA knots have been characterized in detail, and their knot-types have provided insights into the mechanisms of the reactions that produced them.

It is known that the probability that a random knot on a square lattice is the unknot goes to 1 exponentially with the length of the knot [SW88, Pip89].

Further work in this direction attempts to asymptotic knotting and linking of simple curves moving at random in some space under various models. This general body of work provided further theoretical evidence that knotting phenomena occur in long DNA rings. This fact has been validated experimentally, for example Shaw and Wang have measured knotting probability in DNA molecules explicitly as a function of concentration [SW93, SW94].

Recent studies have shown the formation of knots and links in DNA can play an important role in cell replication. Knotted and linked conformations are spatially more compact. However, during replication DNA must untangle. There is some insight into the mechanism whereby this occurs and it may be that knot energetics in DNA rings play an important role in the life cycle of a cell [PPC99, DRZ01].

In [SW93, SW94] Shaw and Wang determined the experimental probability of knotting of DNA molecules of 5.6 kbp and 8.7 kbp in length as a function of the concentration of $NaCl$ and $MgCl_2$ in the reaction. The measured probabilities ranged from close to 0 to roughly 0.08. They also investigated the energetic costs of knotting such chains and observed that the trefoil was disfavoured. They report that the disparity seems to be worse for supercoiled DNA. They conclude by suggesting that the free energy cost in enzyme-catalyzed reactions that lead to knotted products must be compensated for by a favorable term such as that derived from protein-DNA interactions.

Gel electrophoresis has been used to separate some knots and links by crossing number and sometimes by knot type [CKG$^+$94, SKB$^+$96]. Gel velocity in agarose gel is determined by its average tertiary structure as it passes through the field imposed by the gel. Knotted DNA travels faster than unknotted DNA because a knot tends to be more compact. Crossing number can be correlated with gel velocity and in some cases knot type can be further distinguished. This has been realized experimentally for small knots.

There is some interest in the calculation of various knot invariants for application to biological problems. For a small selection see [KM94, BM00, Dar01].

3 Geometry of Knots

A given knot K with n pieces is defined by a sequence of vertices $p_1, ..., p_n$. For convenience, we sometimes say p_{n+1} to mean p_1 and p_0 to mean p_n. Formally the knot consists of the subset of $\mathbb{R}^3$ defined by the union of the segments $s_i = \overline{p_i p_{i+1}}$, for $1 \leq i \leq n$. We say two segments belonging to a knot are disjoint when they do not share an endpoint.

The clearance of a knot K with vertices $p_1, ..., p_n$ is the function

$$clr(K) = \min_{s_i,\ s_j \text{ disjoint}} \{d_{MD}(s_i, s_j)\}.$$

d_{MD} is the minimum distance between a pair of line segments in $\mathbb{R}^3$. This expression measures the distance between disjoint segments. When $clr(K) = 0$, the knot K is called singular. When $clr(K) > \epsilon$ for some $\epsilon > 0$, K is ϵ-non-singular. The self-intersection equations we use are not numerically stable for small values of $clr(K)$ when evaluated with floating point precision. Also, it is assumed that all knots we consider are non-singular.

Given two knots of n pieces, K_a with vertices $p_1, ..., p_n$ and K_b with vertices $q_1, ..., q_n$, we can define a linear interpolation between K_a and K_b parametrized by time t, $K(t)$, that is the knot defined by vertices $r_i(t) = p_i + t \cdot (q_i - p_i)$. This interpolation preserves the topology of the knots in question when the knots $K(t)$ are non-singular for all $t \in [0, 1]$. In particular, existence of such an interpolation between K_a and K_b is a sufficient condition for concluding topological equivalence of K_a and K_b as it implies an explicit ambient isotopy. Detecting the existence of zeroes for $clr(K(t))$ for $t \in [0, 1]$ is the self-intersection detection problem for knot manipulation.

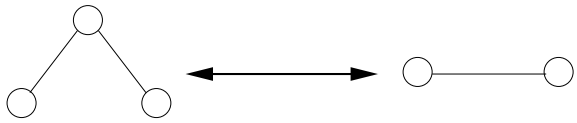

Figure 2: An elementary move.

Adding and removing vertices can be accomplished via a so-called elementary or triangle move, illustrated in Figure 2. A vertex p_i can be removed from knot K with vertices $p_1, ..., p_n$ to obtain knot K' with vertices $p_1, ..., p_{i-1}, p_{i+1}, ..., p_n$ if the triangle $\Delta p_{i-1} p_i p_{i+1}$ is unpunctured by all segments disjoint from s_{i-1} and s_i. Alternately, when three vertices are collinear,

the middle vertex can be removed. This move and its inverse are the elementary moves and any two equivalent knots can be related by a sequence of elementary moves. The best known upper bound on the number of moves required to transform a given unknot by elementary moves to a triangle is $O(2^{ct})$, where c is 10^7 and t is the number of tetrahedrons in a space triangulation where the knot can be drawn on the 1-skeleton [HL]. The author knows of no non-trivial lower bounds in the literature.

4 Untangling Planner

The untangling planner works to reduce the number of vertices in a given knot. In this section, we will describe the design and implementation of our untangling planner. Essentially the core of the planner is a tree-based planner similar to an RRT [KL99, KL00, KL01b, KL01a] or expansive space planner [HLM97, Hsu00, HKLR01, SL01]. The top-level of the method runs the tree-based planner repeatedly until the knot is reduced. The tree-based planner expands the input node to search around the space locally. If any node is produced where an elementary move can be made then the elementary move is made and the process repeats. If after some number of iterations no progress is made then energy methods are used to make progress.

Although similar in spirit to an RRT or expansive space planner, the tree-based expansion we use is geared towards simplicity. The nearest neighbour queries are replaced with a heuristic weighting scheme. Although on longer runs our planner converges to the uniform distribution more slowly, our approach calls the planner often and generates a small number of nearby nodes. Since the dimension of the space is quite high on the interesting benchmarks, we expected the asymptotic advantages of more systematic schemes like RRT or expansive spaces would not have had sufficiently significant impact to make it worth building the necessary geometric data structures. Our experiments in this work and with some toy problems have partially validated this assumption.

Rather than planning a path towards a particular configuration such as the unknot, the planner is used to find a configuration wherein a single vertex can be removed via an elementary move. When no progress is made, the energy of the knot is optimized instead. If no optimization is found, a random move is made instead. Other important design decisions involved the sampling, energy and weighting functions used in the planner and the

choice to avoid configurations where the clearance is very low and that are prone to numerical instability in self-intersection detection.

Configuration Space Let $\mathcal{K}_n$ be the set of all non-singular knots K with n vertices and let $\mathcal{K} = \bigcup_{n \geq 3} \mathcal{K}_n$. This space is the configuration space for the planner. In contrast, to many planning problems, configurations will change over the course of the plan as vertices are removed.

Let K be a knot with n vertices $p_1, ..., p_n$ and let K' be a knot with vertices $q_1, ..., q_n$. We say that $K \sim K'$ if the linear interpolation between them has no intermediate configuration that is singular. In particular, this fact proves that K and K' are the same topological knot.

Given a knot K with vertices $p_1, ..., p_n$ and $\epsilon > 0$ such that $clr(K) \geq \epsilon$, we define two local planners.

Linear Interpolation Planner The first local planner moves a single, given vertex as close as possible towards a target point. Given $1 \leq i \leq n$ and a point $q \in \mathbb{R}^3$, we define a parametric knot $K(t)$ with vertices $p_1, ..., p_{i-1}, p_i + t \cdot (q - p_i), p_{i+1}, ..., p_n$. We find all the zeroes of the intersection equation applied to all pairs of disjoint segments in $K(t)$ where one segment contains the ith vertex. Suppose the zeroes occur at $0 < t_1 < \cdots < t_m < 1$. Beginning at the first zero, they are processed in order. The jth zero, t_j, will occur between some segment s with fixed endpoints and a segment $s'(t)$ which has p_i as endpoint. We calculate $d_{MD}(s, s'(t_j))$ and verify that it is larger than ϵ. If not, we find the smallest integer $r > 0$ such that $clr(K(\frac{t_j}{2^r})) > \epsilon$ and set $t^* = \frac{t_j}{2^r}$. If after processing all the zeroes, t^* is undefined then we calculate the smallest integer $r \geq 0$ such that $clr(K(2^{-r})) > \epsilon$ and set $t^* = 2^{-r}$. This procedure allows us to find a knot $K' = K(t^*)$ where $K \sim K'$ and $clr(K') > \epsilon$. We summarize this construction as $K' = lip_i(K, q)$, lip is a mnemonic for linear interpolation planner. Since we are only moving a single vertex, this calculation can occur in linear time.

Elementary Move Planner The second local planner removes a vertex by an elementary move. Given $1 \leq i \leq n$, we take the midpoint q of p_{i-1} and p_{i+1} and compute t^* as with the previous section. If $t^* = 1$, then the move is successful and we can safely remove the ith vertex with an elementary move to obtain K' with vertices $p_1, ..., p_{i-1}, p_{i+1}, ..., p_n$. The binary function $elm_i(K)$ takes the value 1 if $t^* = 1$ and the value 0 otherwise.

Algorithm 1 Untangle knot K_0

1: Generate a permutation of $1, ..., n$ called $j_1, ..., j_n$.
2: **for** i ranges from 1 to n **do**
3: Check to see if $elm_{j_i}(K_0) = 1$.
4: If so return K_0 with the j_ith vertex removed.
5: **end for**
6: Create a weighted tree T with root K_0 having weight 1.
7: **while** some number of iterations N **do**
8: Choose a knot K at random from T using the weight distribution.
9: Choose a vertex p_i at random from K.
10: Choose a point q distributed uniformly in the spanning sphere of K.
11: Compute $K' = lip_i(K, q)$ and the associated t^*.
12: Divide the weight of K in T by 2.
13: Add K' to T with parent K and weight t^*.
14: Check if for any j, $elm_j(K') = 1$, if so return the path from K_0 to K' and K'' which is K' with the jth vertex removed.
15: **end while**
16: Computes the energies of all nodes of K.
17: If there is a node with energy less than K_0 return it and the path to it from K_0 in T.
18: Choose a knot K at random from T using the weight distribution.
19: Return the path from K_0 to K in T and K.

Vertex Freeing Algorithm 1 below is called with some knot K_0, the input knot, and, N, the number of iterations to run the main loop before aborting. N should be chosen to balance the amount of energy optimization and the aggressiveness of the planner in trying to free a vertex. Implicitly there is a minimum tolerated knot clearance as a parameter as well.

The algorithm is split into three sections. Lines 1-5 are the first section, lines 6-15 the second and lines 16-19 are the third section. Let n be the number of vertices in K_0. The first section exhaustively searches for a vertex that can be removed by an elementary move. The vertices are considered in a random order to eliminate bias. The first section runs in time $O(n^2)$: each call to *elm* costs $O(n)$ geometric comparisons and n calls are made.

The second section consists of a probabilistic tree expanding planner that searches for a move that frees a vertex. This section is more complex and requires certain data structures for a correct implementation. The tree is stored as a directed graph. The distribution of weights and associated probabilities for the tree nodes can be stored in binary heap-like structure which supports $O(\log N)$ weight insertion, weight modification and choose operation. Therefore lines 8, 13 and 14 cost $O(\log N)$. Lines 10 and 11 each require $O(n)$ operations as they require a single traversal of the knot. Line 14 can be implemented in $O(n)$ by maintaining a count of how many segments conflict with each triangle in the knot. The remaining lines in section two can be implemented in constant time. Thus the section can be implemented in time $O(Nn + N \log N)$. The space usage for this section is dominated by the storage for the tree nodes and can be seen to be $O(nN)$. By keeping only incremental updates and memoizing periodically the space can be reduced to closer to $O(N)$ without sacrificing much of the runtime.

Finally, the third section requires energy calculations over all nodes in the tree. We are assuming that the time to calculate the energy function is dominated by an operation on pairs of the disjoint edges. The energy of K_0 can be calculated in $O(n^2)$. Since each other node is constructed by an incremental change to one vertex, the energy can be calculated from the previous node in time $O(n)$ with appropriate book-keeping in the tree. Thus line 17 can be done in time $O(n^2 + nN)$ and this cost dominates section three.

Global Scheme The untangling algorithm is called repeatedly some number of times until no more vertices can be removed from the knot. Depending on the application in mind, it may be enough to simplify a certain amount

or it might be better to search aggressively for a minimal embedding.

Choosing an Energy Function The energy function that we used was based on MD energy but we dropped the length normalizer to obtain

$$E(K) = \sum_{s_i, s_j \text{ disjoint}} \frac{1}{d_{MD}(s_i, s_j)^2}.$$

The energy function we used is non-standard and was chosen for two reasons: it is easy to compute and it does not shrink tangled portions of the knot like MD energy does for certain tangled sections of the knot [LS94, GHK97]. This tendency is very undesirable and is one of the sources of convergence problems for MD energy. The problem of the knot becoming very large is dealt with by sampling in a sphere. We ran 50 trials on a large unknot with MD energy instead of unnormalized MD energy and very little progress was made. We found it surprising that the difference between the two functions was so striking because superficially they seem similar. When visualizing intermediate configurations during the untangling we observed that MD energy tended to shrink the tangled portions of the knot. The unnormalized MD energy tended to separate some large triangles or "ears" from the knot. The planner would have ample space to pass other parts of the knot through these ears. That particular move seemed to be critical to making progress in the untangling.

Intersection Testing The geometry of piecewise linear knots is defined in terms of the geometry of line segments. In this subsection, we briefly describe how we calculate the zeroes for the clearance function. The importance of this description is to ensure numerically robust and efficient computation of segment intersection. The speed and stability of the computation is greatly improved by moving one vertex at a time.

Consider a pair of non-parallel infinite lines in 3-space $l_a(s) = p + s \cdot \vec{u}$ and $l_b(s) = q + s \cdot \vec{v}$. The lines will intersect when they are coplanar, that is to say when they both lie on a plane with normal $\vec{n} = \vec{u} \times \vec{v}$. This solution occurs when

$$\vec{n} \cdot (q - p) = 0. \tag{1}$$

When both lines are moving along known polynomial trajectories, i.e., $p, q, \vec{u}, \vec{v}$ are functions of t, a time variable, equation (1) is a polynomial in t, the zeroes of which occur at the intersection points of l_a and l_b.

The same holds for segments. The only possible intersection points occur at the zeroes of equation (1). At each zero, it can be verified if the segments are indeed intersecting by computing the minimum distance between them. The equations we use for this calculation are detailed in the next section.

Suppose a pair of segments $\overline{p_1(t)p_2(t)}$ and $\overline{p_3(t)p_4(t)}$ are moving along polynomial trajectories where the degree of the ith trajectory is δ_i. The degree of the intersection polynomial from equation (1) is found by making the following simple observation. When

$$deg(\vec{n}) = \max\{\delta_1, \delta_2\} + \max\{\delta_3, \delta_4\}$$

and if $\vec{w}$ is a vector between any pair of points on the opposite segments

$$deg(\vec{w}) \geq \max\{\min\{\delta_1, \delta_2\} + \min\{\delta_3, \delta_4\}\}$$

implies that the degree of the intersection polynomial is the minimum over all choices for $\vec{w}$ of $\deg(\vec{n}) + \deg(\vec{w})$. It is important to know the degree of the resulting polynomial when solving for the zeroes of d_{MD}, a naïve calculation can result in numerical failure.

Calculating d_{MD} A line segment is drawn between two points p and q. Equivalently, a segment is described by a point p and vector $\vec{v}$ where $\vec{v} = q - p$. The minimum distance $d_{MD}(\overline{p_1p_2}, \overline{p_3p_4})$ between a pair of non-parallel line segments can be calculated by taking expressions over dot products of vectors between the points

$$d_{MD}(\overline{p_1p_2}, \overline{p_3p_4}) = d(p_1 + cl(\mu_a)\vec{u}, p_3 + cl(\mu_b)\vec{v}$$

where $\vec{u} = p_2 - p_1$, $\vec{v} = p_4 - p_3$,

$$\mu_a = \frac{d_{1343}d_{4321} - d_{1321}d_{4343}}{d_{2121}d_{4343} - d_{4321}^2},$$

$$\mu_b = \frac{d_{1343} + \mu_a d_{4321}}{d_{4343}},$$

$$cl(x) = \begin{cases} x < 0 & 0 \\ x > 1 & 1 \\ \text{otherwise} & x \end{cases},$$

and

$$d_{ijkl} = (p_i - p_j) \cdot (p_k - p_l).$$

5 Experiments and Results

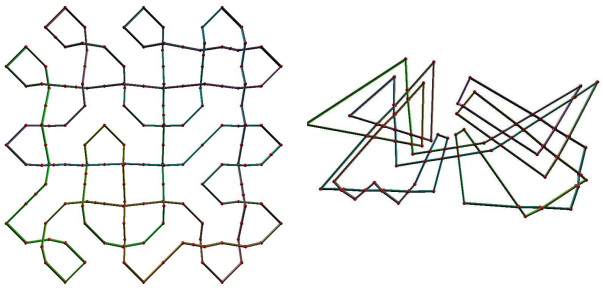

Figure 3: Ligocki-Sethian Example 2 and Twisted Freedman Unknot.

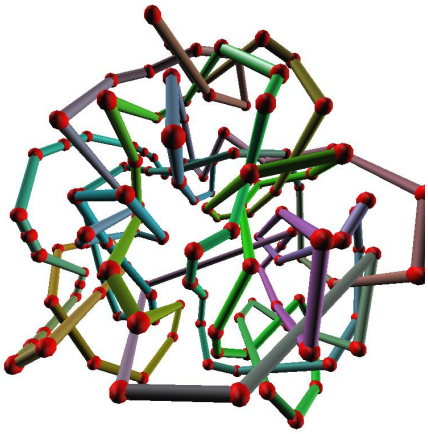

Figure 4: The Ochiai unknot.

This section describes experiments using our untangler. We will show that our planner is significantly faster than the earlier approaches in the literature. The software packages we are comparing against were the simulated annealer of Ligocki-Sethian [LS94], KnotPlot [Sch88], MING knot evolver [Wu96] and stochastic energy optimizer [GHK96, GHK97]. Each of these packages either report some results in a related paper or are publicly distributed.

Ligocki-Sethian Unknot The second Ligocki-Sethian Unknot shown in Figure 3 is a nearly flat weave with 160 vertices that was used as a test case

for a simulated annealer. It was reported that it took more than 16000000 iterations to obtain an embedding that was recognizably a circle[LS94]. Each iteration involved $O(n^2)$ collision checks for each configuration generated. Our untangler solves this problem nearly instantaneously.

Twisted Freedman Unknot The Twisted Freedman unknot is another nearly flat weave with 122 vertices that was used to show that MING evolver is faster and more precise than perturbation methods [Wu96]. It is shown in Figure 3. We ran MING for more than 12 hours on a 300MhZ SGI O2 without the knot untangling to a circular embedding. Our untangler ran on a single cpu of a dual AMD 1900MP and spent an average of approximately 8 minutes for the runs that completed in less than 25 minutes. About a third of the runs ran for more than 25 minutes and were terminated.

Ochiai unknot The Ochiai unknot is Figure 4 is an unknot with 139 vertices that has foiled many energy minimizers. It was originally given as an example of a general construction in a paper by M. Ochiai written in 1990. Since then it has often been used to test unknotting programs. It is reported in [GHK97] that MING running on an SGI O2 untangled it in 108 hours and stochastic minimization took 48 hours. KnotPlot was not able to reduce it [Sch88]. Our untangler averaged about 10 minutes on a single cpu of a dual AMD 1900MP and had roughly one sixth of its runs terminated after 35 minutes.

Random unknots We generated several random unknots in hopes of obtaining more interesting examples to test. We generated unknots by beginning with a circle of vertices and then making a large number of random moves. Even by running the tangler for many hours, we were not able to generate any configurations that were not solved instantaneously by the untangler.

Summary of unknot experiments We summarize the results of our untangling experiments in Figure 5. For each benchmark, we ran the planner several hundred times. Even after factoring in the difference in processor speeds, it is clear that our method is orders of magnitude faster than the energy minimizations in the literature. A second advantage is the probabilistic nature of the algorithm allows it to run many instances in parallel to obtain

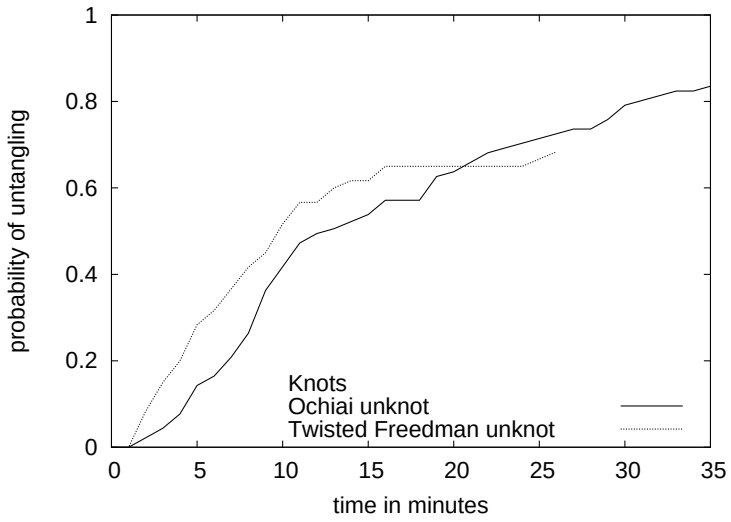

Figure 5: Probability of finding solution versus time in minutes

better times. In the case of the Twisted Freedman and Ochiai examples would mean solution times closer to two or three minutes.

Stick number of non-trivial knots The results we have reported up to now have been for unknots. Another application of the untangler is to minimize the number of segments used to draw a given knot. As an example we took the non-alternating knot 16_{148} shown in Figure 6 from the database of knots stored in the software Knotscape by Hoste and Thistlethwaite and found a 3D presentation of it with 17 sticks after a few minutes of computation. The initial embedding had 50 segments.

6 Discussion

The algorithm we gave engendered several design choices. In this section, we explain the motivations for these choices and discuss implications for other path planning problems. A principal point of interest is that the planner was successful on problems with several hundred degrees of freedom. The focus of the discussion is on isolating design choices that may be effective for high dimensional planning in general.

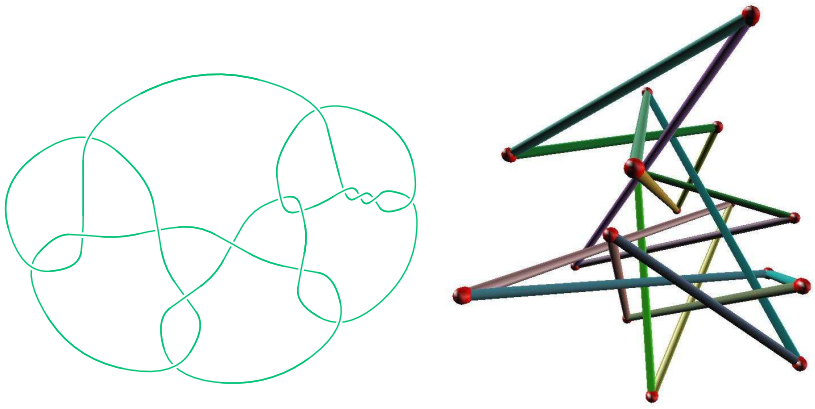

Figure 6: The knot diagram for 16_{148} rendered by Knotscape and 3D drawing with 17 sticks.

Use of trees Knot space has many properties that make it hostile to PRM planners that build a roadmap [KSLO96]. It is very difficult to sample uniformly from a single connected component. Determining whether two knots are in the same component is an instance of UNKNOT or KNOT EQUIVALENCE and is challenging for examples with many pieces. Even finding two random knots K, K' such that $K \sim K'$ for 30 segment knots requires the generation of several million random knots. It is also difficult to find fast algorithmic solutions to knot matching that take into account the symmetries of knots. For these reasons, we use a tree expanding planner in the spirit of other such planners [HLM97, HKLR01, Hsu00, KL99, KL00, KL01b, KL01a, SL01] to avoid the difficulties with global sampling.

Tree expansion In tree-based probabilistic planning, it has been observed that it is undesirable for the tree to be concentrated in a small area of the space. The planner will be much faster if it covers the space more quickly and other tree-based planners have employed a number of heuristics for this reason [HLM97, HKLR01, Hsu00, KL99, KL00, KL01b, KL01a, SL01]. Our planner's expansion scheme is simple but seems to produce similar distributions initially until the more sophisticated heuristics of other methods begin to kick in. Since our planner was a hybrid with an energy minimizer, we found it was more efficient to prune the tree fairly aggressively by energy than to rely on long term space covering heuristics to explore a very high

dimensional space.

When moving all vertices simultaneously along some linear path, the knot is constrained by the most constrained vertex. When moving a single vertex at random n times, we observed that the final position of the knot was typically much further from the start position. Since these two schemes are roughly the same amount of work computationally, moving one vertex at a time has a clear advantage. A further benefit is that the intersection equations have lower degree.

Freeing a vertex Finding a path that takes a given tangled unknot to a triangle requires finding a very long path. Worse, when trying to simplify a knot of unknown type, the final configuration may not be known. One possibility for finding a path to an untangled configuration is to construct a very large search tree. This construction is impractical and is very space intensive. In addition, the planner is also very slow if backtracking after performing an elementary move is possible, even for completely trivial input such as n vertices arranged in a planar regular polygon. This problem occurs because there can be at most $n!$ ways to reduce the knot to a triangle. We disallow backtrack, essentially throwing away the tree but for a path ending in an elementary move, we speed the planner greatly at the cost of having some probability of putting the knot in a bad configuration. We believe that such bad configurations are responsible for the occasional long run times observed in the experiments section. The general approach here is to decompose the problem of finding an untangling into a sequence of local goals consisting of elementary moves. This approach has two advantages: the local goal is much easier to plan for and the final configuration does not need to be known. This idea appears in robotics literature in task planning and in hierarchical planning. In our case, formulating the planning problem in terms of a sequence of local goals was essential to obtain a usable planner. When it is possible to find such formulation, we suggest that it will often be a major improvement to the speed of the planner.

Energy hints versus random There are several planning applications with natural energy functions such as protein folding, molecular docking and realistic deformable robots. Also, there has been some investigation in using energy when planning for such systems [SA01, ASBL01]. After running the loop in Algorithm 1 N times the next configuration to consider is chosen.

In our implementation we used energy to make this choice. By eliminating lines 17 and 18 from the final section of Algorithm 1, thus always executing the random guess on line 19, we obtained very poor results. We ran 50 trials on the Ochiai unknot for nearly an hour with almost no progress. We observed that the knot tended to stay in high energy configurations when moving at random. This observation is consistent with our observation that the planner seems to hampered by high energy knot configurations. We believe that the reason for this behaviour is that high energy configurations are more constrained and it is more difficult for the tree to escape from a high energy region of a high dimensional configuration space. Energy functions with similar properties, either natural or artificially designed, might exist for other planning instances. A more general observation is that using an energy function to bias sampling can be extremely effective. Further investigation and awareness of hybrid energy and planning algorithms may lead to improvements in a variety of planning instances.

7 Future Work and Applications

In the final section, we briefly describe an outstanding issue relating to probabilistic completeness and sketch some possible future directions for our work. Although the principal motivation for this exploration has been to gain insight in motion planning, there are several particular application areas which are more directly connected.

Probabilistic completeness Although we have not shown probabilistic completeness for our planner, we give a conjecture which is both necessary and sufficient for our planner to be probabilistically complete.

Conjecture 7.1. *Let K be a piecewise linear knot drawn with n line segments. Suppose $s(K) < n$. Then there is a sequence of knots with n pieces $K = K_0, K_1, ..., K_m$ with $K_i \sim K_{i+1}$ and K_m has three collinear vertices.*

The proof of probabilistic completeness for the planner follows from this conjecture and proceeds according to the scheme described in [LK02].

Applications to Computational Topology Our planner can be used for stick number calculation of a knot where unknot detection is a special case. Another application is to calculate the unknotting number of a knot:

given some knot, the task is to find a path containing as few crossings as possible that transforms that knot to the unknot.

Applications to Structural Bioinformatics Knots in DNA rings play an important role in the life-cycle of a cell [PPC99, DRZ01]. Determining low energy paths for DNA knotting and unknotting might be used to learn how certain processes occur in cell biology. Our work also might be used to construct realistic examples of certain processes. More broadly, planning for high dimensional closed chains sitting in an energy landscape has many potential applications. Although the details are far more complex in such physical examples, the work presented in this paper may provide some ideas on how to approach the design of physical algorithms in such a context.

Flexible Robots There is a strong similarity with untangling and the general problem of planning with flexible robots: high dimensional configuration spaces that can change parameters over the course of the plan. In this work, we introduced the use of local goals and hybrid planning with potential fields, either of which may be used in a more general setting. An interesting direct extension of this work would be to study the problem of tying knots in physically realistic rope such as that described in [PLK02].

Acknowledgments

Work on this paper by A. Ladd and L. Kavraki has been supported in part by NSF 9702288, NSF 0308237, NSF 0205671, a Whitaker Biomedical Engineering Grant, and a Sloan Fellowship to L. Kavraki. A. Ladd is also supported by FCAR.

References

[ABG+02] M.S. Apaydin, D.L. Brutlag, C. Guestrin, D. Hsu, and J.C. Latombe. Stochastic roadmap simulation: An efficient representation and algorithm for analyzing molecular motion. In *International Conference on Computational Molecular Biology (RECOMB)*, April 2002.

[AHT] I. Agol, J. Hass, and W. Thurston. The computational complexity of knot genus and spanning area. (preprint.).

[Ane99] C. N. Anerizis. *The Mystery of Knots - Computer Programming for Knot Tabulation.* Series on Knots and Everything. World Scientifical Publishing Co. Pte. Ltd., 1999.

[ASBL01] M.S. Apaydin, A.P. Singh, D.L. Brutlag, and J.C. Latombe. Capturing molecular energy landscapes with probabilistic conformal roadmaps. In *IEEE International Conference on Robotics and Automation (ICRA)*, May 2001.

[BBRV00] J. S. Birman, P. Boldi, M. Rampichini, and S. Vigna. Towards an implementation of the b-h algorithm for recognizing the unknot. In *KNOTS-2000*, 2000.

[BEa99] M. Bern, D. Eppstein, and al. Emerging challenges in computational topology, 1999.

[BKRT01] Z. Butler, K. Kotay, D. Rus, and K. Tomita. Cellular automata for decentralized control of self-reconfigurable robots. In *IEEE International Conference on Robotics and Automation*, 2001.

[BLA02] O.B. Bayazit, J-M. Lien, and N.M. Amato. Probabilistic roadmap motion planning for deformable objects. In *IEEE International Conference on Robotics and Automation*, 2002.

[BM00] C. Benham and D. Miller. Writhing and linking densities for closed, circular dna. *Journal of Knot Theory and Its Ramifications*, 9(5):577–585, 2000.

[CKG+94] N. Crisona, R. Kanaar, T. Gonzales, E. Zechiedrich, A. Klippel, and N. Cozzarelli. Processive recombination by wil-type gin and an enhancer-independent mutant. *J. Mol. Biol.*, 243:437–457, 1994.

[Dar01] I. Darcy. Biological distances on dna knots and links. *Journal of Knot Theory and Its Ramifications*, 10(2):269–294, 2001.

[DD00] X. Dai and Y. Diao. The minimum of knot energy functions. *Journal of Knot Theory and its Ramifications*, 9(6):713–724, 2000.

[DER97] Y. Diao, C. Ernst, and J.Van Rensburg. In search of a good polygonal knot energy. *Journal of Knot Theory and its Ramifications*, 6(5):633–657, 1997.

[DRZ01] R.W. Deibler, S. Rahmati, and E.L. Zechiedrich. Topoisomerase iv, alone, unknots dna in escherichia coli. *Genes and Development*, 15:748–761, 2001.

[GHK96] R.P. Grzeszczuk, M. Huang, and L.H. Kauffman. Untangling knots by stochastic energy optimization. *IEEE Visualization*, pages 279–286, 1996.

[GHK97] R.P. Grzeszczuk, M. Huang, and L.H. Kauffman. Physically-based stochastic simplification of mathematical knots. *IEEE Transactions on Visualization and Computer Graphics*, 3(3):262–278, 1997.

[HKLR01] D. Hsu, R. Kindel, J.C. Latombe, and S. Rock. Control-based randomized motion planning for dynamic environments. In *Algorithmic and Computational Robotics: New Directions: The Fourth International Workshop on the Algorithmic Foundations of Robotics*, pages 247–264, 2001.

[HL] J. Hass and J. Lagarias. The number of reidemeister moves needed for unknotting. (preprint.).

[HLM97] D. Hsu, J.C. Latombe, and R. Motwani. Path planning in expansive spaces. In *Proc. IEEE Int'l Conf. on Robotics and Automation*, pages 2719–2726, 1997.

[HLP97] Joel Hass, J. C. Lagarias, and Nicholas Pippenger. The computational complexity of knot and link problems. In *IEEE Symposium on Foundations of Computer Science*, pages 172–181, 1997.

[Hsu00] D. Hsu. *Randomized Single-Query Motion Planning In Expansive Spaces*. PhD thesis, Department of Computer Science, Stanford University, 2000.

[HT98] J. Hoste and M. Thistlethwaite. The first 1,701,936 knots. *Math. Intelligencer*, 20(4):33–48, 1998.

[Jen89] R. Jenkins. A dynamic approach to calculating the homfly polynomial for directed knots and links. Master's thesis, Carnegie Mellon University, 1989.

[JVW90] F. Jaeger, D. L. Vertigan, and D.J.A. Welsh. On the computational complexity of the jones and tutte polynomials. In *Math. Proc. Camb. Phil. Soc.*, 108, pages 35–53, 1990.

[KL99] J. J. Kuffner and S. M. LaValle. Randomized kinodynamic planning. In *Proc. IEEE Int'l Conf. on Robotics and Automation*, pages 473–479, 1999.

[KL00] J. J. Kuffner and S. M. LaValle. RRT-connect: An efficient approach to single-query path planning. In *Proc. IEEE Int'l Conf. on Robotics and Automation*, 2000.

[KL01a] J. J. Kuffner and S. M. LaValle. Randomized kinodynamic planning. *International Journal of Robotics Research*, 20(5):378–400, May 2001.

[KL01b] J. J. Kuffner and S. M. LaValle. Rapidly exploring random trees: Progress and prospects. In *Algorithmic and Computational Robotics: New Directions: The Fourth International Workshop on the Algorithmic Foundations of Robotics*, pages 293–308, 2001.

[KM94] L. H. Kauffman and Y. Magarshak. Graph invariants and the topology of rna folding. *Journal of Knot Theory and Its Ramifications*, 3(3):233–245, 1994.

[KSLO96] 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.*, 12(4):566–580, June 1996.

[Lic97] W.B.R. Lickorish. *An Introduction to Knot Theory.* Springer, 1997.

[LK01] F. Lamiraux and L.E. Kavraki. Planning paths for elastic objects. *International Journal of Robotics Research*, 20(3), 2001.

[LK02] A. Ladd and L. Kavraki. A measure theoretic analysis of PRM. In *IEEE International Conference on Robotics and Automation*, May 2002.

[LS94] T. Ligocki and J. A. Sethian. Recognizing knots using simulated annealing. *Journal of Knot Theory and Its Ramifications*, 3(4):477–495, 1994.

[Pip89] N. Pippenger. Knots in random walks. *Discrete Applied Mathematics*, 25:273–278, 1989.

[PLK02] J. Phillips, A. Ladd, and L. Kavraki. Simulated knot tying. In *IEEE International Conference on Robotics and Automation*, May 2002.

[PPC99] L. Postow, B.J. Peter, and N.R. Cozzarelli. Knot what we thought before: The twisted story of replication. *BioEssays*, 21:805–808, 1999.

[SA01] G. Song and N. Amato. Using motion planning to study protein folding pathways. In *International Conference on Computational Molecular Biology (RECOMB)*, pages 287–296, April 2001.

[Sch88] R. Scharein. *Interactive Topological Drawing.* PhD thesis, University of British Columbia, 1988.

[Sim94] J. Simon. Energy functions for polygonal knots. *J. Knot Theory and its Ramif.*, 3:299–320, 1994.

[SKB+96] A. Stasiak, V. Katritch, J. Bednar, D. Michoud, and J. Dubochet. Electrophoretic mobility of dna knots. *Nature*, 384(6605):142–145, 1996.

[SL01] G. Sánchez and J.-C. Latombe. A single-query bidirectional motion planner with lazy collision checking. In *Proceedings of International Symposium on Robotics*, 2001.

[SW88] D.W. Sumners and S.G. Whittington. Knots in self-avoiding walks. *Journal Physics A: Mathematicals General*, 21:1689–1694, 1988.

[SW93] S. Y. Shaw and J. C. Wang. Knotting of a dna chain during ring closure. *Science*, 260(23):533–536, April 1993.

[SW94] S. Y. Shaw and J. C. Wang. Dna knot formation in aqueous solutions. *Journal of Knot Theory and Its Ramifications*, 3(3):287–298, 1994.

[TPK02] M. Teodoro, G.N. Phillips, and L.E. Kavraki. A dimensionality reduction approach to modeling protein flexibility. In *International Conference on Computational Molecular Biology (RECOMB)*, April 2002.

[VSY02] S. Vassilvitskii, J. Suh, and M. Yim. A complete, local and parallel reconfiguration algorithm for cube style modular robots. In *IEEE International Conference on Robotics and Automation (ICRA)*, May 2002.

[WTA02] J. Walter, B. Tsai, and N. Amato. Choosing good paths for fast distributed reconfiguration of hexagonal metamorphic robots. In *IEEE International Conference on Robotics and Automation*, 2002.

[Wu96] Y-Q. Wu. Ming user manual. www.math.uiowa.edu/~wu/ming/ming.pdf, 1996.