

Computing the Fréchet Distance Between Simple Polygons in Polynomial Time *

Kevin Buchin
Inst. of Computer Science
Freie Universität Berlin
Takustraße 9
14195 Berlin
Germany
buchin@inf.fu-berlin.de

Maike Buchin
Inst. Computer Science
Freie Universität Berlin
Takustraße 9
14195 Berlin
Germany
mbuchin@inf.fu-berlin.de

Carola Wenk
Dept. of Computer Science
University of Texas
at San Antonio
6900 N. Loop 1604 West,
San Antonio, TX 78249-0667
carola@cs.utsa.edu

ABSTRACT

We present the first polynomial-time algorithm for computing the Fréchet distance for a non-trivial class of surfaces: simple polygons. For this, we show that it suffices to consider homeomorphisms that map an arbitrary triangulation of one polygon to the other polygon such that diagonals of the triangulation are mapped to shortest paths in the other polygon.

Categories and Subject Descriptors

F.2.2 [Analysis of Algorithms and Problem Complexity]: Nonnumerical Algorithms and Problems – *geometrical problems and computations*

General Terms

Algorithms

Keywords

Shape Matching, Fréchet Distance, Simple Polygons

1. INTRODUCTION

Shape matching algorithms [4, 12] are an essential component to a wide range of cutting-edge technological sectors such as computer vision, computer aided design, robotics, medical imaging, and drug design. Any shape matching algorithm is based on a distance measure for the shapes under consideration. Due to the lack of other efficient algorithms,

*Work on this paper of the first two authors was supported by the Deutsche Forschungsgemeinschaft within the European graduate program ‘Combinatorics, Geometry, and Computation’ (No. GRK 588/2).

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

SCG’06, June 5–7, 2006, Sedona, Arizona, USA.

Copyright 2006 ACM 1-59593-340-9/06/0006 ...\$5.00.

applications for surface matching in 3D usually use heuristic variants of the Hausdorff distance which are often not well-suited.

The Fréchet distance is a distance measure for continuous shapes such as curves and surfaces, and is defined using reparameterizations of the shapes. Since it takes the continuity of the shapes into account it is generally a more appropriate distance measure than the Hausdorff distance.

Let $f : A \rightarrow \mathbb{R}^d$ and $g : B \rightarrow \mathbb{R}^d$, with $A, B \subseteq \mathbb{R}^k$ homeomorphic, be continuous mappings with $1 \leq k \leq d$. f and g describe two (hyper-)surfaces in \mathbb{R}^d . Their *Fréchet distance* is defined as

$$\delta_F(f, g) = \inf_{\sigma: A \rightarrow B} \sup_{x \in A} \|f(x) - g(\sigma(x))\|,$$

where σ ranges over all orientation-preserving homeomorphisms and $\|\cdot\|$ is the Euclidean norm. Of course, any other metric can be considered as well. We will be considering only compact sets A over which the supremum is attained, and therefore in the following we will use the maximum.

The Fréchet distance has also been defined without requiring the homeomorphisms to be orientation-preserving [8, 11]. We give our results for orientation-preserving homeomorphisms but they hold also for orientation-reversing homeomorphisms and can be extended to general homeomorphisms by considering both cases. We will be considering simple polygons in the plane which we will assume to be counterclockwise orientated. Preserving their orientation is equivalent to preserving the orientation on the boundary.

We say that a homeomorphism σ *realizes* the Fréchet distance of f and g if $\delta_F(f, g) = \max_{x \in A} \|f(x) - g(\sigma(x))\|$. Also the Fréchet distance may be realized by a sequence of homeomorphisms $\sigma_i : A \rightarrow B, i \in \mathbb{N}$, i.e. $\delta_F(f, g) = \inf_{i \in \mathbb{N}} \max_{x \in A} \|f(x) - g(\sigma_i(x))\|$. Our algorithm for computing the Fréchet distance finds a map realizing the Fréchet distance in the above sense. This map is not necessarily a homeomorphism but it can be achieved as the limit of a uniformly converging sequence of homeomorphisms which then realizes the Fréchet distance. In the following when we speak of realizing homeomorphisms we include also this case of such a map.

Note that the Fréchet distance is defined for parameterized shapes, however usually non-parameterized continuous shapes can be meaningfully parameterized by a *natural parameterization* based on the geometric description of the object, such as arc length for the case of curves.

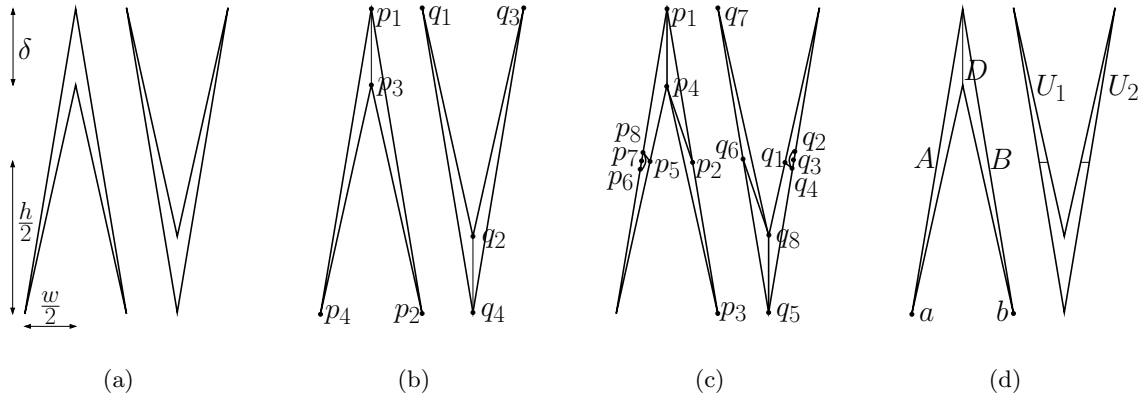


Figure 1: Example for differing Fréchet distances of boundary curves and polygons.

The Fréchet distance between polygonal curves can be computed in polynomial time [3], however computing the Fréchet distance for surfaces is NP-hard [9]. Except for the NP-hardness very little is known so far about the Fréchet distance of surfaces. It is known to be semi-computable [2], but it is not known whether it is computable, and there are no approximation algorithms.

We address this problem by considering a restricted but important class of surfaces, simple polygons, and show that their Fréchet distance can be computed in polynomial time. This is the first polynomial-time algorithm for computing the Fréchet distance for a non-trivial class of surfaces.

The rest of the paper is organized as follows: Section 2 introduces notation and preliminary lemmas that will be used later. In Section 3 we show that it suffices to consider a small well-behaved class of realizing homeomorphisms, i.e., those that map diagonals of a triangulation to shortest paths. In Section 4 we use the result of Section 3 to show how to compute the Fréchet distance between simple polygons.

2. PRELIMINARIES

2.1 Simple Polygons

Let P and Q be two simple polygons in the plane with n and m vertices, respectively. A simple polygon is the area enclosed by a non-self-intersecting closed polygonal curve in the plane. The two polygons may lie in two different planes.

We assume as underlying parameterizations the identity maps $f : P \rightarrow P$ and $g : Q \rightarrow Q$. Then the Fréchet distance simplifies to:

$$\delta_F(P, Q) = \inf_{\sigma: P \rightarrow Q} \max_{t \in P} \|t - \sigma(t)\|,$$

where σ ranges over all orientation-preserving homeomorphisms. In the remainder we will only consider orientation-preserving homeomorphisms and might refer only to σ or to a homeomorphism when the meaning is clear from the context.

The first question that comes to mind is: Is the Fréchet distance of polygons different from the Fréchet distance of their boundary curves?

OBSERVATION 1. *The Fréchet distance of two polygons may be arbitrarily larger than the Fréchet distance of their boundary curves.*

PROOF. Figure 1 (a) gives two example polygons for which, when placed on top of each other, the Fréchet distance between the boundary curves is small (roughly half the width of the polygons) however the Fréchet distance between the polygons is large (roughly half the height of the polygons). Thus the Fréchet distance of the polygons may be arbitrarily larger than the Fréchet distance of their boundary curves.

Figure 1 (b) and (c) indicate homeomorphisms on the curves and polygons, respectively, which realize the respective Fréchet distances. They show where the vertices of the curve and the triangulation, respectively, are mapped to, i.e., p_i is mapped to q_i . The Fréchet distance of the curves cannot be smaller, because half the width of the polygon is also the Hausdorff distance of the curves which is a lower bound for the Fréchet distance.

That the Fréchet distance of the polygons cannot be smaller than half the height of the polygons is indicated in Figure 1 (d). Consider the diagonal D in the left polygon. For the Fréchet distance to be less than half the height of the polygons, D must be mapped to a path that lies completely either in U_1 or U_2 . Then also either A or B must be completely mapped to U_1 or U_2 , in particular the vertex a or b . But a and b have distance more than half the height of the polygons to both U_1 and U_2 . \square

2.2 Fréchet Distance Between Convex Polygons

For the special case of convex polygons the Fréchet distance of the polygons equals the Fréchet distance of their boundary curves.

LEMMA 2. *The Fréchet distance of convex polygons equals the Fréchet distance of their boundary curves, which also equals the Hausdorff distance of the curves.*

PROOF. Let P, Q be two convex polygons, and $\sigma : P \rightarrow Q$ an arbitrary homeomorphism. We show that if we restrict σ to the boundary and extend it to the interior piecewise linearly, we get a map σ' that attains its maximum on the boundary (where it equals σ). σ' is either a homeomorphism or the limit of a uniformly converging sequence of homeomorphisms.

We do this as follows: First we triangulate Q . This yields a convex partition of P by connecting for any diagonal in the triangulation of Q the inverse images under σ of its

end points in P , i.e., we connect $\sigma^{-1}(q)$ and $\sigma^{-1}(q')$ for any diagonal (q, q') . Next we triangulate the convex cells of this partition of P and triangulate Q by connecting for any diagonal in the triangulation of P the images under σ of its end points. We define σ' to be the homeomorphism between these two triangulations. Triangles may degenerate to segments or vertices, in which case σ' can be achieved as a limit of a sequence of homeomorphisms.

The boundary curves of convex polygons are closed convex curves, and for these it is known that the Fréchet distance equals the Hausdorff distance [5, 9]. \square

The above lemma also follows from Corollary 6 and the equality of the Fréchet and the Hausdorff distance of closed convex curves [5, 9].

In fact, the lemma holds for convex polytopes in general. For convex polygons it implies that the Fréchet distance can be computed in polynomial time $O((m+n)\log(m+n))$ [1].

2.3 Shortest Paths in a Simple Polygon

We will compute the Fréchet distance between simple polygons using shortest paths and therefore review an important concept for shortest paths in a simple polygon which was introduced by Guibas et al. [10]: *hourglasses*. If s_1 and s_2 are two segments in a simple polygon, the hourglass of s_1 and s_2 represents all shortest paths between any point t_1 on s_1 and any point t_2 on s_2 . It can be described by the polygon given by the two segments and the two shortest paths between neighboring endpoints of the segments (i.e., if a_1, a_2 and b_1, b_2 are the end points of s_1 and s_2 , respectively, and their order along the boundary of the polygon is a_1, a_2, b_1, b_2 , then the hourglass is the polygon with boundary s_1 , the shortest path between a_2 and b_1 , s_2 , and the shortest path between b_2 and a_1).

2.4 Simplifying a Curve

Given a curve f and a line segment s , Lemma 3 shows that simplifying f by replacing a part of it with a line segment does not increase the Fréchet distance to s .

LEMMA 3. *Let $f : [0, 1] \rightarrow \mathbb{R}^d$ be a curve, let $s : [0, 1] \rightarrow \mathbb{R}^d$ be a line segment, and let $0 \leq t_1 < t_2 \leq 1$. Define $f' : [0, 1] \rightarrow \mathbb{R}^d$ to be equal to f for $t \in [0, t_1] \cup [t_2, 1]$. And for $t \in [t_1, t_2]$ it equals the line segment from $f(t_1)$ to $f(t_2)$. Then $\delta_F(f', s) \leq \delta_F(f, s)$.*

PROOF. First, we see that

$$\begin{aligned} \max_{t \in [0, 1]} \|s(t) - f(t)\| &\geq \max_{t \in [0, t_1] \cup [t_2, 1]} \|s(t) - f(t)\| \\ &= \max_{t \in [0, 1]} \|s(t) - f'(t)\| \end{aligned}$$

holds: the inequality holds because f and f' coincide on $[0, t_1] \cup [t_2, 1]$ and the equality holds because on the missing interval (t_1, t_2) we are taking the maximum distance between two segments, which is attained at segment end points.

Given a homeomorphism $\sigma : [0, 1] \rightarrow [0, 1]$, let $\sigma' : [0, 1] \rightarrow [0, 1]$ be the homeomorphism that equals σ on $[0, t_1] \cup [t_2, 1]$ and maps (t_1, t_2) linearly to $(\sigma_1(t_1), \sigma_1(t_2))$. Using the above inequality we get

$$\max_{t \in [0, 1]} \|s(t) - f(\sigma(t))\| \geq \max_{t \in [0, 1]} \|s(t) - f'(\sigma'(t))\|.$$

And thus for an infinite sequence of homeomorphisms σ_i realizing $\delta_F(f, s)$, the infinite sequence of homeomorphisms σ'_i yields $\delta_F(f', s) \leq \delta_F(f, s)$. \square

3. SHORTEST PATHS LEMMA

In this section we show that it suffices to look at realizing homeomorphisms that map the diagonals of a triangulation of P to shortest paths in Q and are piecewise linear inside triangles. For a triangulation T of P we denote with E_T the set of points lying on all edges of T , i.e., all points on all boundary edges and diagonals (and vertices) of T .

LEMMA 4. *Given two simple polygons P and Q , a triangulation T of P and a homeomorphism $\sigma : P \rightarrow Q$. Then there is a map $\sigma' : P \rightarrow Q$ which fulfills*

- (1) *There is a sequence of homeomorphisms $(\sigma_i)_{i \in \mathbb{N}}$ uniformly convergent to σ such that $\max_{t \in P} \|t - \sigma'(t)\| = \lim_{i \rightarrow \infty} \max_{t \in P} \|t - \sigma_i(t)\|$.*
- (2) *σ' maps the diagonals of the triangulation T to shortest paths in Q and there is a refinement T' of the triangulation T such that σ' acts linearly on the triangles of T' and all vertices of T' lie on the boundary. Therefore $\max_{t \in P} \|t - \sigma'(t)\| = \max_{t \in E_T} \|t - \sigma'(t)\|$.*
- (3) *σ' is “at least as good as σ ”, i.e., $\max_{t \in P} \|t - \sigma'(t)\| \leq \max_{t \in P} \|t - \sigma(t)\|$.*

PROOF. Let σ' be as follows: σ' equals σ on the boundary, it maps diagonals of T to the shortest paths between the corresponding boundary points of Q , and is extended piecewise linearly inside triangles by a refinement T' that adds vertices only on the boundary.

(2) holds by definition of σ' . (1) holds because the shortest paths may be overlapping but non-crossing and can therefore be approximated by arbitrary close homeomorphisms. Note that the shortest paths are non-crossing due to the consistent orientation for orientation-preserving as well as orientation-reversing homeomorphisms. If the shortest paths are overlapping σ' is the limit of a sequence of homeomorphisms but not a homeomorphism itself.

It remains to show (3). Let $\varepsilon = \max_{t \in P} \|t - \sigma(t)\|$. Because of (2) it suffices to show $\max_{t \in E_T} \|t - \sigma'(t)\| \leq \varepsilon$. By definition σ' equals σ on the boundary of P and hence $\max_{t \in \partial P} \|t - \sigma'(t)\| \leq \varepsilon$. So we only need to show that the same holds on all diagonals of T , i.e., that $\max_{t \in D} \|t - \sigma'(t)\| \leq \varepsilon$ holds for an arbitrary diagonal D in T .

For this, let d_0 be the starting point and d_1 be the end point of such a diagonal D . The shortest path $S = \sigma'(D)$ is a polygonal path in Q with starting point $\sigma(d_0)$ and end point $\sigma(d_1)$. We denote the vertices of this polygonal curve by s_0, \dots, s_k , where $s_0 = \sigma'(d_0) = \sigma(d_0)$, $s_k = \sigma'(d_1) = \sigma(d_1)$, and k is the number of edges of the polygonal path as shown in Figure 2.

We iteratively shoot rays along the edges of the shortest path and simplify the curve $\sigma(D)$ using Lemma 3 as follows: Let $C_0 = \sigma(D)$. For each $i = 1, \dots, k$ we do the following (c.f. Figure 2): Let r_i be the ray in direction $s_i - s_{i-1}$ starting at s_{i-1} . By construction s_{i-1} lies on C_{i-1} . The ray r_i cuts the polygon into two parts, s.t. the points s_0 and s_m lie in different parts. Hence the curve C_{i-1} , which is a continuous curve from s_0 to s_m , intersects r_i inside the polygon. Let c_i be the first intersection of r_i with C_{i-1} . Define C_i to be C_{i-1} simplified by exchanging the part of C_{i-1} from s_{i-1} to c_i with the line segment (s_{i-1}, c_i) . By Lemma 3 $\max_{t \in D} \|t - C_i(t)\| \leq \max_{t \in D} \|t - C_{i-1}(t)\|$. Note that s_i lies on the line segment (s_{i-1}, c_i) . Starting with $C_0 = \sigma(D)$ we end with $C_k = S = \sigma'(D)$ and therefore the iteration shows that $\max_{t \in D} \|t - \sigma'(t)\| \leq \max_{t \in D} \|t - \sigma(t)\| \leq \varepsilon$. \square

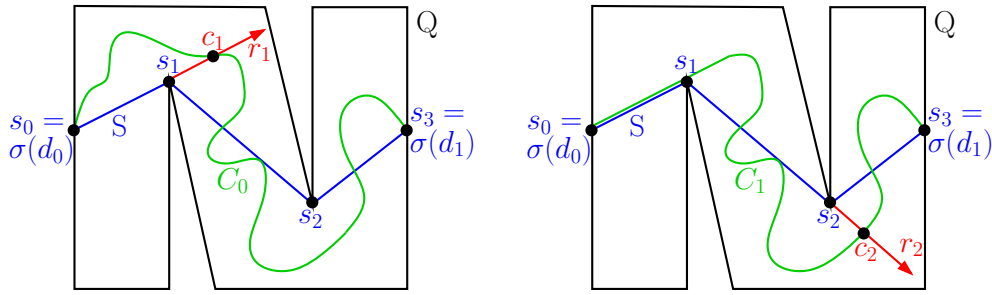


Figure 2: We recursively simplify the curve $C_0 = \sigma(D)$ to the polygonal curve S which is the shortest path from $\sigma(d_0)$ to $\sigma(d_1)$ in Q .

COROLLARY 5. *The Fréchet distance between simple polygons P and Q equals*

$$\inf_{\sigma': P \rightarrow Q} \max_{t \in E_T} \|t - \sigma'(t)\|$$

where T is an arbitrary triangulation of P . σ' ranges over all homeomorphisms from the boundary of P to the boundary of Q which are extended to T by mapping the diagonals of T to the shortest paths between the boundary vertices.

Corollary 5 follows immediately from Lemma 4. Also we get the following corollary:

COROLLARY 6. *The Fréchet distance between two simple polygons, of which one polygon is convex, equals the Fréchet distance between their boundary curves.*

PROOF. We triangulate the possibly non-convex polygon and map these diagonals to shortest paths in the convex polygon (using Lemma 4). However the shortest paths in the convex polygon are also diagonals. From the definition of a free space cell of [3] follows that the Fréchet distance between two line segments equals the maximum distance of its endpoints. Therefore it suffices to compute the distance between the boundary curves. \square

4. COMPUTING THE FRÉCHET DISTANCE

The main result of this section is a polynomial time algorithm for computing the Fréchet distance between simple polygons. In Section 4.7 we present a decision algorithm (which decides whether the Fréchet distance is at most a given parameter ε), and in Section 4.8 we compute the Fréchet distance by a parametric search over a set of critical values.

In Section 4.1, we show which paths in the *free space diagram* correspond to solutions of the Fréchet distance. The free space diagram is the data structure used by Alt and Godau [3] for deciding the Fréchet distance between polygonal curves. Since the boundaries of the polygons are closed polygonal curves we first review the *double free space diagram* and its *reachability structure* in Section 4.2. In Section 4.4 we describe the *combined reachability graph* which extends the reachability structure to also contain the information on diagonal placements.

The decision algorithm uses a dynamic programming approach on the set of diagonals of a triangulation. In Section 4.3 we define the order of the diagonals in a triangulation that we will use for the dynamic programming. In the algorithm we need to test the Fréchet distance between a

diagonal and a whole set of shortest paths, which form an hourglass, as introduced in [10] and reviewed in Section 2.3. We show how to do that in Section 4.5, and we show how to test a whole set of hourglasses at once in Section 4.6.

In the following P and Q always denote two simple polygons, n and m the number of vertices of the boundaries of P and Q , respectively, and ε a real value greater 0. T is a triangulation of P . The decision problem is to decide whether $\delta_F(P, Q) \leq \varepsilon$.

4.1 Feasible Path in the Free Space Diagram

By Lemma 4 the Fréchet distance between two simple polygons is less than ε if there is a homeomorphism $\sigma: \partial P \rightarrow \partial Q$ for which holds:

- (1) σ realizes the Fréchet distance of the boundary curves i.e. $\max_{t \in \partial P} \|t - \sigma(t)\| \leq \varepsilon$
- (2) extending σ to the diagonals of a triangulation by mapping the diagonals to shortest paths in Q yields a map σ' fulfilling $\max_{t \in D} \|t - \sigma'(t)\| \leq \varepsilon$ for all diagonals D .

For condition (1) we use the algorithm and data structure developed by Alt and Godau [3]. They show that the Fréchet distance between two polygonal curves is at most ε if and only if there is a monotone path in the corresponding free space diagram. For closed curves they search for a path in the double free space diagram.

We handle condition (2) as follows: a path in the free space diagram fulfills (1) but it also determines the shortest paths that the diagonals are mapped to because it maps the end points of the diagonals. We call these *diagonal placements*. A path in the free space diagram which places all diagonals correctly, i.e., fulfills (2) for all diagonals in P and their corresponding shortest paths in Q , is called a *feasible path*.

4.2 Free Space Diagram and Reachability Structure

For a given $\varepsilon > 0$ the *free space* of two curves $f, g: [0, 1] \rightarrow \mathbb{R}^d$ is defined as $\{(s, t) \in [0, 1]^2 \mid \|f(s) - g(t)\| \leq \varepsilon\}$. We use the terms free space and *free space diagram* interchangeably. If f and g are polygonal curves of complexity n and m , respectively, then the free space diagram can be represented in a rectangle $[0, n] \times [0, m]$ consisting of n columns and m rows of a total of mn cells. The lower boundary of a free space diagram is considered to correspond to the parameter space of f and the left boundary to correspond to the parameter space of g . The *double free space diagram* is obtained by concatenating two copies of the (single) free space

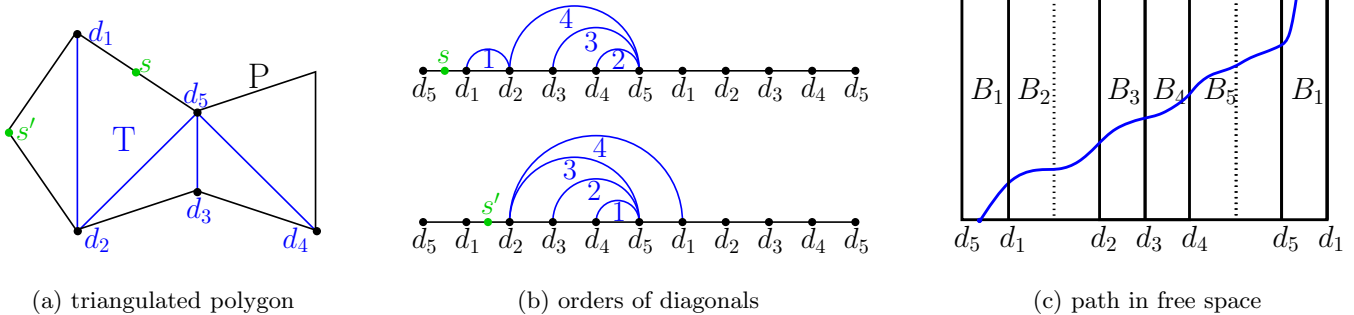


Figure 3: A triangulation (a), two different orders of the diagonals for two different starting points (b) and a path in the free space which is partitioned into blocks (c)

diagram and is represented in $[0, 2n] \times [0, m]$. See [3] for more details on free space diagrams.

The decision problem for closed polygonal curves is solved by computing the *reachability structure*. The reachability structure is a partition of the boundary of the double free space diagram into $O(mn)$ intervals. Each interval on the lower or left boundary is labeled according to whether any part of the upper or right boundary is reachable by a monotone path in the free space originating from this interval. If there is such a path then the interval is also equipped with pointers to the highest and lowest reachable points. Analogously for intervals on the upper and right boundary, for more details see [3].

The reachability structure has complexity $O(mn)$ and can be computed in $O(mn \log mn)$ time. Given the reachability structure one can check in constant time (see [3]) whether there exists a monotone path in the free space between two given points on the boundary of the free space. The decision problem for two closed curves is solved by testing for each interval on the first half of the lower boundary whether the same interval translated by n on the second half of the upper boundary lies in its reachability.

4.3 Order of Diagonals in a Triangulation

The edge set of a triangulation T of P consists of edges on the boundary and in the interior of P , the latter of which are *diagonals*. There are $n - 3$ diagonals with at most $n - 2$ end points on the boundary. We define an order of the diagonals in a given triangulation which we will later use for dynamic programming over the diagonals.

The Fréchet distance of the closed boundary curves is defined as the minimum over all Fréchet distances of the open curves with arbitrary starting (=ending) points. Using the reachability data structure this is achieved by considering all end points of intervals in the reachability structure as starting points. For a fixed starting point s on the boundary of P we define an order over the diagonals as follows: We number the end points d_1, d_2, \dots in counterclockwise order starting at s and order the diagonals as depicted in Figure 3: We cut the boundary at s and draw it as a straight line. We connect end points d_i, d_j of a diagonal by an arc. We order the diagonals, corresponding to arcs, in order left-to-right by second end point and bottom-to-top for equal end points. More formally, if we write a diagonal as an ordered tuple of its end points (d_i, d_j) with $i < j$, we define

$$(d_i, d_j) < (d_k, d_l) \Leftrightarrow (j < l) \vee (j = l \wedge i > k).$$

For two starting points that lie in between the same two diagonal end points the resulting diagonal order is the same. Hence there are at most $n - 2$ different orders of diagonals in total, each characterized by the next diagonal endpoint (in counterclockwise order) to the starting point. We will call areas of the free space that induce the same diagonal order *blocks*. Blocks consists of one or two columns in the free space that lie between the vertical lines corresponding to two neighboring diagonal end points, see Figure 3(c) for an example.

We call two diagonal orders *neighboring* if one starts with the diagonal end point d_i and the other with $d_{(i+1) \bmod e}$, where e is the number of diagonal end points. Two neighboring orders agree on the order of those diagonals which do not have d_i as an endpoint, i.e., they agree on diagonals $D < D'$ if neither D nor D' have d_i as an endpoint. For an example see Figure 3(b).

4.4 Combined Reachability Graph

The *combined reachability graph* combines the reachability information in the free space with valid diagonal placements. First, we define the *reachability graph* to be the reachability structure represented as a graph: its vertices are the reachable intervals of the reachability structure with an edge between two intervals if they can reach each other. The combined reachability graph is a subgraph of the reachability graph. Its vertices are all vertical interval-vertices of the reachability graph with edges between intervals that can be reached by feasible paths (c.f. Section 4.1).

For a fixed order of diagonals the combined reachability graph can be computed by recursively merging the reachability graphs of the blocks in the order of diagonals (described in Section 4.7). Since the reachability structure contains $O(mn)$ intervals both the reachability graph and the combined reachability graph contain $O(mn)$ vertices and $O((mn)^2)$ edges.

We will use the combined reachability graph as follows: When searching for feasible paths starting in block B_1 (as in Figure 3(c)) we compute the combined reachability graph for the order of diagonals starting with d_1 by merging blocks B_2 through B_l (where l is the number of blocks). A feasible path starting in block B_1 consists of an edge in the reachability graph of B_1 from its lower to its right boundary, an edge in the combined reachability graph between the left

boundary of B_2 to the right boundary of B_1 , and an edge in the reachability graph of B_1 from its left to its upper boundary.

4.5 Fréchet Distance Between a Diagonal and an Hourglass

The following lemma shows how to decide the Fréchet distance between a diagonal and a whole set of shortest paths, namely the hourglass of two segments. With a *shortest path in the hourglass* we always refer to a shortest path between two points on the two segments defining the hourglass.

LEMMA 7. *Let an hourglass and a diagonal be given such that each end segment of the hourglass is contained in one of the ε -disks around the endpoints of the diagonal (and not both in the same disk). If there exists one shortest path in the hourglass with Fréchet distance at most ε to the diagonal, then all shortest paths in the hourglass have Fréchet distance at most ε to the diagonal.*

PROOF. Let D be the diagonal, $A = a_1, \dots, a_l$ be a shortest path in the hourglass with $\delta_F(A, D) \leq \varepsilon$, and let $B = b_1 \dots, b_k$ be another shortest path in the hourglass. Define $B' = b_1, a_1, \dots, a_l, b_k$. From a_1, b_1 having distance at most ε to one endpoint of D , a_l, b_k having distance at most ε to the other endpoint, and $\delta_F(A, D) \leq \varepsilon$ it follows that $\delta_F(B', D) \leq \varepsilon$.

Since B' is not the shortest path from b_1 to b_k there exist two points on B' such that simplifying B' by replacing the part of B' between the two points with the line segment between them yields a shorter path B'' from b_1 to b_k in the hourglass. By Lemma 3, $\delta_F(B'', D) \leq \delta_F(B', D)$. Repeating this process and observing that the simplified paths converge to B shows that $\delta_F(B, D) \leq \delta_F(B', D) \leq \varepsilon$. \square

Note that in an open hourglass there always exists (by definition) a shortest path between the end segments which is itself a segment. The Fréchet distance between this segment and the diagonal is at most ε since the Fréchet distance between two segments is the maximum distance of the endpoints.

4.6 Fréchet Distances Between a Diagonal and many Hourglasses

In Section 4.7 we need to decide all Fréchet distances between a diagonal and several hourglasses that have a common end segment. This can be done in $O(m)$ time by choosing an arbitrary vertex on each end segment of the hourglasses and using Lemma 7 and Lemma 8.

LEMMA 8. *Given a diagonal, a polygon with m vertices, and a set of m vertices w_1, \dots, w_m on the boundary of the polygon. Then we can decide all Fréchet distances between the diagonal and the m shortest paths $\pi(w_1, w_i)$ between w_1 and w_i for $i = 1, \dots, m$ in total $O(m)$ time.*

PROOF. For simplicity we assume that the w_i ($i = 1, \dots, m$) are (additional) vertices of the polygon. We run the linear time algorithm for computing the lengths of all shortest paths from one vertex of a simple polygon to all others by Guibas et al. [10]. During the algorithm we store the additional information whether w_i can be reached from w_1 by a monotone path in the free space, for all $i = 2, \dots, m$, and we update this reachability information in constant time while processing each vertex.

The algorithm by Guibas et al. [10] computes the shortest paths starting at w_1 such that when a new vertex is processed all other vertices on its shortest path to w_1 have already been processed and we know the previous vertex on the shortest path. Thus for deciding the Fréchet distance of the shortest path to the new vertex we only need to compute the last cell of the free space diagram. If the previous vertex was not reachable, the new vertex is not either, and we store this for the new vertex. If the previous vertex was reachable, we compute the right boundary of the new cell of the free space diagram. For a “real” vertex of the polygon, i.e. not one of the w_i , we then test and store if the right boundary is reachable from the last cell and store the lowest reachable point on the boundary. For a vertex w_i we test and store if the upper corner of the right boundary is reachable, i.e. the Fréchet distance is at most the value ε that we want to decide. \square

4.7 Decision Algorithm

Algorithm 1: DecideFréchet(P, Q, ε)

Input: Simple Polygons P, Q , $\varepsilon > 0$

Output: Is $\delta_F(P, Q) \leq \varepsilon$?

```

1 Compute a triangulation of  $P$ 
2 Compute all orders of diagonals in the triangulation of  $P$ 
3 Compute a single free space diagram of the boundary curves
4 Compute the reachability graph for all blocks in the free space diagram
5 forall diagonals in the triangulation do
6     forall placements in the free space do
7         test  $\delta_F(\text{diagonal, shortest path}) \leq \varepsilon$  for a
           corresponding shortest path
8     end
9 end
10 forall blocks do
11     forall diagonals, in the order given by the block do
12         if combined reachability graph is not yet
           computed then
13             if previous diagonal nested then
14                 compute combined reachability graph
                   merged with the combined reachability
                   graph of the nested diagonal
15             end
16         else
17             compute combined reachability graph
18         end
19         store combined reachability graph
20     end
21     if diagonal has a left neighbor then
22         merge combined reachability graphs
23     end
24 end
25 Query for a feasible path starting at the lower
   boundary of the block
26 end
27 Answer “yes” if a feasible path has been found, else
   “no”

```

This algorithm decides the Fréchet distance between simple polygons in polynomial time:

THEOREM 9. *Algorithm $\text{DecideFréchet}(P, Q, \varepsilon)$ decides whether the Fréchet distance between simple polygons P, Q is at most ε . The runtime is $O(nT(mn))$, where $T(N)$ is the time to multiply two $N \times N$ matrices, and n and m are the number of vertices of P and Q .*

Note that $T(N) = \Omega(N^2)$ and the currently fastest known matrix multiplication algorithm has a runtime of $T(N) = O(N^{2.376})$ [7]. Also note that the space complexity of this algorithm is $O(n^3m^2)$ since it stores n combined reachability graphs which each have complexity $O((mn)^2)$.

PROOF. The first five steps (lines 1–9) of the algorithm are preprocessing steps: In line 1 we compute a triangulation of P which takes time $O(n)$ [6].

In line 2 we compute all different orders of diagonals for different starting points which takes time $O(n^2)$. In line 3 a single free space diagram is computed in time $O(mn)$. In line 4 the reachability graph is computed from the reachability structure of Alt and Godau [3], reviewed in 4.2, for all blocks. Since each block consists of one or two columns this takes time $O(m^2)$ per block, and $O(m^2n)$ for all blocks.

In lines 5–9 we test for all diagonals their possible placements in the free space. For any end point of a diagonal the free space diagram specifies up to m intervals on the boundary of the other polygon onto which the end point may be mapped. In the free space diagram these are the free space intervals on the vertical line corresponding to the end point. A placement of a diagonal is a mapping of its end points onto these intervals. In lines 6–8 we test which of these placements are valid, i.e., map the diagonal to a shortest path with Fréchet distance at most ε as described in Section 4.6.

In lines 10–26 we loop over all blocks (which is equivalent to looping over all orders of diagonals). The current block is always the starting block of the free space diagram and determines the order of diagonals. We compute the combined reachability graph for this order by merging the reachability graphs for the blocks of the free space diagram according to the order.

Each diagonal is processed as follows:

- (1) If the current diagonal does not have another diagonal nested inside it we compute its combined reachability graph from scratch (line 17). For this we take its reachability graph which we computed in 4 and check for each of its $O((mn)^2)$ edges whether the hourglass between the two intervals contains valid placements for the diagonal by looking up the precomputed result for the containing hourglass between the free space intervals. Either all placements in the hourglass are valid, or none. If none are valid then the edge is deleted, otherwise it is kept.
- (2) If the previous diagonal is nested in the current diagonal we merge the computed combined reachability graph for the previous diagonal into the combined reachability graph for the current diagonal (line 14) as follows: We merge the combined reachability graph of the nested diagonal with the reachability graph of the adjacent block that yields the reachability graph for the current diagonal. The merged graph is the transitive closure of the two graphs which again has complexity $O((mn)^2)$. On the merged graph we check for each edge whether it allows a valid placement of the current diagonal as described in step (1).

If the diagonal has a left neighbor (line 22) then we merge the two combined reachability graphs by computing their transitive closure. *Left neighbors* are for instance the first and fourth diagonals in the first order shown in Figure 3. More formally, we call (d_i, d_j) a left neighbor of (d_k, d_l) if $i < j \leq k < l$ and no diagonal (d_f, d_h) exists with either $f \leq i \wedge j \leq h \leq k$ or $j \leq f \leq k \wedge l \leq h$.

The transitive closure can be computed by multiplying the adjacency matrices of the combined reachability graphs (using boolean operations). Therefore the time complexity of the loop over the diagonals in lines 11–24 is proportional to the time complexity of $O(n)$ matrix multiplications of $mn \times mn$ matrices since for each of the n diagonals we merge at most 3 combined reachability structures (or compute one from scratch in $O((mn)^2)$ time).

The loop over all blocks does not increase the complexity because we store and re-use the combined reachability graphs of subresults. In the double loop over all blocks and diagonals we compute the combined reachability graph for each diagonal only twice.

In line 25 we query for a feasible path starting at the block’s lower boundary, i.e., in any of the $O(m)$ intervals in the reachability structure, as described in Section 4.4. For this we first merge the reachability graph of the single block to the front and to the end of the combined reachability graph which can be done using two matrix multiplications. This adds a total of $O(n)$ matrix multiplications for the whole algorithm. Then we have to query for feasible paths starting at the blocks lower boundary. This is done in $O(m)$ time by checking whether each of the $O(m)$ intervals is connected by an edge in the combined reachability graph to the same interval translated by n on the second half of the upper boundary.

To see that our algorithm is correct we must realize that a feasible path in the free space exists if and only if our algorithm finds such. For this, consider the main part of the algorithm, lines 10 to 26. In the outer loop, lines 10 to 26, we consider all possible starting blocks. Then we loop over all diagonals, according to the order given by the starting block, in lines 11 to 24, and process each diagonal in lines 12 to 23. The processing of each diagonal is simply a merging of the reachability information. More crucial is to realize that the dynamic programming over the diagonals does not exclude feasible paths. This holds because the dynamic programming excludes exactly those paths that place diagonals in such a way that the corresponding shortest paths cross. More specifically, for two nested diagonals, the dynamic programming prohibits placing an endpoint of the inner diagonal ”outside” the outer diagonal. Similarly for two neighboring diagonals, the dynamic programming prohibits placing the endpoints in alternating order. The formal proof of the correctness is deferred to a full version of the paper. \square

4.8 Critical Values for Computation

In order to compute the Fréchet distance instead of only deciding whether it is at most ε , we apply the same technique as for curves [3]: We search a set of critical values using parametric search. In addition to the critical values of the boundary curves we consider critical values for the Fréchet distance between diagonals and shortest paths.

A shortest path is always a polygonal curve where the first and last vertex are arbitrary points on the boundary

of Q and all other vertices are vertices of Q . The distances between the diagonal end points and the boundary of Q are already contained in the critical values for the boundary curves.

Additional critical values can only occur if the Fréchet distance between a diagonal and a shortest path is attained in the interior of the diagonal and the shortest path, i.e., it is attained at a vertex of Q . For the parametric search we sort the interval endpoints in the free space diagram [3]. We get $m \cdot n$ such endpoints, one for any pair of a diagonal and a vertex of Q . In total this yields the following theorem:

THEOREM 10. *The Fréchet distance between two simple polygons can be computed in time $O(nT(mn) \log(mn))$, where $T(N)$ is the time to multiply two $N \times N$ matrices, and n and m are the number of vertices on the boundary of P and Q .*

5. OPEN PROBLEMS

We see two major open problems: improving the runtime and extending the result to more general classes of two-dimensional surfaces. A better runtime might be achieved by a more efficient data structure and merging step for the combined reachability information. In particular, it would be nice to achieve a symmetric runtime. It might also be possible to improve the runtime by carefully choosing the triangulation of P or considering convex partitions instead of triangulations. For extending the result to more general surfaces, triangulated polygons that have been folded along their diagonals into \mathbb{R}^3 are possible candidates.

Acknowledgements

We thank Boris Aronov for suggesting the problem, and we thank Peter Brass and Helmut Alt for fruitful discussions.

6. REFERENCES

- [1] H. Alt, B. Behrends, and J. Blömer. Approximate matching of polygonal shapes. *Ann. Math. Artif. Intell.*, 13:251–266, 1995.
- [2] H. Alt and M. Buchin. Semi-computability of the Fréchet distance between surfaces. In *Proc. 21st European Workshop on Computational Geometry*, pages 45–48, 2005.
- [3] H. Alt and M. Godau. Computing the Fréchet distance between two polygonal curves. *Internat. J. Comput. Geom. Appl.*, 5:75–91, 1995.
- [4] H. Alt and L. Guibas. Discrete geometric shapes: Matching, interpolation, and approximation. In J.-R. Sack and J. Urrutia, editors, *Handbook of Computational Geometry*, pages 121 – 153. Elsevier Science Publishers B.V. North-Holland, Amsterdam, 1999.
- [5] H. Alt, C. Knauer, and C. Wenk. Comparison of distance measures for planar curves. *Algorithmica*, 38(1):45–58, 2004. Special Issue on Shape Algorithmics.
- [6] B. Chazelle. Triangulating a simple polygon in linear time. *Discrete Comput. Geom.*, 6(5):485–524, 1991.
- [7] D. Coppersmith and S. Winograd. Matrix multiplication via arithmetic progressions. *Journal of Symbolic Computation*, 9:251–280, 1990.
- [8] M. Fréchet. Sur la distance de deux surfaces. *Ann. Soc. Polonaise Math.*, 3:4–19, 1924.
- [9] M. Godau. *On the complexity of measuring the similarity between geometric objects in higher dimensions*. PhD thesis, Freie Universität Berlin, Germany, 1998.
- [10] L. Guibas, J. Hershberger, D. Leven, M. Sharir, and R. Tarjan. Linear time algorithms for visibility and shortest path problems inside simple polygons. In *Proc. 2nd Ann. Symp. on Computational Geometry*, pages 1–13, New York, NY, USA, 1986. ACM Press.
- [11] T. Rado. *Length and area*. Amer. Math. Soc. Colloq. Publ. Vol. 30, New York, 1948.
- [12] R. Veltkamp and M. Hagedoorn. State-of-the-art in shape matching. In M. Lew, editor, *Principles of Visual Information Retrieval*. Springer, 2001.