Title
Refolding planar polygons

Permalink
https://escholarship.org/uc/item/2270c21m

Authors
Iben, Hayley N
O'Brien, James F
Demaine, Erik D

Publication Date
2004

DOI
10.1145/1186223.1186341

Supplemental Material
https://escholarship.org/uc/item/2270c21m#supplemental

Peer reviewed

Refolding Planar Polygons

Hayley N. Iben* James F. O'Brien* Erik D. Demaine**
*University of California, Berkeley **Massachusetts Institute of Technology

Introduction

This sketch describes a guaranteed technique for generating intersection-free interpolation sequences between arbitrary, non-intersecting, planar polygons. The computational machinery that ensures against self intersection guides a user-supplied distance heuristic that determines the overall character of the interpolation sequence. Additional control is provided to the user through specifying algebraic constraints that can be enforced throughout the sequence.

Refolding

Our interpolation method builds on a technique presented in [Cantarella et al., 2004]. That technique can efficiently compute a motion for a planar collection of polygons and polylines that *unfolds* the collection to *outer-convex* configurations. Those motions are easily computed and correspond to the downward gradient of a "repulsive" energy function based on the vertex-to-edge distances within the polygon.

Because one can trivially interpolate between any two convex polygons, the unfolding results provide an obvious way to build a path from one polygon to another. The existence of this path guarantees that our refolding algorithm can always construct an intersection-free sequence between any two polygons. This worst-case path is not particularly interesting, so the refolding algorithm described here uses a user-supplied direction heurstic to generate a more desirable path.

The refolding algorithm preserves symmetries in the input polygons by treating all edges equivalently. This is achieved by parameterizing the polygons using the vertex positions directly as opposed to using joint angles and edge lengths. Any desired edge-length preservation is enforced using algebraic constraints.

The interpolation sequence is computed by an iterative process. At each step of the algorithm, the energy is calculated for the two polygons being interpolated between. This energy function is based on the Euclidean distances between all edges and vertices and satisfies the properties described in [Cantarella et al., 2004]. A user-supplied direction heuristic determines a direction that would move the higher energy polygon toward the lower energy one. This direction is then modified to avoid self intersections. This process of selecting the higher energy polygon and moving towards the lower one repeats until the two states converge.

Because the energy function provides a guiding framework, the direction heuristic can be quite simplistic and still produce good results. In fact, Figure 1 was produced using the trivial heuristic that would simply move the vertices on a straight line to their target location. This heuristic alone produces intersecting sequences, but the energy function guides it around intersections.

The user can also control the interpolation by specifying algebraic constraints that should be satisfied by each polygon in the sequence. These constraints include distance constraints, such as constant edge lengths, and changing edge lengths monotonically.

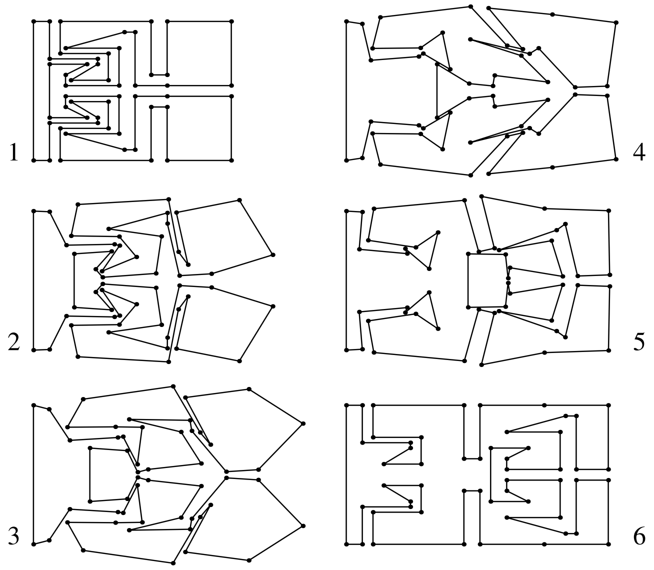

Figure 1: An intersection-free interpolation sequence generated using our algorithm. The first and last frames are the two polygons being interpolated between. All edge lengths were held constant. Total computation time was 3.5 min.

The direction is modified to ensure that each step moves to an equal- or lower-energy configuration. This energy constraint is enforced by projecting the direction so that it is perpendicular to the gradient of the energy function. If the constraints cannot all be satisfied simultaneously, then the energy constraint will be satisfied and the user constraints as much as possible. The energy constraint is treated with higher priority since it is what assures convergence.

As a result of these projections, the final direction may become zero. In this case, the direction is set to the downward energy gradient and projected to honor user constraints.

One possible issue with our method is that it only uses local information from the energy function gradient to avoid collisions. As a result, we cannot guarantee that the path generated is optimal in any sense. However, we can guarantee that we can find a non-self-intersecting path between two polygons. In practice, the algorithm appears to do a good job finding paths that do not detour needlessly.

The intersection avoidance method described here provides a guaranteed method for generating intersection-free interpolation sequences. As demonstrated in the supplemental material, the user-supplied distance heuristic along with the algebraic constraints provides the user with a powerful control mechanism for determining how the interpolation should appear, while still assuring against intersection.

References

CANTARELLA, J. H., DEMAINE, E. D., IBEN, H. N., AND O'BRIEN, J. F. 2004. An energy-driven approach to linkage unfolding. In *The 2004 Symposium on Computational Geometry*. To appear.