Hyperbolic Centroidal Voronoi Tessellation

Guodong Rong* University of Texas at Dallas
Miao Jin† University of Louisiana at Lafayette
Xiaohu Guo‡ University of Texas at Dallas

Abstract

The centroidal Voronoi tessellation (CVT) has found versatile applications in geometric modeling, computer graphics, and visualization. In this paper, we extend the concept of the CVT from Euclidean space to hyperbolic space. A novel hyperbolic CVT energy is defined, and the relationship between minimizing this energy and the hyperbolic CVT is proved. We also show by our experimental results that the hyperbolic CVT has the similar property as its Euclidean counterpart where the sites are uniformly distributed according to given density values. Two algorithms – Lloyd's algorithm and the L-BFGS algorithm – are adopted to compute the hyperbolic CVT, and the convergence of Lloyd's algorithm is proved. As an example of the application, we utilize the hyperbolic CVT to compute uniform partitions and high-quality remeshing results for high-genus (genus$>$1) surfaces.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Geometric algorithms

Keywords: Centroidal Voronoi Tessellation, Hyperbolic Geometry, Geometric Structure, Universal Covering Space

1 Introduction

The Voronoi diagram is a well studied concept in computational geometry, and has a wide usage in different areas in geometric modeling, computer graphics, visualization, etc. [Okabe et al. 1999]. The *centroidal Voronoi tessellation* (CVT) is a special case of the Voronoi diagram, where every site coincides with the centroid of its Voronoi cell. The sites in a CVT are uniformly distributed. This property is conjectured by Gersho in 1979 [Gersho 1979], and has been proved for 2D cases [Fejes Tóth 2001].

In geometric modeling, many applications require a uniform sampling on a surface, or a partition of a surface where every region covers similar area. These tasks can be achieved simultaneously by computing a CVT on the surface where all sites are constrained on the surface. Such a CVT is usually known as the *constrained CVT* (CCVT) [Du et al. 2003]. It is natural to use the geodesic distance to compute the CCVT [Peyré and Cohen 2004], but it is difficult to compute the geodesic distance accurately. Another alternative is to use the 3D Euclidean distance as an approximation [Liu et al. 2009; Rong et al. 2010; Yan et al. 2009], but this may lead to disconnected Voronoi cells if two regions are very close in 3D space but are far apart along the surface. A better approach is to compute the CVT in a 2D parameterization domain of the surface [Alliez et al. 2005]. By assigning appropriate density values, the computed CVT is very close to the CCVT computed using the geodesic distance. This method overcomes the shortages of both prior methods, and is more efficient since the computation is performed in a 2D planar domain.

To parameterize a closed surface to 2D domain, the original surface has to be cut into a genus-0 surface. This makes the sites unable to cross the boundaries in the parameterization domain, and leads to visible artifacts along the cutting edges. In [Alliez et al. 2005], a great deal of special care and delicate strategies, such as minimizing the total cutting edge length and matching the cut graph with the feature skeleton, are required. But if the cut graph does not coincide much with a set of feature edges, the remeshing results become unacceptable for high genus surfaces as indicated in [Alliez et al. 2005].

This cutting problem can be solved by computing the CVT directly on the *universe covering space* (UCS) [Jin et al. 2006] of the surface. A covering space of surface S is a space $\overline{S}$ together with a continuous surjective map $h : \overline{S} \to S$, such that for every $p \in S$ there exists an open neighborhood U of p and $h^{-1}(U)$ (the inverse image of U under h) is a disjoint union of open sets in $\overline{S}$, each of which is mapped homeomorphically onto U by h. A simply connected covering space $\overline{S}$ is called a universal covering space (UCS). For closed genus-1 surfaces, their UCS can be embedded in 2D planar domain, so the computation of the CVT on the UCS is identical as in [Alliez et al. 2005] except that sites can move freely along or cross the cutting boundaries. For closed high-genus (genus$>$1) surfaces, their UCS can be embedded in 2D hyperbolic plane H^2. So computing the CVT in hyperbolic space is required and can lead to new geometric modeling techniques for high-genus surfaces. To the best of our knowledge, the CVT in hyperbolic space has not been studied before. We will systematically analyze it in this paper.

One difficulty for defining the hyperbolic CVT is how to well define the centroid of a given region in 2D hyperbolic space. In this paper, we extend the *model centroid* [Galperin 1993] to define the centroid of a Voronoi cell in 2D hyperbolic space. Another challenge is how to effectively and efficiently compute the hyperbolic CVT. We use two different algorithms – Lloyd's algorithm and the L-BFGS algorithm – to compute the hyperbolic CVT, and prove the convergence of Lloyd's algorithm. Based on our extensive experiments, we conjecture the sites in the hyperbolic CVT are also uniformly distributed with respect to the hyperbolic metric. We also show how to use the hyperbolic CVT to generate uniform partitions and high quality remeshing results for high-genus (genus$>$1) surfaces. Compared with previous method using parameterization in 2D Euclidean space such as [Alliez et al. 2005], the main advantage of using the hyperbolic CVT is that it has no cutting edges on the surface any more.

The main contributions of this paper include:

- We extended the concept of the CVT into hyperbolic space. We defined a CVT energy function in hyperbolic space, and proved the relationship between minimizing the energy function and the hyperbolic CVT. We also demonstrated the uniformity of the sites in the hyperbolic CVT through our experiments.
- We applied Lloyd's algorithm and the L-BFGS algorithm to compute the hyperbolic CVT. We proved the convergence of

*e-mail:guodongrong@utdallas.edu
†e-mail:mjin@cacs.louisiana.edu
‡e-mail:xguo@utdallas.edu

Lloyd's algorithm, and explained the implementation details of both algorithms.

- We used the hyperbolic CVT in the hyperbolic universal covering space to get uniform partitions and high quality remeshing results for high-genus (genus>1) surfaces.

The rest of the paper is organized as follows: Section 2 briefly reviews some related previous work. The formal definition of the hyperbolic CVT is given in Section 3, and two algorithms to compute it are introduced in Section 4. Section 5 applies the hyperbolic CVT on geometric modeling applications. Finally, Section 6 concludes the paper with some possible future work.

2 Related Work

We briefly review some previous work on how to compute the CVT in Euclidean space. We also list applications of the CVT in geometric modeling.

2.1 Centroidal Voronoi Tessellation

The concept of the centroidal Voronoi tessellation was first introduced by Du et al. [1999], but the similar concepts have been studied in different areas long before that, e.g. optimal quantization in signal processing and k-means in pattern recognition.

One of the earliest algorithms to compute the CVT is proposed by MacQueen [1967] which is a probabilistic algorithm. Although the almost sure convergence of this algorithm is proved, its convergence is very slow. Lloyd proposed a deterministic method in 1960s which is officially published later in 1982 [Lloyd 1982]. The convergence of Lloyd's algorithm is later proved by Du et al. [2006]. Due to its simplicity and robustness, Lloyd's algorithm is currently the most widely used algorithm to compute the CVT. Although Lloyd's algorithm is much faster than MacQueen's method, its convergence rate is still linear, and thus is too slow for many applications.

Most recently, Liu et al. [2009] proved the CVT energy function (explained in Section 4) is C^2 continuous, and thus they are able to apply a quasi-Newton method – L-BFGS algorithm – to compute the CVT. This method has a super-linear convergence rate and is the fastest algorithm for the CVT so far.

2.2 Geometric Modeling Applications

Due to the uniformity of the Voronoi cells and sites, the CVT has been widely used in many applications. We only review previous work closely related to geometric modeling. For other applications, please refer to [Du et al. 1999], [Rong et al. 2010] and the references therein.

Many researches on geometric modeling utilize the dual of the CCVT to achieve high quality remeshing results. Surazhsky et al. [2003] computed the CCVT by projecting the 1-ring neighbors of a site in the dual triangle mesh onto the tangential plane, and then finding the centroid of its Voronoi cell in the plane. Contrast to this local parameterization approach, Alliez et al. [2005] used a global parameterization by cutting the surface into a disk-like topology, and computing a 2D CVT in the Euclidean parameterization domain. Valette et al. [2008] directly computed an approximation of the CCVT as clusters of triangles. Yan et al. [2009] first computed a 3D CVT and then found the intersection between the surface and the 3D CVT.

Cohen-Steiner et al. [2004] extended the concept of centroids to planar proxies, and use a flooding scheme to compute the CCVT as a good shape approximation. Lu et al. [2009] computed the CVT for line segments and graphs, and use it to get meaningful segmentations of 3D models.

The CVT in all the above work is computed in 2D or 3D Euclidean space. In this paper, we study the CVT in hyperbolic space, and demonstrate its application for geometric modeling.

3 Centroidal Voronoi Tessellation in Hyperbolic Space

In this section we first give the formal definition of the CVT in Euclidean space, and then extend it into hyperbolic space. We only study the CVT in 2D hyperbolic space in this paper. We use the superscripts E and H to represent notions in Euclidean space and hyperbolic space respectively.

3.1 Notions in Euclidean Space

Given n points (called *sites*) $\mathbf{s}_1$, $\mathbf{s}_2$,..., $\mathbf{s}_n$ in a Euclidean domain $\Omega^E \subset R^2$, the *Voronoi diagram* is the union of n *Voronoi cells* Ω_i^E which contains all points nearer to site s_i than to other sites:

$$\Omega_i^E = \{\mathbf{p} \in \Omega^E | d^E(\mathbf{p}, \mathbf{s}_i) < d^E(\mathbf{p}, \mathbf{s}_j), i \neq j\}, \qquad (1)$$

where $d^E(\mathbf{a}, \mathbf{b}) = \|\mathbf{a} - \mathbf{b}\| = \sqrt{(x_\mathbf{a} - x_\mathbf{b})^2 + (y_\mathbf{a} - y_\mathbf{b})^2}$ is the distance in Euclidean space.

The *centroidal Voronoi tessellation* (CVT) is a special Voronoi diagram where every site $\mathbf{s}_i$ locates exactly at the centroid $\mathbf{c}_i$ of its Voronoi cell. In Euclidean space, the centroid of the Voronoi cell Ω_i is defined as:

$$\mathbf{c}_i^E = \frac{\int_{\Omega_i^E} \rho(\mathbf{p})\mathbf{p}\, \mathrm{d}\sigma}{\int_{\Omega_i^E} \rho(\mathbf{p})\, \mathrm{d}\sigma}, \qquad (2)$$

where $\rho(\mathbf{p}) \geq 0$ is a given density function.

3.2 Voronoi Diagram in Hyperbolic Space

Similar to the Voronoi diagram in Euclidean space, we can also define the Voronoi diagram in a 2D hyperbolic domain $\Omega^H \subset H^2$ using the hyperbolic distance $d^H(\cdot, \cdot)$ to replace $d^E(\cdot, \cdot)$ in (1). A *hyperbolic Voronoi cell* Ω_i^H is thus:

$$\Omega_i^H = \{\mathbf{p} \in \Omega^H | d^H(\mathbf{p}, \mathbf{s}_i) < d^H(\mathbf{p}, \mathbf{s}_j), i \neq j\}. \qquad (3)$$

The *hyperbolic Voronoi diagram* of n sites in Ω^H is the union of hyperbolic Voronoi cells of them.

There are several different models for the hyperbolic geometry. They are all equivalent, but provide different views. Among these models, the Voronoi diagram has been studied in the Upper Half-plane model [Onishi and Takayama 1996], the Poincaré disk model [Nilforoushan and Mohades 2006], and the Klein disk model [Nielsen and Nock 2009]. In this paper, we compute the Voronoi diagram in the Klein disk, a unit disk where any geodesic is a chord. The distance between two points $\mathbf{p}$ and $\mathbf{q}$ in the Klein disk is

$$d_K^H(\mathbf{p}, \mathbf{q}) = \cosh^{-1} \frac{1 - <\mathbf{p}, \mathbf{q}>}{\sqrt{(1 - \|\mathbf{p}\|^2)(1 - \|\mathbf{q}\|^2)}}$$

where $< \cdot, \cdot >$ and $\|\cdot\|$ are inner product and vector norm computed in Euclidean space. Using this distance, it can be proved that, for a set of sites $\mathbf{s}_i$ in the Klein disk, the Voronoi diagram in hyperbolic space is a *power diagram* [Aurenhammer 1987] in Euclidean space.

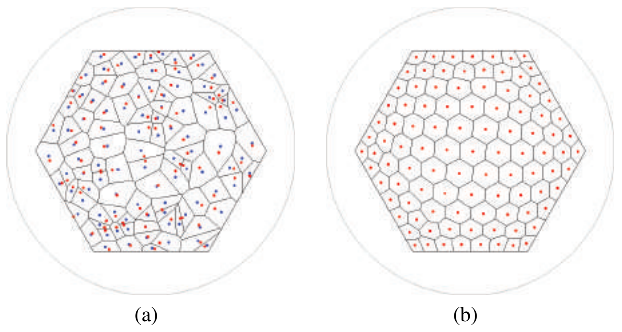
Figure 1: *(a) Voronoi diagram of 100 random initial sites (red dots) in a hexagon in the Klein disk. Blue dots are hyperbolic centroids of corresponding Voronoi cells. (b) Hyperbolic CVT generated from (a). $\rho(\mathbf{p}) \equiv 1$ in this example.*

More specifically, for every site $\mathbf{s}_i$ in the Klein disk, a corresponding weighted point $wp_i = < \mathbf{t}_i, w_i^2 >$ can be created in Euclidean space, where $\mathbf{t}_i = \frac{\mathbf{s}_i}{2\sqrt{1-\|\mathbf{s}_i\|^2}}$ and $w_i^2 = \frac{\|\mathbf{s}_i\|^2}{4(1-\|\mathbf{s}_i\|^2)} - \frac{1}{\sqrt{1-\|\mathbf{s}_i\|^2}}$. The power diagram of weighted points wp_i in Euclidean space is same to the Voronoi diagram of the sites $\mathbf{s}_i$ in the Klein disk. More derivation details can be found in [Nielsen and Nock 2009]. Figure 1(a) shows a Voronoi diagram of 100 random sampled sites in a hexagon in the Klein disk.

3.3 Centroid in Hyperbolic Space

We define the centroid in hyperbolic space following the idea of *model centroid* proposed by Galperin [1993], and extend it from discrete points to a continuous region. The model centroid unified the definition of the centroid in spaces with constant curvature – Euclidean space, hyperbolic space, and spherical space.

Given n discrete points in a k-dimensional space, and n mass values m_i corresponding to these points, the position of the centroid of these points can be located as follows: We find a "model" of the k-dimensional space in $(k+1)$-dimensional Euclidean space. For every point $\mathbf{p}_i$, a vector is built from the origin pointing to the point. The vectors are first scaled by the corresponding mass values, and then are added up. The intersection between the sum vector and the model of the k-dimensional space is defined as the position of the centroid of these points. This definition is proved to be well-defined for any number of discrete points and satisfied the axioms given in [Galperin 1993].

For 2D Euclidean space, the model is the plane $z = 1$ in 3D Euclidean space. When we replace the summation of the n vectors with the integral over a continuous region, it is easy to verify that the model centroid defined in this way is same to the centroid $\mathbf{c}_i^E$ defined in Equation (2).

For 2D hyperbolic space, the model is the Minkowski model, which is the upper sheet of a two-sheeted hyperboloid $-x^2 - y^2 + z^2 = 1$ in 3D Euclidean space. Figure 2 shows the side view of the computation for the centroid of two discrete points.

Since the Voronoi diagram is computed in the Klein disk and the computation of centroids is performed in the Minkowski model, we need to convert positions between these two models. From now on, we will use normal letters to represent points in the Klein disk, and letters with bars for points in the Minkowski model. Furthermore, we use letters with primes to represent points on the plane $z = 0$.

When embed in the plane $z = 1$ with the center on z-axis, the Klein

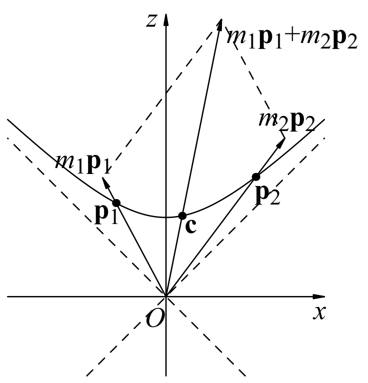

Figure 2: *Side view of the computation for the centroid of two discrete points.*

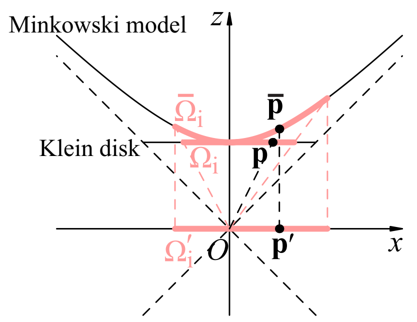

Figure 3: *Illustration of mapping functions φ and ψ.*

disk is the central projection of the Minkowski model with respect to the origin. The correspondence between a point $\mathbf{p}(x_\mathbf{p}, y_\mathbf{p})$ in the Klein disk and a point $\overline{\mathbf{p}}(\overline{x}_\mathbf{p}, \overline{y}_\mathbf{p}, \overline{z}_\mathbf{p})$ in the Minkowski model is given by the following formulas:

$$\begin{cases} x_\mathbf{p} = \overline{x}_\mathbf{p}/\overline{z}_\mathbf{p} \\ y_\mathbf{p} = \overline{y}_\mathbf{p}/\overline{z}_\mathbf{p} \end{cases} \tag{4}$$

$$\begin{cases} \overline{x}_\mathbf{p} = x_\mathbf{p}/\sqrt{1-(x_\mathbf{p}^2+y_\mathbf{p}^2)} \\ \overline{y}_\mathbf{p} = y_\mathbf{p}/\sqrt{1-(x_\mathbf{p}^2+y_\mathbf{p}^2)} \\ \overline{z}_\mathbf{p} = 1/\sqrt{1-(x_\mathbf{p}^2+y_\mathbf{p}^2)} \end{cases} . \tag{5}$$

Formula (5) define a mapping function φ from the Klein disk to the Minkowski model, where $\varphi(\mathbf{p}) = \overline{\mathbf{p}}$. We also define another mapping function ψ which orthogonally projects a point $\overline{\mathbf{p}}$ in the Minkowski model to the plane $z = 0$ to get the point $\mathbf{p}'$, i.e. $\psi(\overline{\mathbf{p}}) = \mathbf{p}'$. Note that these two mapping functions can be naturally extended to Voronoi cells:

$$\varphi(\Omega_i^H) = \bigcup_{\mathbf{p}\in\Omega_i^H} \varphi(\mathbf{p}) = \overline{\Omega}_i^H, \psi(\overline{\Omega}_i^H) = \bigcup_{\overline{\mathbf{p}}\in\overline{\Omega}_i^H} \psi(\overline{\mathbf{p}}) = \Omega_i'^H.$$

Figure 3 illustrates these two mapping functions.

We now extend the definition of the hyperbolic centroid from discrete points to a continuous region in 2D hyperbolic space. We can compute a surface integral over the corresponding region in the Minkowski model and find the intersection between the result and the hyperboloid as the centroid of the region. Particularly, for a Voronoi region $\overline{\Omega}_i^H$ in the Minkowski model, we compute its cen-

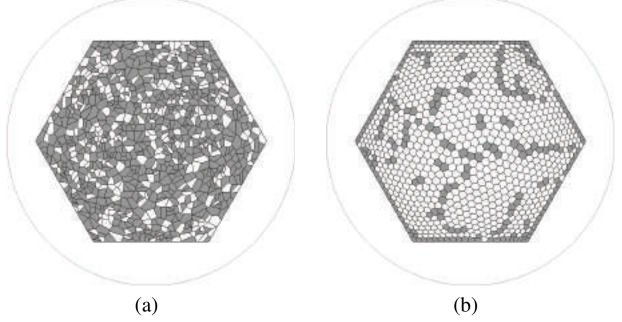

Figure 5: *Hexagonal (unshaded) Voronoi cells in (a) the Voronoi diagram of 1,000 random initial sites in a hexagon in the Klein disk, and (b) the hyperbolic CVT generated from (a).* $\rho(\mathbf{p}) \equiv 1$ *in this example.*

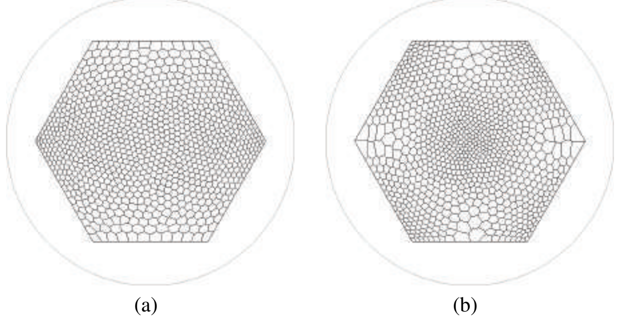

Figure 6: *Hyperbolic CVTs generated from same initial sites in Figure 5 with (a)* $\rho(\mathbf{p}) = e^{-2x_{\mathbf{p}}^2-10y_{\mathbf{p}}^2}$ *and (b)* $\rho(\mathbf{p}) = e^{-20x_{\mathbf{p}}^2-20y_{\mathbf{p}}^2} + 0.05\sin^2(\pi x_{\mathbf{p}})\sin^2(\pi y_{\mathbf{p}})$.

troid as:

$$\mathbf{c}_i^H = \frac{\int_{\overline{\Omega}_i^H} \rho(\overline{\mathbf{p}})\mathbf{v}(\overline{\mathbf{p}})\,\mathrm{d}\overline{\sigma}}{\| \int_{\overline{\Omega}_i^H} \rho(\overline{\mathbf{p}})\mathbf{v}(\overline{\mathbf{p}})\,\mathrm{d}\overline{\sigma}\|_M}, \tag{6}$$

where $\|\overline{\mathbf{p}}\|_M = \sqrt{-\overline{x}_{\mathbf{p}}^2 - \overline{y}_{\mathbf{p}}^2 + \overline{z}_{\mathbf{p}}^2}$ is the norm in the Minkowski model. Figure 1(a) shows the hyperbolic centroids of all Voronoi cells marked as blue dots.

3.4 Hyperbolic Centroidal Voronoi Tessellation

Once we have definitions of the Voronoi diagram and the centroid in hyperbolic space, it is straightforward to combine them to define the centroidal Voronoi tessellation in hyperbolic space (hyperbolic CVT) where every site locates on the centroid of its Voronoi cell. Figure 1(b) shows the hyperbolic CVT generated from the 100 initial sites in Figure 1(a).

Gersho's conjecture states that the sites in a Euclidean CVT are evenly distributed in the space. We conjecture that it is also true for the hyperbolic CVT. Since it is difficult to visually tell the uniformity of the sites in a hyperbolic CVT, we measure the geometrical uniformity of the sites and Voronoi cells in a hyperbolic CVT. For every site $\mathbf{s}_i$, we define the radius r_i of its Voronoi cell, the distance d_i to its nearest neighbors, and the area a_i of its Voronoi cell as follows:

$$r_i = \max_{\mathbf{p}\in\Omega_i^H} d_K^H(\mathbf{p},\mathbf{s}_i),\ d_i = \min_{j\neq i} d_K^H(\mathbf{s}_i,\mathbf{s}_j),\ a_i = Area^H(\Omega_i^H),$$

where $Area^H(\cdot)$ denotes the area of a polygon in hyperbolic space. For every quality measure, smaller variance means better uniformity of the sites. Figure 4 compares the three measures computed for the Voronoi diagram of 100 initial sites (Figure 1(a)) and the hyperbolic CVT (Figure 1(b)). It is clear that the sites in the hyperbolic CVT are very uniformly distributed.

In 2D Euclidean space, it has been proved that most of Voronoi cells in a CVT are hexagons [Newman 1982]. We have also experimentally confirmed the same conclusion for 2D hyperbolic CVTs. One example is shown in Figure 5 where we show hexagonal Voronoi cells as unshaded. It is clear that most of Voronoi cells in the CVT result are hexagons.

Although the above CVT results are all computed with a constant density value $\rho(\mathbf{p}) \equiv 1$, our definition of the hyperbolic CVT is general for any density values. Figure 6 shows two examples with non-constant density values generated from the same initial sites as in Figure 5. As we can see, similar to the behavior in Euclidean space, in a hyperbolic CVT, sites tend to cluster near the regions with higher density values. This property is critical for many applications in geometric modeling (see Section 5 for details).

4 Computational Algorithms

Lloyd's algorithm is the most widely used algorithm to compute the CVT in Euclidean space. We will apply it to compute the hyperbolic CVT, and prove its convergence in hyperbolic space. We also investigate applying the L-BFGS algorithm on computing the hyperbolic CVT.

4.1 Lloyd's Algorithm

Lloyd's algorithm is an iterative algorithm to minimize the CVT energy (defined below). It starts with an arbitrary set of initial sites. In every iteration of Lloyd's algorithm, the Voronoi diagram of current sites is first computed. Next, the centroids of Voronoi cells are computed and used as new sites for next iteration. This procedure is repeated until certain stopping condition is satisfied (e.g. the moving distance of every site is smaller than a threshold). Since both the Voronoi diagram and the centroid have been defined in hyperbolic space, we can directly apply Lloyd's algorithm to compute the hyperbolic CVT. The only concern is whether Lloyd's algorithm will converge in hyperbolic space.

Lloyd's algorithm has been proved to be convergent when computing the CVT in Euclidean space R^2 [Du et al. 2006]. We now prove that Lloyd's algorithm also converges in hyperbolic space H^2.

In Euclidean space, the CVT energy of the ordered set of sites $\mathbf{S} = (\mathbf{s_1}, \mathbf{s_2}, \ldots, \mathbf{s_n})$ is defined as:

$$\begin{aligned} F^E(\mathbf{S}) &= \sum_{i=1}^{n} \int_{\Omega_i^E} \rho(\mathbf{p}) d^E(\mathbf{p},\mathbf{s}_i)^2 \,\mathrm{d}\sigma \\ &= \sum_{i=1}^{n} \int_{\Omega_i^E} \rho(\mathbf{p}) \|\mathbf{p}-\mathbf{s}_i\|^2 \,\mathrm{d}\sigma. \end{aligned} \tag{7}$$

As pointed in [Stahl 2007], many formulas in hyperbolic space can be achieved by replace the distance d^E with $\sinh(d^H)$ in the corresponding formulas in Euclidean space. So we define the CVT energy of $\mathbf{S}$ in hyperbolic space as:

$$F^H(\mathbf{S}) = \sum_{i=1}^{n} \int_{\Omega_i^H} \rho(\mathbf{p}) \sinh^2(d_K^H(\mathbf{p},\mathbf{s}_i))\,\mathrm{d}\sigma. \tag{8}$$

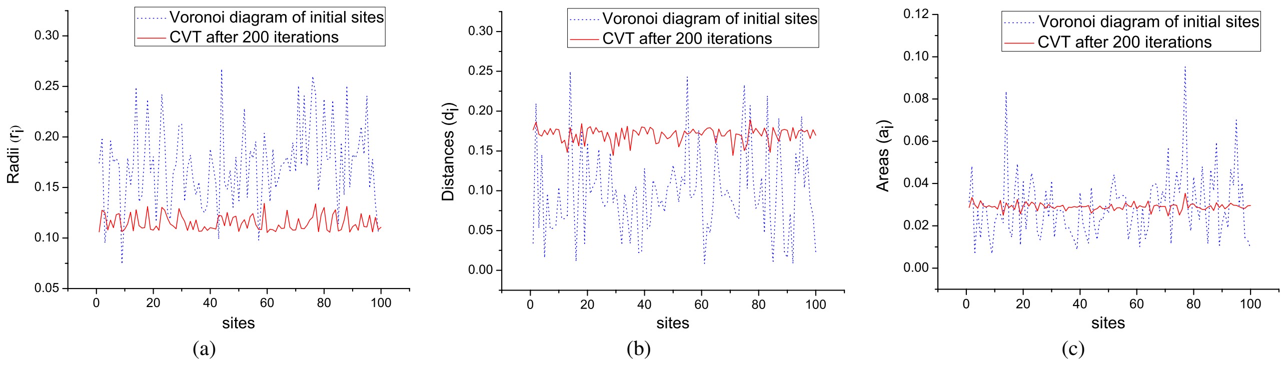

Figure 4: *Comparison of geometrical measures of the Voronoi diagram of 100 initial sites shown in Figure 1(a) (blue dotted curve) and the hyperbolic CVT shown in Figure 1(b) (red solid curve).*

We first consider the situation where the sites are fixed and the tessellation varies:

Lemma 1 *When the sites are fixed, the CVT energy $F^H(\mathbf{S})$ is minimized when the tessellation is the Voronoi diagram of the sites.*

Next, we study the situation where the tessellation is fixed (as the Voronoi diagram) and the sites moves:

Lemma 2 *When the tessellation is fixed, the CVT energy $F^H(\mathbf{S})$ is minimized when the sites locate at centroids of their Voronoi cells.*

The proofs of the above two lemmas are given in the appendix. With the results of the above two lemmas, we can use the same proofs of Theorem 2.3, Corollary 2.4, Theorem 2.5, and Theorem 2.6 in [Du et al. 2006] to prove the convergence of Lloyd's algorithm for the hyperbolic CVT. Note that same to the case in Euclidean space, the minimum found by Lloyd's algorithm may be either a local minimum point or a saddle point of the CVT energy.

4.2 L-BFGS Algorithm

The L-BFGS (Limited memory Broyden-Fletcher-Goldfarb-Shanno) algorithm is a quasi-Newton algorithm which can quickly find the minimum of the CVT energy. It has been successfully used in Euclidean space to compute the CVT, and has faster convergence speed than Lloyd's algorithm. We also apply it for the hyperbolic CVT.

A Newton algorithm to minimize an energy function requires to compute a full Hessian matrix of the function. This is usually difficult for the CVT energy. The L-BFGS algorithm only uses the energy value and its first order partial derivatives computed in (maximum) m (a user-given parameter) preceding iterations to approximate the inverse of Hessian matrix.

More specifically, suppose $\mathbf{S}_k$ and ∇F_k, two vectors in R^{2n}, are the ordered set of n sites and the gradient of the CVT energy $F^H(\mathbf{S}_k)$ in k-th iteration, we define $\mathbf{x}_k = \mathbf{S}_k - \mathbf{S}_{k-1}$, and $\mathbf{y}_k = \nabla F_k - \nabla F_{k-1}$. The approximated Hessian matrix is computed recursively as:

$$\mathbf{H}_k = \mathbf{V}_k^T \mathbf{H}_{k-1} \mathbf{V}_k + \rho_k \mathbf{x}_k \mathbf{x}_k^T,$$

where $\rho_k = 1/(\mathbf{y}_k^T \mathbf{x}_k)$ and $\mathbf{V}_k = \mathbf{I} - \rho_k \mathbf{y}_k \mathbf{x}_k^T$. This equation is recursively evaluated for m (or k, if $k < m$) previous preceding iterations. After $\mathbf{H}_k$ is evaluated, the new sites are updated as $\mathbf{S}_{k+1} = \mathbf{S}_k - \mathbf{H}_k \nabla F_k$.

So the key point of applying the L-BFGS algorithm in hyperbolic space is how to evaluate the partial derivatives of $F^H(\mathbf{S})$. $F^H(\mathbf{S})$ is the sum of n partial energy terms. In its partial derivative $\partial F^H(\mathbf{S})/\partial \mathbf{s}_i$, only the i-th term has non-zero value. So we can perform the computation only on the Voronoi cell Ω_i^H. According to [Okabe et al. 1999], the first order partial derivative of the CVT energy with respect to the site $\mathbf{s}_i$ is:

$$\frac{\partial F^H(\mathbf{S})}{\partial \mathbf{s}_i} = \frac{\partial F^H(\mathbf{s}_i)}{\partial \mathbf{s}_i} = \int_{\Omega_i^H} \frac{2\rho(\mathbf{p})(\mathbf{s}_i - \mathbf{p})}{(1 - \|\mathbf{p} - \mathbf{s}_i\|^2)^2}\, \mathrm{d}\sigma, \tag{9}$$

where the later part is computed by setting the origin of the coordinate system at $\mathbf{s}_i$.

4.3 Implementation Details

Both Lloyd's algorithm and the L-BFGS algorithm need to compute the Voronoi diagram as the first step. As discussed above, this can be easily accomplished in the Klein disk by computing a power diagram. In our implementation, we use the CGAL library [Fabri 2001] to compute the power diagram.

For Lloyd's algorithm, the next step is to compute the centroid for every Voronoi cell. In hyperbolic space, although the centroid of a triangle with constant density is proved to be coincident with the common point of its three medians [Stahl 2007], for a triangle with non-constant density, it is not known how to get a close-form solution of its centroid defined by Equation (6). As the result, we cannot compute the centroid of a Voronoi cell by dividing it into several triangles as we did in Euclidean space. Instead, we use summations to approximate the integral.

For every Voronoi cell Ω_i^H, we first apply a Möbius transformation, the rigid motion in hyperbolic plane, to move its site $\mathbf{s}_i$ to the origin to achieve a relatively small numerical error. Then we use the mapping functions φ and ψ to map the Ω_i^H from the Klein disk to the Minkowski model, and then to the plane $z = 0$, to get ${\Omega'}_i^H = \psi(\phi(\Omega_i^H))$. The plane $z = 0$ is uniformly sampled using a regular grid. We perform the summation over all samples located within ${\Omega'}_i^H$ to approximate the integral in Equation (6).

The integral for the L-BFGS algorithm in Equation (9) is computed in the Klein disk. For every Voronoi cell Ω_i^H, we also perform a rigid transformation to move $\mathbf{s}_i$ to the origin. We uniformly (in Euclidean sense) sample the Klein disk, and sum over all samples located within Ω_i^H. Note that since the metric in the Klein disk is different to that in Euclidean space, we need to compute the delta area for every sample at (x, y) as:

$$\Delta\sigma = \frac{\Delta x \Delta y}{\sqrt{(1 - x^2 - y^2)^3}}, \tag{10}$$

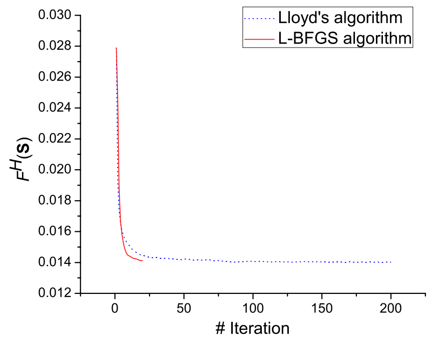

Figure 7: *CVT energy against the number iterations for Lloyd's algorithm (blue dotted curve) and the L-BFGS algorithm (red solid curve) starting from the 100 initial sites shown in Figure 1(a).*

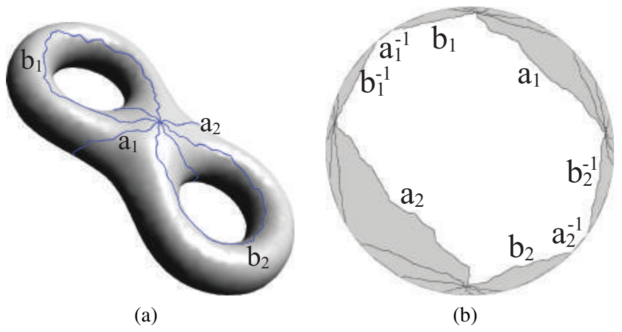

Figure 8: *Canonical fundamental group generators on (a) the surface and (b) the Klein disk. Center domain is shown unshaded, and neighbor domains shaded.*

where Δx and Δy are sample intervals along x- and y-directions.

We plot in Figure 7 the CVT energy $F^H(\mathbf{S})$ against the number of iterations for both Lloyd's algorithm and the L-BFGS algorithm starting from the 100 sites shown in Figure 1(a). Similar to in Euclidean space, the energy decreases dramatically for the first several iterations, and converges gradually to a local minimum. Also, the L-BFGS algorithm converges much faster than Lloyd's algorithm, which is same to its behavior in Euclidean space. Note although the CVT energy should monotonically decrease, there is slight perturbation in the curve for Lloyd's algorithm due to our approximation of the integral for centroid computation.

5 Application for High-Genus Surfaces

Let S be a surface embedded in R^3. g is the Riemannian metric induced from the Euclidean metric of S. Suppose $u : S \to R$ is a scalar function defined on S. Then $\overline{g} = e^{2u}g$ is also a Riemannian metric on S and is conformal to the original one, which means the new metric preserves angles and locally only differs a scaling with the original one. According to the Uniformization theorem [Klingenberg 1982], any surface admits a Riemannian metric of constant Gaussian curvature, which is conformal to the original metric. Such metric is called uniformization metric. Surfaces with negative Euler numbers admit hyperbolic uniformization metric with constant -1 Gaussian curvature and their UCS can be isometrically embedded in 2D hyperbolic plane based on the uniformization metric.

Since conformal metric only introduces area distortion, which can be easily compensated by modifying the density function expressed in the embedding domain [Alliez et al. 2005], we use hyperbolic uniformization metric to parameterize high genus surfaces to hyperbolic plane. The usage of the universal covering space allows sites to move freely everywhere on the surface: a site can cross the boundary of the center domain, and come into the center domain from the "opposite" side of the boundary. So there is no artifacts along the cutting edges any more. This is also the major advantage of our method over Alliez et al.'s algorithm [Alliez et al. 2005].

For a given closed high genus surface, we use the method introduced in [Jin et al. 2006] to compute its hyperbolic uniformization metric and construct the embedding of its UCS in the Klein disk.

A double torus model (genus 2) is given in Figure 8(a), with a set of canonical fundamental group generators (blue loops a_1, b_1, a_2, b_2) which cut the surface into a topological disk, the so called fundamental domain, with $4g$ sides: a_1, b_1, a_1^{-1}, b_1^{-1}, a_2, b_2, a_2^{-1}, b_2^{-1}. The fundamental domain is isometrically embedded in the Klein disk with marked sides as shown in the unshaded region in Figure 8(b). A finite copies of the fundamental domain (shaded regions in Figure 8(b)) are glued coherently by applying corresponding Möbius transformations. Note that any two domains in UCS only differ a rigid motion in hyperbolic plane, the Möbius transformation, which can be computed quickly as in [Jin et al. 2006].

We adapt Lloyd's algorithm to compute the CVT of a set of initial sites $\mathbf{S}$ in the center domain (the original embedded fundamental domain) of the UCS. By applying Möbius transformations on sites in $\mathbf{S}$, we compute the corresponding points in neighbor domains (those sharing an edge or a vertex with the center domain) and denote the union of them by $\mathbf{S}'$. The Voronoi diagram of $\mathbf{S} \cup \mathbf{S}'$ is computed, and the centroids of Voronoi cells of sites in $\mathbf{S}$ are located. If a centroid is outside of the center domain by the side, say a_1 (see Figure 8(b)), by performing a corresponding Möbius transformation we move it to the "opposite" side a_1^{-1} of the center domain. The adjusted centroids are inside the center domain and are used as the new sites in the next iteration.

Back to the example of the double torus model and its conformal embedding of the UCS in the Klein disk, Figure 9(a) shows the Voronoi diagram of 100 random initial sites marked with red inside the center domain and their corresponding points in neighbor domains in UCS marked with green. The adjusted centroids for Voronoi cells of sites in $\mathbf{S}$ are all inside the center domain, marked with blue.

Density Value. As discussed before, conformal parametrization introduces area distortion only, which can be compensated by assigning an appropriate density value at each point expressed in the embedding domain.

The *conformal factor* cf is defined to measure local scaling of conformal mapping. For each vertex $\mathbf{v}$ on the surface, cf can be computed as the ratio of the area sum of its incident triangles in 3D space and in 2D hyperbolic plane, i.e. $cf(\mathbf{v}) = \frac{A_{3D}(\mathbf{v})}{A_{2D}(\mathbf{v})}$. For a non-vertex point on the surface, cf can be computed by linearly interpolating the computed conformal factors of the three vertices of the triangle containing that point.

We then define a *sizing field* where every point $\mathbf{p}$ on the surface has a desired size $\mu(\mathbf{p})$. For every triangle t, its sizing ratio is computed as $sr(t) = \frac{longest\_edge(t)}{\mu(centroid(t))}$. For a given sizing field, we say a triangle mesh satisfies it if the sizing ratio of every triangle is less than or equal to 1. For the ideal case, the sizing ratios of all triangles should be 1 to minimize the number of triangles used. So the length of the longest edge of every triangle is approximately equal to the sizing at its centroid. The area of the triangle is thus proportional to the square of the sizing at its centroid, i.e. $A(t) \sim$

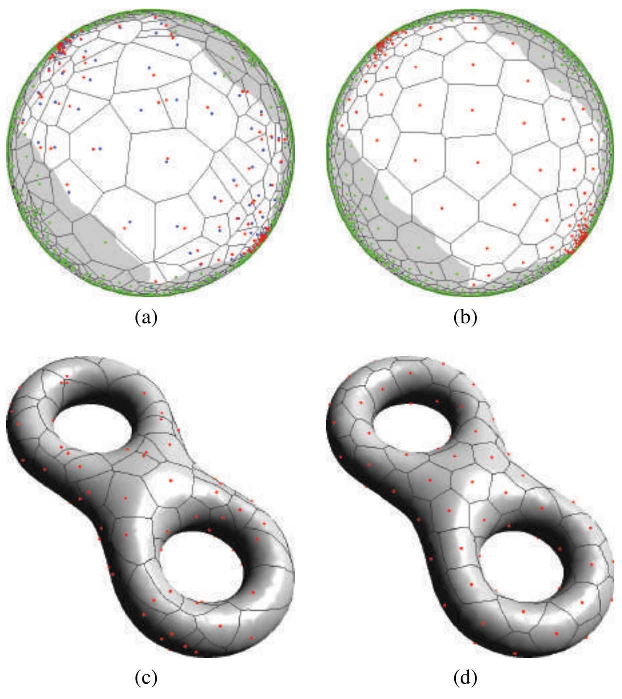

Figure 9: *Voronoi diagram of 100 random initial sites in (a) the UCS, and the centroids for Voronoi cells of sites in* $\mathbf{S}$*. Red dots are sites in* $\mathbf{S}$*, green dots are sites in* $\mathbf{S}'$*, and blue dots are centroids. (b) CVT result generated from (a). (c) Initial Voronoi diagram on surface. (d) CVT result on surface.*

$\mu(centroid(t))^2$. Since we want the sites uniformly distributed on the surface, the sizing filed on the surface should be a constant value everywhere. After a proper normalization, we have $A_{3D}(\mathbf{p}) \sim 1$ and $A_{2D}(\mathbf{p}) \sim \mu(\mathbf{p})^2$, thus the conformal factor $cf(\mathbf{p}) = \frac{1}{\mu(\mathbf{p})^2}$. It is pointed out by Du and Wang [2006] that the dual mesh of a CCVT with density values $\rho(\mathbf{p}) = \frac{1}{\mu(\mathbf{p})^{d+2}}$ will satisfy the given sizing field, where d is the dimension of the problem. So, in 2D, we assign the density value at point $\mathbf{p}$ as:

$$\rho(\mathbf{p}) = \frac{1}{\mu(\mathbf{p})^4} = cf(\mathbf{p})^2. \tag{11}$$

So the uniformity of the sites in the CVT result is compensated by the distortion, and thus uniformly distributed on the surface.

With the modulated density values (instead of a constant), the CVT result generated by 100 Lloyd's iterations from initial sites shown in Figure 9(a) is given in Figure 9(b). The Voronoi diagram of initial sites and the CVT results on the original surface are shown in Figure 9(c) and 9(d) respectively. As we can see the final sites are evenly distributed on the surface.

A CVT result of the surface of Amphora with 1,000 sites is shown in Figure 10(b). Compared with the Voronoi diagram of the random initial sites (Figure 10(a)), we can clearly see the improvement of the uniformity of Voronoi cells. We can also modulate the density values using some properties required by applications. One example is to use the *local feature size* (LFS, distance to the medial axis) as in [Yan et al. 2009]:

$$\rho(\mathbf{p}) = cf(\mathbf{p})^2/lfs(\mathbf{p})^2. \tag{12}$$

By modulating the density values by the LFS, the CVT result contains more sites where there are more details and vice versa. Figure 10(c) shows the CVT result with LFS-modulated density values using the same initial sites as in Figure 10(a).

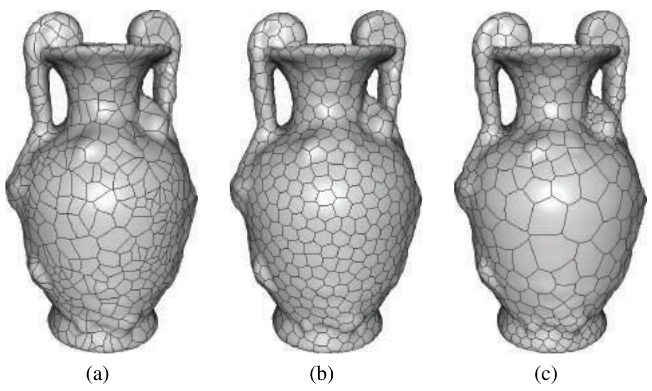

Figure 10: *Results for the surface of Amphora (genus-2): (a) The Voronoi diagram of 1,000 random initial sites, (b) the CVT with density values computed using Equation (11), and (c) the CVT with density values modulated by LFS.*

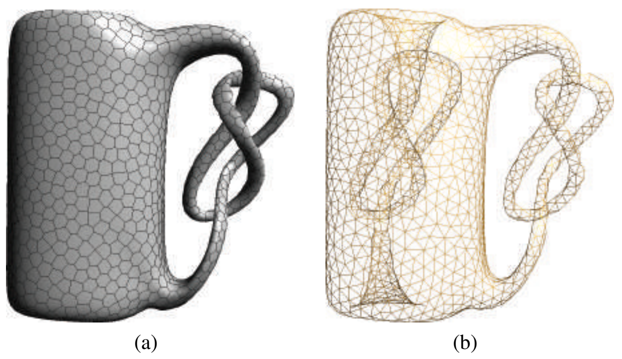

Figure 11: *(a) The CVT of 2,000 sites on the surface of Knot (genus-2). (b) Dual triangle mesh for Knot. (The wire frame mode is used to show the complex inner structure.)*

Figure 11 shows results for a genus-2 surface with much more complicated shape. It is clear that our algorithm using the hyperbolic CVT can generate high-quality results for relatively few sites on complicated surfaces. Two more results for genus-3 surfaces are shown in Figure 12.

Figure 13 compares our algorithm with Yan et al.'s fast restricted Voronoi diagram algorithm (FRVD) [Yan et al. 2009], which uses Euclidean distance to compute a 3D Voronoi diagram, and then finds the intersection between the 3D Voronoi diagram and the surface. For the surface of Knot, part of the inner knot is very close to the outer surface, but they are far apart along the surface. From our experiments, even with 4,000 sites, some Voronoi cells in the FRVD result still spans over the inner knot and the outer surface. As the result, the inner knot is glued with the outer surface by some non-manifold vertices and edges (shown in red in Figure 13(c)). As a comparison, our method using the hyperbolic CVT has no problem to separate these two parts with much fewer sites.

Although our method cannot always guarantee a valid triangulation, we did not have any problems in our experiments with a reasonable number of sites. In our method, the hyperbolic CVT is computed in the running-time, but all other steps, including computing the uniformization metric, embedding, and computing the distortion can

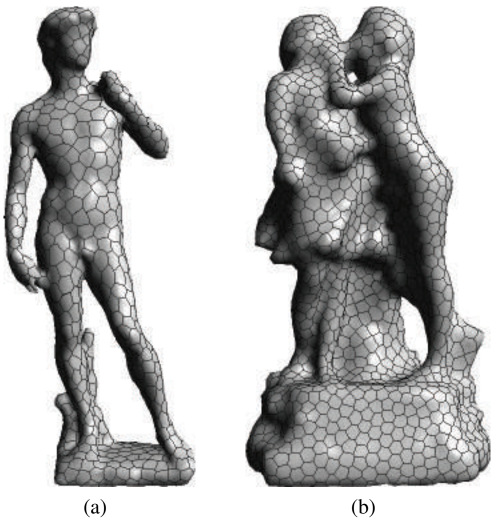
Figure 12: *(a) The CVT of 1,000 sites on the surface of David (genus-3) and (b)The CVT of 2,000 sites on the surface of Sculpture (genus-3).*

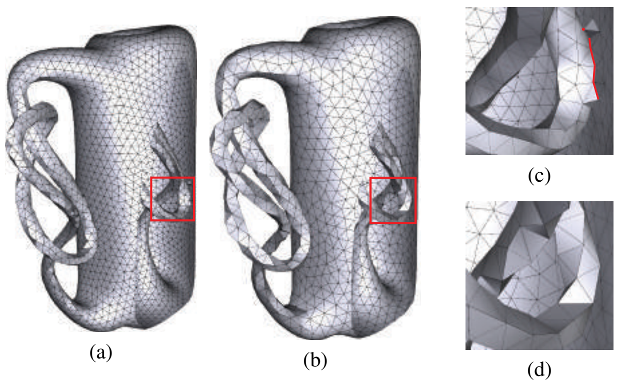
Figure 13: *Comparing the dual triangle meshes generated by (a) FRVD with 4,000 sites and (b) our algorithm with 2,000 sites. The regions where the inner tube is very close to the outer surface (marked by red boxes) are enlarged in (c) and (d). Non-manifold vertex and edges in (c) are shown in red.*

all be pre-computed. The time complexity of computing the hyperbolic Voronoi diagram is same to the that in Euclidean space, i.e. $O(n \log n)$ where n is the number of sites. As a comparison, the algorithms using the exact geodesic distance need to compute the geodesic Voronoi diagram in every iteration, which is much more expensive than the computation of the power diagram.

6 Conclusion and Future Work

We extended the concept of the centroidal Voronoi tessellation from Euclidean space to hyperbolic space. We showed the hyperbolic CVT has the same property with its Euclidean counterpart, i.e. the sites are uniformly distributed. We explained how to apply Lloyd's algorithm and the L-BFGS algorithm to compute the hyperbolic CVT, and proved the convergence of Lloyd's algorithm in hyperbolic space. Finally, we used the hyperbolic CVT to get uniform partitions and high quality remeshing results for high-genus (genus>1) surfaces. We believe this newly proposed hyperbolic CVT would have many more applications in difference areas such as surface processing, scientific visualization, pattern recognition, etc.

Note that we only handle closed high-genus surfaces in this paper, but our method can also be generalized to surfaces with boundaries, by using the double covering technique (gluing two copies of a same open surface along their boundaries) to convert them to closed symmetric surfaces. We are also investigating conformally embedding high-genus surfaces with boundaries in the Klein disk.

Due to lack of a close-form solution for the hyperbolic centroid of a triangle with non-uniform density, we can only use summation to approximate the integral in the computation of the hyperbolic CVT. This not only leads to degraded quality, but also slow down the performance of our programs. How to directly compute the integral as in Euclidean space is one of our major future work.

In our current implementation, the Voronoi diagram on the UCS is computed by using all points in $\mathbf{S} \cup \mathbf{S}'$. For a genus-$g$ surface, there are $16g^2 - 8g$ neighbor domains. So the number of sites is quite large. This makes the computation of the Voronoi diagram very slow, which becomes the bottleneck of our current program. For a genus-3 surface with 2,000 sites, our current program needs more than one minute to compute the Voronoi diagram for a single iteration. Optimizing this procedure is critical to increase the program speed. We may borrow the idea of the periodic triangulation package in CGAL [Caroli and Teillaud 2009] to accelerate this procedure.

Another possible acceleration approach is to utilize the parallel computability of the programmable graphics processing unit (GPU). GPU has already been used to accelerate the computation of the CVT in Euclidean space [Vasconcelos et al. 2008; Rong et al. 2010]. But the metric of the Klein disk is highly non-uniform in the unit Euclidean disk, and is not easy (if not impossible) to be represented by a 2D texture. So none of the existing GPU algorithms can be straightforwardly extended to compute the hyperbolic CVT. We would like to investigate new data structures on the GPU for hyperbolic geometry.

Acknowledgements

Guodong Rong and Xiaohu Guo are partially supported by the National Science Foundation under Grant No. CCF-0727098. Miao Jin is partially supported by Louisiana Board of Regents under Grant No. RD-A-21.

References

Alliez, P., de Verdière, É. C., Devillers, O., and Isenburg, M. 2005. Centroidal Voronoi diagrams for isotropic surface remeshing. *Graphical Models 67*, 3, 204–231.

Aurenhammer, F. 1987. Power diagrams: properties, algorithms and applications. *SIAM Journal on Computing 16*, 1, 78–96.

Caroli, M., and Teillaud, M. 2009. Computing 3D periodic triangulations. Research Report RR-6823, INRIA.

Cohen-Steiner, D., Alliez, P., and Desbrun, M. 2004. Variational shape approximation. *ACM Transationcs on Graphics 23*, 3, 905–914.

Du, Q., and Wang, D. 2006. Recent progress in robust and quality Delaunay mesh generation. *Journal of Computational and Applied Mathematics 195*, 1, 8–23.

DU, Q., FABER, V., AND GUNZBURGER, M. 1999. Centroidal Voronoi tessellations: Applications and algorithms. *SIAM Review 41*, 4, 637–676.

DU, Q., GUNZBURGER, M. D., AND JU, L. 2003. Constrained centroidal Voronoi tessellations for surfaces. *SIAM Journal on Scientific Computing 24*, 5, 1488–1506.

DU, Q., EMELIANENKO, M., AND JU, L. 2006. Convergence of the Lloyd algorithm for computing centroidal Voronoi tessellations. *SIAM Journal on Numerical Analysis 44*, 1, 102–119.

FABRI, A. 2001. CGAL - the computational geometry algorithm library. In *Proceedings of the 10th International Meshing Roundtable*, 137–142.

FEJES TÓTH, G. 2001. A stability criterion to the moment theorem. *Studia Scientiarum Mathematicarum Hungarica 38*, 1-4, 209–224.

GALPERIN, G. A. 1993. A concept of the mass center of a system of material points in the constan curvature spaces. *Communications in Mathematical Physics 154*, 1, 63–84.

GERSHO, A. 1979. Asymptotically optimal block quantization. *IEEE Transactions on Information Theory 25*, 4, 373–380.

JIN, M., LUO, F., AND GU, X. 2006. Computing surface hyperbolic structure and real projective structure. In *ACM Symposium on Solid and Physical Modeling*, 105–116.

KLINGENBERG, W. P. A. 1982. *Riemannian Geometry*. Walter de Gruyter, New York.

LIU, Y., WANG, W., LÉVY, B., SUN, F., YAN, D.-M., LU, L., AND YANG, C. 2009. On centroidal Voronoi tessellation — energy smoothness and fast computation. *ACM Transactions on Graphics 28*, 4, 1–17.

LLOYD, S. P. 1982. Least squares quantization in PCM. *IEEE Transactions on Information Theory 28*, 2, 129–137.

LU, L., LEVY, B., AND WANG, W. 2009. Centroidal Voronoi tessellations for line segments and graphs. technical report, INRIA-ALICE.

MACQUEEN, J. B. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, University of California Press, 281–297.

NEWMAN, D. J. 1982. The hexagon theorem. *IEEE Transactions on Information Theory 28*, 2, 137–139.

NIELSEN, F., AND NOCK, R. 2009. Hyperbolic Voronoi diagrams made easy. *ACM Computing Research Repository abs/0903.3287*.

NILFOROUSHAN, Z., AND MOHADES, A. 2006. Hyperbolic Voronoi diagram. In *Computational Science and Its Applications - ICCSA 2006*, vol. 3984 of *Lecture Notes in Computer Science*. Springer-Verlag, 735–742.

OKABE, A., BOOTS, B., SUGIHARA, K., AND CHIU, S. N. 1999. *Spatial tessellations: concepts and applications of Voronoi diagrams*, 2nd ed. John Wiley & Sons.

ONISHI, K., AND TAKAYAMA, N. 1996. Construction of Voronoi diagram on the upper half-plane. *IEICE transactions on fundamentals of electronics, communications and computer sciences E79-A*, 4, 533–539.

PEYRÉ, G., AND COHEN, L. 2004. Surface segmentation using geodesic centroidal tesselation. In *Proceedings of 2nd International Symposium on 3D Data Processing, Visualization, and Transmission,*, IEEE Computer Society, Washington, DC, USA, 995–1002.

RONG, G., LIU, Y., WANG, W., YIN, X., GU, X., AND GUO, X. 2010. GPU-assisted computation of centroidal Voronoi tessellation. *IEEE Transactions on Visualization and Computer Graphics.* to appear.

STAHL, S. 2007. Mass in the hyperbolic plane. *ACM Computing Research Repository abs/0705.3448.*

SURAZHSKY, V., ALLIEZ, P., AND GOTSMAN, C. 2003. Isotropic remeshing of surfaces: A local parameterization approach. In *Proceedings of the 12th International Meshing Roundtable*, 215–224.

VALETTE, S., CHASSERY, J.-M., AND PROST, R. 2008. Generic remeshing of 3D triangular meshes with metric-dependent discrete Voronoi diagrams. *IEEE Transactions on Visualization and Computer Graphics 14*, 2, 369–381.

VASCONCELOS, C. N., SÁ, A., CARVALHO, P. C., AND GATTASS, M. 2008. Lloyd's algorithm on GPU. In *Proceedings of the 4th International Symposium on Visual Computing*, 953–964.

YAN, D.-M., LÉVY, B., LIU, Y., SUN, F., AND WANG, W. 2009. Isotropic remeshing with fast and exact computation of restricted Voronoi diagram. *Computer Graphics Forum 28*, 5, 1445–1454. (Proceedings of Symposium on Geometry Processing 2009).

A Proof of Lemma 1

Lemma 1 *When the sites are fixed, the CVT energy $F^H(\mathbf{S})$ is minimized when the tessellation is the Voronoi diagram of the sites.*

Proof. Suppose we have another tessellation $\hat{\Omega}$ which is different to the Voronoi diagram Ω, and we define another energy function $\hat{F}^H(\mathbf{S})$ over it:

$$\hat{F}^H(\mathbf{S}) = \sum_{i=1}^{n} \int_{\hat{\Omega}_i} \rho(\mathbf{p}) \sinh^2(d_K^H(\mathbf{p}, \mathbf{s}_i))\, \mathrm{d}\sigma. \tag{13}$$

For a particular point $\mathbf{p}$ belonging to the Voronoi cell Ω_i, suppose in the tessellation $\hat{\Omega}$ it belongs to a cell $\hat{\Omega}_j$. Because of the definition of the Voronoi diagram, we always have

$$d_K^H(\mathbf{p}, \mathbf{s}_i) \le d_K^H(\mathbf{p}, \mathbf{s}_j),$$

and thus

$$\rho(\mathbf{p}) \sinh^2(d_K^H(\mathbf{p}, \mathbf{s}_i)) \le \rho(\mathbf{p}) \sinh^2(d_K^H(\mathbf{p}, \mathbf{s}_j)).$$

Since $\hat{\Omega}$ is not a Voronoi diagram, the above formula must hold with strict inequality on some cells. Thus,

$$F^H(\mathbf{S}) < \hat{F}^H(\mathbf{S})$$

so that $F^H(\mathbf{S})$ is minimized when the tessellation is the Voronoi diagram of the sites. □

B Proof of Lemma 2

Lemma 2 *When the tessellation is fixed, the CVT energy $F^H(\mathbf{S})$ is minimized when the sites locate at centroids of their Voronoi cells.*

Proof. First, we compute the *partial energy* over the Voronoi cell of the site $\mathbf{s}_i$

$$F^H(\mathbf{s}_i) = \int_{\Omega_i} \rho(\mathbf{p}) \sinh^2(d_K^H(\mathbf{p}, \mathbf{s}_i)) \, \mathrm{d}\sigma.$$

Since the energy is coordinate-independent, we can set the origin of the coordinate system at $\mathbf{s}_i$. By doing so, the distance $d_K^H(\mathbf{p}, \mathbf{s}_i)$ can be simplified to

$$d_K^H(\mathbf{p}, \mathbf{s}_i) = \frac{1}{2} \ln \frac{1 + \|\mathbf{p} - \mathbf{s}_i\|}{1 - \|\mathbf{p} - \mathbf{s}_i\|} = \tanh^{-1}(\|\mathbf{p} - \mathbf{s}_i\|). \quad (14)$$

So

$$\begin{aligned} F^H(\mathbf{s}_i) &= \int_{\Omega_i^H} \rho(\mathbf{p}) \sinh^2(\tanh^{-1}(\|\mathbf{p} - \mathbf{s}_i\|)) \, \mathrm{d}\sigma \\ &= \int_{\Omega_i^H} \rho(\mathbf{p}) \left(\frac{\|\mathbf{p} - \mathbf{s}_i\|}{\sqrt{1 - \|\mathbf{p} - \mathbf{s}_i\|^2}} \right)^2 \mathrm{d}\sigma. \quad (15) \end{aligned}$$

Denote $\mathbf{q} = \mathbf{p} - \mathbf{s}_i$ and $\Phi_i^H = \bigcup_{\mathbf{p} \in \Omega_i^H} \mathbf{q}$, we have:

$$F^H(\mathbf{s}_i) = \int_{\Phi_i^H} \rho(\mathbf{q}) \left(\frac{\|\mathbf{q}\|}{\sqrt{1 - \|\mathbf{q}\|^2}} \right)^2 \mathrm{d}\sigma$$

where $\rho(\mathbf{q}) = \rho(\mathbf{p})$ because the rigid transform does not change the density values. Next, using the mapping functions φ and ψ (as illustrated in Figure 3), we can map the above integral from the Klein disk to the plane $z = 0$ as:

$$\begin{aligned} F^H(\mathbf{s}_i) &= \int_{\Phi_i^H} \rho(\mathbf{q}) \left(\frac{\|\mathbf{q}\|}{\sqrt{1 - \|\mathbf{q}\|^2}} \right)^2 \mathrm{d}\sigma \\ &= \int_{\Phi'^H_i} \rho(\mathbf{q}') \|\mathbf{q}'\|^2 \, \mathrm{d}\sigma'. \quad (16) \end{aligned}$$

where $\mathbf{q}' = \psi(\phi(\mathbf{q}))$ and $\Phi'^H_i = \bigcup_{\mathbf{q} \in \Phi_i^H} \mathbf{q}'$.

Since Φ'^H_i is in 2D Euclidean space $z = 0$, it is known that the above energy function is minimized when its centroid is at the origin [Du et al. 1999]. The x-coordinate of the centroid is $x'_c = \frac{\int_{\Phi'^H_i} \rho(\mathbf{q}') x'_{\mathbf{q}} \, \mathrm{d}\sigma'}{\int_{\Phi'^H_i} \rho(\mathbf{q}') \, \mathrm{d}\sigma'}$. So to minimize $F^H(\mathbf{s}_i)$, we need to have $x'_c = 0$, and thus its numerator

$$\int_{\Phi'^H_i} \rho(\mathbf{q}') x'_{\mathbf{q}} \, \mathrm{d}\sigma' = 0.$$

Since the mapping function ψ dose not change the x-coordinate, $\overline{x}_{\mathbf{q}} = x'_{\mathbf{q}}$. Furthermore, $\rho(\overline{\mathbf{q}})\mathrm{d}\overline{\sigma} = \rho(\mathbf{q}')\mathrm{d}\sigma'$ due to the conservation of mass. So we have

$$\int_{\overline{\Phi}_i^H} \rho(\overline{\mathbf{q}}) \overline{x}_{\mathbf{q}} \, \mathrm{d}\overline{\sigma} = \int_{\Phi'^H_i} \rho(\mathbf{q}') x'_{\mathbf{q}} \, \mathrm{d}\sigma' = 0.$$

Comparing with Equation (6), this indicates the x-coordinate of $\mathbf{c}_i^H$ (the hyperbolic centroid of Φ_i^H) is 0. Similarly, we have the y-coordinate of $\mathbf{c}_i^H$ is 0. This means the numerator of Equation (6) is on z-axis. So after the normalization, $\mathbf{c}_i^H$ is at the point $(0, 0, 1)$, i.e. the center of the Klein disk. Now we have proved that when $\mathbf{s}_i$ locates at $\mathbf{c}_i^H$, the partial energy $F^H(\mathbf{s}_i)$ is minimized.

Applying the same proof for all the Voronoi cells, we have the conclusion that when all sites locate at the centroids of their Voronoi cells, the CVT energy function $F^H(\mathbf{S}) = \sum_{i=1}^n F^H(\mathbf{s}_i)$ is minimized. $\square$