Morphing Planar Graph Drawings with Bent Edges

Anna Lubiw [a,1] Mark Petrick [a,2]

[a] *David R. Cheriton School of Computer Science, University of Waterloo, Waterloo, ON, Canada, N2L 3G1*

Abstract

We give an algorithm to morph between two planar drawings of a graph, preserving planarity, but allowing edges to bend. The morph uses a polynomial number of elementary steps, where each elementary step is a linear morph that moves each vertex in a straight line at uniform speed. Although there are planarity-preserving morphs that do not require edge bends, it is an open problem to find polynomial-size morphs. We achieve polynomial size at the expense of edge bends.

Keywords: Please list keywords for your paper here, separated by commas.

1 Introduction

A *morph* from one drawing of a planar graph to another is a continuous transformation from the first drawing to the second that maintains planarity. Developments in the theory of morphing run parallel to the developments in graph drawing, though they lag behind. In particular, the milestones in the history of planar graph drawing are: the existence results for straight line planar drawings due to Wagner, Fáry and Stein; Tutte's algorithm to construct such drawings; and, in 1990, the polynomial time algorithms of de Frayssiex, Pach, Pollack [2] and independently Schnyder [5] to construct a straight-line drawing of an n-vertex planar graph on an $O(n) \times O(n)$ grid.

[1] Email: `alubiw@uwaterloo.ca`
[2] Email: `mdtpetrick@uwaterloo.ca`

Mirroring these, the first result on morphing planar graph drawings was an existence result: between any two planar straight-line drawings there exists a morph in which every intermediate drawing is straight-line planar. This was proved for triangulations, by Cairns [1] in 1944, and extended to planar graphs by Thomassen [6] in 1983. Both proofs are constructive—they work by repeatedly contracting one vertex to another. Unfortunately, they use an exponential number of steps, and are horrible for visualization purposes since the graph contracts to a triangle and then re-emerges.

The next development was an algorithm to morph between any two planar straight-line drawings, given by Floater and Gotsman [3] in 1999 for triangulations, and extended to planar graphs by Gotsman and Surazhsky [4] in 2001. The morphs are not given by means of explicit vertex trajectories, but rather by means of "snapshots" of the graph at any intermediate time t. By choosing sufficiently many values of t, they give good visual results, but there is no proof that polynomially many steps suffice. Furthermore, the morph suffers from the same drawbacks as Tutte's original planar graph drawing algorithm in that there is no nice bound on the size of the grid needed for the drawings.

The history of morphing planar graph drawings has not progressed to the analogue of the small grid results of de Fraysseix et al.: *It is an open problem to find a polynomial size morph between two given drawings of a planar graph.*

In this paper we solve this problem provided that edges are allowed to bend during the course of the morph. We give a polynomial-time algorithm to find a planarity preserving morph between two drawings of a planar graph on n vertices, where the morph is composed of a sequence of $O(n^6)$ linear morphs.

Terminology. A *planar drawing* of a graph $G = (V, E)$ assigns to each vertex $v \in V$ a distinct point $p(v)$ in the plane, and to each edge $e = (u, v)$ a path from $p(u)$ to $p(v)$ so that paths intersect only at a common endpoint. A *plane graph* is one that has a planar drawing. Two planar drawings of a graph are *combinatorially identical* if they have the same outer face and the same cyclic order of edges around vertices. We will consider drawings in which edges are drawn as polygonal paths. A point where such a polygonal path changes direction is called a *bend*.

A *morph* from a drawing P of a graph G to a drawing Q of the graph is a continuous family of drawings $P(t)$, indexed by time $t \in [0, 1]$ where each $P(t)$ is a drawing of G, and $P(0) = P$ and $P(1) = Q$. A morph *preserves planarity* if each $P(t)$ is planar.

2 The Morphing Algorithm

We give an algorithm that takes two combinatorially identical planar straight line drawings P and Q of a graph G, and finds a planarity-preserving edge-bending morph from P to Q using a polynomial number of elementary steps.

Conceptually, the morph is simple. If $v_1, v_2, \ldots, v_n$ are the vertices of Q ordered by x-coordinate, locate v_1 in P—it must be on the outer face—and "pull it out" of the drawing until it is at the far left, allowing the edges of P to bend in compensation. Repeat with v_2, v_3 and so on until the vertices of P appear in the same x-ordering as those of Q. The edges of P become monotone polygonal paths which are then straightened via a linear morph.

In more detail: Add vertical lines to the drawing P, one through each vertex and n new lines $L_1, \ldots L_n$ at the left. We will move v_i to L_i, thus ordering the vertices. To specify the route taken by v_i, we augment Q with an extra vertex v_0 at the far left, and with extra straight-line edges so that every vertex v_i is joined by some edge e_i to a vertex earlier in the ordering. We augment P to match by routing each new edge as a polygonal path. The ith main step of the algorithm morphs P by pulling v_i along the path of e_i line by line until it reaches L_i. Suppose v_i lies on line l_2, and e_i goes from v_i to the previous line l_1. Let l_3 be the next line. Before moving v_i from l_2 to l_1 we perform a *straightening step* (details deferred) that enforces the property that all *incoming edges* (from lower index vertices) to v_i arrive from l_1, and there are no other vertices/bends along l_1 between the incoming edges.

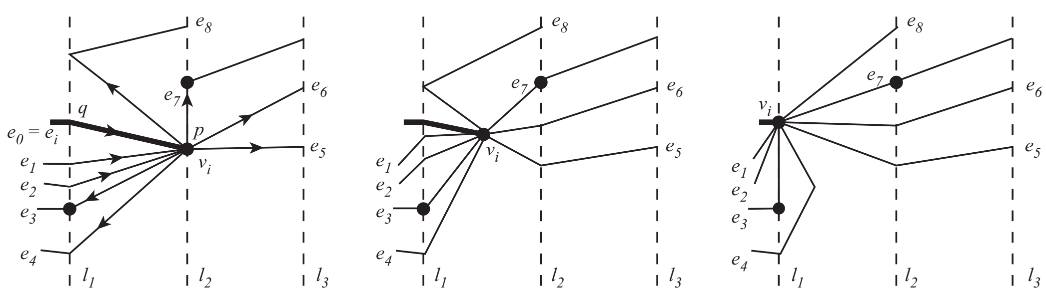

Fig. 1. The main step of the algorithm moves v_i along one segment of e_i.

The morph that moves v_i from l_2 to l_1 operates locally (see Fig. 1):

Vertex v_i moves along e_i to l_1.

Incoming edges arrive from bends on l_1. Move these bends along l_1 to q. (See edges e_1 and e_2 in Fig. 1.)

Outgoing edges have several cases: Any edge to l_3 acquires a new bend on l_2, initially at p, and with final positions nicely spaced on l_2. (See e_5 and

e_6.) There may be an edge to a vertex on l_2—leave the vertex fixed. (See e_7.) Finally, consider an edge to a bend/vertex t on l_1. If t is a bend, and *the interval along* l_1 *between* q *and* t *contains only bends connected to* v_i (*) then move t to q. (See e_8.) If t is a vertex and property (*) holds then t stays fixed and the edge (v_i, t) morphs to lie along l_1. (See e_3.) Otherwise (*) fails which means that there is an *intervening* bend/vertex along l_1 between q and t. In this case t stays fixed and the segment (v_i, t) morphs to a two-segment path that bends around the intervening point(s). (See e_4.)

We defer further details and justification of planarity to the full paper. Note that after all vertices lie in the correct x-order, there is a final linear morph, whose justification is also deferred.

3 Analysis

The run time of the algorithm depends on the number of bends (or lines), but the main step introduces new bends (see the right-hand pane of Fig. 1). We handle this by counting *turns*, which are bends where the path changes x-direction. We argue that turns are *propagated* rather than created—intuitively, pulling v_i along e_i causes each outgoing edge to follow the path of e_i. Further details of the following theorem are defered.

Theorem 3.1 *The algorithm uses* $O(n^6)$ *linear morphs.*

References

[1] S. S. Cairns. Deformation of plane rectilinear complexes. *American Mathematical Monthly*, 51:247–252, 1944.

[2] H. de Fraysseix, J. Pach, and R. Pollack. How to draw a planar graph on a grid. *Combinatorica*, 10(1):41–51, 1990.

[3] M. S. Floater and C. Gotsman. How to morph tilings injectively. *Journal of Computational and Applied Mathematics*, 101:117–129, 1999.

[4] C. Gotsman and V. Surazhsky. Guaranteed intersection-free polygon morphing. *Computers and Graphics*, 25:67–75, 2001.

[5] W. Schnyder. Embedding planar graphs on a grid. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, pages 138–148, 1990.

[6] C. Thomassen. Deformation of plane graphs. *Journal of Combinatorial Theory Series B*, 34:244–257, 1983.