An Energy-Driven Approach to Linkage Unfolding

Jason H. Cantarella* — cantarel@math.uga.edu
Erik D. Demaine** — edemaine@mit.edu
Hayley N. Iben*** — iben@eecs.berkeley.edu
James F. O'Brien*** — job@eecs.berkeley.edu

*University of Georgia
**Massachusetts Institute of Technology
***University of California, Berkeley

Abstract

We present a new algorithm for unfolding planar polygonal linkages without self-intersection based on following the gradient flow of a "repulsive" energy function. This algorithm has several advantages over previous methods. (1) The output motion is represented explicitly and exactly as a piecewise-linear curve in angle space. As a consequence, an exact snapshot of the linkage at any time can be extracted from the output in strongly polynomial time (on a real RAM supporting arithmetic, radicals, and trigonometric functions). (2) Each linear step of the motion can be computed exactly in $O(n^2)$ time on a real RAM where n is the number of vertices. (3) We explicitly bound the number of linear steps (and hence the running time) as a polynomial in n and the ratio between the maximum edge length and the initial minimum distance between a vertex and an edge. (4) Our method is practical and easy to implement. We provide a publicly accessible Java applet [1] that implements the algorithm.

Categories and Subject Descriptors: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling

General Terms: Algorithms, Theory

Keywords: Carpenter's rule problem, linkage reconfiguration, unfolding, gradient flow, knot energy, computational geometry.

1 Introduction

1.1 Linkage Reconfiguration

Consider a planar linkage of rigid bars connected at flexible joints to form a collection of tangled but noncrossing arcs and cycles (polygonal chains). The linkage may move in any way that preserves the bar lengths and causes no two bars to cross. Figure 1 shows four frames from an example of such a motion.

1.2 Motivation

Arc and cycle linkages and their motions arise throughout science and engineering in a variety of contexts, including:

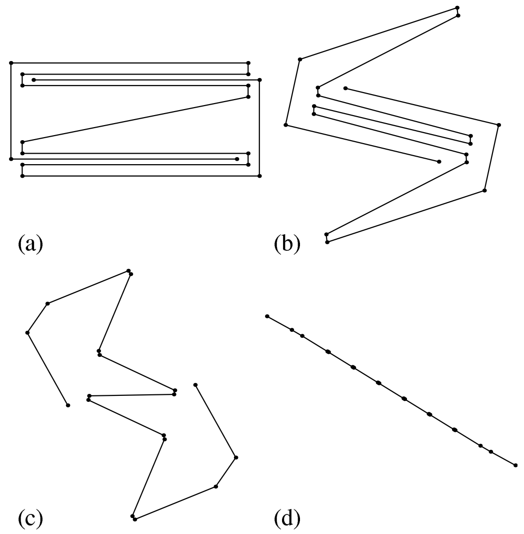


Figure 1. A sample unfolding of a polygonal arc produced by our algorithm. By following the gradient of a repulsive energy function, the linkage evolves from its initial configuration shown in (a), through a series of non-intersecting intermediate configurations represented by (b) and (c), to a final straight configuration (d). Throughout the motion all segments preserve their length, but the figure uniformly scales each configuration to fit in the same image area.

1. robotic-arm folding, where the goal is to fold the arm from one configuration to another;
2. hydraulic tube bending, where the goal is to manufacture a particular shape out of an initially straight tube;
3. protein folding, where the backbone of the protein can be modeled as an arc or cycle, and the goal is to understand how the amino acids quickly and precisely fold into a minimum-energy configuration; and
4. computer graphics, where the goal in key-frame animation is to smoothly interpolate between two shapes of an underlying skeleton (linkage).

In the past few years, tremendous progress has been made on understanding convexifying motions for arc and cycle linkages, specifically in FOCS 2000 [9, 12]. However, the algorithms behind these motions are relatively complex and slow. The goal of this paper is to improve this situation by presenting a simple and efficient method for computing convexifying motions of planar arc and cycle linkages.

1.3 Existence of Motions

A natural question asks for a characterization of the shapes into which a linkage can fold. The most fundamental version of this

SCG'04, June 8–11, 2004, Brooklyn, New York, USA.

question asks whether the linkage can fold into *every* non-self-intersecting configuration. In the context of arcs and cycles, this question is equivalent to whether the arcs can be straightened and the cycles can be convexified. This fundamental question has been resolved in all cases: every valid configuration is reachable for every arc or cycle in 2D [9, 12] and in 4D and higher dimensions [7], whereas not every configuration is reachable for some arcs and cycles in 3D [5, 6]. Intuitively, 4D chains have a "lot of space" (comparing the dimensions of the configuration space and the barriers preventing a motion), 3D chains can be geometrically "knotted" (but still topologically trivial), and 2D chains can be *expanded* until they unfold (consequently avoiding crossings).

1.4 Algorithms

In 4D, we have an essentially ideal situation: there are strongly polynomial-time algorithms to compute a polynomial number of succinctly describable moves (algebraic curves of constant degree) for an arc or cycle [7]. (*Strongly polynomial time* means that the running time on a real RAM is polynomial in the number n of vertices in the linkage, and independent of the bit complexity of the input.) In 3D, it is PSPACE-hard to decide whether a 3D arc can be folded from one configuration to another [3], though it remains open how quickly we can determine whether an arc can be straightened [5].

In contrast, the algorithmic side remains relatively undeveloped in 2D. The original 2D theorem of [9] is algorithmic but requires solving an ordinary differential equation where the right-hand side is defined implicitly by a convex optimization. This motion is "canonical", in particular preserving any symmetries present in the original linkage; it also expands all distances between pairs of vertices. Although the algorithm is finite for any specified output error tolerance (and even output error can likely be avoided), no time bounds have been established. The alternative approach of [12] gives a motion involving polynomially many algebraic motions of degree $\Theta(n)$. This motion is expansive and involves conceptually simple motions, but does not preserve symmetries in the linkage. Unfortunately, computing each algebraic motion requires exponential time and is accurate only up to a specified error tolerance. Nonetheless, that exponential bound is the current best time bound on any algorithm for this problem.

1.5 Our Results

In this paper, we introduce a novel energy-driven approach for straightening 2D arcs and convexifying 2D cycles that establishes stronger algorithmic, practical, and mathematical results.

On the algorithmic side, we obtain the first polynomial-time algorithm for linkage unfolding where the polynomial depends on n and geometric features of the initial configuration.[1] Specifically, the running time is $O(n^{79} r^{26})$ where r is the ratio of the maximum edge length over the minimum elliptic distance between a vertex and an edge in the initial configuration. (Elliptic distance is defined in Section 3.2.) In particular, if the input vertices are chosen from an integer $N \times N$ grid, then this time bound is *pseudopolynomial* in the sense that it is polynomial in n and N. This algorithm is also the first that outputs an explicit, exact representation of a motion, in the sense that an exact snapshot of the linkage at any time during the motion can be extracted from the output in strongly polynomial time. Specifically, the motion is piecewise-linear in angle space. Each linear step in the motion can be computed in $O(n^2)$ time, whereas previous approaches required linear programming or convex programming to compute even an infinitesimal motion, which take weakly polynomial time. The running time of the algorithm is strongly polynomial in the *output size* (n times the number of steps in the output motion), and we prove that the output size is polynomial in n and the geometric features mentioned above.

On the practical side, our algorithm is simple and easy to implement, involving a straightforward computation of the gradient of an energy function. We have implemented the algorithm as a Java applet [1] and in C++. Our timings indicate that our algorithm runs dramatically faster than an implementation of [9]. (The algorithm of [12] has not been implemented to our knowledge.) The algorithm is inspired by a natural physical process, in which vertices repel edges (and vice versa) as if they all were objects with similar electrostatic charges.

On the mathematical side, our techniques construct a natural C^∞ unfolding motion. In contrast, the motions of [9] and [12] are piecewise-C^1 and piecewise-C^∞, respectively. Our motions are not always expansive, but this seems key to achieving our results.

1.6 Overview

The basic idea of our approach is to define an *energy function* on the configurations of a linkage, satisfying four properties:

1. expansive motions decrease energy;
2. the energy is infinite when the linkage crosses itself;
3. the energy is minimum when the linkage is in the desired configuration (straight or convex);
4. as two connected components of the linkage grow in distance, their interaction energy decreases.

The first property, together with the existence of expansive motions [9], establishes the existence of motions that decrease energy. We follow the negative gradient flow to find a motion that decreases energy. The second property implies that this energy-decreasing motion will avoid self-intersection. The third property along with the existence of energy-decreasing motions implies that we eventually reach the desired configuration. The fourth property prevents multiple components from flying apart from each other so quickly that they never actually straighten or convexify.

We begin in Section 2 with background and definitions. Then in Section 3 we define the precise constraints we need of an energy function and give examples of such energy functions. Section 4 establishes the main mathematical result, that gradient flow produces the desired smooth motion. Section 5 describes the algorithm to find an exact piecewise-linear motion and proves that its running time is finite. Section 6 gives explicit bounds on the running time in terms of n and geometric features of the input. Section 7 describes experiments with an implementation of our approach, and shows the resulting animations and running times. We conclude in Section 8

2 Background: Arc-and-Cycle Sets

We now define the objects of interest. An *arc-and-cycle set* A is a finite collection of planar polygonal arcs and polygonal closed curves. A *configuration* $V = [v_1, v_2, \dots]$ of A is an assignment of coordinates to vertices such that the edge lengths match those in A. If A has n vertices, the *configuration space* of A, denoted $X(A)$, can be viewed as the algebraic subvariety of $\mathbb{R}^{2n}$ determined by fixing the length of each edge. The *embedded* configurations of A—configurations without self-crossing—are denoted $EX(A)$.

A configuration of an arc-and-cycle set is *outer-convex* if each outermost connected-component of A is either straight (when it is

[1] Our model of computation is a real RAM supporting $+, -, \times, \div, \sqrt{\ }$, sin, and arcsin.

an arc) or convex (when it is a cycle). A motion of a configuration is *strictly expansive* if it does not decrease any vertex-to-vertex distance, and strictly increases all of the vertex-to-vertex distances between pairs of vertices that are not forced to have constant distance because they are connected by a straight chain of edges or because they are on or inside a common convex cycle. A motion is merely *expansive* when it does not decrease any vertex-vertex distance, and increases at least one such distance.

The main result of [9] establishes the existence of such motions, which we use extensively:

THEOREM 1. *Any arc-and-cycle set admits a strictly expansive motion until it is outer-convex.*

3 Energy Functions

Next we consider energy functions whose minimization forces the linkage to "repel itself". The gradient of any such function will then define a motion of the linkage towards an outer-convex configuration that avoids crossings as desired.

3.1 Definition and Required Properties

An *energy function* is a function from embedded configurations $EX(A)$ to the nonnegative real numbers $\mathbb{R}^+$. We call an energy function *admissible* if it has four properties defined below: it must be C^2, charge, repulsive, and separable. (We can define a version of admissibility for $C^{1,1}$ functions instead of C^2, but it is much harder to work with.)

3.1.1 Charge

An energy function E is *charge* if it approaches $+\infty$ on the boundary of $EX(A)$, that is, if it becomes infinite as the linkage approaches any self-crossing configuration.

This requirement is an adaptation of an idea from the literature of knot energies (cf. [10]) to capture the idea that our energy functional must avoid self-crossing configurations. The inspiration for the name "charge" comes from electrostatics, where it takes an infinite amount of work to pull a pair of point charges together until they coincide.

3.1.2 Repulsive

An energy function E is *repulsive* if it decreases to first order under any strictly expansive motion of A.

This requirement captures the idea that the vertices and edges of the linkage should roughly repel each other under the gradient flow of the energy.

3.1.3 Separable

For an arc-and-cycle set A with connected components $A_1, \dots,$ A_n, an energy function E is *separable* if it can be written in the form

$$E(A) = \sum_{i,j=1}^{n} E_{ij}(A_i, A_j), \tag{1}$$

where each *two-component energy* E_{ij} is an energy function on the arc-and-cycle set $A_i \cup A_j$ that itself is C^2, repulsive, and charge; and furthermore the contribution of E_{ij} to the gradient of E approaches zero as the distance between A_i and A_j grows.

This requirement enforces that, as connected components of A become far away from each another, the repulsion between them has little impact on the gradient of the energy.

3.2 Example

We now give an example of an energy function that obeys our criteria. The basic idea is to sum powers of reciprocals of distances between vertices and edges of the arc-and-cycle set. This idea immediately leads to the charge property: as a distance approaches zero, the reciprocal approaches $+\infty$. We use a particular definition of distance between a vertex and edge so that the energy function is C^∞.

Specifically, the *elliptic-distance energy* of an arc-and-cycle set A with vertex set V and edge set E is defined by

$$E(A) := \sum_{\substack{\text{edge } \{v,w\} \\ \text{vertex } u \notin \{v,w\}}} \frac{1}{(\|u-v\| + \|u-w\| - \|v-w\|)^2}. \tag{2}$$

where the denominator is the squared *elliptic distance* between vertex u and edge $\{v, w\}$. For any edge $\{v, w\}$, the level sets of the summand in the elliptic-distance energy, as we vary the position of vertex u, are a family of ellipses with foci at v and w which converge at zero to the edge $\{v, w\}$.

PROPOSITION 1. *Elliptic-distance energy is admissible.*

PROOF. This energy is C^∞ on the interior of $EX(A)$ and is therefore also C^2. Because the denominator of the summand vanishes precisely when vertex u is on the edge $\{v, w\}$, the energy is charge. Any expansive motion cannot increase any of the summands, and it must increase a positive term in at least one of the denominators, while leaving all negated terms alone. Thus the energy is repulsive. Finally, because we can split the sum up according to which connected-component of A the edge $\{v, w\}$ and the vertex u belong to, while the derivative of the summand approaches zero as the distances $\|u - v\|$ and $\|u - w\|$ become large, the energy is separable. □

4 Gradient Flow Almost Unfolds Linkages

This section proves our main mathematical result: for any $\epsilon > 0$, the negative gradient flow of any admissible energy functional moves any linkage configuration to within distance ϵ of an outer-convex configuration in finite time.

4.1 Existence of Gradient Flow

We first observe that the gradient flow is well-defined:

PROPOSITION 2. *Given any embedded arc-and-cycle set A, the downhill gradient flow $A(t)$ of A under any admissible energy function E exists for all time $t \geq 0$ and is as smooth (in t) as the energy function E (in space).*

PROOF. Because energy only decreases under gradient flow, we can restrict to the closed subspace $EX^+(A)$ of $EX(A)$ where $E \leq E(A)+1$. Because E is C^2, the integral curve $V(t)$ of $-\nabla E$ through A exists for all time, unless it approaches the boundary of this space. But energy approaches $+\infty$ along the boundary and energy strictly decreases along the path, so this cannot happen. □

4.2 Main Theorem

We now prove our main theorem:

THEOREM 2. *If A is an arc-and-cycle-set and E is an admissible energy function on $EX(A)$, then for any $\epsilon > 0$ the motion $A(t)$ defined by the downhill gradient flow of E carries $A(0)$ to within ϵ of an outer-convex configuration in finite time.*

PROOF. A standard result in dynamical systems says that any trajectory of the negative gradient flow $A(t)$ either weakly converges to some configuration of A that is critical for E or $A(t)$ leaves any compact neighborhood of $A(0)$ in finite time.

Because E is repulsive, Theorem 1 implies that any critical configuration of A is outer-convex. So in the first case there is nothing more to prove.

We focus on the second case. We can split A into n sublinkages $A_i(t)$, so that the components of each A_i remain within a bounded distance of one another for all time. In this case $A_i(t)$ remains within a compact subspace of $EX(A_i)$. We define a compact subspace of this space by restricting our attention to the space $EX^+(A_i)$ of configurations with $E_{ii} \leq E(A(0)) + 1$. Here we have used separability of E to write $E(A) = \sum_{i,j} E_{ij}(A_i, A_j)$ where each E_{ij} is a C^2, repulsive, charge energy function on $EX(A_i \cup A_j)$.

Now removing an ϵ-neighborhood of the outer-convex configurations leaves a subspace S_i on which $\|\nabla E_{ii}\|$ is bounded below by some $G_i > 0$, because this removes a neighborhood of the critical configurations for E_{ii} (by Theorem 1 and because E_{ii} is repulsive).

Because the A_i are drifting further apart, and E is separable, for each E_{ij} there is some finite time after which each $\|\nabla E_{ij}\| < G_i/2n$. After this point, the gradient flow of E must reduce each E_{ii} at rate at least $G_i/2$. But each $E_{ii}(A_i(t))$ is finite at this point and must always be non-negative, so for all t greater than some t_i, $A_i(t)$ must be outside S_i.

By definition, the complement of S_i consists of configurations with $E_{ii} > E(A_i(0))$ and configurations within ϵ of an outer-convex configuration. But $E_{ii}(A_i(t_i)) < E(A_i(0))$, so we must be in the second case: $A_i(t)$ is close to an outer-convex configuration for $t > t_i$. So for any $t > \max_i t_i$, $A(t)$ is close to an outer-convex configuration, completing the proof. □

5 Algorithm

This section presents an algorithm for computing a piecewise-linear motion from an initial configuration to an outer-convex configuration. This path is computed by first selecting a particular admissible energy function, expressing the energy function in terms of a suitable parameterization, and then applying Euler integration along the downward gradient path to get a series of "snapshots" of our linkage with decreasing energy which can be joined by linear interpolation in our parameter space. The algorithm terminates when we are sufficiently close to an energy-critical configuration to complete the motion by linear interpolation. As shown in Section 4, any critical configuration is guaranteed to correspond to an outer-convex configuration as desired.

5.1 Parameterizing the Configuration Space of an Arc

We start by considering the case when A consists of a single arc of $n-1$ edges. Refer to Figure 2(a). Let $V = [v_1, v_2, \ldots, v_n]$ denote the positions of the n vertices and let e_i denote the edge between vertices v_i and v_{i+1}. We parameterize the system by $\Theta = [\theta_1, \theta_2, \ldots, \theta_{n-1}]$ where θ_i measures the angle between edge e_i and the x axis. The locations of the other vertices are given by

$$v_{i+1} = v_i + \ell_i[\cos\theta_i, \sin\theta_i], \quad i \in \{1, \ldots, n-1\}, \tag{3}$$

where ℓ_i is the constant length of the edge e_i, and v_1 is arbitrarily set to the origin. If we wish, we may also assume θ_1 to be zero.

The major virtue of this parameterization is that it is exact: any set of parameter values corresponds precisely to a linkage configuration in $X(A)$, and linear interpolation between two "snapshot" positions in angle space yields a one-parameter family of exact linkage configurations joining snapshots.

We can define a norm of the angle parameterization as follows. If $\Theta' = [\theta'_1, \ldots, \theta'_{n-1}]$, then

$$\|\Theta - \Theta'\| = \sum_i \min\{|\theta_i - \theta'_i|, 2\pi - |\theta_i - \theta'_i|\} \tag{4}$$

This norm is different from the norm on $X(A)$ as a subvariety of $\mathbb{R}^{2n}$: a small angular move is magnified by the length of the edge it turns. However, we can relate the two norms as follows. Let $\ell_{\max}$ be the maximum edge length, $\max_i \ell_i$. Let $V' = [v'_1, v'_2, \ldots, v'_n]$ denote the point parameterization of the configuration represented by angle parameterization Θ'. Then

$$\|V - V'\| < n^2 \ell_{\max} \|\Theta - \Theta'\|. \tag{5}$$

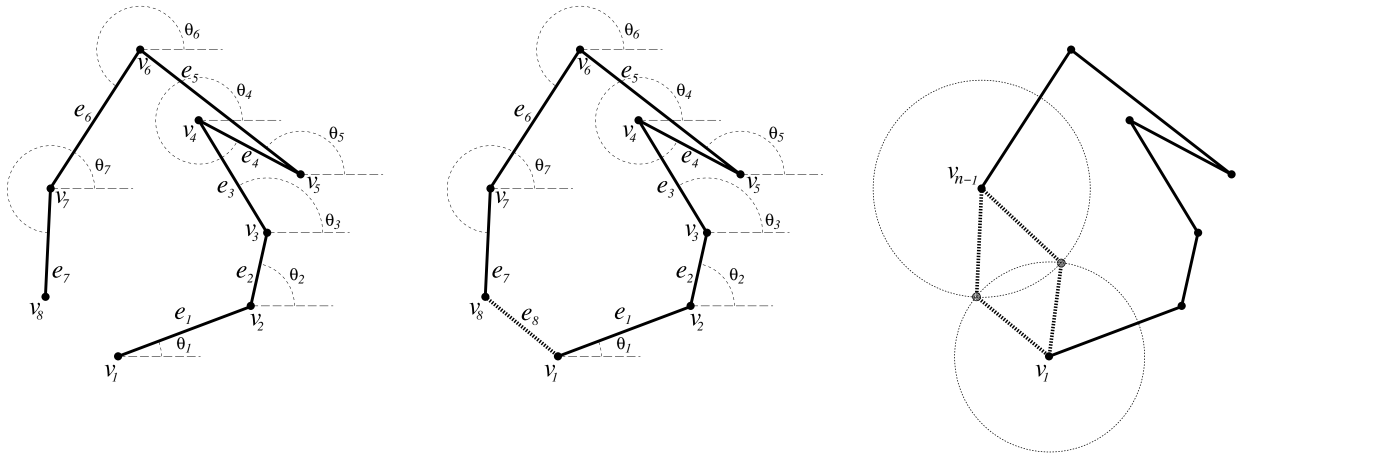
5.2 Parameterizing Cycles

For cycles, the situation is more complicated: we must change our parameterization to ensure that the length of the closing edge $e_n = \{v_n, v_1\}$ is preserved. Refer to Figure 2(b)–2(c). We remove θ_{n-1} from the set of independent variables, and determine it in terms of the other independent variables, $[\theta_1, \theta_2, \ldots, \theta_{n-2}]$, by computing the location of v_n as the intersection of the circle of radius ℓ_n centered at v_1 with the circle of radius ℓ_{n-1} centered at v_{n-1}. We can compute v_n with

$$v_n = v_1 + d\frac{l_n^2 - l_{n-1}^2 + \|d\|^2}{2\|d\|^2} \pm d^\perp \sqrt{\frac{l_n^2}{\|d\|^2} - \frac{(l_n^2 - l_{n-1}^2 + \|d\|^2)^2}{4\|d\|^4}} \tag{6}$$

where $d = v_{n-1} - v_1$ and $\cdot^\perp$ denotes rotation by 90 degrees. Because v_1 and v_{n-1} are not co-located (no self-intersections), there will be zero, one, or two real solutions for v_n depending on whether $\|v_1 - v_{n-1}\|$ is greater than, equal to, or less than $\ell_{n-1} + \ell_n$. When there are two possible solutions, one will cause v_n to be a convex vertex, and the other will cause v_n to be reflex.

We arrange for there always to be two solutions and choose among those solutions by the following procedure. At the initial configuration A_0, we let v_n be the vertex of maximum absolute turn angle, and use this to define an angle-space parameterization Θ_0. Now any closed n-sided polygon has a vertex whose absolute turn angle is at least $2\pi/n$, so we may assume the absolute turn angle at v_n is at least $2\pi/n$. If the polygon has minimum edge length $\ell_{\min}$, a calculation reveals that v_n remains convex or reflex in all configurations V' with $\|V_0 - V'\| < 2\ell_{\min}/n$ (in the vertex-space norm). So if the next angle-space position is Θ_1, and $\|\Theta_0 - \Theta_1\| < 2\ell_{\min}/(n^3\ell_{\max})$, then by Equation 5 there will still be two real solutions for v_n and maintaining the convex/reflexness of v_n will let us interpolate continuously between Θ_0 and Θ_1. Then in $O(n)$ time we choose a new angle-space parameterization of A_1 so that v_n is again the vertex of maximum absolute turn angle in A_1 and continue. Iterating this procedure yields a well-defined angle-space parameterization for any snapshot A_i, and retains the property that linear interpolation between these angle space positions yields a one-parameter family of exact linkage configurations joining snapshots, as long as the vertex-space distance between successive configurations remains less than $2\ell_{\min}/n$.



(a) Open chain. (b) Closed chain. (c) Forcing closure in a closed chain.

Figure 2. Parameterization of chains in terms of angles.

5.3 Computing the Gradient of Energy at a Configuration

Simple inspection shows that the elliptic energy function requires computing $O(n^2)$ terms. Naïve computation of the gradient in angle space for an arc would require computing the derivative of those $O(n^2)$ terms with respect to each of the $O(n)$ parameterization variables, for a total cost $O(n^3)$. However, there are many common subexpressions and after some algebraic manipulation the total work to compute the gradient can be reduced to $O(n^2)$. (We omit the details.)

For closed cycles, the contribution of θ_n is distributed to the rest of Θ by applying the chain rule to Equations 3 and 6. We use this gradient for our theoretical results, but from a practical standpoint it is numerically inefficient. The partials of Equation 6 can over emphasize the motion of v_n, slowing convergence. Instead we can compute the gradient of the closed-chain energy using the open parameterization with edge e_n accounted for by explicit constraint projection. We then discard the gradient term for θ_{n-1} and determine v_n with Equation 6. This procedure still preserves all edge-lengths exactly, but is numerically more efficient.

5.4 Picking Step Size to Avoid Self-Intersection

Before we can generate snapshots by following the gradient, we must show that we can choose step sizes to ensure that we can linearly interpolate between snapshot configurations while avoiding self-intersections. Suppose our initial configuration has energy E. Because the energy functional is charge, the Euclidean distance between the compact set of configurations with energy $\leq E$ and the compact set of non-embedded configurations of A is strictly larger than some $D_s > 0$. By Equation 5, we obtain a corresponding distance bound $D_s/(n^2\ell_{\max}) > 0$ in angle space. (In Section 6 we explicitly compute these bounds for elliptic-distance energy.)

We use two consequences of this fact. First, if the energy decreases monotonically on the sequence of snapshots Θ_i, and the distance between successive snapshots Θ_i and Θ_{i+1} is less than $D_s/(n^2\ell_{\max})$, then the path of exact configurations interpolating between the snapshots avoids self-intersection. Second, if any snapshot is within $D_s/(n^2\ell_{\max})$ of an outer-convex configuration, then the algorithm may terminate: we can move to the outer-convex configuration by linear interpolation and this motion avoids self-intersection.

5.5 Generating Snapshot Configurations

By Theorem 2, the negative gradient flow of any admissible energy moves every arc-and-cycle set to an outer-convex configuration. We now demonstrate a discretized version of this flow which generates a piecewise-linear path $\Theta_0, \Theta_1, \ldots, \Theta_K$ to an outer-convex configuration in a bounded number of steps. We generate this path by using Euler integration to trace the streamline in the gradient field downward from Θ_0. Because Euler integration will accumulate positional error as it advances, our path may diverge substantially from the true streamline, and the two only converge as the step-size approaches zero. Regardless of how well the discrete path matches the streamline, it is constructed so that is still arrives at an outer-convex configuration in a bounded number of steps.

Our primary goal is to choose our steps so that $E(\Theta_i) - E(\Theta_{i+1}) > \Delta E > 0$ for some ΔE. Once we can establish such a bound on energy decrease, our algorithm will terminate after at most $E(\Theta_0)/\Delta E$ steps because the energy is initially $E(\Theta_0)$ and is always nonnegative.

As in the proof of the main theorem, we can restrict our attention to embedded configurations Θ_i whose energy is at most $E(\Theta_0)$ and whose distance to an outer-convex configuration is at least $D_s/(n^2\ell_{\max})$. Such configurations form a compact subset S of $EX(A)$. Because ∇E can vanish only on outer-convex configurations, by compactness there are positive constants G and C so that $\|\nabla E\| > G$ and $\|\nabla^2 E\| < C$.

Define $\Theta_{i+1} = \Theta_i - \Delta t \cdot \nabla E(\Theta_i)/\|\nabla E(\Theta_i)\|$. Then we can expand $E(\Theta_{i+1})$ using Taylor's Theorem around Θ_i:

$$\begin{aligned} E(\Theta_{i+1}) &= E(\Theta_i) - \Delta t \|\nabla E(\Theta_i)\| \\ &+ \frac{1}{2}(\Delta t)^2 \nabla^2 E(\Theta_i - (\Delta t)^* \nabla E)\|u\|^2. \end{aligned}$$

for some $0 \leq (\Delta t)^* \leq \Delta t$ and where u is the unit vector given by $-\nabla E(\Theta_i)/\|\nabla E(\Theta_i)\|$. If $\Delta t < G/C$, then the first-order term is at least twice the second-order term, so the decrease in energy ΔE is at least $(\Delta t)(G/2)$.

We now have three distinct a priori upper bounds on Δt: $2\ell_{\min}/(n^3\ell_{\max})$, $D_s/(n^2\ell_{\max})$, and G/C. The minimum U of all three of these bounds is the largest allowed step size.

A basic form of our algorithm is as follows:

1. Set $\Delta t := U$, $k := 0$, and Θ_0 to the angle parameterization of A.
2. Until Θ_k leaves S:
   (a) Compute the gradient ∇E at Θ_k.
   (b) Set $\Theta_{k+1} := \Theta_k - \Delta t \cdot \nabla E(\Theta_k)/\|\nabla E(\Theta_k)\|$
   (c) In the output motion, linearly interpolate from Θ_k to Θ_{k+1}.
   (d) Recompute the angle-space parameterization so that v_n has maximum absolute turn angle.
3. In the output motion, linearly interpolate from Θ_k to the closest outer-convex configuration.

The discussion above and our choice of Δt proves that the motion avoids self-intersection and that the algorithm terminates after at most $2E(\Theta_0)/(G\Delta t)$ steps.

In practice, this gradient descent can be implemented in many more efficient ways, although it is difficult to obtain stronger worst-case bounds. For example, instead of moving at a distance Δt along the gradient direction, we can perform binary search around U to find the $\Delta t < \min\{\ell_{\min}/(n^2\ell_{\max}\pi), D_s/(n^2\ell_{\max})\}$ that decreases energy the most (steepest descent). This approach is taken by our implementations. Although it is easy to show that the number of steps is no more than the straightforward algorithm, the worst-case bound remains the same. Another more sophisticated approach, conjugate gradient, likely converges even faster, but we have not yet experimented with it. We note that by the analysis above, any method of choosing steps which decreases energy and respects the step-size bounds required for valid linear interpolation is an unfolding algorithm.

6 Bound on Number of Steps

In this section we give explicit bounds on the number of steps taken by the algorithm described in the previous section for elliptic-distance energy on an arc or cycle linkage. Our bound is in terms of the following geometric parameters of the configuration Θ_i:

1. $\ell_{\max}$: maximum edge length, $\max_i \ell_i$.
2. $d_{\min}(\Theta_i)$: minimum elliptic distance between a vertex and an edge, $\min_{i,j}(\|v_i - v_j\| + \|v_i - v_{j+1}\| - \|v_j - v_{j+1}\|)$.
3. $w(\Theta_i)$: width of the linkage, i.e., the minimum width of a strip, bounded by two parallel lines, that contains the linkage.

We also define more convenient forms for two of the parameters: $L(\Theta_i) = \max\{1, \ell_{\max}(\Theta_i)\}$ and $D(\Theta_i) = \min\{1, d_{\min}(\Theta_i)\}$.

THEOREM 3.
$\|\nabla E(\Theta_i)\| \geq d_{\min}(\Theta_i) w^3(\Theta_i)/(5328\, n^{8.5}\ell_{\max}^6)$.

PROOF. Recall that $\|\nabla E(\Theta_i)\|$ is the rate at which the energy E decreases under normalized gradient motion in the direction $-\nabla E(\Theta_i)/\|\nabla E(\Theta_i)\|$. We bound this quantity by first proving a lower bound on the energy decrease under any normalized expansive infinitesimal motion. Then the result follows because the normalized gradient motion must decrease energy to the first order faster than any other normalized motion.

Consider a normalized expansive infinitesimal motion defined at time $t = 0$ that fixes the edge $e_1 = \{v_1, v_2\}$. Observe that the diameter of the arc or cycle linkage, i.e., the maximum distance between two vertices, is at most $n\ell_{\max}$. Then [8, Lemma 15] tells us that

$$\max_i \left\|\frac{dv_i}{dt}\right\|_{t=0} \leq 666\left(\frac{n\ell_{\max}}{w(\Theta_i)}\right)^3 \sum_{j,k} \left.\frac{d\|v_j - v_k\|}{dt}\right|_{t=0},$$

i.e.,

$$\sum_{j,k} \left.\frac{d\|v_j - v_k\|}{dt}\right|_{t=0} \geq \frac{w^3(\Theta_i)}{666\, n^3\ell_{\max}^3} \max_i \left\|\frac{dv_i}{dt}\right\|_{t=0}.$$

Because each term in the sum is nonnegative, we know that some term in the sum, say j, k, is at least the average. The number of terms is at most n^2. Thus

$$\left.\frac{d\|v_j - v_k\|}{dt}\right|_{t=0} \geq \frac{w^3(\Theta_i)}{666\, n^5\ell_{\max}^3} \max_i \left\|\frac{dv_i}{dt}\right\|_{t=0}. \quad (7)$$

Now we consider the first-order change in energy under this motion. Because the motion is expansive, no term in the energy function increases. Thus the absolute first-order change in energy is at least the absolute first-order change in a term involving $\|v_j - v_k\|$. Suppose $\{v_i, v_j\}$ is a bar incident to v_j but not v_k. (If such a bar does not exist, v_j is an end of a chain and v_k is its neighbor; we interchange the labels of v_j and v_k and then such a bar exists.) Then we have

$$\begin{aligned}
\left.\frac{dE}{dt}\right|_{t=0} &\leq \left.\frac{d(\|v_k - v_i\| + \|v_k - v_j\| - \ell_{ij})^{-2}}{dt}\right|_{t=0} \\
&= \left(\underbrace{\left.\frac{d\|v_k - v_i\|}{dt}\right|_{t=0}}_{\geq 0} + \underbrace{\left.\frac{d\|v_k - v_j\|}{dt}\right|_{t=0}}_{\text{Eq. 7}} - \underbrace{\left.\frac{d\ell_{ij}}{dt}\right|_{t=0}}_{=0}\right) \\
&\quad \cdot (-2) \cdot (\|v_k - v_i\| + \|v_k - v_j\| - \ell_{ij})^{-3} \\
&\leq \frac{-w^3(\Theta_i)}{333\ell_{\max}^3 n^5 d_{\max}^3(\Theta_i)} \cdot \max_m \left\|\frac{dv_m}{dt}\right\|_{t=0},
\end{aligned}$$

where $d_{\max}(\Theta_i)$ is the maximum elliptic distance between a vertex and an edge in Θ_i. We can upper bound elliptic distances in terms of vertex-vertex distances using the triangle inequality:

$$\begin{aligned}
&\|v_k - v_i\| + \|v_k - v_j\| - \ell_{ij} \quad (8) \\
\leq\ &\|v_k - v_i\| + \|v_k - v_i\| + \|v_i - v_j\| - \ell_{ij} \\
=\ &2\|v_k - v_i\|.
\end{aligned}$$

Thus, $d_{\max}(\Theta_i)$ is at most twice the maximum distance between two vertices, which was earlier observed to be at most $n\ell_{\max}$. So $d_{\max}(\Theta_i) \leq 2n\ell_{\max}$.

Next we bound $\max_m \|dv_m/dt\|_{t=0}$. Because the expansive motion is normalized, $\sum_m \|dv_m/dt\|_{t=0}^2 = 1$, and so we have $\max_m \|dv_m/dt\|_{t=0}^2 \geq 1/n$. This result in turn tells us that $\max_m \|dv_m/dt\|_{t=0} \geq 1/\sqrt{n}$.

Combining all bounds, we obtain that

$$\left.\frac{dE}{dt}\right|_{t=0} \leq \frac{-w^3(\Theta_i)}{2664\, n^{8.5}\ell_{\max}^6}.$$

As described above, this bound on energy decrease holds also of the normalized gradient motion over point space, $-\nabla E(V)/\|\nabla E(V)\|$. To convert this derivative from point space to angle space, we use the chain rule twice—once to convert from vertex space $V = [v_1, v_2, \ldots, v_n]$ to real-vertex space $W = [v_1, v_2, \ldots, v_{n-1}]$, and again to convert from real-vertex space W to angle space Θ:

$$\frac{\partial E}{\partial \Theta} = \frac{\partial E}{\partial V} \cdot \frac{\partial V}{\partial W} \cdot \frac{\partial W}{\partial \Theta}.$$

The first term $\partial E/\partial V$ is what we already bounded: $dE/dt|_{t=0}$.

The second term $\partial V/\partial W$ is a Jacobian providing a scale factor between vertex space V and real-vertex space W. The $2(n-1) \times$

$2(n-1)$ submatrix $[\partial v_i/\partial v_j]_{i,j<n}$ is an identity matrix. The rest of the Jacobian is just two additional columns which can only increase the scale factors.

The third term $\frac{\partial W}{\partial \Theta}$ is a Jacobian providing a scale factor between point space and angle space. Each entry $\frac{\partial v_i}{\partial \theta_j}$ is a vector whose length matches the bar controlled by angle θ_j. Thus, each $\frac{\partial v_i}{\partial \theta_j}$ is at least $\ell_{\min}$, the length of the shortest bar. By Equation 8, $\ell_{\min} \geq \frac{1}{2} d_{\min}(\Theta_i)$.

Thus, $\partial E/\partial \Theta \geq \frac{1}{2} d_{\min}(\Theta_i)(\partial E/\partial V)$ and the theorem follows. □

THEOREM 4.
$\|\nabla^2 E(\Theta_i)(u,u)\| \leq 61920\, n^6 L^7(\Theta_i)/D^{12}(\Theta_i)$.

The proof of this bound is essentially a much more tedious computation along the lines of Theorem 3. One of the main challenges is that the relation $\partial V/\partial W$ between vertex space V and real-vertex space W must be bounded above. We omit the details from this abstract.

These bounds are almost all we need. However, we are interested in the values of $d_{\min}$ and w at the initial configuration Θ_0, $d_{\min}(\Theta_0)$ and $w(\Theta_0)$, not their values at some intermediate configuration Θ_i. Fortunately, we can bound the change of these parameters. ($\ell_{\min}$ does not change.)

LEMMA 1. *The elliptic-distance energy of any configuration Θ_i is at most $n^2/d_{\min}(\Theta_0)$.*

PROOF. The energy of any Θ_i is at most the initial energy $E(\Theta_0)$. There are at most n^2 terms in the energy expression $E(\Theta_0)$, and each term is at most $1/d_{\min}(\Theta_0)$. □

LEMMA 2. *For any configuration Θ_i, $d_{\min}(\Theta_i) \geq d_{\min}(\Theta_0)/n^2$.*

PROOF. By Lemma 1, $E(\Theta_i) \leq n^2/d_{\min}(\Theta_0)$. Hence the maximum term in $E(\Theta_i)$ is at most $n^2/d_{\min}(\Theta_0)$, so the minimum elliptic distance between a vertex and an edge in Θ_i is at least $d_{\min}(\Theta_0)/n^2$. □

LEMMA 3. *The Euclidean distance between any valid configuration Θ_i and any self-intersecting configuration is at least $D_s = d_{\min}(\Theta_0)/(2n^2)$.*

PROOF. By Lemma 2, the minimum elliptic distance between a vertex and an edge in Θ_i is at least $d_{\min}(\Theta_0)/n^2$. Now for any ellipse with foci e_1 and e_2, the closest points on the ellipse to the line segment joining the foci are the endpoints of the major axis. But at these points, this distance is half of the elliptic distance. Thus the minimum (Euclidean) distance between any vertex of Θ_i and any edge not incident to that vertex is $d_{\min}(\Theta_0)/(2n^2)$, and some vertex of Θ_i must move at least this far to cause a self-intersection. □

LEMMA 4. *For any configuration Θ_i whose angle-space distance to an outer-convex configuration is at least $D_s/(n^2\ell_{\max})$, the width $w(\Theta_i)$ is at least $2d^2_{\min}(\Theta_i)/(n\ell_{\max})$.*

PROOF. First we argue that some vertex in the linkage has absolute turn angle bounded away from 0. If the linkage contains a cycle, then at least one vertex v_i has absolute turn angle T at least $2\pi/n$. If the linkage consists only of arcs, let T be the maximum absolute turn angle of all vertices (excluding endpoints of arcs). The linkage has angular distance at most Tn^2 from an outer-convex configuration, because the absolute angle θ_k of each edge e_k needs to rotate at most Tn to reach the same angle as the first edge in that arc (and hence straighten). Hence, $Tn^2 \geq D_s/(n^2\ell_{\max})$. Thus, the absolute turn angle at some vertex v_i is at least $D_s/(n^5\ell_{\max})$. Therefore, in either case, we have a vertex v_i whose absolute turn angle T is at least $\min\{2\pi/n, D_s/(n^5\ell_{\max})\}$.

Consider the two neighbors v_{i-1} and v_{i+1} of v_i that form the angle at v_i. The width of the linkage is at least the width of the triangle formed by these three vertices v_{i-1}, v_i, v_{i+1}, which in turn is at least twice the in-radius of the triangle. The in-radius of the triangle is the area divided by half the perimeter. The perimeter is at most $4\ell_{\max}$. It remains to prove a lower bound on the area of the triangle.

If $T \leq \pi/2$, then the interior angle at v_i is between $\pi/2$ and $\pi - 2\pi/n$. Suppose among v_{i-1} and v_{i+1} that v_{i+1} has the larger interior angle in the triangle. Then the interior angle θ of v_{i+1} is between π/n and $\pi/2$. Because $\theta \leq \pi/2$, $\sin\theta \geq 2\theta/\pi \geq 2/n$. The altitude from v_i is $\|v_i - v_{i+1}\|\sin\theta \geq 2\ell_{\min}/n \geq d_{\min}/n$ by Equation 8. The base of this altitude is $\|v_{i-1} - v_{i+1}\| \geq d_{\min}/2$ by Equation 8. Thus the area is at least $d^2_{\min}/(2n)$ in this case.

If $T \geq \pi/2$, then the altitude of one of the other vertices, say v_{i-1}, is inside the triangle. Hence the altitude from v_{i-1} is also the Euclidean distance between vertex v_{i-1} and edge $\{v_i, v_{i+1}\}$. By Lemma 3, this distance is at least D_s. The base of this altitude is $\|v_{i+1} - v_i\| \geq \ell_{\min} \geq d_{\min}/2$ by Equation 8. Thus the area is at least $D_s d_{\min}/2$ in this case.

Hence in either case the area of the triangle is at least $\min\{D_s, d_{\min}(\Theta_i)\} d_{\min}(\Theta_i)/(2n)$. By Lemmas 2 and 3, this area lower bound equals $d^2_{\min}(\Theta_i)/(2n)$, concluding the proof. □

As described in Section 5.5, the number of decent steps is at most $E(\Theta_0)/(G\Delta t)$. Using the observation that $D \geq d_{\min}(\Theta_i) \geq d_{\min}(\Theta_0)/n^2$, a computation shows that $w^3 d^{13}_{\min}(\Theta_0)/(5320 \cdot 61920)n^{38.5}L^{13}$ is a lower bound for Δt. Substituting this and our bounds for $E(\Theta_0)$ and G into the bound $E(\Theta_0)/(G\Delta t)$, and writing w in terms of $d_{\min}(\Theta_0)$, n, and L by Lemma 4, we arrive at the following final bound:

COROLLARY 1. *The number of steps in our algorithm is at most $1752484608000\, n^{79} L^{25}/D^{26}(\Theta_0)$.*

This statement of the bound has the disadvantage of being large when the linkage is scaled very small or very large, because the definitions of L and D force values of at least and at most 1, unnecessarily blowing up the ratio L/D. Fortunately, the actual number of steps made by our algorithm is invariant under scaling of the linkage, so we can choose a scaling that avoids this disadvantage:

COROLLARY 2. *The number of steps in our algorithm is at most $117607251220365312000\, n^{79}(\ell_{\max}/d_{\min}(\Theta_0))^{26}$.*

PROOF. Uniformly scale the linkage and the initial configuration Θ_0 by a factor of $1/\ell_{\max}(\Theta_0)$. The resulting configuration Θ'_0 has $\ell_{\max}(\Theta'_0) = 1$ and $d_{\min}(\Theta'_0) = d_{\min}(\Theta_0)/\ell_{\max}(\Theta_0)$. In particular, $L(\Theta'_0) = \ell_{\max}(\Theta'_0)$. Next we compare $D(\Theta'_0)$ and $d_{\min}(\Theta'_0)$. Consider any vertex v_j connected by bars to two other vertices v_i and v_k. By Equation 8, the elliptic distance between vertex v_k and bar $\{v_i, v_j\}$ is at most $2\|v_k - v_j\| = 2\ell_{jk} \leq 2\ell_{\max}(\Theta'_0) = 2$. Therefore $d_{\min}(\Theta'_0) \leq 2$ and thus $D(\Theta'_0) \geq \frac{1}{2} d_{\min}(\Theta'_0)$. Applying Corollary 1, we obtain an upper bound of

$$\begin{aligned} & 1752484608000\, n^{79}/(2d_{\min}(\Theta'_0))^{26} \\ = \; & 67108864 \cdot 1752484608000\, n^{79}/d^{26}_{\min}(\Theta'_0) \\ = \; & 117607251220365312000\, n^{79}(\ell_{\max}(\Theta_0)/d_{\min}(\Theta_0))^{26}. \end{aligned}$$

□

	Doubled tree ($n = 50$)				Teeth ($n = 29$)			
Method	#steps	Time (sec)	Time/step	Error	#steps	Time (sec)	Time/step	Error
CDR	463	5,927.0	12.8010	0.654%	322	187.6	0.5826	14.131%
Energy	79,681	289.2	0.0036	n/a	5,032	7.9	0.0016	n/a
Ratio	0.0058	20.49	3,555	n/a	0.0639	23.74	364.13	n/a

Table 1. Running times for the examples in Figure 3, measured in CPU seconds. Computation times for both methods were measured in CPU seconds on a 930 MHz Pentium III. CDR running times just measure time spent during the CPLEX barrier optimizer for quadratic programs, which ignores the (relatively short) time to prepare the input to CPLEX. Energy running times measure the entire execution of a C++ program. Our C++ implementation runs about 6 times faster than our Java implementation which is accessible on the Internet [1].

7 Experiments

7.1 Comparison to CDR

We compared a C++ implementation of our energy approach to an implementation of [9] based on the CPLEX barrier solver for quadratic programs[2], on two examples of closed chains. The resulting animations and running times are shown in Figure 3 and Table 1, respectively. The running times are measured imperfectly, as described in the caption of Table 1, but in a way that only favors the CDR method.

The comparison in Table 1 is difficult to interpret, because the methods we are comparing have fundamental differences. At the superficial level, for each example, the CDR method uses many fewer steps, but the cost for computing each step is several orders of magnitude slower, so that overall the CDR method is much slower than the energy method. But a more careful analysis shows that the energy method is even better.

In particular, the number of steps are chosen in fundamentally different ways with the two methods. In the energy method, we can move conservatively in accordance with the step bound used in Section 6 or we can use a more aggressive numerical method. Regardless of how the steps are chosen the link lengths are preserved exactly. In the CDR method however, the steps are approximating a complex motion, and small steps are necessary to keep the approximation close and preserve the edge lengths. Because the CDR implementation does not include constraint stabilization, the edge lengths drift, and this drift accumulates over the motion. The final configurations have more than 10% relative error in the edge lengths. To obtain much smaller errors with the CDR method would require many more steps, and significantly more computation time.

The time per step is easier to compare, although again this comparison is not necessarily the "right" thing. The primary cost in the CDR method is solving a convex program with $\Theta(n^2)$ linear constraints, where n is the number of joints. Such a program can be solved up to error tolerance ϵ in $O(n^4/\epsilon)$ worst-case time by the classic ellipsoid method [11], or in $O(n^2/\epsilon)$ time with high probability by a new random-sampling method [4]. In contrast, the running time of the energy method to compute a step depends quadratically on n, and does not depend on any error tolerance.

7.2 Additional Examples

To illustrate the scalability of the energy approach, we show some additional examples and their performance in Figure 4 and Table 2, respectively. Some of these example contain several hundred vertices and would have been prohibitively expensive to run using the CDR method.

[2]The convex objective function in [9] is not exactly quadratic, but CPLEX does not support such objective functions. We use a quadratic relaxation of the objective function because it is much faster to compute, in particular because we can then use CPLEX. This relaxation makes the running times of [9] only smaller. It is also perhaps a fairer comparison, because the objective function in [9] was not chosen with efficiency in mind, but rather for convenience in the proof.

	Energy method		
Example	#steps	Time (sec)	Time/step
Spiral ($n = 34$)	5,080	7.7	0.0015
Tentacle ($n = 380$)	2,481	1,159.0	0.4672
Spider ($n = 380$)	2,274	870.7	0.3829
Cover ($n = 17$)	10,390	4.1	0.0004

Table 2. Running times for the C++ implementation of the energy method, applied to the examples in Figures 1 and 4, measured in CPU seconds on a 930 MHz Pentium III.

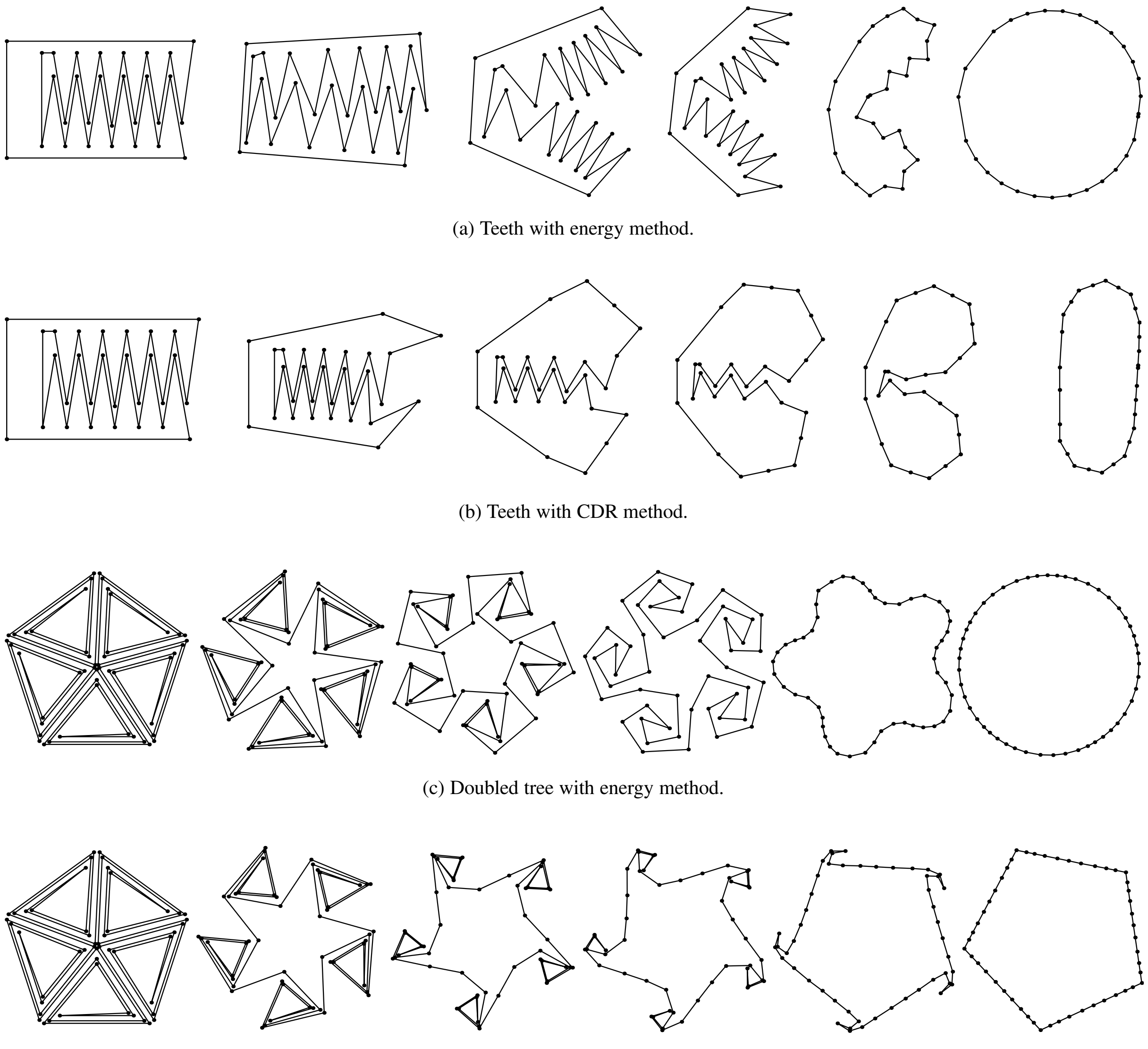
8 Conclusion

We have presented a simpler, more efficient, and more practical method to unfold linkages made up of arcs and cycles. While the motion is not globally expansive, its minimization of energy attempts to balance distances and reconfigure the linkage more "organically".

One interesting question about our motion is to determine the shape of the final minimum-energy configuration of a cycle. In contrast to [9] or [12], which have unpredictable final configurations, we might expect that our energy method results in a cycle that best approximates a regular polygon, that is, causes the joints to lie on a common circle. See [2] for other results along these lines. From our experiments, this expectation seems false, but a combination of our energy function with a term involving the area of the polygon may lead to such a result.

For even faster algorithms, an interesting approach which we plan to explore is minimizing the energy function with a more sophisticated optimization procedure such as conjugate gradient. This direction should lead to motions that involve fewer steps and would be faster overall. We also note that our repulsive energy behaves very much like light energy or gravitational attraction as it radiates. It is quite likely that the same hierarchical multipole methods that have been used for large n-body gravitational simulations, photo-realistic rendering, and fast evaluation of radial splines could be used to efficiently solve very large linkage systems as well.

In Section 5.3 we briefly touched on the idea that any of many downward directions can be used in our minimization procedure. In particular, different choices of admissible energy function and parameterization will yield different gradient directions that could be used to construct a variety of unfolding motions. These choices can be engineered to have different desirable properties such as numerical stability or symmetry preservation. For example, the methods described above do not preserve symmetries, but a method based on a parameterization using positions with explicit algebraic constraints for each edge does preserve symmetries.



(a) Teeth with energy method.

(b) Teeth with CDR method.

(c) Doubled tree with energy method.

(d) Doubled tree with CDR method.

Figure 3. A comparison of convexification by our method and by CDR. To maximize visibility, the animation zooms as time proceeds; in fact, all edge lengths remain constant.

Perhaps the most exciting direction for further research, which we are actively pursuing, is the *linkage refolding* problem: given two configurations of a linkage, find the "shortest" motion connecting them. This problem is important in several of the applications mentioned in the introduction, including key-frame animation and robotic-arm folding. We believe that our approaches apply to this problem as well, by defining an energy function on the space of motions instead of the space of configurations, and following the gradient of the motion to produce the shortest motion, forming a geodesic in the space of motions. Details will appear in a forthcoming paper.

Acknowledgments

We thank Robert Connelly, Alan Edelman, Jeff Erickson, Rob Ghrist, R. Andrew Hicks, Jonathan Shewchuk, and Ileana Streinu for helpful discussions. Demaine was supported in part by the National Science Foundation under grant CCF-03-47776 and Cantarella under grant DMS-02-04862. Iben was supported by an NSF Fellowship. Iben and O'Brien were supported in part by NSF CCR-02-04377, State of California MICRO 02-055, and by generous support from Pixar Animation Studios, Intel Corporation, Sony Computer Entertainment America, the Okawa Foundation, and the Alfred P. Sloan Foundation.

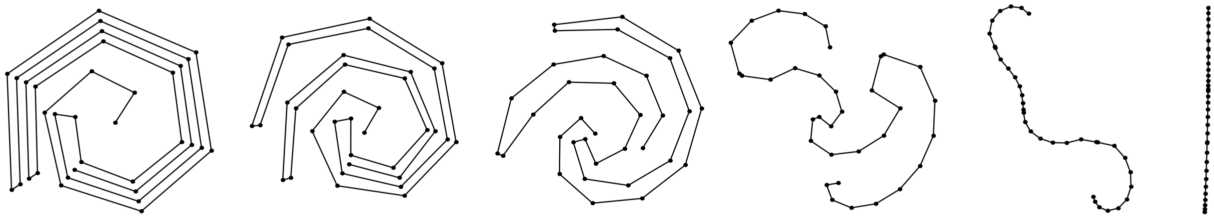
References

[1] http://www.cs.berkeley.edu/~job/Projects/Unfold/ .

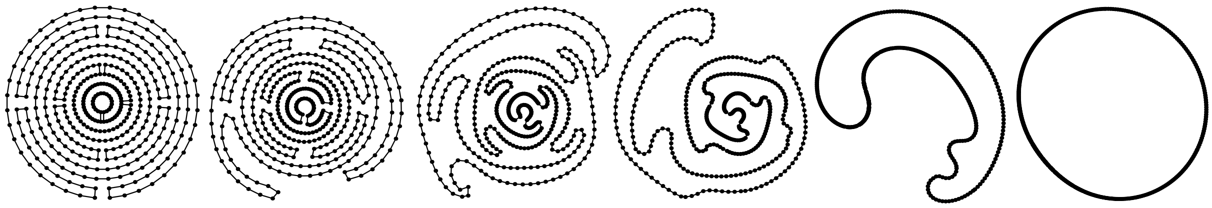
[2] Aaron Abrams, Jason Cantarella, Joseph H.G. Fu, Mohammad Ghomi, and Ralph Howard. Circles minimize most knot energies. *Topology*, 42(2):381–394, 2002.

[3] Helmut Alt, Christian Knauer, Guenter Rote, and Sue Whitesides. The complexity of (un)folding. In *Proceedings of the 19th ACM Symposium on Computational Geometry*, San Diego, California, June 2003. To appear.

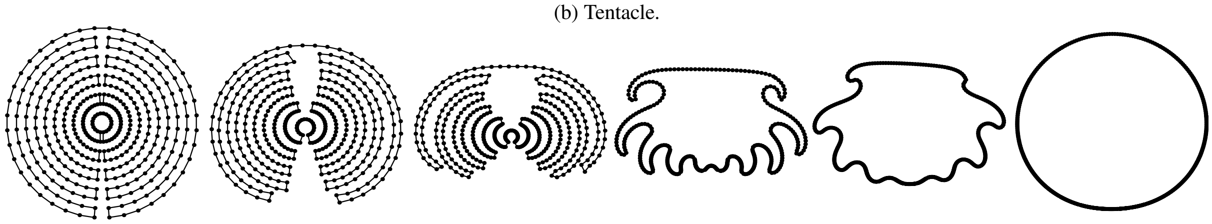
[4] Dimitris Bertsimas and Santosh Vempala. Solving convex programs by random walks. In *Proceedings of the 34rd ACM*



(a) A spiral open chain.



(b) Tentacle.



(c) Spider.

Figure 4. Other examples of straightening and convexification computed with our method. To maximize visibility, the animation zooms as time proceeds; in fact, all edge lengths remain constant.

Symposium on the Theory of Computing, pages 109–115, 2002.

[5] 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 three dimensions. *Discrete & Computational Geometry*, 26(3):283–287, October 2001.

[6] Jason Cantarella and Heather Johnston. Nontrivial embeddings of polygonal intervals and unknots in 3-space. *Journal of Knot Theory and Its Ramifications*, 7(8):1027–1039, 1998.

[7] Roxana Cocan and Joseph O'Rourke. Polygonal chains cannot lock in 4D. *Discrete & Computational Geometry*, 20(3):105–129, November 2001.

[8] Robert Connelly, Erik D. Demaine, and Günter Rote. Straightening polygonal arcs and convexifying polygonal cycles. Technical Report B 02-02, Fachbereich Mathematik und Informatik, Serie B - Informatik, Freie Universität Berlin, February 2002.

[9] Robert Connelly, Erik D. Demaine, and Günter Rote. Straightening polygonal arcs and convexifying polygonal cycles. *Discrete & Computational Geometry*, 30(2):205–239, September 2003. Preliminary version appeared at FOCS 2000.

[10] Yuanan Diao, Claus Ernst, and E.J. Janse Van Rensburg. Properties of knot energies. In *Topology and Geometry in polymer science (Minneapolis, MN 1996)*, pages 37–47. Springer, New York, 1998.

[11] Martin Grötschel, Laszlo Lovász, and Alexander Schrijver. *Geometric Algorithms and Combinatorial Optimization*, volume 2 of *Algorithms and Combinatorics*. Springer-Verlag, Berlin Heidelberg, 1988. 2nd edition 1994.

[12] Ileana Streinu. A combinatorial approach to planar noncolliding robot arm motion planning. In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, pages 443–453, Redondo Beach, California, November 2000.