Computing Strongly Homotopic Line Simplification in the Plane

Shervin Daneshpajouh* Mohammad Ali Abam† Lasse Deleuran‡ Mohammad Ghodsi§

Abstract

We study a variant of the line-simplification problem where we are given a polygonal path $P = p_1, p_2, \ldots, p_n$ and a set S of m point obstacles in a plane, and the goal is to find the optimal strongly-homotopic simplification, that is, a minimum subsequence $Q = q_1, q_2, \ldots, q_k$ ($q_1 = p_1$ and $q_k = p_n$) of P defining a polygonal path which approximates P within a given error ε and every link $q_l q_{l+1}$ corresponding to the shortcut $p_i p_j$ is homotopic to the sub-path $p_i, \ldots, p_j$ of P. We present a general method running in time $O(n(m+n)\log(nm))$ for identifying every shortcut $p_i p_j$ that is homotopic to the sub-path $p_i, \ldots, p_j$ of P, called a homotopic shortcut. In the case of x-monotone paths we propose an efficient and simple method to compute all homotopic shortcuts in $O(m\log(nm) + n\log n\log(nm) + k)$ time where k is the number of homotopic shortcuts.

1 Introduction

Introduction

Suppose we want to visualize a large geographical map as a collection of non-intersecting chains representing some features like rivers or country borders, and points representing places like cities, at different levels of details. This leads us to simplify the map. A simplified map must resemble the original map, that is, it must satisfy conditions (i) the distance between each point on a original chain and the simplified chain is within a given error tolerance, and (ii) each original chain and the simplified chain must be in the same homotopy class; roughly speaking it means that if a point (city for instance) is below a chain (river for example), the point must remain below the simplified chain. We however, pay our attention to a simpler variant of the above problem where we are given a polygonal path $P = p_1, p_2, \ldots, p_n$ and a set S of m point obstacles in a plane, and we wish to simplify the path P such that the simplified path is close enough to stay within the given error tolerance of the original path, while still homotopic to the original path.

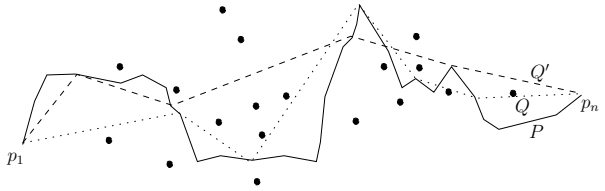

Figure 1: Two simplifications Q and Q' of path P. Only Q is strongly homotopic to P.

In this paper we study the restricted version of the line-simplification, in which the vertices of Q are a subsequence of P and each $q_l q_{l+1}$ is called a *link*. Each link $q_l q_{l+1}$ of the simplified path corresponds to a shortcut $p_i p_j$ (with $j > i$) of the original path P, and the error of the link is defined as the distance between $p_i p_j$ and the sub-path $p_i, \ldots, p_j$ denoted by $P(i,j)$. To measure the distance between $p_i p_j$ and $P(i,j)$ the Hausdorff distance and the Fréchet distance are often used. The *error* of the simplification Q denoted by $\text{error}(Q,P)$ is now defined as the maximum error of any of its links. A simplification Q is homotopic to path P if it is continuously deformable to P without passing over any points of S while keeping its end-vertices fixed. A path P and its simplification Q are *strongly homotopic* if for any link $q_l q_{l+1}$ of Q corresponding to the shortcut $p_i p_j$, the sub-path $P(i,j)$ and $p_i p_j$ are homotopic. For ease of presentation, such a shortcut is called a *homotopic shortcut*. Fig.1 illustrates two simplifications Q and Q'. The simplification Q is strongly homotopic to P but Q' is not; so only Q is of our interest. If a path P and its simplification Q are strongly homotopic, they are clearly homotopic as well but if P and Q are homotopic, they are not necessarily strongly homotopic. However, if P is x-monotone, we can simply conclude P and Q are strongly-homotopic if they are homotopic.

Related work. De Berg *et al.*[2, 3] were first to study homotopic line simplification in the restricted model for the Hausdorff distance under the Euclidean metric. Their algorithm running in $O(n(n+m)\log n)$ time finds the optimal simplification provided that P is x-monotone; otherwise the simplification is not guaranteed to be optimal. Guibas *et al.*[8] showed that the optimal homotopic line-simplification prob-

*Computer Engineering Department, Sharif University of Technology, daneshpajouh@ce.sharif.edu

†Technische Universität Dortmund, mohammad-ali.abam@tu-dortmund.de

‡MADALGO, Center for Massive Data Algorithmics, a Center of the Danish National Research Foundation. Computer Science Department, Aarhus University, ld@madalgo.au.dk

§Computer Engineering Department, Sharif University of Technology, ghodsi@sharif.edu

This is an extended abstract of a presentation given at EuroCG 2011. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear in a conference with formal proceedings and/or in a journal.

lem in the unrestricted model is NP-hard when the simplification is forced to be non-self-intersecting like the original one. A general version, subdivision simplification, was later proved to be MIN PB-complete, that is even hard to be approximated, and some heuristic algorithms were given by Estkowski and Mitchell [7].

Our results. We propose general methods to efficiently compute all homotopic shortcuts which combined with the Imai and Iri's general framework, give us the optimal strongly-homotopic simplification. For x-monotone paths P of length n, our first algorithm computes all homotopic shortcuts in $O(m\log(nm) + n\log n\log(nm) + k)$ time where k is the number of homotopic shortcuts. In the case of general paths P of length n, we propose an $O(n(m+n)\log(nm))$-time algorithm to compute all homotopic shortcuts.

2 Optimal strongly-homotopic simplification

Our approach, like almost all algorithms given in the restricted model, is based on Imai and Iri's general framework in which for a given error ε, an unweighted directed graph $G_\varepsilon(P)$ (or simply G_ε) is defined as follows: $G_\varepsilon = (V, E_\varepsilon)$ where $V = \{p_1, \ldots, p_n\}$ and $E_\varepsilon = \{(p_i, p_j) | d_F(p_ip_j, P(i,j)) \leqslant \varepsilon\}$ where $d_F(p_ip_j, P(i,j))$ is the F distance of shortcut p_ip_j and $P(i,j)$, for some error function F. Each simplification Q with the error of at most ε corresponds to a path in G_ε from p_1 to p_n and therefore, the optimal simplification is the shortest path in G_ε from p_1 to p_n in time $O(|G_\varepsilon|)$. Our general algorithm is as follows. We first compute G_ε in the absence of the obstacles, and then test whether each edge p_ip_j in G_ε is homotopic to $P(i,j)$ in the presence of the obstacles. We remove non-homotopic shortcuts from G_ε and finally compute the shortest path in G_ε from p_1 to p_n by a breath first search to obtain the optimal strongly-homotopic simplification. The main step of our algorithm is how to compute homotopic shortcuts; the rest exists in the literature as already mentioned. This step can be done in $O((n^3 + n^2m)\log(nm))$ time by individually applying the homotopy-testing algorithm given in [4] to each edge in G_ε and its corresponding sub-path. This of course is far from being efficient. Therefore, the challenging question is whether it is possible to efficiently compute all homotopic shortcuts. We affirmatively answer this question using some novel ideas and nice observations and obtain the following results using Lemmas 3 and 6.

Theorem 1 *Suppose P is a non-self-intersecting polygonal path with size n in a plane containing m point obstacles. Suppose that $T_F(n)$ is the time needed to compute G_ε under the error function F in the absence of the obstacles. Therefore (i) if P is x-monotone, the optimal strongly-homotopic simplification can be computed in $T_F(n) + O(m\log(nm) + n\log n\log(nm)+k)$ time where k is the number of homotopic shortcuts. (ii) if P is a general path, the optimal strongly-homotopic simplification can be computed in $T_F(n) + O(n(m+n)\log(nm))$ time.*

2.1 Computing homotopic shortcuts for x-monotone paths

The main idea behind our x-monotone algorithm is to enclose P inside a simple polygon $\Psi(P,S)$ such that p_ip_j is a homotopic shortcut if and only if p_i can see p_j inside the simple polygon $\Psi(P,S)$.

We define the simple polygon $\Psi(P,S)$ as follows. We first determine the aboveness relation of the obstacles and path P in $O(m\log n)$ time by locating each obstacle with respect to the path P in $O(\log n)$ time. We then compute in $O(n+m)$ time an axis-parallel rectangle containing the path vertices and obstacles in its interior. Let λ be a sufficiently small number less than the x-coordinate difference of any two points of the path vertices and obstacles—λ can simply be computed in $O((n+m)\log(nm))$ time. For each obstacle s_i above P, we define two points s_i^+ and s_i^- on the upper boundary of the rectangle with x-coordinates $x(s_i) + \lambda$ and $x(s_i) - \lambda$ respectively where $x(s_i)$ is the x-coordinate of s_i. We connect s_i to s_i^+ and s_i^- and remove the part of the rectangle joining s_i^+ to s_i^-. Similarly, we define s_i^+ and s_i^- and the associated edges for obstacles s_i below P. All together, this gives us the simple polygon $\Psi(P,S)$.

Lemma 2 *A shortcut p_ip_j is a homotopic shortcut if and only if p_i can see p_j inside $\Psi(P,S)$.*

The above lemma reduces the problem to computing the visibility graph of n points inside a simple polygon with size $3m+4$. Thanks to Ben-Moshe *et al.*[1] who presented an $O(m + n\log n\log(mn) + k)$-time algorithm to compute all visible pairs in $O(n + m + k)$ space, where k is the number of visible pairs. Putting it all together we get the following result:

Lemma 3 *Let $P = p_1, \ldots, p_n$ be an x-monotone polygonal path and let S be a set of m point obstacles in a plane. All homotopic shortcuts p_ip_j can be computed in $O(m\log(nm) + n\log n\log(nm) + k)$ time and $O(n+m+k)$ space where k is the number of homotopic shortcuts.*

2.2 Computing homotopic shortcuts for general paths

In this section we describe an algorithm to compute all homotopic shortcuts p_ip_j in $O(n(m+n)\log(nm))$ time. We fix $i = 1$ and show all homotopic shortcuts p_1p_j can be together computed in $O((m+n)\log(nm))$

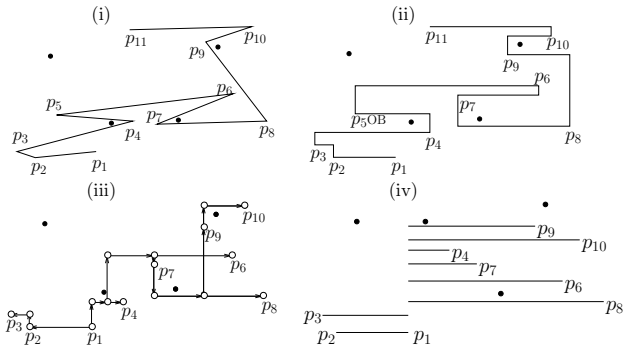

Figure 2: (i) The original path $P = p_1, \ldots, p_{11}$. (ii) The rectified path. (iii) The tree $\mathcal{T}$. The nodes of $\mathcal{T}$ are denoted by empty circles. (iv) The rectified shortcuts.

time which trivially can be extended to any i obtaining our main result as i changes from 1 to n.

One way of representing a path α is to write the sequence of obstacles that it passes above (overbar) and below (underbar). An adjacent pair of repeated obstacles can be deleted by deforming the path without changing the homotopy class. This deletion can be continued until a *canonical sequence* is obtained. A repeated obstacle with opposite bars is called a turn obstacle of α. The path α can be represented in the rectified form, so-called *rectified path*, where each maximal x-monotone sub-path is treated as a horizontal segment whose y-coordinate is its rank in the total order. We define the *canonical rectified path* (denoted by CRP(α)) to be the shortened form of the rectified path α which represents the canonical sequence of α.

Rectifying P. Since P is non-self-intersecting, the edges of P and the obstacles of S induce an aboveness relation which is acyclic and computable in time $O((n+m)\log(nm))$ [9]. The aboveness relation indeed defines a partial order. This partial order can be easily extended to a total order. Let $\text{rank}_p(\mathcal{O})$ be the rank of an object $\mathcal{O}$ (obstacle or edge) in the total order. By letting the y-coordinate of any point of object $\mathcal{O}$ be $\text{rank}_p(\mathcal{O})$, the path P gets rectified, i.e, each edge is represented as a horizontal segment. We join two horizontal segments corresponding to two consecutive edges in P by a vertical segment in order to maintain the original connectivity—see Fig. 2(ii).

Orthogonal-range queries. Our algorithm relies on a three-sided orthogonal range-query data structure defined over m point obstacles in order to compute the the *canonical rectified path*, denoted by CRP($P(1,i)$).This data structure is used to find the closest obstacle to any given vertical segment. We use Chazelle's data structure [5] that preprocesses the obstacles in time $O(m \log m)$ while using $O(m)$ space in order to answer three-sided orthogonal-range queries in $O(\log m)$ time.

Canonical rectified paths. After rectifying P and constructing the orthogonal-range-query data structure we are ready to compute all CRP(i) inductively, i.e, CRP($i+1$) is obtained from CRP(i). Since separately maintaining CRP(i) may require $O(n^2)$ space, we implicitly store them in a tree $\mathcal{T}$ (See Fig. 2(iii)) such that CRP(i) can be extracted by traversing $\mathcal{T}$ from the root to either a leaf or an internal node of $\mathcal{T}$, provided that it is x-monotone and it is known that if CRP(i) is not x-monotone, it can not be homotopic with its corresponding shortcut p_1p_i. Due to lack of space we omit the construction of tree $\mathcal{T}$ from this version but it can be found in the full version[6]. To this end we state the final result in the following lemma.

Lemma 4 *All x-monotone* CRP(i) *can be encoded into a tree $\mathcal{T}$ of size $O(n)$ where each x-monotone* CRP(i) *corresponds to a path from the root to a node or leaf in the tree $\mathcal{T}$.*

Rectifying shortcuts. Let $\mathcal{I}$ be the set of indices i such that CRP(i) is x-monotone. Indeed $\mathcal{I}$ presents all shortcuts remaining candidates to be homotopic shortcuts after computing all CRP(i). Consider all shortcuts p_1p_i where $i \in \mathcal{I}$ and m obstacles in S. They induce an aboveness relation defining a partial order which can be simply extended to a total order. Let $\text{rank}_s(\mathcal{O})$ be the rank of an object $\mathcal{O}$ (obstacle or shortcut) in this total order. We set the y-coordinate of any point of object $\mathcal{O}$ to be $\text{rank}_s(\mathcal{O})$. This rectifies all shortcuts. Note that there is not any relation between $\text{rank}_p(s_i)$ and $\text{rank}_s(s_i)$ for obstacle s_i as they are coming from two different total orders.

Testing homotopy for tree $\mathcal{T}$ and the rectified shortcuts. To summarize, we have two planes: (i) the plane $\mathcal{H}_1$ includes the obstacles and tree $\mathcal{T}$, and (ii) the plane $\mathcal{H}_2$ includes the obstacles and the rectified shortcuts corresponding to x-monotone CRP(i) encoded in $\mathcal{T}$. The tree $\mathcal{T}$ has $O(n)$ edges and all its edges embedded in $\mathcal{H}_1$ are either horizontal or vertical—note that $\mathcal{T}$ is not necessary non-self-intersecting and therefore a partial order of paths encoded in $\mathcal{T}$ may not exist. The y-coordinates of all horizontal segments and the obstacles in $\mathcal{H}_1$ come from rank_p while in $\mathcal{H}_2$ come from rank_s.To this end, we recall that $\mathcal{I}$ is the set of indices i such that CRP(i) is x-monotone. For a horizontal edge e of $\mathcal{T}$, let above(e) (below(e)) be the set of obstacles s_j satisfying (i) $\text{rank}_p(s_j) > \text{rank}_p(e)$ ($\text{rank}_p(s_j) < \text{rank}_p(e)$) and (ii) the x-coordinate of s_j lies between the x-coordinates of the endpoints of e.

It is easy to see the following lemma.

Lemma 5 *For any $i \in \mathcal{I}$, the rectified shortcut p_1p_i and* CRP(i) *are homotopic if and only if for any horizontal edge e of $\mathcal{T}$ appearing on* CRP(i) *and any obstacle $s_j \in$ above(e) ($s_j \in$ below(e)), the condition (A) $\text{rank}_s(p_1p_i) < \text{rank}_s(s_j)$ ($\text{rank}_s(p_1p_i) > \text{rank}_s(s_j)$) is satisfied.*

This lemma simply implies that any edge e and any object $s_j \in$ above(e) ($s_j \in$ below(e)) violating the condition (A) removes p_1p_i of being a homotopic shortcut and if there is not such a pair e and s_j, the shortcut p_1p_i is definitely a homotopic shortcut. However, testing condition (A) for each edge e and any $s_j \in$ above(e) is costly. The following useful observation shows among all obstacles either above or below e, at most two obstacles together with e are necessary to be tested whether they satisfy condition (A).

Observation 1 *It suffices to verify the condition (A) for each horizontal edge e of $\mathcal{T}$ and the obstacle s_j which has the minimum (maximum)* rank_s *in* above(e) (below(e))*; let* $\min(\text{above}(e)) = \text{rank}_s(s_j)$ ($\max(\text{below}(e)) = \text{rank}_s(s_j)$).

Now, the main question is how fast we can perform the verification of condition (A) for any horizontal edge e of $\mathcal{T}$ to remove non-homotopic shortcuts as we know above(e) or below(e) may contain many obstacles and $\mathcal{I}(e)$ may contain several indices where $\mathcal{I}(e)$ is the set of all indices $i \in \mathcal{I}$ that e appears in CRP(i). From now on, we pay our attention to above(e) as below(e) can be handled similarly.

In order to collectively verify the condition (A) in which an edge e is involved, we define a new ordering of elements in $\mathcal{I}$ such that elements in $\mathcal{I}(e)$ become consecutive. This ordering is obtained by an in-order traversal of $\mathcal{T}$—note that for any $i \in \mathcal{I}$, there is a node in $\mathcal{T}$ labeled with p_i. Let $\sigma(i)$ be the rank of $i \in \mathcal{I}$ in this ordering. With each $i \in \mathcal{I}$, we associate the point $(\sigma(i), \text{rank}_s(p_1p_i))$ in a new plane $\mathcal{H}$. Each edge e and its associated $\min(\text{above}(e))$ define a three-sided orthogonal range in $\mathcal{H}$. It is easy to see every $i \in \mathcal{I}$ whose corresponding point lies in this range, violates the condition (A) and consequently must be removed from $\mathcal{I}$. As these three-sided ranges are available in advance, we can sweep points and ranges in $\mathcal{H}$ from top to bottom and maintain a binary tree over the points based on their x-coordinates (i.e. their σ-coordinate). Upon processing a three-sided range, we simply remove $O(\log n)$ subtrees from the binary tree. Therefore the sweeping takes $O(n \log n)$ time in total. After handling below(e) for horizontal edges e of $\mathcal{T}$ in a similar manner, any remaining shortcut is a homotopic shortcut. Putting it all this together we get the following result.

Lemma 6 *Let $P = p_1, \ldots, p_n$ be a non-self-intersecting polygonal path and let S be a set of m point obstacles in a plane. All homotopic shortcuts p_ip_j can be computed in $O(n(m+n)\log(nm))$ time and $O(n+m+k)$ space where k is the number of homotopic shortcuts.*

3 Conclusion

We have proposed algorithms to compute an optimal line simplification Q (in the restricted model) of a polygonal path P with size n being strongly-homotopic to P with respect to m point obstacles in a plane. For non-self-intersecting x-monotone and general path P, our algorithms run in $T_F(n) + O(m\log(nm) + n\log n\log(nm) + k)$ and $T_F(n) + O(n(m+n)\log(nm))$ time respectively where k is the number of homotopic shortcuts and $T_F(n)$ is the time needed to compute G_ε under the error function F .

References

[1] Ben-Moshe, B., Hall-Holt, O., Katz, M., Mitchell, J.: Computing the visibility graph of points within a polygon. *In Proc. Annual Symposium on Computational Geometry*, 27–35, 2004.

[2] de Berg, M., van Kreveld, M., Schirra, S.: A new approach to subdivision simplification. *International Symposium on Computer Assisted Cartography*, 04, 79–88, 1995.

[3] de Berg, M., van Kreveld, M., Schirra, S.: Topologically correct subdivision simplification using the bandwidth criterion. *Cartography and GIS*, 25, 243–257, 1998.

[4] Cabello, S., Liu, Y., Mantler, A., Snoeyink, J.: Testing homotopy for paths in the plane. *In Proc. Annual Symposium On Computational Geometry*, 160–169, 2002.

[5] Chazelle, B.: An algorithm for segment-dragging and its implementation. *In Algorithmica*, 3, 205–221, 1988.

[6] Daneshpajouh, Sh., Abam, M. A., Deleuran, L., Ghodsi, M.: Computing Homotopic Line Simplification in a Plane (Full version). `http://www.daimi.au.dk/~danesh/publications/chls.pdf`

[7] Estkowski, R., Mitchell, J. S.: Simplifying a polygonal subdivision while keeping it simple. *In Proc. Annual Symposium on Computational Geometry*, 40–49, 2001.

[8] Guibas, L.J., Hershberger, J.E., Mitchell, J.S.B., Snoeyink, J.S.: Approximating polygons and subdivisions with minimum link paths. *International Journal of Computatioanl Geometry and Applications*, 3, 383–415, 1993.

[9] Palazzi, L., Snoeyink, J. Counting and reporting red/blue segment intersections. *In CVGIP: Graph. Models Image Process.*, 56(4), Academic Press, Inc., Orlando, FL, USA, 1994.