Locked and Unlocked Polygonal Chains in 3D*

T. Biedl† E. Demaine† M. Demaine† S. Lazard‡
A. Lubiw† J. O'Rourke§ M. Overmars¶ S. Robbins‖
I. Streinu§ G. Toussaint‖ S. Whitesides‖

February 6, 2001

Abstract

This paper studies movements of polygonal chains in 3D whose links are not allowed to cross or change length. Our main result is an algorithmic proof that any simple closed chain that initially takes the form of a planar polygon can be made convex in 3D. Other results include an algorithm for straightening open chains having a simple orthogonal projection onto some plane, and an algorithm for making convex any open chain initially configured on the surface of a polytope. All our algorithms require only $O(n)$ basic "moves".

1 Introduction

A *polygonal chain* $P = (v_0, v_1, \ldots, v_{n-1})$ is a sequence of consecutively joined segments $e_i = v_i v_{i+1}$ (also called edges or links) of fixed lengths $\ell_i = |e_i|$, embedded in space.[1] A chain is *closed* if $v_{n-1} = v_0$; otherwise, it is *open*. A closed chain is also called a *polygon*. A chain is *simple* if its edges are pairwise disjoint except for adjacent edges, which share only their common endpoint. Unless stated otherwise, *chain* means "simple polygonal chain." For an open chain, our goal is to straighten it; for a closed chain the goal is to *convexify* it, i.e., to reconfigure it to a planar convex polygon. Both goals are to be achieved by continuous motions that maintain link lengths and simplicity of the chain throughout.

A *locked* chain is one that cannot be straightened or convexified. Since a chain in 3D can be continuously moved between any of its unlocked configurations via straightened or

*This research was initiated at a workshop at the Bellairs Res. Inst. of McGill Univ., Jan. 31–Feb. 6, 1998, organized by A. Lubiw and S. Whitesides. This is a revised and expanded version of abstract [BDD+99a]. Research supported in part by FCAR, NSERC, and NSF.

†University of Waterloo, Waterloo, Canada. {biedl, eddemaine, mldemaine, alubiw}@uwaterloo.ca.
‡INRIA Lorraine, France. lazard@loria.fr.
§Smith College, Northampton, USA. {orourke, streinu}@cs.smith.edu.
¶Utrecht University, The Netherlands. markov@cs.ruu.nl.
‖McGill University, Montreal, Canada. {stever, godfried, sue}@cs.mcgill.ca.

[1] All index arithmetic throughout the paper is mod n.

convexified intermediate configurations, the property of being unlocked is of fundamental importance. Nontrivial knots provide examples of closed chains that are locked. However, as Figs. 1 and 2 show, even open chains and unknotted closed chains may be locked in 3D. Section 2 provides details.

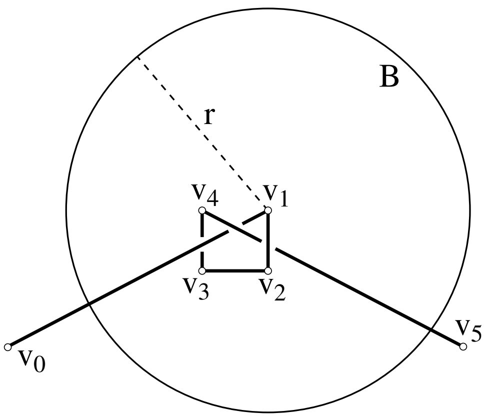

Figure 1: A locked, open chain K with long "knitting needles" at the ends.

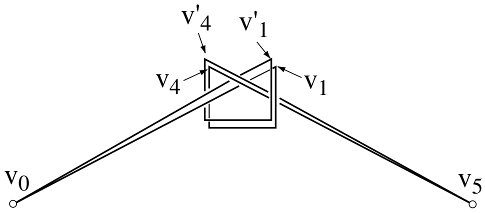

Figure 2: K' (K doubled): a locked but unknotted closed chain.

In Section 3, we give an algorithmic proof that any open chain with a simple orthogonal projection can be made straight in 3D, as well as an algorithmic proof that any open chain embedded in the surface of a polytope can be straightened. Section 4 presents our main result, an algorithmic proof that any closed chain initially taking the form of a polygon lying in the plane can be made convex.

We describe our algorithms in terms of "moves". Throughout the paper, a " move" is a continuous motion of a chain in which only $O(1)$ angles at joints (vertices) change at once, and only $O(1)$ dihedral angles at edges change at once. (The dihedral angle of an edge is the angle between the plane it determines with one of its neighboring edges and the plane it determines with the other.) Our algorithms make easily described moves that change angles at a very small number of vertices and edges at once.

After we reported our work in abstract form ([BDD$^+$99a]), Connelly, Demaine, and Rote [CDR00] and Streinu [Str00] reported two approaches for convexifying polygons in the plane using motions just in the plane. Our algorithm (Section 4) convexifies planar polygons using motions in 3D, which at first glance appears a less general result. However, the motions in all our algorithms are significantly simpler, consisting of a linear number of moves. (Our technical report [BDD$^+$99b] argues that the sequence of moves can be computed in $O(n)$ time on an extended real RAM model of computation). On the other hand, the motions of [CDR00] are described implicitly by the solution of a differential equation, and the motions of [Str00] are given by a high-degree algebraic curve. Both of these types of motions are difficult to compute and represent explicitly.

Background

The study of movement problems for polygonal chains goes at least as far back as A. Cauchy's 1813 theorem on the rigidity of polyhedra [Cro97, Ch. 6]. Mechanical engineers have studied planar linkages, which exhibit a graph structure whose links are allowed to cross, since Peaucellier's 1864 linkage or earlier. Edge weighted graph embedding and reconfiguration problems, with or without a simplicity requirement, have arisen in many contexts, including molecular conformation, mechanical design, robotics, animation, rigidity theory, algebraic geometry, random walks, and knot theory.

Basic questions concerning open and closed chains have proved surprisingly difficult. For example, the question of whether every planar, simple open chain can be straightened in the plane while maintaining simplicity went unsolved for several years. In piecewise linear knot theory, complete classification of the 3D embeddings of closed chains with n edges has been found to be difficult, even for $n = 6$ ([CJ98]).

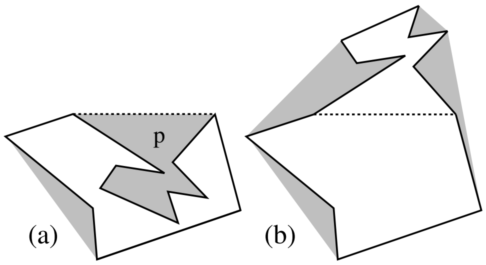

Figure 3: (a) A pocket p; (b) The polygon after flipping p.

Any simple polygon lying in the plane may be made convex in 3D by "flipping" out the reflex pockets, i.e., rotating the pocket chain into 3D and back down to the plane; see Fig. 3. This simple procedure was suggested by Erdős [Erd35] and proved to work by de Sz. Nagy [Nag39]. The number of flips, however, cannot be bounded as a function of the number of vertices n of the polygon, as Joss and Shannon [Grü95] first proved. See [Tou99] for the complex history of these results.

Previous computational geometry research on the reconfiguration of chains (e.g., [Kan97], [KSW96], [HJW85], [Whi92]) typically studies chains with *crossing* links, moving in 2D, sometimes in the presence of obstacles; [Sal73] and [LW95] study configurations of closed chains with crossing links in all dimensions $d \geq 2$. For more on weighted graph embedding and reconfiguration problems, see also, for example [HJW84], [Kor85], [CH88], [Whi97], and [Can87]. In answer to an open problem we posed in a preliminary version [BDD$^+$99a] of this paper, [CO99] have shown that locked chains do not exist in dimension greater than 3.

2 Chains that Lock

Here we discuss the examples shown in Figs. 1 and 2, which show that open chains and closed, unknotted chains may be locked. Both of these results were obtained independently by other researchers [CJ98]. We give particularly simple proofs.

Consider the chain $K = (v_0, \ldots, v_5)$ configured as in Fig. 1, where the lengths ℓ_0 and ℓ_4 of the extreme edges are each much longer than the sum of the lengths of the intermediate edges. Let B be a sphere centered at v_1 of radius r, where $(\ell_1+\ell_2+\ell_3) < r < \ell_4-2(\ell_1+\ell_2+\ell_3)$. Thus, no matter how the chain moves, v_1, v_2, and v_3 lie strictly inside B, whereas v_0 and v_5 lie strictly outside B. Join v_0 and v_5 outside B with a long string to form a knot. The string should be long enough not to impede the motions of v_0 and v_5, and the string should form an unknot when its ends are identified. Alternatively, it is not difficult to design ways to join v_0 and v_5 with an unlocked chain P consisting of a few long links that can track any motions of v_0 and v_5 while avoiding collisions. If chain K could be straightened, then the knot $K \cup P$ could be made convex, a contradiction.

Next, "double" chain K by adding vertices v_i' near v_i, for $1 \leq i \leq 4$, and connecting all these vertices to form a chain $K' = (v_0, \ldots, v_5, v_4', \ldots, v_1')$. See Fig. 2. Any convexifying motion for K' brings K to a convex configuration from which, removing the second copy of K, it can be straightened, a contradiction. Thus K' is also locked.

3 Two Unlocked Families of Open Chains

3.1 Straightening Open Chains with Simple Projections

Let P be an open chain in 3D with a simple orthogonal projection P' onto the xy-coordinate plane, denoted Π_{xy}[2]. This subsection describes an algorithm to straighten P. In answer to open problem #3 in our technical report [BDD+99b], Calvo et al. [CKMST] showed the more general result that any closed chain with a simple orthogonal projection P' on the plane can be convexified by using the method of [CDR00] as a subroutine. Our algorithm is much simpler.

We call lines and planes parallel to the z-axis "vertical", and in this subsection, we label projections in Π_{xy} with a prime. Let $P[i,j]$ denote the subchain $(v_i, v_{i+1}, \ldots, v_j)$, including endpoints v_i and v_j. Let the projection P' of P on Π_{xy}, let d_{min} denote the minimum distance between a vertex v' and a non-adjacent edge e', and choose a positive $r < d_{min}$. Let C_i denote a cylinder of radius r with a vertical axis passing through v_i. Finally, let Π_i denote the vertical plane containing e_i and e_i'.

For $0 \leq i < n$, the goal of step i is to squeeze the links of $P[0, i+1]$ into C_{i+1} so that v_0 lies within distance $(i+1)r/n$ of the axis of C_{i+1}, and so that $P[0, i+1]$ is *monotone* with respect to the line $\Pi_i \cap \Pi_{xy}$. By monotone, we mean that the intersection of the subchain with each vertical line in C_{i+1} is either empty or a single point. Step $i = 0$ achieves this goal by rotating e_0 in Π_0 about its endpoint v_1.

To initialize for step $i+1$, subchain $P[0,i]$ is kept internally rigid and rotated as an object about the axis of C_i. In particular, $P[0,i]$ is rotated away from e_i until $P[0,i]$ rotates into the vertical plane Π_i of e_i. To achieve the goal configuration of step $i+1$, edge e_i is rotated in Π_i about its endpoint v_{i+1} until its other endpoint, v_i, lies within distance r/n of the axis of C_{i+1}; the angle between the edges incident to v_i is changed so that any configuration of $P[0,i]$ during the rotation of e_i is a translate of the configuration of $P[0,i]$ when the rotation of e_i began. See Fig. 5.

Once the entire chain has been moved to a planar, monotone configuration in C_n, the joints may be straightened one by one, working from one end of the chain to the other.

The algorithm performs $O(n)$ moves. To compute descriptions of these rotations on an extended real RAM requires computation of the cylinder radius r. This may be done in $O(n^2)$

[2][BGRT96] provides a polynomial-time algorithm to determine whether P admits a simple orthogonal projection onto a plane and to output a projection plane if it exists.

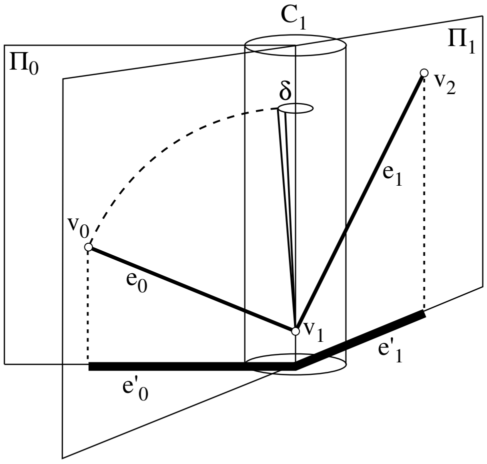

Figure 4: Step 0 rotates e_0 in Π_0 about v_1 to bring e_0 into C_1; step 1 begins by rotating e_0 about the C_1 axis until $\Pi_0 = \Pi_1$.

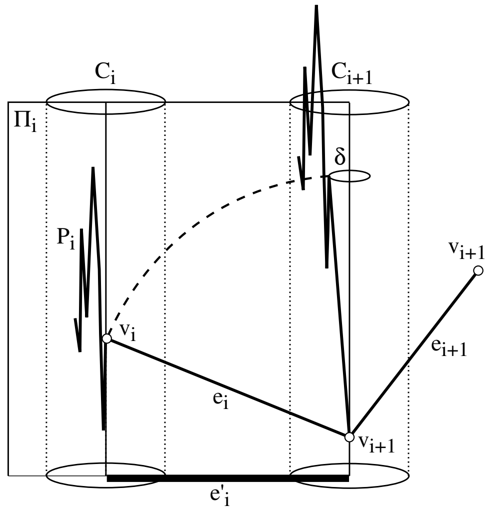

Figure 5: Step $i+1$ rotates e_i about v_{i+1} in Π_i, dragging along P_i.

time by computing each vertex-vertex distance and each edge-edge distance; in [BDD$^+$99b] we describe a method for computing r in $O(n)$ time.

3.2 Straightening Open Chains on a Polytope

In this subsection, we show that any open chain P embedded on the surface of a convex polytope may be straightened. We start with the case of an open chain lying in Π_{xy}, which we straighten (essentially as in Section 3.1) as follows. Rotate e_0 about v_1 within Π_0 until e_0 is vertical; now v_0 projects into v_1 on Π_{xy}. In general, rotate e_i about v_{i+1} within Π_i until v_i sits over v_{i+1}. Throughout this motion, keep the previously straightened chain $P_i = P[0, i]$ above v_i in a vertical ray through v_i.

An easy generalization of this "pick-up into a vertical ray" idea permits straightening any open chain lying on the surface of a convex polytope $\mathcal{P}$. The same procedure is followed, except that the surface of $\mathcal{P}$ plays the role of Π_{xy}, and surface normals play the roles of vertical rays. Suppose that a vertex v_i of the polygonal chain P lies on an edge or at a corner of $\mathcal{P}$, touching d faces $f_1, \ldots f_d$. Let $R_1 \ldots R_d$ denote the rays through v_i that are normal to $f_1, \ldots f_d$, respectively. For each v_i, choose a preferred normal. When such a vertex v_i is encountered, the line containing P_i is rotated to the preferred normal for v_i.

This algorithm performs $O(n)$ rotations, and a description of these can be computed in $O(n)$ time on an extended real RAM. Note that it is possible to draw on a polytope surface a chain that has no simple orthogonal projection to a plane, so this algorithm handles some cases not covered by the algorithm of the previous subsection.

4 Convexifying Planar Simple Polygons in 3D

This section offers a new algorithm for convexifying planar closed chains. It is more complicated than the flipping method of Erdős illustrated in Fig. 3 but has the advantage that it uses only a bounded number of moves, in fact $O(n)$ moves. Furthermore, in contrast to [CDR00] and [Str00], our algorithm remains conceptually simple and the moves it performs are easily described.

We consider a closed chain P, initially in the form of a simple polygon in the xy-plane, Π_{xy}. We assume that P has no more than two consecutive collinear vertices, as otherwise we can freeze and eliminate middle joints. For $\epsilon > 0$, let Π_ϵ be the plane $z = \epsilon$ parallel to Π_{xy}. Throughout this Section, primed labels indicate positions of vertices that have been lifted to Π_ϵ or above, and $\Pi_z(p, q)$ denotes the vertical plane containing points p and q.

4.1 Algorithm Overview

The algorithm lifts vertices one by one out of Π_{xy}, accumulating them into a vertical convex "arch" A with base vertices in Π_ϵ. The unlifted vertices remain fixed in their initial positions. Fig. 6 shows an arch A with base points v_0' and v_i' in Π_ϵ (not shown). Arch A lies above Π_ϵ in a vertical plane $\Pi_z(v_0', v_i')$. Edges $v_i' v_{i+1}$ and $v_0' v_{n-1}$ connect the base points v_0' and v_i' of A to $P[i+1, n-1]$ in Π_{xy}.

After a new vertex v_{i+1} (and typically, its adjacent edge $v_i' v_{i+1}$) is lifted to Π_ϵ, the arch is flipped down to Π_ϵ. There the arch together with the new vertex v_{i+1}' and its adjacent edge is reconvexified, with the future base points v_0' and v_{i+1}' held fixed during the process. The new arch is then flipped back up, rotating about the line through $v_0' v_{i+1}'$, to a new vertical plane $\Pi_z(v_0', v_{i+1}')$.

Performing reconvexification in Π_ϵ assures collisions with the links remaining in the plane, and with the two connecting links to the arch, are avoided. Similarly, raising the arch to vertical while lifting a new vertex to Π_ϵ prevents collisions with arch edges.

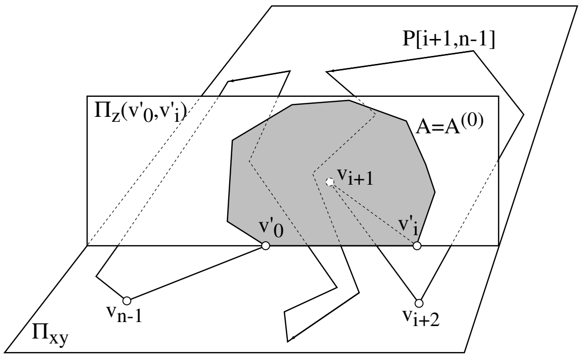

Figure 6: The arch A after incorporating $P[0, i]$; v_0' and v_i' lie in Π_ϵ (not shown).

4.2 Procedures

Our algorithm[3] makes use of the following procedures S_0–S_4: an initialization procedure S_0 that raises three vertices and creates a two-link arch; a vertex lifting procedure S_1 that lifts a new vertex v_i to Π_ϵ; an arch lowering procedure S_2 that rotates the current arch A about the line through its endpoints into Π_ϵ; a reconvexification procedure S_3 that convexifies the arch together with the vertex just lifted; and an arch raising procedure S_4 that returns the new arch to vertical.

At the end of the initialization S_0, the arch contains v_0', v_1', v_2'. It then makes repeated passes through S_1–S_4. Here, for reference, are the hypotheses that are intended to hold at the beginning of the pass through S_1–S_4 that lifts v_{i+1}, for $2 \leq i \leq n-2$. (See Fig. 6.) These hypotheses will be used in the proof of correctness.

H1: Vertices v_0' and v_i' lie in Π_ϵ, and $v_{i+1}, \ldots, v_{n-1}$ lie in Π_{xy} in their original positions.

H2: Subchain $P[0,i]$ has now been moved up to form a convex arch A in $\Pi_z(v_0', v_i')$; A rises above Π_ϵ and intersects Π_ϵ only in v_0' and v_i'.

H3: Points v_0' and v_i' project down to Π_{xy} within distance δ of their original positions v_0 and v_i. (In Section 4.3, we compute $\delta > 0$ from the initial configuration of P such that at each vertex v_j, the disk of radius δ centered at v_j includes no other vertices and intersects only the two edges incident to v_j, and such that any perturbed polygon obtained by displacing the vertices within their disks, ignoring the fixed link lengths, remains simple and without straight vertices.)

Next, we describe each of the procedures S_0–S_4 in detail, noting what constraints they impose on the choice of ϵ and δ each time they are performed. Choosing ϵ and δ that satisfy the constraints will enable us to prove that $H1$-$H3$ hold as intended. Later we show that it is possible to calculate values for ϵ and δ that satisfy all the constraints.

Note that procedures S_1–S_4 below do not apply until arch A contains at least three vertices.

S_0: initialization Select an arbitrary (strictly) convex vertex v_1, and lift v_0, v_1, v_2 as follows. Rotate v_1 about the line through $v_0 v_2$ up to Π_ϵ, and call its new position v_1''; rotate v_0 about the line through $v_{n-1} v_1''$ up to Π_ϵ, and call its new position v_0'; rotate v_2 about the line through $v_1'' v_3$ up to Π_ϵ, and call its new position v_2'; finally, rotate v_1'' about the line through $v_0' v_2'$ upwards until it lies in the plane $\Pi_z(v_0', v_2')$, and let v_1' denote its new position above Π_ϵ. This last rotation rotates a triangle from a horizontal to a vertical plane to create the initial arch A, so the triangle must be nondegenerate.

S_0**-constraints:** during the initialization procedure, vertices v_0, v_1, v_2 must reach Π_ϵ, and v_1'' should not become straight (to prevent a degenerate arch).

S_1: lifting v_i The purpose of S_1 is to lift[4] a vertex v_i $(i > 2)$ from Π_{xy} to Π_ϵ. To do this, rotate v_i about the line through v_{i-1}' and v_{i+1}. Thus v_i moves on a circle C with center on $v_{i-1}' v_{i+1}$. See Fig. 7. Note that circle C might lie partially below Π_{xy}, and that unless v_i is the last vertex, v_{n-1}, the plane that contains C is not vertical.

[3] We call this the *St. Louis Arch Algorithm* because of the resemblance to the arch in St. Louis, Missouri.

[4] When referring to previous Fig. 6, note that the vertex to be lifted next in that Figure is v_{i+1}.

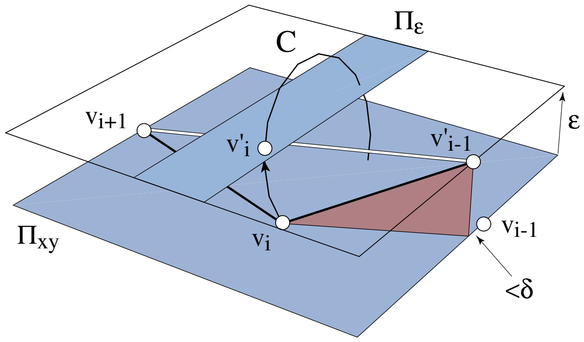

Figure 7: v_i rotates around C up to Π_ϵ; v'_{i-1} projects to within δ of v_{i-1}.

S_1**-constraints:** When vertex lifting is applied to a vertex v_i, for $2 < i \le n-1$, ϵ must be chosen so that C intersects Π_ϵ, and so that v'_i projects to Π_{xy} no more than δ away from v_i (to satisfy H3).

S_2: lowering the arch After v_i has been raised by S_1, edge $v'_{i-1}v'_i$ lies in Π_ϵ. If H2 holds, arch A is a convex chain lying (except for its base vertices) above Π_ϵ in $\Pi_z(v'_0, v'_{i-1})$. Procedure S_2 now rotates the arch A about the line through $v'_0v'_{i-1}$ away from edge $v'_{i-1}v'_i$ until A reaches plane Π_ϵ. It is possible that $\triangle v'_0v'_{i-1}v'_i$ is degenerate; in any case, the chain $P(v'_0, v'_1, \ldots, v'_{i-2}, v'_{i-1})$, without its endpoints, lies strictly to one side of the line through $v'_0v'_{i-1}$. See Fig. 8.

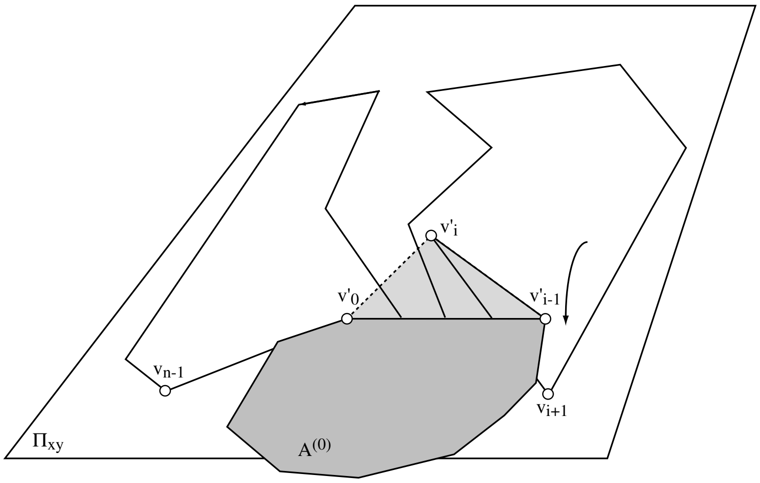

Figure 8: The arch A and $\triangle v'_0v'_{i-1}v'_i$ lie in plane Π_ϵ (not shown), floating ϵ above Π_{xy}.

S_3: reconvexifying the arch Consider a pass through S_1–S_4 that began by lifting v_i. At the start of S_3, the arch A lies in Π_ϵ, to one side of the line through $v'_0v'_{i-1}$, and v'_i lies either on this line, or on the opposite side of this line. Vertices v'_0 and v'_{i-1} will become the base points for the new arch A and are held fixed while reconvexification is performed.

If v_i lies on the line through the base points, S_3 resolves this degeneracy. There are two cases, depending on whether v'_i lies between v'_0 and v'_{i-1} or whether v'_{i-1} lies between v'_0 and

v_i'. (Assuming hypotheses H1–H3 held at the start of this pass through S_1–S_4, and assuming δ is chosen as described in H3, it is not possible for v_0' to lie between v_{i-1}' and v_i'.)

S_4 rotates edge e_{i-1}' about v_i' (held fixed) in such a way that v_{i-1}' moves to the open half-plane containing the other arch vertices. Edges e_{i-2}' and e_{i-1}' rotate slightly about their respective endpoints v_{i-3}' and v_{i-2}' to track the motion of v_{i-1}'.

If v_i' lay between v_0' and v_{i-1}', then the entire polygonal chain $P(v_0', \ldots, v_i')$ has become convex and S_4 is done.

In the other case, v_{i-1}' lay between v_0' and v_i'. Now the chain $P(v_0', \ldots v_{i-1}')$ is convex, and $\triangle v_0', v_{i-1}', v_i'$ is no longer degenerate. Adding a phantom edge $v_0'v_i'$ to $P(v_0', \ldots v_i')$ creates a simple "barbed" polygon. A "barbed" polygon is one that contains a diagonal that splits the polygon into two convex pieces, one of which is a triangle.

S_3 convexifies the barbed polygon. We do this by a simple inductive process. Choose a diagonal through v_0' that partitions the barbed polygon into a convex polygon and a quadrilateral. Reconfigure the quadrilateral to a triangle, with one edge containing a joint that has become straight. Freeze this joint. Should the triangle be degenerate, freeze both straight joints, and remove the degeneracy as above. This leaves a barbed polygon on fewer vertices. (See also [Sal73] and [ELR$^+$98]).

In case the convexification of the barbed polygon results in a polygon with more than two vertices on the line through $v_0'v_i'$, then S_4 rotates e_0' and/or e_{i-1}' a small amount about v_0' and/or v_i', respectively. If e_0' and/or e_i' is part of a longer edge containing joints that are frozen straight, then the entire frozen edge is rotated. This will restore H2 when the arch is lifted.

S_4: raising the arch S_4 rotates the convexified arch with base points v_0' and v_i' about the line through these points up into the vertical plane $\Pi_z(v_0', v_i')$. Because of strict convexity at v_0' and v_i', the remaining vertices of the arch move to positions above Π_ϵ.

4.3 ϵ and δ

Here we compute values for ϵ and δ such that the S_0-constraints will be satisfied and such that, whenever the hypotheses H1-H3 are satisfied at the beginning of a pass through S_1–S_4, the S_1-constraints on ϵ and δ incurred during that pass will be satisfied.

Let α_j be the angle at v_j that is less than π. Also, let $\beta_j = \pi - \alpha_j$, the deviation from straightness at joint v_j; so $\beta_j > 0$ for all j since by assumption, P has no straight joints.

The fact that P is simple and that each $\beta_j > 0$ guarantees that a $\delta > 0$ as described in H3 exists. It is a straightforward exercise to compute a δ. Our technical report gives a simple formula.

With δ chosen, let σ_{min} denote the minimum separation $|v_jv_k|$ for all positions of v_j and v_k within their δ disks, for all distinct j and k. Let β_{min} denote the minimum of all β_j for all positions of v_j within their δ disks. By i) and ii), σ_{min} and β_{min} are positive.

It is a straightforward exercise, based on examination of the double cone determined by the rotation of v_i about the line through its neighbors (see Fig. 7), to compute an ϵ value from σ_{min} and β_{min} so that the S_0-constraints are satisfied, and so that the S_1-constraints are satisfed under the assumption that H1-H3 hold.

4.4 Correctness and complexity

The St. Louis Arch Algorithm first computes a suitable ϵ and δ, as in Section 4.3, then performs the initialization process S_0, and then makes repeated passes through S_1–S_4 until all vertices have been lifted into a convex arch.

Theorem The St. Louis Arch Algorithm convexifies a planar simple polygon.

proof: We prove by induction that the hypotheses H1–H3 are satisfied at the beginning of each pass through S_1–S_4.

Clearly H1–H3 hold at the beginning of the first pass through S_1–S_4, which lifts $v_{i+1} = v_3$ to Π_ϵ, by the choice of δ and ϵ. If H1–H3 hold at the beginning of the pass through S_1–S_4 that lifts vertex v_i, for $3 \le i < n-1$, then, by the choice of ϵ and δ, H1–H3 also hold at the beginning of the next pass. For the last pass, which lifts v_{n-1}, the S_1-constraints are satisfied, and the algorithm terminates following the reconvexification procedure S_3.

Clearly collisions do not occur during initialization S_0. Using the fact that H1-H3 hold at the beginning of each pass through S_1–S_4, it is easily seen that no collisions occur during any of these proceedures. $\square$

Only a constant number of moves are used during each execution of procedures S_0, S_1, S_2, and S_4. Each execution of the reconvexification procedure S_3 can be done with $O(n)$ moves, so the resulting procedure can be accomplished by $O(n^2)$ moves. Jeff Erikson observed that the "freezing" of a joint that becomes straight during the reconvexification procedure S_3 can only be done n times. This implies, by an amortizationi argument, that the St. Louis Arch Algorithm makes $O(n)$ moves. Moreover, our technical report (([BDD$^+$99b]) argues that ϵ and δ and the entire sequence of moves can be computed in $O(n)$ time in the extended real RAM model of computation.

5 Open problems

Our results suggest several open questions, including:

1. Characterize the unlocked open and closed chains in 3D. What is the complexity of deciding whether a chain in 3D is unlocked?

2. The algorithm of Section 3.1 applies to chains with simple orthogonal projections to a plane. In what ways can this approach be generalized to other projections?

3. Can an open chain of unit-length links lock in 3D? Cantarella and Johnson show in [CJ98] that the answer is *no* if $n \le 5$.

Acknowledgements

We thank Jeff Erickson for the amortization argument and Hazel Everett for useful comments. The idea and terminology for the knitting needles example arose in conversation with William Lenhart.

References

[Ada94] C. C. Adams. *The Knot Book.* W. H. Freeman, New York, 1994.

[BDD+99a] 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, 2 pages.

[BDD+99b] 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. Archived by arXiv.org as arXiv:cs.CG/9910009 8 Oct. 1999 (see http://xxx.lanl.gov/help/general for information on how to access); also appeared as Smith College Technical Report TR060, 1999, 29 pages.

[BGRT96] P. Bose, F. Gomez, P. Ramos, and G. T. Toussaint. Drawing nice projections of objects in space. In *Graph Drawing (Proc. GD '95)*, volume 1027 of *Lecture Notes Comput. Sci.*, pages 52–63. Springer-Verlag, 1996.

[CKMST] J. Calvo, D. Krizanc, P. Morin, M. Soss, and G. Toussaint. Convexifying polygons with simple projections. *Information Processing Letters*, to appear.

[Can87] J. Canny. *The Complexity of Robot Motion Planning.* ACM – MIT Press Doctoral Dissertation Award Series. MIT Press, Cambridge, MA, 1987.

[CH88] G. Crippen and T. Havel. *Distance Geometry and Molecular Conformation.* Research Studies Press, Letchworth, UK, 1988.

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

[CO99] R. Cocan and J. O'Rourke. Polygonal chains cannot lock in 4d. In *Proc. 11th Canad. Conf. Comput. Geom.*, 1999. Extended abstract. Full version: LANL/CoRR paper cs.CG/9908005.

[Cro97] P. Cromwell. *Polyhedra.* Cambridge University Press, 1997.

[CSW95] F. Chin, J. Snoeyink, and C.-A. Wang. Finding the medial axis of a simple polygon in linear time. In *Proc. 6th Annu. Internat. Sympos. Algorithms Comput.*, volume 1004 of *Lecture Notes Comput. Sci.*, pages 382–391. Springer-Verlag, 1995.

[CDR00] R. Connelly, E. Demaine, and G. Rote. Straightening polygonal arcs and convexifying polygonal cycles. In *Proc. IEEE Symp. Foundations of Comput. Science (FOCS)*, Nov. 12-14, 2000, pages 432–442.

[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.

[Erd35] P. Erdős. Problem 3763. *Amer. Math. Monthly*, 42:627, 1935.

[GN86] C. C. Gibson and P. E. Newstead. On the geometry of the planar 4-bar mechanism. *Acta Applicandae Mathematicae*, 7:113–135, 1986.

[Grü95] B. Grünbaum. How to convexify a polygon. *Geombinatorics*, 5:24–30, July 1995.

[HJW84] J. E. Hopcroft, D. A. Joseph, and S. H. Whitesides. Movement problems for 2-dimensional linkages. *SIAM J. Comput.*, 13:610–629, 1984.

[HJW85] J. E. Hopcroft, D. A. Joseph, and S. H. Whitesides. On the movement of robotic arms in 2-dimensional bounded regions. *SIAM J. Comput.*, 14:315–333, 1985.

[Hun78] K. H. Hunt. *Kinematic geometry of mechanisms.* Oxford University Press, 1978.

[Kan97] V. Kantabutra. Reaching a point with an unanchored robot arm in a square. *Internat. J. Comput. Geom.*, 7(6):539–550, 1997.

[Kor85] J. U. Korein. *A Geometric Investigation of Reach.* ACM distinguished disserations series. MIT Press, 1985.

[KSW96] M. van Kreveld, J. Snoeyink, and S. Whitesides. Folding rulers inside triangles. *Discrete Comput. Geom.*, 15:265–285, 1996.

[Liv93] C. Livingston. *Knot Theory.* The Mathematical Association of America, 1993.

[LW92] W. J. Lenhart and S. H. Whitesides. Reconfiguration with line tracking motions. In *Proc. 4th Canad. Conf. Comput. Geom.*, pages 198–203, 1992.

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

[Nag39] B. de Sz. Nagy. Solution to problem 3763. *Amer. Math. Monthly*, 46:176–177, 1939.

[Sal73] G. T. Sallee. Stretching chords of space curves. *Geometriae Dedicata*, 2:311–315, 1973.

[SS83] J. T. Schwartz and Micha Sharir. On the "piano movers" problem II: General techniques for computing topological properties of real algebraic manifolds. *Adv. Appl. Math.*, 4:298–351, 1983.

[Str00] I. Streinu. A combinatorial approach to planar non-colliding robot arm motion planning. In *Proc. IEEE Symp. Foundations of Comput. Science (FOCS)*, Nov. 12-14, 2000, pages 443–453.

[Tou99] G. T. Toussaint. The Erdős-Nagy theorem and its ramifications. In *Proc. 11th Canad. Conf. Comput. Geom.*, 1999. Extended abstract.

[Whi92] S. H. Whitesides. Algorithmic issues in the geometry of planar linkage movement. *Australian Comput. J.*, pages 42–50, 1992.

[Whi97] W. Whiteley. Rigidity and scene analysis. In Jacob E. Goodman and Joseph O'Rourke, editors, *Handbook of Discrete and Computational Geometry*, chapter 49, pages 893–916. CRC Press LLC, Boca Raton, FL, 1997.