Polygonal Chains Cannot Lock in 4D

Roxana Cocan and Joseph O'Rourke*

Abstract

We prove that, for all dimensions $d \geq 4$, every simple open polygonal chain may be straightened, and every simple closed polygonal chain may be convexified. Both can be achieved by algorithms that use polynomial time in the number of vertices, and result in a polynomial number of "moves." These results contrast to those known for $d = 3$, where open and closed chains can be "locked."

1 Introduction

1.1 Summary

A *polygonal chain* $P = (v_0, v_1, \ldots, v_n)$ is a sequence of consecutively joined segments (or edges) $e_i = v_i v_{i+1}$ of fixed lengths $\ell_i = |e_i|$, embedded in space. A chain is *closed* if the line segments are joined in cyclic fashion, i.e., if $v_n = v_0$; otherwise, it is *open*. A chain is *simple* if only adjacent edges intersect, and only then at the endpoint they share. We study reconfigurations of simple polygonal chains, continuous motions that preserve the lengths of all edges while maintaining simplicity throughout. One basic goal is to determine if an open chain can be *straightened*—stretched out in a straight line, and whether a closed chain can be *convexified*—reconfigured to a planar convex polygon. If neither is possible, the chain is called *locked*. This terminology is borrowed from [BDD+99].

It is an open problem to determine if every open (or closed) chain in 2D can be straightened (or convexified).[1] See [ELR+98] for partial results. In 3D it is known that there exist both open and closed chains that are locked [CJ98, BDD+99, Tou99], and several special cases are known to be unlocked [BDD+99]. In this paper we prove that, for all dimensions $d \geq 4$, neither open nor closed chains can lock. We partition our results into three main theorems:

Theorem 1 *Every simple open chain in 4D may be straightened, by an algorithm that runs in $O(n^3\alpha(n))$ time, and which accomplishes the straightening in $O(n)$ moves.*

Here "move" is used in the sense defined in [BDD+99]. Essentially each move is a simple monotonic rotation of a few joints. We have implemented this algorithm for the case when the vertices are in general position, when it is straightforward.

Theorem 2 *Every simple closed chain in 4D may be convexified, by an algorithm that runs in $O(n^6 \log n)$ time, and which accomplishes the straightening in $O(n^6)$ moves.*

Theorem 3 *Both Theorems 1 and 2 hold for all dimensions $d \geq 4$, i.e., polygonal chains cannot lock in dimensions greater than three.*

We partition the results this way because the proofs of these three theorems are rather different. In particular, the proof of Theorem 3 is not difficult. In this extended abstract, we will sketch the proofs of the theorems. More detailed proofs are contained in [Coc99], and we are currently preparing a full version for publication.

We summarize our results in the context of earlier work in the table below.

Dimension	(Un)Locked
2	?
3	$\exists$ locked chains
$d \geq 4$	Cannot lock

2 Straightening Open Chains in 4D

Let P be a simple, open polygonal chain in 4D with $n \geq 2$ vertices v_i. Each vertex is also called a *joint* of the chain. Let $s_i = v_i v_{i+1}$ be the i-th segment or *link* of the chain. We say a joint v_i is *straightened* if (v_{i-1}, v_i, v_{i+1}) are collinear and form a simple chain; in this case, the angle at v_i is π.

We prove Theorem 1 by straightening the first joint v_1, "freezing" it, and repeating the process until the entire chain has been straightened. This is a procedure which, of course, could not be carried out in 3D. But there is much more room for maneuvering in 4D.

Let $s_0^g = v_0^g v_1$ be the *goal position* for segment s_0: the position that represents straightening of joint v_1. Let w_0 be the *goal direction*: a vector orthogonal to s_0^g that represents the direction in which s_0 should be rotated to move it to its goal position. Note that rotation is in a plane in d dimensions, in this case the plane determined by s_0 and w_0.

We distinguish three possibilities:

*Dept. of Computer Science, Smith College, Northampton, MA 01063, USA. {rcocan,orourke}@cs.smith.edu. Research supported by NSF Grant CCR-9731804.

[1] An example of J. Mitchell [Personal communcation, Feb. 1999.] makes it seem likely that there are locked chains in 2D as well.

1. The goal position is *intersected* by some other link of the chain.

2. The goal direction is *obstructed* in that rotation of s_0 in the direction w_0 will hit some link of the chain along the way to the goal position. Note that if the goal position is intersected, the goal direction is obstructed because s_0 will hit the intersecting segments when it reaches the goal position.

3. The goal direction is *free* when it is not obstructed (and so the goal position is not intersected).

We consider these possibilities in reverse order, easiest to most difficult. A high-level view of the algorithm is as follows:

repeat until chain straightened do
  1: if w_0 is *free* then
    Rotate s_0 directly to s_0^g.
  else if w_0 is *obstructed* then
    Rotate s_0 to new position whose goal direction is free.
    goto 1.
  else if s_0^g is *intersected* then
    Move v_1 so that the goal position is not intersected.
    goto 1.

In this abstract we will concentrate only on skirting obstructions, the middle step of the algorithm. This employs what we call the "obstruction diagram." First we describe the space in which the obstruction diagram is embedded.

Consider the space of possible directions from which s_0 might approach s_0^g. In 3D, this set of unit vectors forms a circle $\mathbb{S}^1$, which can be viewed as orthogonal to and centered on s_0^g. Similarly, in 4D, the set of possible approach directions toward s_0^g forms a 2-sphere $\mathbb{S}^2$. Every point on this sphere represents a direction of approach to s_0^g. In dimension d, this set of directions form the unit sphere $\mathbb{S}^{d-2}$. Note we consider approach directions to s_0^g rather than departure directions from s_0, because the former is fixed on the unmoving s_0^g, whereas we will be moving s_0.

Let $\hat{w}$ be the unit vector along a direction w. The *obstruction diagram* $\mathrm{Ob}(s_0^g)$ is the set of vectors $\hat{w}$ representing obstructed goal directions for s_0^g. The obstructions arise from the particular configuration of the other links in the chain P (aside from s_0). We now argue for this key lemma:

Lemma 1 *The obstruction diagram $Ob(s_0^g)$ consists of a union of n arcs on $\mathbb{S}^2$.*

Consider first the obstruction diagram for a 3D chain. Each segment of the chain potentially obstructs a set of approach directions for s_0 that form an arc on the circle $\mathbb{S}^1$. This can be seen by projecting a segment s_i onto the plane containing $\mathbb{S}^1$, which is orthogonal to s_0^g, and then viewing that projected segment s_i' from the center of $\mathbb{S}^1$. The union of these n arcs is $\mathrm{Ob}(s_0^g)$. It is clear that such a union could cover all of $\mathbb{S}^1$, which means that every approach direction is obstructed. And indeed it is easy to arrange this by surrounding s_0^g with a "cage" of segments, so that s_0 cannot approach the goal position without colliding with some segment of the cage.

Now turn to the case at hand, the obstruction diagram $\mathrm{Ob}(s_0^g)$ for a 4D chain. Take an arbitary segment s_i of the chain, and project it into the 3D subspace containing the sphere $\mathbb{S}^2$ representing all directions orthogonal to s_0^g. From the center of $\mathbb{S}^2$, this projected segment s_i' corresponds to an arc on the sphere. See Fig. 1. Note we are being conservative by including an arc in $\mathrm{Ob}(s_0^g)$ even if the obstruction is "behind" s_0 and so does not present a true obstacle to rotation.

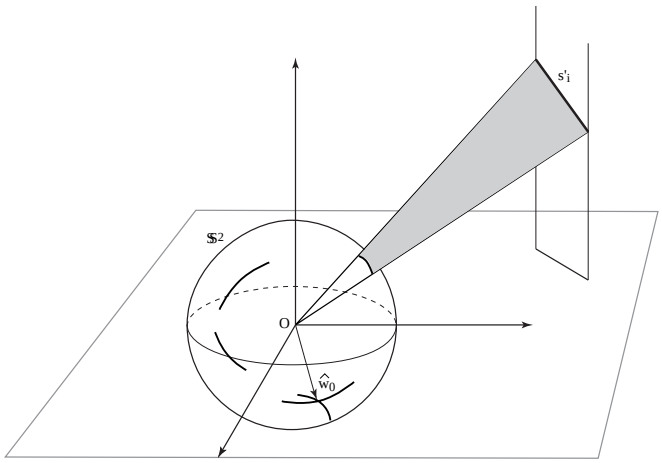

Figure 1: In 4D, s_i projects to s_i' in the 3D subspace containing $\mathbb{S}^2$, and produces an arc of the obstruction diagram determined by the intersection of the triangle (O, s_i') with $\mathbb{S}^2$.

The implication of Lemma 1 is that not all directions can be obstructed, for a finite collection of arcs cannot cover all of $\mathbb{S}^2$. If w_0 is obstructed, then $\widehat{w}_0$ touches one or more arcs of the obstruction diagram (as in Fig. 1). It is now relatively easy to find a Δw that will move w_0 to be free: choose a direction on $\mathbb{S}^2$ that steps off of the obstructed spot, and choose a step length that maintains the simplicity of the chain. Move s_0 in the direction Δw; and from its new position s_0^*, its goal direction $w_0^* = w_0 + \Delta w$ is unobstructed. Now move s_0^* directly to the goal. No further details will be presented.

The total number of moves used by the algorithm is at most $3n = O(n)$. It is easy to achieve the claimed time complexity of $O(n^3\alpha(n))$ as follows. For each of the n steps, construct the relevant obstruction diagrams as arrangements of circular arcs on a sphere. This arrangement can be constructed in $O(n^2\alpha(n))$ time and $O(n^2)$ space [EGP+92, Hal97]. This then establishes Theorem 1.

3 Convexifying Closed Chains in 4D

Our algorithm for convexifying closed chains employs the *line tracking* motions introduced in [LW95]. Indeed our algorithm mimics theirs in that we repeatedly apply line tracking motions, each of which straightens at least one joint, until a triangle is obtained (which is a planar convex polygon, as desired). Although the overall design of our algorithm is identical, the details are quite different, for there are two major differences with [LW95]: (1) They permitted self-intersections of the chain; (2) Their choice of a line in the line tracking motion is different than ours.

Let $(v_0, v_1, v_2, v_3, v_4)$ be five consecutive vertices of a closed polygonal chain. We allow $v_0 = v_4$. A *line tracking* motion of v_2 moves v_2 along some line L in space, while keeping both v_0 and v_4 fixed. As long as the angle at joints

v_1 and v_3 (the *elbows*) are neither π (straight) nor 0 (folded), such a motion is possible. Neither angle can be 0 because that would violate the simplicity of the chain. Straightening one joint is precisely our goal, so we assume that neither joint is straight; and therefore a line tracking motion is possible.

We will choose L and a direction along it so that the movement increases the distance from v_2 to both v_0 and v_4 simultaneously. This necessarily opens both elbow angles. The motion stops when one elbow straightens. The only issue is whether this can be done while maintaining simplicity. We prove this lemma:

Lemma 2 *For a simple 4D chain $(v_0, \ldots, v_4)$, there exists a line tracking motion of v_2 that straightens either v_1 or v_3 (or both) while maintaining simplicity of the chain throughout the motion.*

As before, we start by thinking about the situation in 3D. Let $\mathbb{R}_{[0,1)}$ be the interval $[0, 1)$ on the real line. We will parametrize the location of v_2 along L by $t \in [0, 1)$, with $t = 0$ the start, and $t = 1$ chosen to be the first time at which a joint, which we take to be v_1 without loss of generality, straightens. Let $\mathcal{C}$ be the configuration space of the four-link system in isolation, permitting intersections between the links. It should be clear that

$$\mathcal{C} = \mathbb{S}^1 \times \mathbb{S}^1 \times \mathbb{R}_{[0,1)}. \tag{1}$$

This can be seen as follows. Fix some t so that v_2 is fixed. Then each of v_1 and v_3 is free to rotate (independently) on a circle in 3D centered on v_0v_2 and v_2v_4 respectively. As t varies from 0 to 1, these circles move in space, and grow and shrink in radius. At $t = 1$ the v_1 circle shrinks to a point. But for $t \in [0, 1)$, both circles retain a positive radius. Thus the configuration space $\mathcal{C}$ has the topology of $\mathbb{S}^1 \times \mathbb{S}^1$ for each t, and thus the claim follows.

Turning now to 4D, an elbow at the join of two links has a space of possible motions in 4D that is topologically $\mathbb{S}^2$, for it is the intersection of two 3-spheres. Thus the configuration space of our four-link chain, ignoring self-intersections, is

$$\mathcal{C} = \mathbb{S}^2 \times \mathbb{S}^2 \times \mathbb{R}_{[0,1)}. \tag{2}$$

As in Section 2, we incorporate the obstacles representing the other links via an "obstruction diagram." We start by ignoring the four moving links as obstructions, and only consider the remaining links of the polygonal chain as obstacles. We develop the obstruction diagram first for fixed t, so that the relevant configuration space is $\mathbb{S}^2 \times \mathbb{S}^2$. Because we are ignoring the moving links as obstructions, movement on the two spheres is independent, so it suffices to determine the obstruction diagram on one $\mathbb{S}^2$, that for v_1: $\mathrm{Ob}(v_1)$. Our key observation is the following lemma:

Lemma 3 *In 4D, if $(v_2 - v_0) \cdot (v_1 - v_0) \neq 0$ and $(v_2 - v_0) \cdot (v_2 - v_1) \neq 0$, then a single segment contributes at most four points to $Ob(v_1)$.*

Let $p_1(t)$ represent the position of v_1 on its sphere $\mathbb{S}^2$ at a particular time t. The goal is for the links (v_0, v_1, v_2) to avoid all obstacles, which means that $p_1(t)$ should avoid points of the obstruction diagram. The import of the above lemma is that at a fixed t avoiding the special cases mentioned in the lemma, the diagram is a finite set of points. Letting t vary, the points move. It is always possible to "run away from" a finite set of moving points on the surface of a 2-sphere. This freedom permits us to avoid self-intersections, and therefore to straighten a joint.

We offer an informal geometric argument for Lemma 3, which is key to establishing Theorem 2. Let s be a segment, and let C be a cone spun out by the link v_0v_1. We first explore the situation in 3D, and then argue in a parallel manner for 4D. In 3D:

1. If C does not lie in a plane, i.e., the apex angle of C is not π.
   (a) Let s lie in a plane Π parallel to the base of C, such that s does not contain the apex a of the cone. See Fig. 2a. Then $\Pi \cap C$ is a circle, and it is clear that s intersects this circle in at most two points.
   (b) Now suppose s is in a plane Π not parallel to the base, such that s does not contain the apex a of the cone. Then $\Pi \cap C$ is an ellipse, and again s intersects this ellipse in at most two points.
   (c) If s lies in a line through the apex a of C, then it may intersect C in a segment.
2. If C lies in a plane Π [Case omitted here].

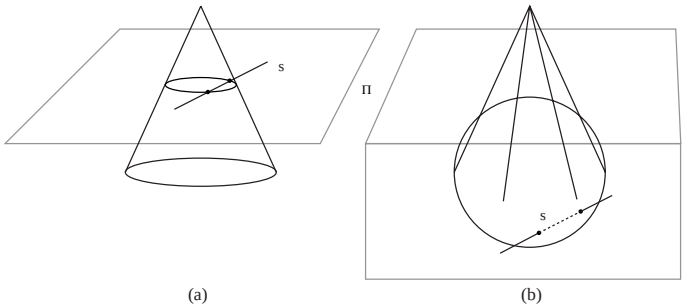

Figure 2: (a) A segment intersects a cone in two points in 3D; (b) A segment intersects a cone in two points in 4D.

Now to 4D. Just as the base of a cone in 3D is a 1-sphere, the base of a cone in 4D is a 2-sphere. This base sphere lies in a 3D subspace orthogonal to the axis of the cone.

1. If C does not lie in a 3D subspace.
   (a) Let s lie in a 3D subspace Π parallel to the base of C, i.e., orthogonal to the cone axis, and s does not contain the apex a of the cone. Then $\Pi \cap C$ is a sphere, and s intersects this sphere in at most two points. See Fig. 2b.
   (b) Let s lie in a 3D subspace Π not parallel to the base, and s does not contain the apex a of the cone. Then $\Pi \cap C$ is an ellipsoid, and s intersects this ellipsoid in at most two points.
   (c) If s lies in a line through the apex a of C, then it may intersect C in a segment.
2. If C lies in a 3D subspace Π [Case omitted here].

So except for the special cases where s contains the apex, or C is degenerate, s intersects the cone in at most two points. Lemma 3 follows because the chain (v_0, v_1, v_2) sweeps out two cones.

The case in the preconditions of Lemma 3 refers to the situation in which one cone is degenerately flat. As Fig. 3 illustrates, here a segment might obstruct a range of rotations of $v_2 - v_1$, producing an arc in $\mathrm{Ob}(v_1)$. But because these preconditions can only hold once as t varies from 0 to 1, they present no impediment to connectivity.

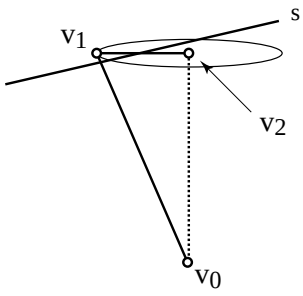

Figure 3: $(v_2 - v_0) \cdot (v_2 - v_1) = 0$ and segment s (which lies in the plane of the circle) contributes an arc to the obstruction diagram $\mathrm{Ob}(v_1)$.

3.1 Motion Planning

Skipping the details that establish that our initial and final positions, $t = 0$ and $t = 1$, are connected in the configuration space $\mathcal{C}$, we know a path that avoids self-intersection exists, i.e., either the joint v_1 or v_3 can be straightened. The next step is to compute such a path algorithmically. General motion planning algorithms here yield a polynomial-time algorithm.

Our "robot" consists of the four links $(v_0, v_1, v_2, v_3, v_4)$ moving in the 5-dimensional configuration space $\mathcal{C}$, Eq. (2). The subspace $\mathcal{C}_0$ that avoids self-intersection between the four links is some semialgebraic subset of $\mathcal{C}$, semialgebraic because the constraints on self-intersection may be written in Tarski sentences (see, e.g., [Mis97]). The free configuration space $\mathcal{F}$ is composed of the points of $\mathcal{C}_0$ that avoid the obstacles, which is again a semialgebraic set. Then motion planning between two points of $\mathcal{F}$ in the same connected component may be achieved by any general motion planning algorithm [Sha97, Sec. 40.1.1]. For example, Canny's Roadmap algorithm achieves a time and space complexity of $O(n^k \log n)$, where n is the number of obstacles, and k the number of degrees of freedom in the robot's placements. In our case, $k = 5$. His algorithm produces a piecewise algebraic path through $\mathcal{F}$, of $O(n^k)$ pieces. Each piece constitutes a constant number of moves, with the constant depending on the algebraic degree of the curves, which is bounded as a function of k. Therefore each joint straightening can be accomplished in $O(n^5)$ moves. Repeating the planning and straightening n times leads to $O(n^6)$ moves in $O(n^6 \log n)$ time, proving Theorem 2.

4 Higher Dimensions

We have already shown that every simple open/closed chain in 4D can be straightened/convexified. Our task is to prove that this result holds for higher dimensions, using the results from 4D.

For an open chain, we straighten four links at a time and then repeat the procedure until the chain is straight. If the chain contains less than four links, then it determines a space that has at most dimension three, and it can be included in a 4D-space. For a closed chain, our algorithm also moves four links at a time. Four links determine a space that is at most 4-dimensional, which means that it can be included in a 4D-subspace of our d-dimensional space $\mathbb{R}^d$, $d \geq 4$.

We have already shown that these four links, for both open and closed chains, can be straightened in 4D; therefore, they can be straightened in this 4D-subspace of $\mathbb{R}^d$. We only have to worry about the pieces of the remainder of the chain that intersect this 4D-subspace. But since we are dealing with segments, their intersection with a 4D-subspace is either a point or a segment. But these are the kind of obstructions we have proven that can be avoided in 4D. Therefore, the straigtening of these four links can be completed in the 4D-subspace, and therefore in the d-dimensional space $\mathbb{R}^d$, while maintaining rigidity and simplicity.

The complexity for the algorithms in d-dimensional spaces, $d \geq 4$, is the same as for the algorithms in 4D.

Acknowledgement. We thank Erik Demaine for helpful comments.

References

[BDD+99] T. Biedl, E. Demaine, M. Demaine, S. Lazard, A. Lubiw, J. O'Rourke, M. Overmars, S. Robbins, I. Streinu, G. Toussaint, and S. Whitesides. Locked and unlocked polygonal chains in 3D. In *Proc. 10th ACM-SIAM Sympos. Discrete Algorithms*, pages 866–867, January 1999.

[CJ98] J. Cantarella and H. Johnston. Nontrivial embeddings of polygonal intervals and unknots in 3-space. *J. Knot Theory Ramifications*, 7:1027–1039, 1998.

[Coc99] Roxana Cocan. Polygonal chains cannot lock in 4D. Undergraduate thesis, Smith College, 1999.

[EGP+92] H. Edelsbrunner, Leonidas J. Guibas, J. Pach, R. Pollack, R. Seidel, and M. Sharir. Arrangements of curves in the plane: Topology, combinatorics, and algorithms. *Theoret. Comput. Sci.*, 92:319–336, 1992.

[ELR+98] H. Everett, S. Lazard, S. Robbins, H. Schröder, and S. Whitesides. Convexifying star-shaped polygons. In *Proc. 10th Canad. Conf. Comput. Geom.*, pages 2–3, 1998.

[Hal97] D. Halperin. Arrangements. In J. E. Goodman and J. O'Rourke, editors, *Handbook of Discrete and Computational Geometry*, chapter 21, pages 389–412. CRC Press LLC, Boca Raton, FL, 1997.

[LW95] W. J. Lenhart and S. H. Whitesides. Reconfiguring closed polygonal chains in Euclidean d-space. *Discrete Comput. Geom.*, 13:123–140, 1995.

[Mis97] B. Mishra. Computational real algebraic geometry. In J. E. Goodman and J. O'Rourke, editors, *Handbook of Discrete and Computational Geometry*, chapter 29, pages 537–558. CRC Press LLC, Boca Raton, FL, 1997.

[Sha97] M. Sharir. Algorithmic motion planning. In J. E. Goodman and J. O'Rourke, editors, *Handbook of Discrete and Computational Geometry*, chapter 40, pages 733–754. CRC Press LLC, Boca Raton, FL, 1997.

[Tou99] G. T. Toussaint. A new class of stuck unknots in Pol_6. Technical Report SOCS-99.1, School of Comput. Sci., McGill Univ., 1999.