Computer-Aided Design 43 (2011) 1449–1456

Contents lists available at SciVerse ScienceDirect

Computer-Aided Design

journal homepage: www.elsevier.com/locate/cad

Computing shortest words via shortest loops on hyperbolic surfaces

Xiaotian Yin [a,*], Yinghua Li [d], Wei Han [a], Feng Luo [b], Xianfeng David Gu [c], Shing-Tung Yau [a]

[a] Mathematics Department, Harvard University, 1 Oxford Street, Cambridge, MA 02138, USA
[b] Mathematics Department, Rutgers University, 110 Frelinghuysen Rd., Piscataway, NJ 08854, USA
[c] Computer Science Department, Stony Brook University, Stony Brook, NY 11794-4400, USA
[d] Department of Applied Mathematics and Statistics, Stony Brook University, Stony Brook, NY 11794-3600, USA

ARTICLE INFO

Keywords:
Shortest word
Shortest loop
Fundamental group
Hyperbolic metric
Universal covering space

ABSTRACT

Given a loop on a surface, its homotopy class can be specified as a word consisting of letters representing the homotopy group generators. One of the interesting problems is how to compute the shortest word for a given loop. This is an NP-hard problem in general. However, for a closed surface that allows a hyperbolic metric and is equipped with a canonical set of fundamental group generators, the shortest word problem can be reduced to finding the shortest loop that is homotopic to the given loop, which can be solved efficiently. In this paper, we propose an efficient algorithm to compute the shortest words for loops given on triangulated surface meshes. The design of this algorithm is inspired and guided by the work of Dehn and Birman–Series. In support of the shortest word algorithm, we also propose efficient algorithms to compute shortest paths and shortest loops under hyperbolic metrics using a novel technique, called transient embedding, to work with the universal covering space. In addition, we employ several techniques to relieve the numerical errors. Experimental results are given to demonstrate the performance in practice.

1. Introduction

1.1. Motivations

Two loops on a surface are homotopic if one can be deformed to the other continuously on the surface. The equivalent classes of homotopic loops form a group, the so-called *fundamental group* of the surface, which is one of the most important topological invariants of the surface. Given a basis (i.e. a set of generators) of this group, any loop can be expressed as a product (or a word) of these generators. This word conveys homotopy information of the loop and has profound importance in topology [1–4].

The generators of the fundamental group have non-trivial relations in general, thus the word representation for a loop is not unique. For example, the loop in Fig. 2 (left) could be represented as either $a_1a_1b_2a_2^{-1}b_2^{-1}a_2b_1b_1^{-1}b_1a_1a_1^{-1}$ or $a_1b_1a_1$. Such ambiguities give rise to the needs of computing the shortest word representation for a loop, which is a highly challenging problem.

The goal of this work is to develop efficient and practical algorithms to actually compute the shortest words on triangulated surfaces, which are commonly used in computer graphics and geometric modeling society. In particular, we focus on high genus oriented closed surfaces; such surfaces admit hyperbolic uniformization metrics and have some special topological properties that we can utilize in the computation.

1.2. Related work

This problem has been studied in both mathematical and computational communities. This problem was essentially solved in theory by Dehn [3,1] for general groups using algebraic and topological methods. Later on Birman and Series [4] revisited Dehn's algorithm and brought geometric insights into this topological problem. According to [4], the shortest word problem can be converted to a shortest path problem if two conditions are satisfied.

First, the surface S has to be equipped with a hyperbolic metric with constant curvature -1 everywhere. There are mature results on this topic in the literature. The Ricci flow is one of the most powerful tools to compute metrics based on prescribed curvatures, which was first proposed by Hamilton in the study of 3-manifold topology and geometry [5] and later on for surfaces [6]. In [7] Chow and Luo developed the theories of combinatorial surface Ricci flow, which were later implemented and applied to surface parameterization in [8]. We use this method to compute a hyperbolic metric.

Second, a canonical basis must be given (Fig. 3). That is, we need a set of base loops that each represent a simple homotopy class and they intersect and only intersect at a single base point. There are

* Corresponding author. Tel.: +1 631 830 5555.
E-mail addresses: xyin@math.harvard.edu, xiaotian.yin@gmail.com (X. Yin), yinli@ic.sunysb.edu (Y. Li), weihan@math.harvard.edu (W. Han), fluo@math.rutgers.edu (F. Luo), gu@cs.sunysb.edu (X.D. Gu), yau@math.harvard.edu (S.-T. Yau).

0010-4485/$ – see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cad.2011.08.017

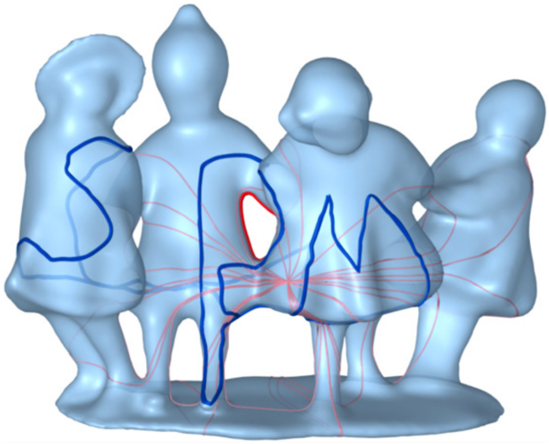


Fig. 1. Shortest word (red) for a general loop (blue) on a genus-8 hyperbolic surface with a canonical basis of 16 shortest loops (pink). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

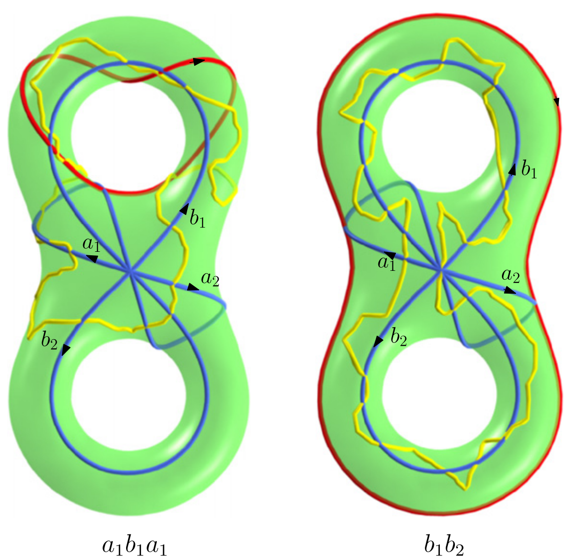


Fig. 2. Shortest words (red) for general loops (yellow) using hyperbolic metrics and a canonical geodesic basis (blue). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

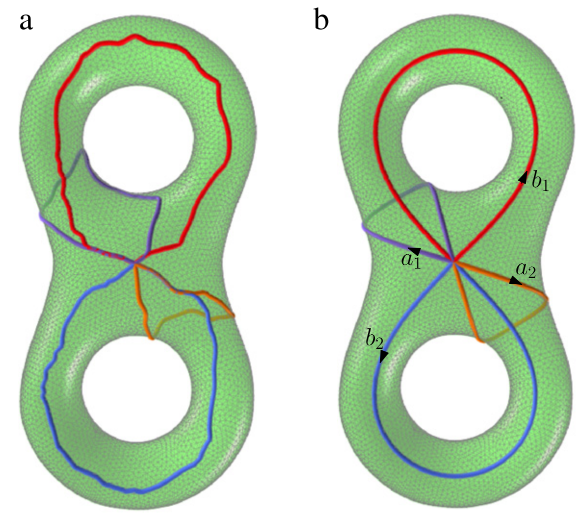


Fig. 3. General basis (a) vs canonical basis (b) of the fundamental group for a genus two surface.

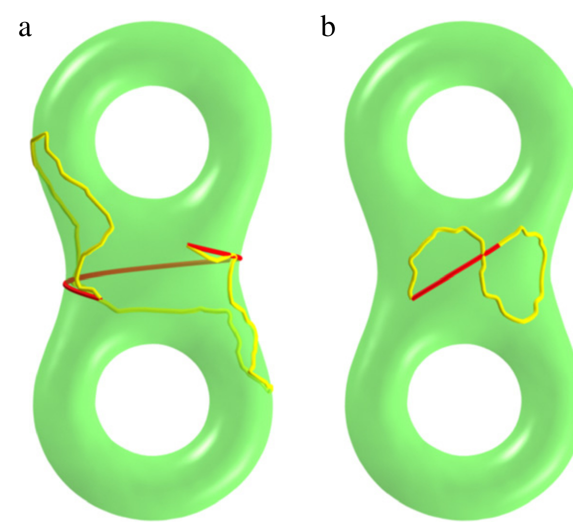


Fig. 4. Homotopic shrink of general paths (yellow) to shortest paths (red). Paths in (a) and (b) connect the same pair of end points, but they are not homotopic and therefore the shortest paths are not the same. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

several remarkable works related to this. To name a few, Colin de Verdière and Lazarus [9] proposed an algorithm to find such a set of optimal loops on orientable surfaces. Erickson and Whittlesey [10] gave a very efficient greedy algorithm to compute a system of shortest loops relaxing the homotopy condition. Dey et al. [11] proposed an algorithm to compute tunnel loops and handle loops that are both topologically correct and geometrically relevant. In this work we present an alternative method as a byproduct of our solution to the shortest path problem.

Our work is closely based on the work of Birman and Series [4], but is different to theirs in several aspects. First, our algorithm works on triangulated surfaces rather than smoothed surfaces, it takes into accounts the numerical errors that are unavoidable in actual computation and can be applied directly in such a discrete setting. Second, we focus on the fundamental group (rather than general groups) of a special type of hyperbolic surfaces where the fundamental group generators come in pairs as handle loop a_i and tunnel loop b_i. This avoids certain general groups that requires extra efforts to handle in [4] and can greatly simplify the proof. Furthermore, in this case the occurrence of a letter a_i (or b_i) in a word corresponds to an intersection of the loop in question with the paired base loop b_i (or a_i). This allows an algorithm that is simpler than [4] and more intuitive in geometry.

There are other algorithms that use algebraic methods to compute the shortest words. For example, Epstein et al. [12] proposed a symbolic algorithm that builds up an automaton based on the given relations of the generators and uses that to reduce a word that is given as input. Our algorithm instead is more geometric. We start from a given loop on a surface without knowing its initial word representation, end up with a shrunk loop and the shortest word representation.

Another related work is by Ferguson and Rockwood [13]. They looked at the word problems in terms of deck transformations in the universal covering space, and suggested to build a lookup table of all possible words on a genus g surface for every number g. While retrieving a word from the table could be very fast, it takes exponential time and space to build the table and therefore does not scale well for large g in practice. In contrast, our method only embeds a small neighborhood of the given loop in the universal covering space and is therefore not limited by the genus number g.

Regarding the computational complexity, finding the shortest word for an arbitrary group has been proved to be NP-hard by Parry [14] using a reduction to the Traveling Salesman Problem (TSP), although checking whether a word is equivalent to the identity is polynomial. However, it has been shown that the problem can be less hard in certain cases. Epstein et al. [2] proved that it takes polynomial time to find the length of the shortest word for free groups and hyperbolic groups. In this work, we show that the problem can be solved efficiently in our special setting if a hyperbolic metric is given.

Our work reduces the shortest word problem to shortest loop problem on surfaces, the latter has been studied by various researchers. Birkhoff [15] proposed a shortening process to approximate a loop with a piecewise geodesic polygon and update the polygon iteratively by connecting midpoints of edges with geodesic segments. Hass and Scott [16] extended Birkhoff's curve shortening process to a disk flow algorithm, and proved that for negatively curved close surfaces, this flow converges to the closed geodesic that is well known to be unique under hyperbolic metrics. In our shortest loop algorithm, we use a variation of the above process where the piecewise geodesic approximation has only one geodesic piece that both end points coincide, and the convergence of this algorithm is guaranteed by the results in [16].

1.3. Contributions

In this work, we propose an efficient and robust solution to the shortest word problem for the fundamental group of triangulated hyperbolic surfaces that are oriented and closed. We divide the problem into three sub-problems.

- The Shortest Path Problem: given a path on a triangulated surface S equipped with a hyperbolic metric, how to compute the shortest path that can be homotopically deformed to the original one with both end points fixed? And as a special case, what if the end points of the path coincide and form a loop based at a fixed point?
- The Shortest Loop Problem: given a loop on a triangulated surface S equipped with a hyperbolic metric, how to compute the shortest loop (under hyperbolic metric) that is homotopic to the original one?
- The Shortest Word Problem: on a triangulated surface S equipped with a hyperbolic metric and a canonical basis of the fundamental group, how to compute the shortest word representation for an arbitrary loop on S?

For each sub-problem, we provide solutions as follows:

- For the shortest path problem, we propose a linear algorithm (Section 3.1) that employs a novel *transient embedding* scheme to lift a path into the universal covering space and shrink it there. Compared to the previously published methods, this method only embeds the one-ring neighborhood of a given loop rather than multiple copies of the fundamental domain, and therefore only takes linear time and linear space. This basic algorithm is not only used to deform a non-canonical basis to a canonical one (with base point), but is also used to compute the shortest loop without base point.
- For the shortest loop problem, we employ a midpoint shortening algorithm (Section 3.2) that calls the shortest path algorithm iteratively. Due to the properties of hyperbolic metric, this algorithm converges quickly to the global minimum, i.e. the shortest loop.
- For the shortest word problem, we propose a simple yet effective algorithm (Section 3.1) based on the above two. We utilize the topological properties of high genus oriented closed surfaces to simplify the work in [4].

In addition, to handle the numerical errors that are unavoidable in real computations, we employ a series of techniques (Section 4) to ensure the robustness of the algorithm and the correctness of the final output. By all these considerations we try to bridge the gap between some beautiful results from classical topology and geometry and real computations in geometric modeling.

In the rest of the paper, we first introduce some background knowledge in Section 2, then explain the detailed algorithms in Section 3 and the numerical issues in Section 4, followed by a brief conclusion in Section 5.

2. Preliminary

In this section we briefly review some basic concepts in algebraic topology and Riemannian geometry. The interested readers should refer to text books such as [17,18] for details.

2.1. Fundamental group and representation of homotopy class

Let S be a topological surface, p be a point on S. Consider the set of continuous functions $f : [0, 1] \to S$ with the property $f(0) = p = f(1)$. These functions are called *loops with base point* p. Any two such loops, say f and g, are considered equivalent if there is a continuous function $h : [0, 1] \times [0, 1] \to S$ with the property that, for all $0 \leq t \leq 1$, $h(t, 0) = f(t)$, $h(t, 1) = g(t)$ and $h(0, t) = p = h(1, t)$. Such an h is called a *homotopy* from f to g, and the corresponding equivalence classes are called *homotopy classes*.

The product of two loops f and g is defined as $(f \cdot g)(t) := f(2t)$, if $0 \leq t \leq \frac{1}{2}$ and $(f \cdot g)(t) := g(2t - 1)$ if $\frac{1}{2} \leq t \leq 1$. Thus the loop $f \cdot g$ first goes along f at a double speed and then goes along g at double speed. The product of two homotopy classes for loop $[f]$ and $[g]$ is then defined as $[f \cdot g]$, and it can be shown that this product does not depend on the choice of representatives.

With the above product, the set of all homotopy classes of loops with base point p forms the *fundamental group* of S at point p and is denoted as $\pi_1(S, p)$ or simply $\pi(S, p)$. The identity element is the constant map at the base point, and the inverse of a loop f is the loop g defined by $g(t) = f(1 - t)$. That is, g follows f backwards.

Suppose S is a genus g closed surface. A *canonical set of generators* of $\pi(S, p)$ consists of

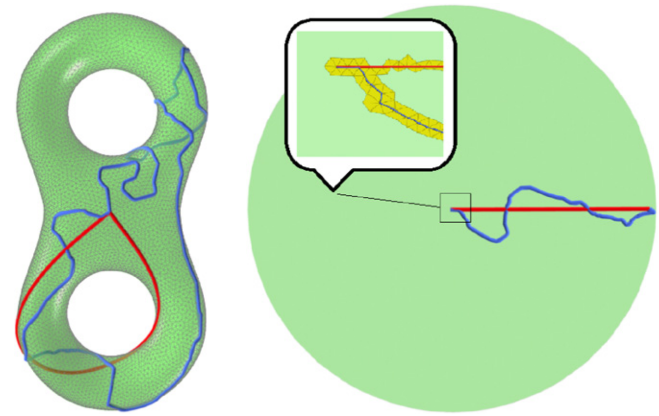
$$\{a_1, b_1, a_2, b_2, \ldots, a_g, b_g\}$$

such that any one of them intersects and only intersects with all the other generators at a single base point. See Fig. 3 for a comparison of canonical basis and non-canonical basis.

2.2. Universal covering space

A *covering space* of S is a space $\bar{S}$ together with a continuous surjective map $h : \bar{S} \to S$, such that for every $h \in S$ there exists an open neighborhood U of p such that $h^{-1}(U)$ (the inverse image of U under h) is a disjoint union of open sets in $\bar{S}$ each of which is mapped homeomorphically onto U by h. The map h is called the *covering map*. A connected covering space is a *universal covering space* if it is simply connected. Suppose $\gamma \subset S$ is a loop through the base point p on S. Let $\bar{p}_0 \in \bar{S}$ be a preimage of the base point $\bar{p}_0 \in h^{-1}(p)$, then there exists a unique path $\bar{\gamma} \subset \bar{S}$ lying over γ (i.e. $h(\bar{\gamma}) = \gamma$) and $\bar{\gamma}(0) = \bar{p}_0$. $\bar{\gamma}$ is a *lift* of γ. Fig. 5 shows a loop on a genus two surface and its lifting in the universal covering space.

A *deck transformation* of a cover $h : \bar{S} \to S$ is a homeomorphism $f : \bar{S} \to \bar{S}$ such that $h \circ f = h$. All deck transformations form a group, the so-called *deck transformation group*. A *fundamental domain* of S is a simply connected domain, which intersects each orbit of the deck transformation group only once. A fundamental domain can be obtained by slicing a surface S open along a canonical set of fundamental group generators. Deck transformations map fundamental domains to fundamental domains. The deck transformation



(a) S. (b) $\bar{S}$.

Fig. 5. General loop (blue) vs shortest loop (red) with a fixed base point on a hyperbolic surface (a), and their lifting in the universal covering space (b) where only the one-ring neighbors (yellow) of the paths are embedded. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

group $Deck(S)$ is isomorphic to the fundamental group $\pi_1(S, p)$. Let $\bar{p}_0 \in h^{-1}(p)$, $\phi \in Deck(S)$, $\bar{\gamma}$ is a path in the universal covering space connecting $\bar{p}_0$ and $\phi(\bar{p}_0)$, then the projection of $\bar{\gamma}$ is a loop on S, ϕ corresponds to the homotopy class of the loop, $\phi \to [h(\bar{\gamma})]$. This gives the isomorphism between $Deck(S)$ and $\pi_1(S, p)$.

2.3. Hyperbolic metric

The surface *uniformization theorem* postulates that any closed surface admits a special Riemannian metric, which induces constant Gaussian curvature, the constant is one of $\{+1, 0, -1\}$ depending on the Euler number. Such a metric is called the *uniformization metric*. The universal covering space of the surface with the uniformization metric can be isometrically embedded into the sphere $\mathbb{S}^2$, the plane $\mathbb{R}^2$ or the hyperbolic plane $\mathbb{H}^2$.

In this work, we use Poincaré disk to model the hyperbolic space $\mathbb{H}^2$, which is the unit disk $|z| < 1$ on the complex plane with the metric

$$ds^2 = \frac{4dzd\bar{z}}{(1 - z\bar{z})^2}.$$

The rigid motion here is given by Möbius transformations

$$z \to e^{i\theta}\frac{z - z_0}{1 - \bar{z}_0 z}.$$

A hyperbolic circle $(\mathbf{c}, r)$ is also a Euclidean circle $(\mathbf{C}, R)$ with

$$\mathbf{C} = \frac{2 - 2\mu^2}{1 - \mu^2\mathbf{c}\bar{\mathbf{c}}}\mathbf{c}, \qquad R^2 = \mathbf{C}\bar{\mathbf{C}} - \frac{\mathbf{c}\bar{\mathbf{c}} - \mu^2}{1 - \mu^2\mathbf{c}\bar{\mathbf{c}}},$$

where $\mu = \tanh\frac{r}{2}$.

Given two points p and q on $\mathbb{H}^2$, the unique geodesic in between is a circular arc through them and perpendicular to the unit circle.

3. Algorithms

This section is devoted to the algorithmic details of computing shortest words. We first present a linear algorithm to compute the shortest path homotopic to a given one (Section 3.1). Then we present an algorithm to compute the shortest loop (without any fixed point) homotopic to a given loop (Section 3.2), which utilizes the shortest path algorithm. In the end we show an algorithm to compute the shortest word of a given loop over a canonical basis (Section 3.3); this algorithm calls the shortest path algorithm to shrink non-shortest basis and calls the shortest loop algorithm to shrink the given loop.

In all these algorithms, we suppose the input surface is triangulated as $S = (V, E, F)$, and a hyperbolic metric is also given $L : E \to R^+$. As a remark, we are looking for paths or loops that are shortest under the given hyperbolic metric rather than the original Euclidean metric on S.

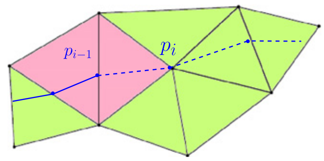
3.1. Computing shortest paths

Computing the shortest path is a basic procedure that is used extensively in later computations. Here we present a simple yet efficient algorithm to shrink a path homotopically. The input to the algorithm is a path $\gamma = p_1p_2\cdots p_n$ on S, where p_i is either a vertex $v_j \in V$ or on an edge $e_k \in E$. The output is a new path $\gamma_s = q_1q_2\cdots q_m$ that is homotopic to γ and is shortest between p_1 and p_n ($q_1 = p_1$ and $q_m = p_n$).

The algorithm works on both the surface S and its universal covering space $\bar{S}$. It follows a common philosophy in topology: lift the given path $\gamma \subset S$ to $\bar{\gamma} \subset \bar{S}$ (steps 1–3) and shrink it in $\bar{S}$ (steps 4–7). But the novelty of this algorithm is that it uses a transient embedding scheme for both the lifting and shrinking process. As a consequence only the one-ring neighbors of γ and $\bar{\gamma}$ need to be embedded in $\bar{S}$. The fundamental domain and its copies are conceptually used but not actually embedded in this work.

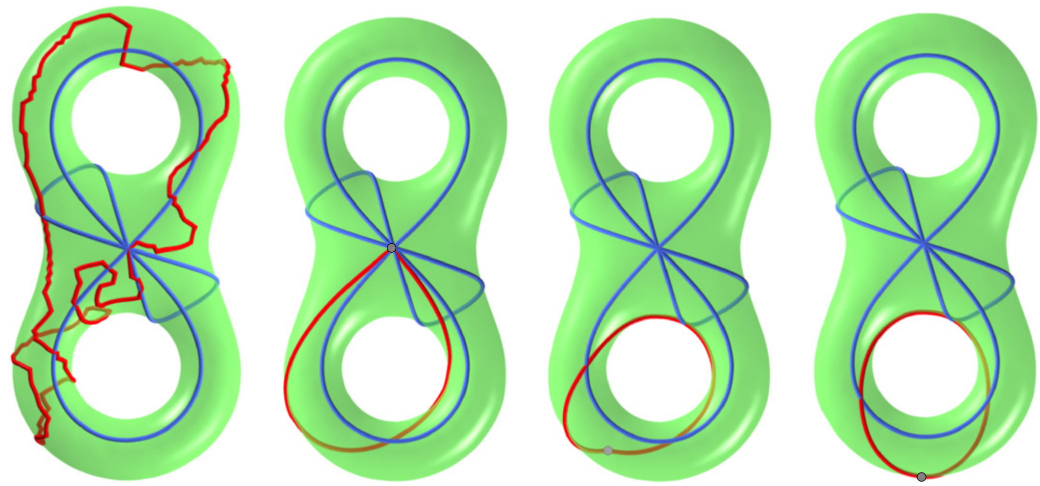
Algorithm 3.1 (*The Shortest Path Algorithm*).

1. Embed one-ring neighbor $N(p_1)$ into $\bar{S}$;
2. Repeat the following for each remaining point p_i on the path:
   (a) Address its one-ring neighbor $N(p_i) = f_{i,1}, , f_{i,d_i}$.
   (b) For each $\bar{f}_j \in N(\bar{p}_i)$, ignore it if it is embedded in the previous iteration or embed it otherwise.
3. Record the coordinates of $\bar{p}_1$ and $\bar{p}_n$.
4. Apply a rigid motion to $\bar{S}$ to align $\bar{p}_1$ with the origin and $\bar{p}_n$ into the X-axis; rename them as $\bar{q}_1$ and $\bar{q}_n$.
5. Re-embed $N(\bar{q}_1)$.
6. Trace out a ray $\bar{q}_1\cdots\bar{q}_m$ along the X-axis from $\bar{q}_1$ (i.e. the origin) toward $\bar{q}_n$; do the following for each $\bar{q}_j$:
   (a) Embed the un-embedded part of the one-ring neighbor $N(\bar{q}_j)$ around $\bar{q}_j$;
   (b) Advance the ray from $\bar{q}_j$ along X-axis until it hits a point $\bar{q}_{j+1}$ on the boundary of $N(\bar{q}_j)$;
7. Denote the trace ray as $\bar{\gamma}_s = \bar{q}_1\cdots\bar{q}_m$. Project it back onto S as $\gamma_s = q_1q_2\cdots q_m$.



(a) $i-1$th. (b) ith.

Fig. 6. Transient embedding in two consecutive iterations. Faces in red are marked as "embedded" at that moment, while those in green are marked as "un-embedded" and allow (re-)embedding in the next iteration. Note that some red faces in the $(i-1)$th iteration become green in the ith iteration, which distinguishes the transient embedding from the previously published embedding methods.



(a) Original. (b) 1 iteration. (c) 2 iterations. (d) 4 iterations.

Fig. 7. The midpoint (marked gray) shortening process converges quickly to the shortest loop without fixed point.

This algorithm follows a similar work flow of previous published embedding algorithms to construct universal covering space. It first embeds a small number of faces (one face or a one-ring neighbor) at specific positions (usually around the origin) using the given edge lengths (i.e. hyperbolic metric). Then the remaining faces are embedded using a breadth-first search, so that for each face f to be embedded, there must be two vertices v, $v' \in f$ that are already embedded and only the third one v'' needs to be considered. In this process, all the faces and vertices that are already embedded are marked as "embedded"; any face with this mark will be skipped from being embedded again.

However, there is a fundamental difference between our algorithm and other existing ones. In a previously published algorithm, the "embedded" marker is kept permanently and globally in a fundamental domain; once a face is marked, it will remain marked until the whole fundamental domain is embedded. On the other hand, our algorithm only keeps this marker transiently and locally. It means that an "embedded" marker is only kept for two consecutive iterations and only within a neighborhood of up to two points (see Fig. 6). Once the iteration for p_i is done, the markers for faces in $N(p_{i-1}) - N(p_i)$ (i.e. those exclusively embedded in the iteration for p_{i-1}) will be cleared, and these faces will be treated as "un-embedded" and therefore become immediately available to be re-embedded.

We name such a scheme *transient embedding*. When a path is lifted from S into $\bar{S}$, only its one-ring neighbor needs to be embedded in $\bar{S}$. Therefore the algorithm is linear in both time and space complexity with respect to the length of the given path. It avoids embedding one or more copies of the fundamental domain that are traversed by the path being lifted.

Nevertheless, transient embedding does not sacrifice the homotopy invariance during a path lifting. One possible concern is that, by keeping such a small window for embedded faces, a homotopically non-trivial loop might be falsely lifted as a trivial one. This case most possibly happens around a tiny handle loop (e.g. Fig. 9) or a tiny tunnel loop (e.g. fig] Fig. 8). However, in a normal triangular surface mesh, any handle or tunnel would be covered by at least three triangles, which is greater than our window size that is at most two. Therefore the transient scheme will always extend a loop correctly on a non-degenerate mesh.

Once the input path is correctly traced in the universal covering space $\bar{S}$, the only information we need is the position of its end points $\bar{p}_1$ and $\bar{p}_n$, because this determines the homotopy class. And the shortest path we are looking for is now just a geodesic (i.e. a Euclidean arc connecting $\bar{p}_1$ and $\bar{p}_n$). To simplify the computation, we apply a rigid transformation to the whole universal covering space (conceptually) so that this geodesic becomes a straight line segment between the origin and some point on the positive X-axis. Then we just trace out this shortest path $\bar{\gamma}_s$ along X-axis, and embed the one-ring neighbors on the fly so that it can be projected back onto S. In this process, we use transient embedding again.

This algorithm not only works for open paths (Fig. 4), but also works for loops with one point fixed (Fig. 5). And in the latter case, this algorithm can be used to shrink base loops if they are not given as geodesic ones. In fact, if all the base loops are shrunk this way with the same base point fixed for each loop, it can be concluded that this base point will be the only intersection point among these base loops, which is a natural consequence of using hyperbolic metrics.

3.2. Computing shortest loops

The above shortest path algorithm can be used to shrink a loop, but has to fix one point in the loop. In many cases, such as the shortest word computation, we need a shortest loop without any point fixed. Here we propose a shortest loop algorithm that uses the shortest path algorithm iteratively. The input to this algorithm is a general loop $\gamma = p_1 p_2 \cdots p_n p_1$ on the original surface. The

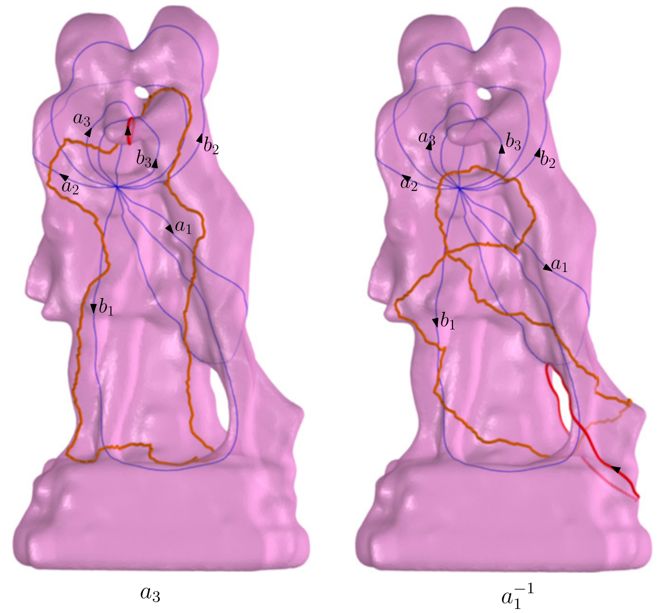


Fig. 8. Shortest words on a genus-3 sculpture model that has a tiny tunnel loop along the arms. The algorithm can handle original loops with self-intersections.

output is a loop $\gamma_S = q_1 q_2 \cdots q_n q_1$ that is homotopic to γ and is shortest under the hyperbolic metric associated with S.

Algorithm 3.2 (*The Shortest Loop Algorithm*).

1. Let $p = p_1$ be the initial fixed point.
2. Repeat the following on γ until its length becomes stable:
   (a) Shrink γ keeping p fixed using the shortest path algorithm in Section 3.1;
   (b) Check the length of γ (under hyperbolic metric), and stop the iteration if its difference to the previous iteration falls below a threshold;
   (c) Set p to a new fixed point on γ and go to the next iteration;
3. Set $\gamma_S = \gamma$ and exit.

In this algorithm, we construct a sequence of shortest loops with fixed points to approach the shortest loop without fixed points. In each iteration, we shrink γ with fixed point p; pick a new point $p' \neq p$ to serve as the fixed point and let p be free to deform. This process is actually a variation of the midpoint shortening process in [15], which has been proved in [16] to converge to the unique shortest loop under hyperbolic metrics and will not get stuck in any local minimum.

One thing that needs care is the choice of the fixed point, which will impact the speed of the convergence. As an easy observation, if we pick a new fixed point p' very close to the old one p, the next iteration of shrinking will not bring much changes to the shape of the loop. In contrast, if we take the point farthest to p, the algorithm will converge very quickly. In other words, the new fixed point p' should be the midpoint of the geodesic path that starts from p and loops back to p. Fig. 7 shows the convergence of this process on a genus two surface.

3.3. Computing shortest words

We have shown that the shortest path algorithm (Section 3.1) can deform a non-canonical set of base loops to shortest loops that intersect at a single base point. We have also shown that a general loop can be shrunk to a homotopic shortest loop without any fixed point using the shortest loop algorithm (Section 3.1). Now we can use the following algorithm to find the shortest word representation for the given loop under this basis.

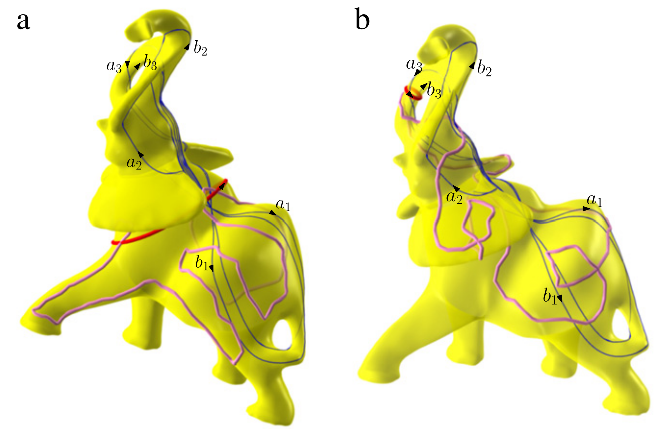


Fig. 9. Shortest words on a genus-3 elephant model that has tiny handle loops around the tail and the teeth.

The input to the algorithm is a set of shortest loops $\mathbf{B} = \{a_1, b_1, \ldots, a_g, b_g\}$ that only intersect at a base point p, and another shortest loop $\gamma_S = p_1 p_2 \cdots p_n p_1$. The output is a shortest word representation of γ. Note that all the loops are directed ones on S, and the algorithm works completely on S and does not need $\bar{S}$ any more.

Algorithm 3.3 (*The Shortest Word Algorithm*).

1. Let $\mathbf{W}$ be the word representation of γ_S, initialized as empty.
2. Go through γ_S to find all its intersections with loops in $\mathbf{B}$; save these intersection points in order as $\mathbf{S} = \{s_1, s_2, \ldots, s_k\}$.
3. For each intersection $s_i \in \mathbf{S}$ do the following:
   (a) If $s_i \neq p$, append a corresponding letter to $\mathbf{W}$ (explained in below).
   (b) Otherwise, perturb γ_S to go around p from the side with less (or equal) base loops, and append corresponding letters $\mathbf{W}$ in the order of intersections.
4. Output word $\mathbf{W}$.

In this algorithm, we need to append a corresponding letter C to the word $\mathbf{W}$ whenever the loop encounters a base loop, where the letter C could be any one from

$$\{a_1, b_1, \ldots, a_g, b_g, a_1^{-1}, b_1^{-1}, \ldots, a_g^{-1}, b_g^{-1}\}.$$

Note that for high genus closed surface, the fundamental group generators always appear in pairs of (a_i, b_i). A fact is that when a loop crosses one generator, it includes the paired generator in its word. Based on this observation, the letter C is determined as follows. Keep in mind that all the loops are directed.

- If γ_S crosses a_i from the left, set $C = b_i$;
- If γ_S crosses a_i from the right, set $C = b_i^{-1}$;
- If γ_S crosses b_i from the left, set $C = a_i$;
- If γ_S crosses b_i from the right, set $C = a_i^{-1}$.

We show several examples of shortest words in Figs. 1, 2, 8 and 9.

4. Numerical robustness

Our algorithms work with edge lengths of triangulated surfaces and are therefore subject to numerical errors, which is unavoidable for any numerical method of the same nature. For example, the given hyperbolic metric may not be a hundred percent accurate, the floating point representations with limited bits will cause accumulated truncation errors, etc. However, it can be shown that our shortest word computation is robust under numerical inaccuracy. This is partly due to the fact that although we use geometric information in the algorithm, the problem itself is a topological one. On the other hand, in the computation of shortest paths and shortest words, we employ a couple of techniques to relieve the numerical errors (Sections 4.1–4.4). We illustrate the algorithm robustness using a wide range of experiments, covering surfaces with various topological invariants and geometric features and loops with different homotopy classes and geometric complexities.

4.1. Floating point representations

Our algorithms do a lot of computations using real numbers, such as the edge lengths of a triangular mesh S, the point coordinates in the universal covering space $\bar{S}$, and etc. In computers these real numbers can only be approximated by floating-point numbers with a limited amount of bits, such as 32 bits or 64 bits, which will give rise to truncation errors and their accumulation throughout the whole process.

It turns out this issue is especially important to our algorithms, which uses the Poincaré disk as the hyperbolic model to embed part of the universal covering space. In such a model, as a point gets close to the boundary of the unit disk, the metric around it will become highly compressed and the mesh will appear extremely dense there. When we trace or shrink a path from a point around the origin out to a point close to the disk boundary, the higher precision we use to represent a floating-point number, the more accurate we can embed the one-ring neighbors and locate the paths. In this work, we use 64-bit representations whenever real number computations are involved.

4.2. Accuracy of hyperbolic metrics

As an input to the algorithm, a hyperbolic metric is supposed to have a constant curvature of -1 everywhere on S so that the universal covering space $\bar{S}$ can be accurately embedded in the Poincaré disk. However, inaccuracies are unavoidable in all existing numerical methods for computing a hyperbolic metric on a triangulated surface. In fact, the larger the error is, the more impact it will produce to the accuracy of the paths traced in the Poincaré disk. In this work, we use the discrete curvature flow method introduced in [7] to produce a hyperbolic metric, where the error can be well controlled by setting an upper bound threshold as an input. In the experiments, a threshold of 10^{-7} or lower is used to achieve satisfactory results.

4.3. Accuracy of shortest paths

By using high resolution floating point representations and highly accurate hyperbolic metrics, numerical errors can be largely reduced but still exist to impact the shortest path computation. In particular, when we lift a path $\gamma = p_1 \cdots p_n$ (starting from p_1) in the universal covering space $\bar{S}$, numerical errors could accumulate during tracing and result in inaccurate coordinates for the last point in $\bar{S}$; therefore if we project the shrunk path back to S as $\gamma_s = q_1 \cdots q_m$, the new ending point q_m may not coincide with the original one p_n.

In order to remove this gap, we propose the following *double tracing* technique. In a double tracing, we trace the given path twice in opposite directions:

$$\gamma = p_1 \cdots p_n$$
$$\gamma^{-1} = p_n \cdots p_1$$

shrink them in $\bar{S}$ separately and project them back onto S respectively:

$$\gamma_A = q_1 \cdots q_m$$
$$\gamma_B = q'_{m'} \cdots q'_1.$$

Our shortest path algorithm guarantees that the original path γ will coincide with γ_A at $q_1 = p_1$ and with γ_B at $q'_{m'} = p_n$. Now we construct a new path γ_s that interpolates these two paths as follows.

On the original surface S, we trace a path τ_1 connecting q'_1 to q_1 and path τ_2 connecting q_m to $q'_{m'}$ by a simple breadth-first search starting from q'_1 and q_m respectively. Note that in practice τ_1 and τ_2 will be very short and $\gamma_A\tau_2\gamma_B\tau_1$ will bound a thin strip of surface patch $S_s \subset S$. Then we embed S_s into the Poincaré disk as $\bar{S}_s$, where γ_A and γ_B is lifted to $\bar{\gamma}_A$ and $\bar{\gamma}_B$ respectively. In $\bar{S}$ we parameterize these two paths uniformly by t ($0 \leqslant t \leqslant 1$) (i.e. arc length parameterization) and interpolate them linearly within $\bar{S}_s$:

$$\bar{\gamma}_s(t) = (1 - t) \cdot \bar{\gamma}_A(t) + t \cdot \bar{\gamma}_B(1 - t).$$

Projecting $\bar{\gamma}_s$ back onto S, the resulting path γ_s will connect the original end points p_1 and p_n seamlessly and can serve as the final shortest path between them. Note that as a special case, this technique also applies to a loop that starts from and ends at the same base point.

4.4. Accuracy of shortest words

When computing the shortest word for a given loop, the homotopy class of the loop should be kept invariant. If due to numerical errors a loop deviates from the right position, it might result in a wrong homotopy class. This impact is especially severe if the loop passes near the base point. To avoid this, in step 3 of the shortest word algorithm, we not only perturb the loop when it passes through the base point, but also check and consider a perturbation when the loop is close enough to the base point.

Once this is resolved, the algorithm will be much more robust to such deviation errors. This is because finding the shortest word itself is still a combinatorial problem; in order to result in a false homotopy class, the deviation has to be bigger than the diameter of the fundamental domain. But after all the above error reductions, this will rarely happen on a triangulated surface that is sufficiently subdivided and has no extremely thin or extremely tiny handles.

5. Conclusion

In this work we propose a numerical solution to finding the shortest words in the fundamental group of triangulated hyperbolic surfaces that are closed and oriented. The basic philosophy is to use geometric approaches to solve a topological problem. In particular, we develop a transient embedding scheme to compute shortest paths on surfaces, which only requires embedding the one-ring neighbor of the given path in the universal covering space and is therefore linear (with a small coefficient) in the length of the path. Then a variation of the Birkhoff midpoint shortening process is proposed to deform a piecewise geodesic loop (with only one piece actually) iteratively, which is guaranteed to converge to a shortest loop that is the unique global minimum. Finally a shortest word representation of a given loop is induced from its shortest loop image using a simplified version of the revised Dehn's algorithm. Numerical accuracy is taken into the consideration throughout the whole process. Several techniques are used to relieve numerical errors and increase the robustness.

Although only high genus closed surfaces are considered in this work, the algorithms proposed here can be potentially extended to cover other surfaces (e.g. with boundaries) that also admit hyperbolic metrics, which will be an interesting topic to explore in the future.

Acknowledgments

The authors would like to acknowledge and thank the anonymous reviewers for their insightful and constructive comments. The first two authors were supported by Harvard University Mathematics Department Funds. This work was also partially supported by ONR N000140910228, NSF CCF-1081424, NSF Nets 1016829, NIH R01EB7530 and DARPA 22196330-42574-A.

References

[1] Dehn M. Transformation der Kurven auf zweiseitigen Flächen. Mathematische Annalen 1912;72(3):413–21.
[2] Epstein DBA, Cannon JW, Holt DF, Levy SVF, Patterson MS, Thurston WP. Word processing in groups. AK Peters; 1992. ISBN 978-0-86720-244-1.
[3] Dehn M. Über unendliche diskontinuierliche Gruppen. Mathematische Annalen 1911;71(1):116–44.
[4] Birman JS, Series C,. Dehn's algorithm revisited with applications to simple curves on surfaces. In: Combinatorial group theory and topology. Utah: Alta; 1987. p. 451–78.
[5] Hamilton RS. Three-manifolds with positive Ricci curvature. Journal of Differential Geometry 1982;17:255–306.
[6] Hamilton RS. The Ricci flow on surfaces. In: Mathematics and general relativity (Santa Cruz, CA, 1986). Contemporary Math, Providence, RI: American Math Society; 1988. p. 71.
[7] Chow B, Luo F. Combinatorial Ricci flows on surfaces. Journal of Differential Geometry 2003;63(1):97–129.
[8] Jin M, Kim J, Luo F, Gu X. Discrete surface Ricci flow. IEEE Transaction on Visualization and Computer Graphics 2008;14(5):1030–43.
[9] de Verdière ÉC, Lazarus F. Optimal system of loops on an orientable surface. In: Proceedings of the 43rd annual IEEE symposium on foundations of computer science. 2002. p. 627–36.
[10] Erickson J, Whittlesey K. Greedy optimal homotopy and homology generators. In: Proceeding of the 16th annual ACM-SIAM symposium on discrete algorithms. 2005. p. 1038–46.
[11] Dey TK, Li K, Sun J, Cohen-Steiner D. Computing geometry-aware handle and tunnel loops in 3D models. ACM Transactions on Graphics 2008;27(3):1–9.
[12] Epstein DBA, Holt DF, Rees SE. The use of Knuth–Bendix methods to solve the word problem in automatic groups. Journal of Symbolic Computation 1991; 12(4–5):397–414.
[13] Ferguson H, Rockwood A. Multiperiperidic functions for surface design. Computer Aided Geometric Design 1993;10:315–28.
[14] Parry W. Growth series of some wreath products. Transactions of the American Mathematical Society 1992;331(2):751–9.
[15] Birkhoff GD. Dynamical systems with two degrees of freedom. Transaction of American Mathematical Society 1917;18:199–300.
[16] Hass J, Scott P. Shortening curves on surfaces. Topology 1994;33(1):25–43.
[17] Petersen P. Riemannian geometry. Springer; 1997.
[18] Hatcher A. Algebraic topology. Cambridge University Press; 2006.