Tutorial on Discrete Ricci Flow for Global Parameterizations

David X. Gu
Center of Visual Computing
State University of New York at Stony Brook
gu@cs.sunysb.edu

Feng Luo
Mathematics Department
Rutgers University

Shing-Tung Yau
Mathematics Department
Harvard University

August 30, 2005

Abstract

This work introduces the concepts and methods for Ricci flow for computer scientists and engineers. Readers can understand the background theories as well as the implementation details, such that they can make Ricci flow software easily and find potential applications in graphics field.

First, the basic concepts from local differential geometry are briefly introduced, the concepts of metric, curvature are explained in details. Then different energies are defined to quantitative measure the distorison of parameterizations. The conformal parameterizations are emphasized.

Second, the theories from global differential geometry are thoroughly explained, such as manifolds, affine atlas, Riemann surfaces, Riemann uniformization theorem. Then Ricci flow is introduced to conformally deform surfaces, such that the solution surfaces have constant Gaussian curvatures.

Third, the concepts and methods from continuous geometry are systematically translated to the discrete setting via circle packing metric. The discrete Ricci flow is thoroughly explained, the existence of the solution, the exponential convergence, the variational energy, the Newton's method are explained.

Finally, discrete Ricci flow is implemented based a common mesh library. The details of the algorithms are illuminated. Experimental results are illustrated and discussed.

Readers who are only interested in the implementation of Ricci flow can skip the first two chapters.

Contents

1 Introduction

Shape representation and deformation are fundamental problems in computer graphics. Ricci flow is a theoretic solid and piratical simple method for tackling these problems.

Ricci Flow was first introduced in differential geometry by Hamilton [3] in $1980's$. Later, Hamilton generalize Ricci flow for 3-manifolds. It has been broadly studied and developed by pure mathematicians and has recently been applied to prove the famous Poincare conjecture on the topology of 3-manifolds [7].

Circle packing was introduced by Thurston [9], which is a bridge to transfer conformal mappings from smooth surface case to combinatorial graphs.

Chow and Luo [1] combined Ricci flow with circle packing and established the theoretic foundations of combinatorial Ricci flow.

Gu and Luo [] implemented the Ricci flow algorithms and improved the efficiency by changing gradient flow to Newton's method. The method has been applied for global parameterizations, and further manifold splines.

In the following discussion, we briefly draw the big pictures in both continuous setting and discrete setting. They are systematically "dual" to each other.

1.1 Motivation

Shape representation and deformation are the central problems in computer graphics and geometric modelling.

In engineering fields, triangular meshes are commonly used to represent shapes, its connectivity models the topology, the edge lengths describes the metric (intrinsic geometry), the dihedral angles further determine the embedding of the mesh in $\mathbb{R}^3$.

The edge lengths determine the curvature on each vertex. But the inverse is much more difficult,

Problem 1 *Given curvature on the mesh, how to find compatible edge lengths?*

This problem has fundamental importance. The solution to this problem will allow the users to model the shapes by designing their curvature.

For example, surface parameterizations have played an important role in graphics. Many real applications in graphics heavily rely on parameterizations, such as texture mapping, shape comparison, fluid simulation, geometric morphing and so on. Surface parameterization is equivalent to find a special configuration of edge lengths, such that the curvatures of vertices are zero, namely, the mesh is flat.

Another example is for parametric surface, especially splines. In order to model natural shapes with manifold structure, special parameter atlas need to be constructed, such that all the chart transition functions are affine. Finding the affine atlas is equivalent to find a flat metric of the mesh.

For surface fairing, it is desirable to distribute the curvature more uniformly on vertices. It is straightforward to compute the resulting curvature, but difficult to find the edge length and the embedding of the mesh.

Ricci flow is the powerful tool to solve the problem. It offers the freedom to traverse the intrinsic shape space (all the admissible configurations of edge lengths)that can be represented by a mesh, enable the users to model shapes by designing their curvature distributions. The most direct real applications include global surfaces parameterizations, manifold splines, surface fairing, shape matching, shape morphing etc.

1.2 Continusing Setting

A surface in the Euclidean space $\mathbb{R}^3$ has three level information,

- Topology,
- Riemannian Metric,
- Embedding.

Topology is determined by the number of boundaries and handles of the surface. Metric is a structure such that the lengths and angles of tangent vectors can be measured. Embedding is the way the surface sits in $\mathbb{R}^3$.

Gaussian curvature is the measurement of how close a neighborhood of a point on the surface to a plane, it is solely determined by the Riemannian metric, and independent of the embedding of the surface. But, the Gaussican curvature is confined by the topology of the surface.

A topological surface can be equipped with different Riemannian metrics. Two metrics are conformal or angle preserving if for any two tangent vectors, the angles between them are the same measured by the different metrics. Therefore, all possible Riemannian metrics of a surface can be classified by this conformal equivalence relation.

Any surface embedded in $\mathbb{R}^3$ has a unique metric induced by the Euclidean metric of $\mathbb{R}^3$. The surface can be equipped by a unique metric, which is conformal equivalent to the induced metric, and it has constant Gaussican curvature. One can

ask a much broader question:
Given a function satisfying the topological constraint, can one find a Riemannian metric, such that the Gaussican curvature induced by the metric equals to the function? If it exists, how to compute it?

The answer to these questions are the main focus of this tutorial, roughly speaking,
The metric exists, it is unique in each conformal class. It can be computed using Ricci flow.

The basic idea of Ricci flow is to deform the current metric conformally driven by the difference between current Gaussian curvature and the target Gaussian curvature pointwisely. The flow will converge to the desired metric, the curvature error shrinks exponentially fast.

The above problem can be modelled as a variational problem, such that by minimizing the energy, the desired metric can be reached. The energy function is convex, therefore it has unique global optima. Ricci flow is just the gradient flow. By using Newton's method, the convergence speed can be further improved.

1.3 Discrete Setting

In computer graphics and geometric modeling, general surfaces in $\mathbb{R}^3$ are represented as triangular meshes. Each mesh has three level information,

- Topology, indicated by the connectivity of the mesh.
- Riemannian Metric, the edge lengths.
- Embedding, the dihedral angles for edges.

The Gaussian curvature of a vertex is the measurement of the difference of its one ring neighbor with the plane. It is defined as the difference between the summation of its adjacent angles and 2π. The Gaussian curvature is solely determined by the edge lengths. The total Gaussian curvature of all vertices equals to $2\pi\chi$, where χ is the Euler number of the mesh.

In smooth case, a conformal deformation has the following crucial properties,

1. It transform an infinitesimal circle to an infinitesimal circle.
2. It preserves the intersection angles among the infinitesimal circles.

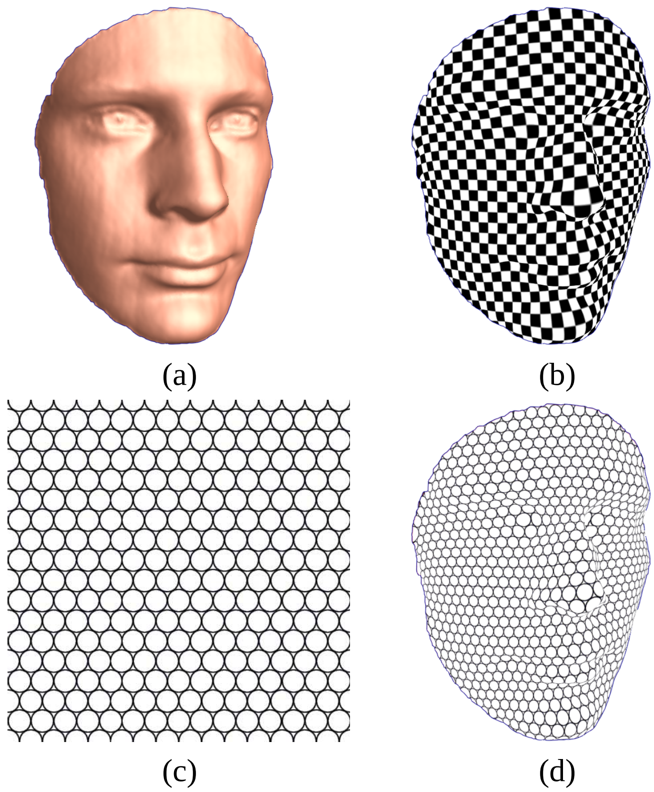


Figure 1: **Conformal mapping and its properties.** Conformal mappings preserves angles, the right angles of checkers in are preserved in (b). Conformal mapping transforms the infinitesimal circles on the texture plane to the infinitesimal circles on the surface, it also preserves the tangency of circles.

A cone is associated with each vertex, such that the cone angle equals to the curvature of the vertex, the boundary of each cone is a circle. Each edge connecting 2 vertices, the corresponding 2 circles intersect each other. The edge length is determined by the radii of the circles and their intersection angle. We call this kind of edge lengths a *circle packing metric* of the mesh.

One can change the circle radii, preserving the intersection angles of a circle packing metric. This kind of deformation is the analogy of conformal deformation in smooth case, and called the discrete conformal deformation.

Given a closed mesh equipped with a circle packing metric, one can conformally deform its metric such that the final metric can be realized in constant Gaussican curvature spaces. Namely, a closed genus zero mesh can be embedded in a sphere, each edge is a geodesic, the length equals to the metric; a closed genus one mesh can be embedded in the plane, each edge is realized by a line segment with the length of the metric; a high genus closed mesh can be realized in the hyperbolic disk, each edge is a geodesic with the length specified by the circle packing metric.

One can ask a much broader question:
Given a function satisfying the topological constraint defined on the vertices, can one find a circle packing metric, such that the Gaussican curvature induced by the metric equals to the function? If it exists, how to compute it?

The answer to these questions are the main focus of this tutorial, roughly speaking,
The curvature function has more constraints than topological constraint. If the curvature function satisfies all the constraints, then the circle packing metric exists, it is unique in each conformal class. It can be computed using discrete Ricci flow.

The basic idea of discrete Ricci flow is to deform the vertex radii driven by the difference between current Gaussian curvature and the target Gaussian curvature on each vertex. The flow will converge to the desired metric, the curvature error shrinks exponentially fast.

The above problem can be modelled as a variational problem, such that by minimizing the energy, the desired metric can be reached. The energy function is convex, therefore it has unique global optima. Ricci flow is just the gradient flow. By using Newton's method, the convergence speed can be further improved.

2 Smooth Ricci flow

This section introduces the concepts and theoretic results of smooth surface Ricci flow.

We first introduce the major relevant concepts from local differential geometry, then from global differential geometry.

2.1 Local Differential Geometry

Suppose a surface $S \subset \mathbb{R}^3$ has a parametric representation,

$$\mathbf{r}(u,v) = (x(u,v), y(u,v), z(u,v))$$

for points (u,v) in some domain in $\mathbb{R}^2$. The parameterization is *regular* if

1. $x_1(u,v), x_2(u,v), x_3(u,v)$ are smooth functions.
2. The tangent vectors
$$\mathbf{r}_u = \frac{\partial \mathbf{r}}{\partial u}, \mathbf{r}_v = \frac{\partial \mathbf{r}}{\partial v},$$
are linearly independent at every point.

Therefore, the *normal* vector

$$\mathbf{n}(u,v) = \frac{\mathbf{r}_u \times \mathbf{r}_v}{|\mathbf{r}_u \times \mathbf{r}_v|}$$

is well defined everywhere.

The first fundamental form of S is defined as

$$ds^2 = Edu^2 + 2Fdudv + Gdv^2,$$

where

$$E = \mathbf{r}_u \cdot \mathbf{r}_u, F = \mathbf{r}_u \cdot \mathbf{r}_v, G = \mathbf{r}_v \cdot \mathbf{r}_v.$$

Suppose two tangent vectors at (u,v) are

$$d\mathbf{r}_1 = \mathbf{r}_u du_1 + \mathbf{r}_v dv_1, d\mathbf{r}_2 = \mathbf{r}_u du_2 + \mathbf{r}_v dv_2,$$

then the inner product of them is defined as

$$< d\mathbf{r}_1, d\mathbf{r}_2 >_g = \begin{pmatrix} du_1 & dv_1 \end{pmatrix} \begin{pmatrix} E & F \\ F & G \end{pmatrix} \begin{pmatrix} du_2 \\ dv_2 \end{pmatrix}.$$

Thus, the length and angles of tangent vectors can be measured by the first fundamental form. First fundamental form is also called the *Riemannian metric* of the surface. The geometry determined by the metric is called the *intrinsic geometry*, which is independent of the embedding of the surface in $\mathbb{R}^3$, such as the geodesics.

The surface embedding is described by the second fundamental form,

$$II = Ldu^2 + 2Mdudv + Ndv^2,$$

where

$$L = \mathbf{r}_{uu} \cdot \mathbf{n}, M = \mathbf{r}_{uv} \cdot \mathbf{n}, N = \mathbf{r}_{vv} \cdot \mathbf{n}.$$

First fundamental form and the second fundamental form together determines the surface uniquely up to rotation and translation in $\mathbb{R}^3$.

The the map between the surface $\mathbf{r}(u,v)$ to the normal vector $\mathbf{n}(u,v)$ is called *Gauss map*, its derivative map $W : d\mathbf{r}(u,v) \to d\mathbf{n}(u,v)$ is called the *Weingarten map*, which is a linear map, and can be represented easily as $\lambda \mathbf{r}_u + \mu \mathbf{r}_v \to \lambda \mathbf{n}_u + \mu \mathbf{n}_v$,

$$W = \begin{pmatrix} L & M \\ M & N \end{pmatrix} \begin{pmatrix} E & F \\ F & G \end{pmatrix}^{-1}$$

The determinant of the matrix W represents the area distortion of the Gauss map, and is defined as *Gaussian curvature*,

$$K = |W| = \frac{LN - M^2}{EG - F^2}.$$

By definition, Gaussian curvature requires the embedding of the surface (L, M, N), but in fact, it can be computed solely using the metric (E, F, G), namely, it is intrinsic. The formula is

$$K = -\frac{1}{\sqrt{EG}}[(\frac{(\sqrt{E})_v}{\sqrt{G}})_v + (\frac{(\sqrt{G})_u}{\sqrt{E}})_u]$$

Consider a curve on surface $(u(s), v(s))$, assume the tangent direction of the curve has an angle $\theta(s)$ with $\mathbf{r}_u$, then the *geodesic curvature* of the curve is defined as

$$k_g = \frac{d\theta}{ds} - \frac{1}{2\sqrt{G}}\frac{\partial \ln E}{\partial v}\cos\theta + \frac{1}{2\sqrt{E}}\frac{\partial \ln G}{\partial u}\sin\theta.$$

Geodesic curvature is also intrinsic.

Suppose there are two surface patches, $S_1(u,v)$ and $S_2(u,v)$, the map $\phi : S_1(u,v) \to S_2(u,v)$ is called a *conformal mapping*, if

$$\frac{E_1(u,v)}{E_2(u,v)} = \frac{F_1(u,v)}{F_2(u,v)} = \frac{G_1(u,v)}{G_2(u,v)} = \lambda(u,v),$$

where $\lambda(u,v)$ is called the *conformal factor*. It can be easily verified that, any two intersecting curves on S_1 will be mapped to S_2, and the intersection angle doesn't change. Therefore, conformal mapping is also called *angle preserving mapping*.

Especially, the parameterization (u,v) of $S(u,v)$ is a *conformal parameterization*, if the metric can be represented as

$$ds^2 = \lambda(u,v)^2(du^2 + dv^2).$$

Many geometric computations will be simplified using conformal parameter, such as the Gaussian curvature

$$K(u,v) = \Delta_g \ln \lambda(u,v).$$

where

$$\Delta_g = \frac{1}{\lambda^2}(\frac{\partial^2}{\partial u^2} + \frac{\partial^2}{\partial v^2}),$$

is the Laplace-Beltrami operator. Conformal mapping preserves the *shapes* of the parameter domain, it is highly desirable to use conformal parameterization for graphics applications, such as texture mapping.

Although, conformal parameterization has no angle distorsion, it will introduce *area distortion*. If conformal factor function equals to one everywhere, there will be no area distortion at all, the resulting map is an *isometric map*. In general two surfaces have an isometric map between them, share the same metric, so the Gaussian curvature functions are equal. It is impossible, because surfaces are usually curved, and the parameter plane is flat.

In order to measure the area distortion for a conformal parameterization, we define the following *area distortion energy*

$$\int_S (\lambda(u,v) - 1)^2 dA = \int_S (\lambda(u,v) - 1)^2 \lambda^2(u,v) dudv,$$

with the assumption, both the parametric domain area equals to the surface area and equals to one, otherwise, we can normalize the surface first, namely

$$\int_S dA = 1, \int_D dudv = 1.$$

where D is the parameter domain.

2.2 Global Differential Geometry

General surfaces can not be covered by a single parameter domain, instead, they may need many local parameters overlapping one another. Therefore, one region on the surface may be covered by several parameter charts. All the geometric meaningful quantities should be consistent under different parameterizations.

A *manifold* of dimension n is a connected Hausdorff space M for which every point has a neighborhood U that is homeomorphic to an open subset V of $\mathbb{R}^2$. Such a homeomorphism

$$\phi : U \to V$$

is called a *coordinate chart*. An *atlas* is a family of charts $\{(U_\alpha, \phi_\alpha)\}$ for which U_α constitute an open covering of M.

Suppose $\{(U_\alpha, \phi_\alpha)\}$ and $\{(U_\beta, \phi_\beta)\}$ are two charts on a manifold M, $U_\alpha \cap U_\beta \neq \emptyset$, the chart transition is

$$\phi_{\alpha\beta} : \phi_\alpha(U_\alpha \cap U_\beta) \to \phi_\beta(U_\alpha \cap U_\beta).$$

If all chart transition functions are affine maps on $\mathbb{R}^2$, the atlas is called an *affine atlas*.

Theorem 2 *A surface admits an affine atlas, if and only if it has zero Euler number.*

Affine atlas plays an crucial role for manifold splines. A surface admits a manifold spline if and only if it admits an affine atlas. In practice, it is important to compute an affine atlas. This can be accomplished by Ricci flow.

If all chart transition functions are holomorphic functions, the atlas is called an *conformal atlas*. If a metric surface admits an conformal atlas, it is called a *Riemann surface*.

Theorem 3 *All metric surfaces are Riemann surfaces.*

Therefore, we can use conformal parameter charts to cover the whole surface. Riemann surface has special vector fields, the so-called holomorphic 1-forms, which have zero curl and divergence, and have been applied for global conformal surface parameterization [].

The global definition of a metric is as the following: on chart $\{U_\alpha, \phi_\alpha\}$, the metric has the form

$$\begin{pmatrix} g_{11}^\alpha & g_{12}^\alpha \\ g_{21}^\alpha & g_{22}^\alpha \end{pmatrix}$$

on another chart $\{U_\beta, \phi_\beta\}$, the metric is

$$\begin{pmatrix} g_{11}^\beta & g_{12}^\beta \\ g_{21}^\beta & g_{22}^\beta \end{pmatrix} = \begin{pmatrix} \frac{\partial u_\beta}{\partial u_\alpha} & \frac{\partial u_\beta}{\partial v_\alpha} \\ \frac{\partial v_\beta}{\partial u_\alpha} & \frac{\partial v_\beta}{\partial v_\alpha} \end{pmatrix} \begin{pmatrix} g_{11}^\alpha & g_{12}^\alpha \\ g_{21}^\alpha & g_{22}^\alpha \end{pmatrix} \begin{pmatrix} \frac{\partial u_\beta}{\partial u_\alpha} & \frac{\partial u_\beta}{\partial v_\alpha} \\ \frac{\partial v_\beta}{\partial u_\alpha} & \frac{\partial v_\beta}{\partial v_\alpha} \end{pmatrix}^T.$$

We simply use (M, g) to denote a manifold M equipped with a Riemannian metric g.

Suppose M is a manifold, it can be equipped with different metrics. But all metrics satisfy the following Gauss-Bonnet formula,

Theorem 4 *A surface with a Riemannian metric* (M, g)*, then*

$$\int_M K dA + \int_{\partial M} K_g ds = 2\pi\chi(M),$$

where K *is the Gaussican curvature induced by* g*,* K_g *is the geodesic curvature,* ∂M *is the boundary of* M*,* $\chi(M)$ *is the Euler number of* M*.*

This means the metric has the topological constraints.

Any closed metric surface (M, g) has a special metric $\bar{g}$, such that $\bar{g}$ is conformal equivalent to g, and $\bar{g}$ has constant Gaussian curvature everywhere.

Theorem 5 *Any closed metric surface* (M, g) *admits a canonical metric* $\bar{g}$*, such that*

1. $\bar{g}$ *is conformally equivalent to* g*,* $\bar{g} = \lambda g$*,*
2. $\bar{g}$ *induces constant Gaussian curvature.*

Namely, closed genus zero surface can be conformally deformed to a spherical metric, with $+1$ curvature; genus one surface can be conformally deformed to have flat metric, with 0 curvature; high genus surfaces can have a hyperbolic metric conformally equivalent to its original metric, with -1 curvature.

Suppose the desired Gaussian curvature $\bar{K}$ is assigned to the surface M, we would like to find a desired metric $\bar{g}$, such that it induces $\bar{K}$. Ricci flow is able to accomplish it.

Definition 6 *A Ricci flow for a surface* (M, g) *is defined as*

$$\frac{dg_{ij}}{dt} = (\bar{K} - K) g_{ij}.$$

Basically, the metric is deformed by the difference between the current curvature and the target curvature.

Theorem 7

1. *For all the time, the solution to the Ricci flow exists and unique.*
2. *The convergence is exponentially fast.*
3. *The metrics of the solutions are conformal equivalent to the original metric.*

4. *If $\bar{K} \equiv 0$, and the surface area is normalized to be a constant, then the final metric will induce a constant curvature.*

Basically, suppose (M, g) is a metric surface equipped with a Riemannian metric g, $\lambda > 0$ is a positive functions defined on M. Then the Gaussian curvature map $\Pi : \lambda g \to K$is a homeomorphism between the conformal metric space

$$G = \{\lambda g | \lambda : M \to \mathbb{R}^+\},$$

and the curvature space

$$K = \{K | K : M \to \mathbb{R}, \int_M K dA + \int_{\partial M} K_g ds = 2\pi\chi(M)\}.$$

The inverse map $\Pi^{-1} : K \to \lambda g$ can be computed using Ricci flow.

The solution to the Ricci flow

$$\frac{g_{ij}}{dt} = -K g_{ij},$$

conformally deform the metric

$$g_{ij}(t) = g_{ij}(0) e^{-\int_0^t K(\tau) d\tau}.$$

The conformal factor $\lambda = e^{-\int_0^\infty K(\tau) d\tau}$ is the global minimum of the following energy,

$$E(\lambda) = \int_M K_{\lambda g} dA^2_{\lambda g},$$

with the normalized area, such that

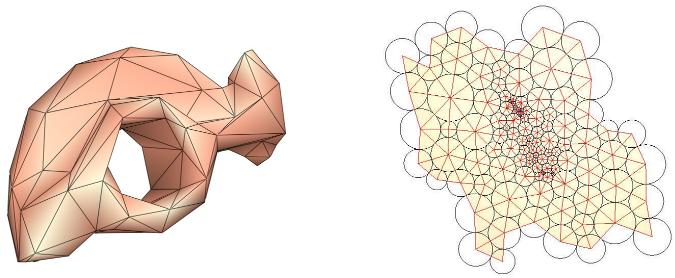
$$\int_M dA_{\lambda g} = 1.$$

where $K_{\lambda g}$ is the Gaussian curvature under the metric λg, $dA_{\lambda g}$ is the area element under the metric λg.

3 Discrete Ricci Flow

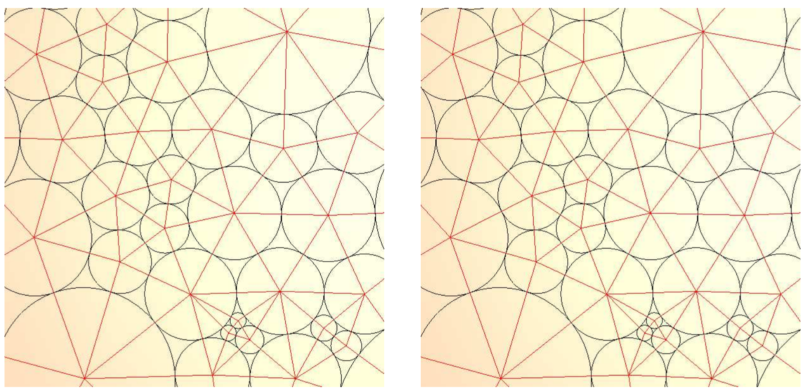
Continuous Surface	Discrete Mesh
Metric (First fundamental form)	edge length
Second fundamental form	dihedral angle
For convex surfaces, metric determines the embedding	For convex meshes, edge lengths determine the dihedral angles
Gaussian curvature $K = -\frac{1}{\sqrt{EG}}[(\frac{(\sqrt{E})_v}{\sqrt{G}})_v + (\frac{(\sqrt{E})_u}{\sqrt{E}})_u]$	Discrete Gaussian curvature $K_i = 2\pi - \sum_{f_{ijk}\in F} \theta_i^{jk}$
Geodesic Curvature	Discrete geodesic curvature $K_i = \pi - \sum_{f_{ijk}\in F} \theta_i^{jk}, v_i \in \partial M$
Conformal equivalent metrics $\lambda : M \to \mathbb{R}^+, \{\lambda g\}$	circle packing metrics based on (M, Φ) $\{(M, \Phi, \Gamma)\}$
conformal mapping	a mapping between (M, Φ, Γ_1) and (M, Φ, Γ_2), preserves edge angles Φ
Gauss-Bonnet formulae $\int_M KdA + \int_{\partial M} K_g ds = 2\pi\chi$	Discrete Gauss-Bonnet formulae $\sum_{v_i \notin \partial M} K_i + \sum_{v_i\in\partial M} K_i = 2\pi\chi$
None	Combinatorial constraints for Gaussian curvature on vertices
Ricci flow $\frac{dg_{ij}}{dt} = (\bar{K} - K)g_{ij}$	discrete Ricci flow $\frac{d\gamma_i}{dt} = (\bar{K}_i - K_i)\gamma_i$
Ricci flow energy $\int_M K^2_{\lambda g} dA_{\lambda g}$	discrete Ricci flow energy $\int \sum_i K_i dln\gamma_i$
The solution to Ricci flow exists, unique. The flow exponentially converges.	the solution to discrete Ricci flow exists unique. The flow exponentially converges.
The flow $\frac{dg_{ij}}{dt} = -Kg_{ij}$ with normalized total area leads to a metric with constant Gaussian curvature	The flow $\frac{d\gamma_i}{dt} = -K_i\gamma_i$ with normalized total area leads to a metric with constant Gaussian curvature.

Table 1: Concepts and theories in continuous setting and discrete setting



(a) genus one mesh (b) A flat circle packing metric

Figure 2: **A flat circle packing metric for a genus one mesh.**



(a) circle packing metric (b) Zoomed in of (a)

Figure 3: **Circle packing metric.**

The major theoretic results of Ricci flow for smooth surfaces can be systematically translated to the discrete setting. The bridge from smooth surface to triangular mesh is the so called *circle packing* metric [9].

In this section, we explain the theoretical background of discrete Ricci Flow and show its exponential convergence rate. Theoretic proofs can be found in [1]. Conventional Ricci flow is a gradient flow of some energy form, we introduce a novel algorithm based on the Newton's method, it converges much faster.

A two dimensional simplicial complex (triangular mesh) is denoted $M = (V, E, F)$, where V is the set of all vertices, E is the set of all non-oriented edges and F the set of all faces. We use $v_i, i = 1, \cdots, n$ to denote its vertices, e_{ij} to denote an oriented edge from v_i to v_j, and f_{ijk} to denote an oriented face with vertices v_i, v_j, v_k which are ordered counter-clockwise such that the face normals

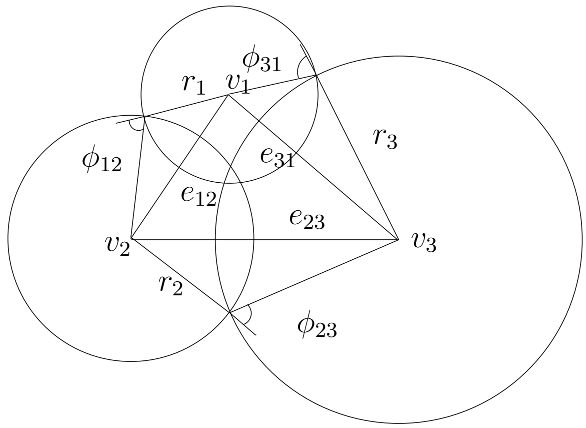


Figure 4: **Circle packing metric for a triangle**. Triangle $[v_1, v_2, v_3]$ has vertices v_1, v_2, v_3, and edges e_{12}, e_{23}, e_{31}. Three circles centered at v_1, v_2, and v_3, with radii γ_1, γ_2 and γ_3 intersect one another, with intersection angles of Φ_{12}, Φ_{23} and Φ_{31}, which are the weights associated with the edges. The edge lengths of the triangle are determined by γ_i and Φ_{ij} by the cosine law.

toward outside.

3.1 Circle Packing

The following key observation plays vital role for systematically translating smooth Ricci flow to discrete Ricci flow,

Observation 8 *A conformal mapping has the following two properties,*

1. *It transform infinitesimal circles to infinitesimal circles.*

2. *It preserves the intersection angles between two infinitesimal circles.*

Figure 1 illustrates these properties of a conformal mapping. In order to translate conformal mappings from smooth surface category to discrete mesh category, Thurston defined *circle packing* as the followings,

1. Change infinitesimal circles to circles with finite radii.

2. Each circle is centered at a vertex like a cone, the radius is denoted as γ_i for vertex v_i.

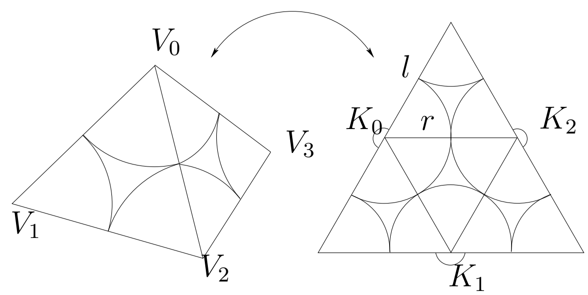


Figure 5: **Circle packing metric and curvature.** For a canonical tetrahedron, the edges lengths are all $l = 1.0$, and the radius at each vertex is $r = 0.5$. The curvature on each vertex equals to $K_i = \pi$. The weights on all edges are $\Phi = 0$.

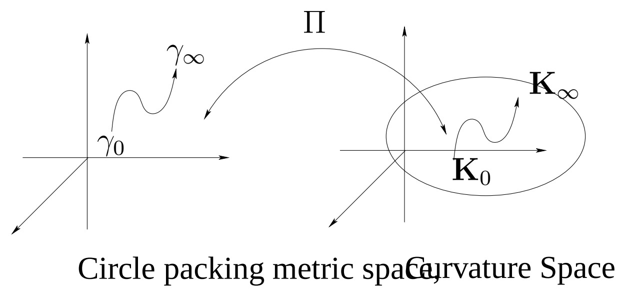


Figure 6: **Gaussian curvature is a homeomorphism between the circle packing metric space and the curvature space. The inverse map can be computed using Ricci Flow.** Starting from the known metric γ_0 and the known curvature $\mathbf{k}_0$, using Ricci Flow, as we flow to the target curvature $\mathbf{K}_\infty$, the metric flows to the corresponding metric $\gamma_\infty = \Pi^{-1}(\mathbf{K}_\infty)$.

3. An edge has two vertices, the two circles intersect each other with an intersection angle, the angle is denoted as Φ_{ij} for edge e_{ij}, and called the *weight*.

The way to determine the radii γ_i and the intersection angle Φ_{ij} is to make them *compatible* to the induced metric from $\mathbb{R}^3$.

Definition 9 *A mesh with circle packing* (M, Γ, Φ)*, where* M *represents the triangulation (connectivity),* $\Gamma = \{\gamma_i, v_i \in V\}$ *are the vertex radii,* $\Phi = \{\Phi_{ij}, e_{ij} \in E\}$ *are the angles associated with each edge. A* discrete conformal mapping

$$\tau : (M, \Gamma, \Phi) \to (M, \bar{\Gamma}, \Phi),$$

solely changes the vertex radii Γ*, but preserves the intersection angles* Φ*.*

In really, a discrete conformal mappings can approximate a smooth conformal mapping with arbitrary accuracy. If we keep subdividing the mesh and construct refiner and refiner circle packing, the discrete conformal mappings will converge to the smooth conformal mapping. For a rigorous proof, we refer the readers to [8].

In graphics and geometric modeling applications, meshes are usually embedded in $\mathcal{R}^3$, and the metrics are induced from those of $\mathbf{R}^3$. We can find the optimal weight Φ with initial circle radii Γ, such that the circle packing metric (M, Φ, Γ) is as close as possible to the Euclidean metric in the least square sense.

Namely, we want to find (M, Φ, Γ) by minimizing the following functional

$$\min_{\Gamma, \Phi} \sum_{e_{ij} \in E} |l_{ij} - \bar{l}_{ij}|^2, \tag{1}$$

where $\bar{l}_{ij}$ is the edge length of e_{ij} in $\mathbb{R}^3$.

After finding the optimal circle packing (M, Φ, Γ), we will use discrete Ricci flow to adjust the vertex radii Γ to deform the mesh, therefore, the deformation is discrete conformal.

3.2 Discrete Metric and Gaussian Curvature

We first define the circle packing metric and the Gaussian curvature for the mesh.

Definition 10 *A* metric *on the triangular mesh M is an edge length function*

$$l : E \to \mathbb{R}^+,$$

satisfying the triangle inequality, namely for each face f_{ijk},

$$l_{ij} + l_{jk} > l_{ki}.$$

The intersection angle associated with each edge is defined as the weight of the edge,

Definition 11 *A* weight *on the mesh is a function defined on edges $\Phi : E \to [0, \frac{\pi}{2}]$.*

Definition 12 *A* radius *function assigns to each vertex v_i a positive number γ_i, $\Gamma : V \to \mathcal{R}^+$.*

A circle packing (M, Φ, Γ) uniquely determines a metric, defined as

Definition 13 *A circle packing* (M, Φ, Γ) *deduces a metric. The edge length* l_{ij} *is associated with the edge* e_{ij} *is computed using the Cosine law,*

$$l_{ij} = \sqrt{\gamma_i^2 + \gamma_j^2 + 2\gamma_i\gamma_j \cos \Phi_{ij}}. \tag{2}$$

This metric is called the circle packing metric *of* (M, Φ, Γ)*. The metrics deduced from* Φ, Γ *using the Cosine Law are called the* circle-packing metrics *based on* (M, Φ).

Figure 4 illustrates a circle packing metric for a triangle f_{ijk}. The triangle is formed by the centers of three circles of radii γ_i, γ_j and γ_k intersecting at angles Φ_{ij}, Φ_{jk} and Φ_{ki}.

The discrete Gaussian curvature is defined as the difference between the one-ring neighbor of the vertex and the plane.

Definition 14 *Given a metric, let the angle of vertex* v_i *in* f_{ijk} *be denoted by* θ_i^{jk}*. Then the discrete Gaussian curvature* K_i *at an interior vertex* v_i *is defined as*

$$K_i = 2\pi - \sum_{f_{ijk} \in F} \theta_i^{jk}, \qquad v_i \notin \partial M, \tag{3}$$

while the discrete Gaussian curvature for a boundary vertex v_i *is defined as*

$$K_i = \pi - \sum_{f_{ijk} \in F} \theta_i^{jk}, \qquad v_i \in \partial M. \tag{4}$$

Figure 5 demonstrates the circle packing metric for mesh formed by a tetrahedron, where all the edge weights are zero, all the vertex radii are 0.5, and all the vertex curvatures are π.

The Gaussian curvature at each individual vertex is arbitrary, but the *total* curvature is determined by the topology of the surface, as indicated by the Gauss-Bonnet theorem:

Theorem 15 (Gauss-Bonnet) *Suppose* M *is a mesh, the total discrete Gaussian curvature equals to the product of* 2π *and its Euler number,*

$$\sum K_i = 2\pi\chi. \tag{5}$$

where $\chi = |V| + |F| - |E|$.

Furthermore, for any circle packing metric (M, Φ, Γ), $\Phi : E \to [0, \frac{\phi}{2}]$ and any proper subset I of vertices V,

$$\sum_{i \in I} K_i(r) > - \sum_{(e,v) \in Lk(I)} (\pi - \Phi(e)) + 2\pi\chi(F_I), \tag{6}$$

where F_I is the set of all faces in M whose vertices are in I, $Lk(I)$ is the link of I, the set of pairs (e, v) of an edge e and a vertex v satisfying

1. the end points of e are not in I,

2. the vertex v is in I, and (3) e and v form a triangle.

The following fundamental theorem states that the map between the vertex radii Γ and the discrete Gaussian curvature K is a homeomorphism(a one to one continuous map, the inverse is also continuous):

Theorem 16 (Vertex Radii and Curvature) *Let P be the set of normalized vertex radii, such that the product of the radii is one*

$$P = \{\gamma = (\gamma_1, \gamma_2, \cdots, \gamma_n) \in \mathbb{R}^n_{>0} | \Pi_{i=1}^n \gamma_i = 1\}$$

The Gaussian curvature map

$$\Pi : P \to \mathbb{R}^n$$

sends the vertex radii γ to its curvature

$$\Pi(\gamma) = (K_1(\gamma), K_2(\gamma), \cdots, K_n(\gamma)).$$

By the Gauss-Bonnet Theorem, its image lies in the convex polytope Y defined by the set of all linear Inequalities 6 on the hyperplane defined by Equation 5. The map $\Pi : P \to Y$ is a homeomorphism.

This theoretic result is very useful for practical applications. It allows the user to design surfaces by designing their vertex curvatures first, then finding the corresponding edge lengths using discrete Ricci flow, and finally finding the positions of vertices.

3.3 Ricci Flow

One can assign discrete values for Gaussian curvature $\bar{K}$ for a weighted mesh (M, Φ) as long as $\bar{K}$ satisfies the Conditions 5 and 6. Having done so, we wish to find the unique circle packing metric $\bar{\gamma}$ which induces the curvature $\bar{K}$. For this purpose, we introduce the discrete Ricci Flow.

Definition 17 (Discrete Ricci Flow) *We define the discrete Ricci Flow as*

$$\frac{d\gamma_i}{dt} = (\bar{K}_i - K_i)\gamma_i, \tag{7}$$

where $\bar{K}_i$ is the desired discrete Gaussian curvature at vertex v_i.

Definition 18 (convergence) *A solution to Equation 7 exists and is* convergent *if*

1. $\lim_{t\to\infty} K_i(t) = \bar{K}_i$ *for all* i,
2. $\lim_{t\to\infty} \gamma_i(t) = \bar{\gamma}_i \in \mathcal{R}^+$ *for all* i.

A convergent solution converges exponentially *if there are positive constants* c_1, c_2, *so that for all time* $t \geq 0$

$$|K_i(t) - \bar{K}_i| \leq c_1 e^{-c_2 t},$$

and

$$|\gamma_i(t) - \bar{\gamma}_i| \leq c_1 e^{-c_2 t}.$$

In theory, the discrete Ricci Flow is guaranteed to be exponentially convergent.

Theorem 19 *Suppose (M, Φ) is a closed weighted mesh. Given any initial circle-packing metric based on the weighted mesh, the solution to the discrete Ricci Flow 7 in the Euclidean geometry with the given initial value exists for all time and converges exponentially fast. The solution converges to the metric $\Pi^{-1}(\bar{K})$.*

The basic idea to show the convergence is to use variational approach. Special energy form is constructed, Ricci flow is the neigative gradient flow of the energy. The energy form is convex (namely, the Hessian matrix is positive definite everywhere), therefore, global minima exists and unique. For detailed theoretic proof, we refer the readers to [1].

3.4 Variational Approach

Discrete Ricci Flow is the solution to a variational problem, namely, it is the negative gradient flow of some convex energy, and therefore we can use Newton's method to further improve the convergence.

Let $u_i = \ln \gamma_i$. Under this change of variable, the Ricci Flow in Equation 7 takes the following form:

$$\frac{du_i}{dt} = -(K_i - \bar{K}_i),$$

The corresponding energy form is defined as

$$f(\mathbf{u}) = \int_{\mathbf{u}_0}^{\mathbf{u}} \sum_{i=1}^{n} (K_i - \bar{K}_i) du_i,$$

where $\mathbf{u} = (u_1, u_2, \cdots, u_n)$, $\mathbf{u}_0$ is $(0, 0, \cdots, 0)$.

Thus $\frac{\partial f}{\partial u_i} = K_i - \bar{K}_i$, that is, the Ricci Flow 7 is the negative gradient flow of the energy f.

The Heissian matrix of the energy f is

$$\frac{\partial^2 f}{\partial u_i \partial u_j} = \frac{\partial K_i}{\partial u_j},$$

$$\frac{\partial K_i}{\partial u_j} = \gamma_j \frac{\partial K_i}{\partial \gamma_j} = \begin{cases} \gamma_j \sum_k \frac{B_k^{ij}}{\sqrt{1-(A_k^{ij})^2}} & i = j \\ 0 & i \neq j, e_{ij} \notin E \\ \gamma_j \sum_K \frac{C_k^{ij}}{\sqrt{1-(A_k^{ij})^2}} & i \neq j, e_{ij} \in E \end{cases}$$

where

$$\begin{aligned} A_k^{ij} &= 1 - \frac{2\gamma_j\gamma_k}{(\gamma_i+\gamma_k)(\gamma_i+\gamma_j)} \\ B_k^{ij} &= \frac{2\gamma_j\gamma_k(\gamma_i+\gamma_k+2\gamma_j)}{(\gamma_i+\gamma_k)^2(\gamma_i+\gamma_j)^2} \\ C_k^{ij} &= -\frac{2\gamma_i\gamma_j^2}{(\gamma_i+\gamma_k)(\gamma_i+\gamma_j)^2} \end{aligned}$$

which can be verified to be positive definite. As f is strictly convex, it therefore has a unique global minimum, so both the Gradient Descent method and Newton's method can be used to stably find this minimum.

4 Implementation

Discrete Ricci flow can be easily implemented using common mesh libraries based on halfedge data structure, such as OpenMesh, CGAL etc. The implementation is very simple, it takes a graduate student 15 minutes for coding.

4.1 Data structure

First, we define the following traits for vertices, edges, half-edges,

- Corner angle, representing the angles. Suppose a corner connecting vertex v_i, and in face f_{ijk}, then the corner angle is denoted as θ_i^{jk}. Each half-edge represents a corner. Corner angle is a trait associated with half-edges.
- Edge weight Φ, representing the intersection angles of circle packing, denoted as Φ_{ij} for edge e_{ij}.
- Edge length, representing the discrete metric on the mesh, denoted as l_{ij} for edge e_{ij}.
- Vertex radius γ, denoted as γ_i for vertex v_i.
- Vertex Gaussian curvature, denoted as K_i for vertex v_i.
- The parameter for a vertex, denoted $\mathbf{p}_i$ for vertex v_i.

4.2 Ricci flow algorithm

The Ricci Flow algorithm is summarized as the following steps:

1. Assign the weight for each edge and the radii for each vertex by minimizing the energy in Equation 1. If the faces of the initial mesh are close to right angles, or the application does not require conformality, we can instead simply set
$$\Phi(e_{ij}) \equiv 0, \quad \forall e_{ij} \in E, \qquad \gamma_i \equiv 1, \quad \forall v_i \in V.$$
2. Compute the current metric l_{ij}, using the Cosine law
$$l_{ij}^2 = \gamma_i^2 + \gamma_j^2 + 2\gamma_i\gamma_j \cos \Phi_{ij}.$$
davidxgu

3. Compute corner angles for all corners of the mesh,

$$\theta_k^{ij} = \cos^{-1} \frac{l_{jk}^2 + l_{ki}^2 - l_{ij}^2}{2 l_{jk} l_{ki}}$$

4. Compute the discrete Gaussian curvature for each vertex,

$$K_i = 2\pi - \sum_{f_{ijk} \in F} \theta_i^{jk},$$

5. Update the vertex radii using

$$\gamma_i = \gamma_i + \epsilon \times (\bar{K}_i - K_i) \times \gamma_i \tag{8}$$

   where ϵ is a step length. In practice, the step length can be varied dynamically to improve the efficiency.

6. Normalize the vertex radii, such that the product of all γ_i is equal to one.

$$p = \Pi_i^n \gamma_i, \gamma_i = \frac{\gamma_i}{\sqrt[n]{p}},$$

7. Check the deviation between each K_i and $\bar{K}_i$, and find the maximum error

$$error = \max_i |K_i - \bar{K}_i|$$

   If the error is less than a predetermined threshold, stop. Otherwise, goto Step 2.

The procedure converges fast. By fixing a vertex v_i, the error curve

$$y(t) = |K_i(x) - \bar{K}_i(x)|$$

is an exponential curve, as depicted in figure 7.

5 Global parameterizations

Discrete Ricci flow gives engineers the freedoms to model the surfaces by design curvatures first. In many applications, it is straightforward to find the desired curvature first, then find the metric and the embedding.

Global surface parameterization problem can be interpreted as finding a special metric, such that the curvature is zero at every vertex except for several singularities or boundary vertices.

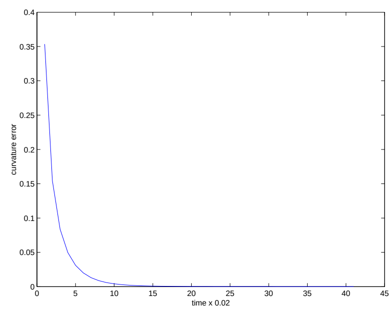


Figure 7: **Curvature error curve.**

5.1 Pipeline

In conventional global conformal surface parametrization a special metric is computed on the mesh such that at $2g-2$ singularities, the curvature is equal to -2π, and at all other vertices, the curvature is zero. The positions of the singularities can not be assigned arbitrarily, as they are determined by the geometry of the surface—see Figure **??**, for example. The singularities are the centers of the octagons. Their positions are determined by the conformal structure of the surface and the cohomologous type of the holomorphic 1-form. For detailed explanation, we refer readers to [**?**].

The Ricci Flow method allows the user to freely assign singularities for global parameterizations, as long as they satisfy the Gauss-Bonnet Theorem 5 and the connectivity condition 6.

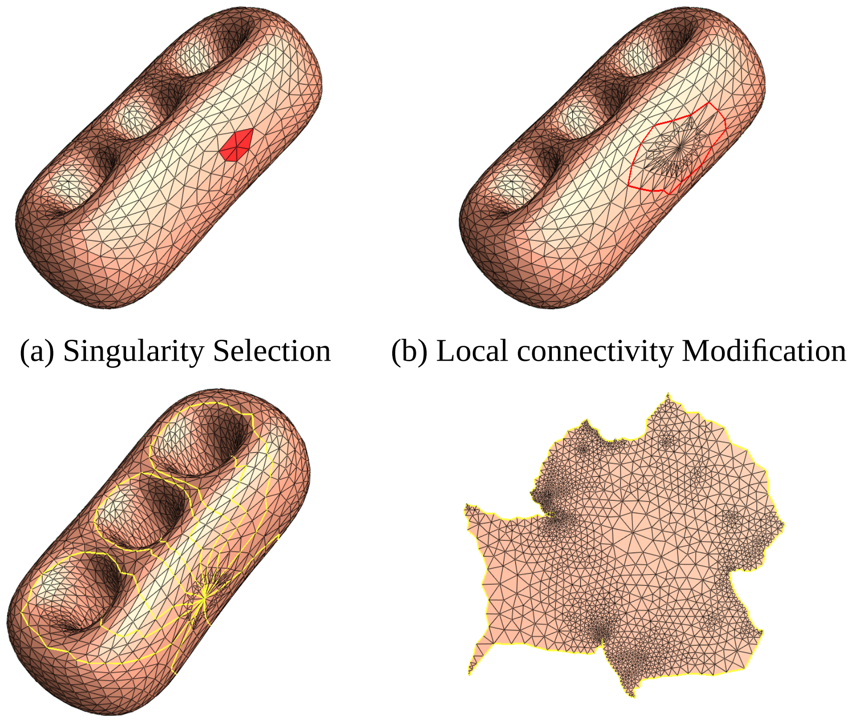
We formulate the *constraint global parameterization problem* as the following,

Definition 20 *Given a mesh M, a set of singular vertices $\{v_1, v_2, \cdots, v_m\}$, the target Gaussian curvature of the singularities are given $\{\bar{K}_1, \bar{K}_2, \cdots, \bar{K}_m\}$, the* constraint global parameterization *is to find a metric, such that it induces the Gaussian curvature K_i,*

1. *For a singular vertex v_i, $K_i = \bar{K}_i$.*
2. *For a nonsingular vertex v_i, $K_i = 0$.*

An affine atlas can be deduced from a flat metric, the atlas can be used for real applications, such as texture mapping, texture synthesis, geometry images, and manifold splines.

The whole **pipeline of constructing the affine atlas** can be summarized as



(a) Singularity Selection (b) Local connectivity Modification

(c) Chartification (cut graph) (d) Planar Embedding

Figure 8: **3 holes model with $10k$ faces, one singularity with curvature -8π.**

1. Singularity selection.
2. Connectivity Modification.
3. Ricci flow.
4. Mesh chartification.
5. Planar Embedding of each chart.

5.2 Singularity selection

The choice of the singularities will affect the quality of the global parameterizations. Because Ricci flow deforms the metric conformally, the final parameterization has no angle distorsion, but the area distorsion energy 9 varies very much.

Figure 19 demonstrates that the choice of the position of the singularity affect the area distortion energies.

The discrete area distortion energy is an analogy of the smooth area distortion energy 9,

$$E(\Phi, \Gamma) = \sum_{f_{ijk} \in M} (\frac{\bar{A}_{ijk}}{A_{ijk}} - 1)^2 A_{ijk}, \tag{9}$$

where $\bar{A}_{ij}$ is the area of face f_{ijk} using the flat metric, A_{ijk} is the area of the face using original metric induced from the Euclidean $\mathbb{R}^3$ metric,

$$s = \frac{1}{2}(l_{ij} + l_{jk} + l_{ki}), A_{ijk} = \sqrt{s(s - l_{ij})(s - l_{jk})(s - l_{ki})}.$$

It is a challenging problem to adjust the singularities to minimize the area distortion,

Problem 21 *Given a mesh embedded in $\mathbb{R}^3$, how to determine the number, the positions and the curvature distributions of singularities in order to minimize the area distortion energy 9.*

If the surface is a closed genus one mesh, then no singular vertex is needed. For high genus mesh, we could select only one singular vertex and concentrate all the curvature at it. If the mesh is open, we can assign target curvatures of zero to all interior vertices and only update the radii at interior vertices. In this way, the curvature will be automatically distribute itself along the boundary. We call this a *free boundary condition*.

5.3 Local Connectivity modifications

In order to determine the desired flat metric, the combinatorial constraints for the curvature in Equation 6 have to be satisfied.

If both the initial curvature configuration and the target curvature configuration satisfy these constraints, any intermediate curvature configuration during the Ricci Flow will also satisfy the constraints. Thus, it is enough to only consider the target curvature.

If some singularities violate the combinatorial constraits, we need to modify the local connectivity in their neighborhood. The mesh connectivity can be easily modified using conventional mesh operations, such as *edge collapse*, *edge swap*, *edge split* [5].

In practice, we use the following heuristic method to modify the connectivity around each singular vertex, such that the following criteria are met:

1. The topological valence of a singular vertex v is no less than $4 - \frac{2\bar{K}(v)}{\pi}$, thus the average corner angle around v is less than $\frac{\pi}{2}$.

2. For all the vertices in the first n-rings of neighbors of the singular vertex, their valences are no less than 6. n is a small integer, which in our implementation is choose to be between 1 and 3.

The Figure **??** illustrates the connectivity of the 2-ring neighborhood of the singular vertex on a genus 3 mesh.

5.4 Mesh chartification

Mesh chartificaion refers to find an open covering of the mesh, such that different charts overlap each other. The basic idea is to find a set of curves G, which go through all the singularities, such that the mesh M can be sliced open along the curves to form a topological disk. Such curves form a *cut graph*, as introduced in the work on *geometry images* [2]. If there is only one singularity, the cut graph G can be constructed using a set of canonical homology bases through the singularity [4].

The mesh is now cut open along the cut graph to form a chart $\bar{M}$, which is called the *central chart*. Vertices on the cut graph with valence $\neq 2$ are called the *nodes*. All the singularities are also nodes. The cut graph is separated into segments, each of which connects two nodes: $G = \cup_k s_k$, where s_k denotes a segment. We cover each segment s_k by a chart

$$U_k = \cup_{v_i \in s_k} N_i, \quad N_i = \cup f_{ijk}.$$

where N_i represents the one ring neighborhood of vertex v_i. We call U_k's *boundary charts*. Then the central chart $\bar{M}$ covers all the faces of the mesh, the boundary charts covers the boundaries of the central chart, the whole mesh is covered by all the charts. The transition functions among the charts are just translation and rotation in $\mathbb{R}^2$.

The algorithm for computing an open covering of M is as follows:

1. Compute a cut graph G using a canonical homology basis or e.g. the method used for constructing geometry images.

2. Slice the mesh along the cut graph to form a topological disk $\bar{M}$.

3. Find the nodes in the cut graph, and separate the cut graph into segments.

4. For segment, compute the union of the one ring neighborhoods of all its vertices.

5. The open covering of M is formed by $\bar{M}$ and U_k, where

$$M \subset \bar{M} \cup_k U_k.$$

5.5 Planar embedding

Each chart can be flattened on the plane face by face using the flat metric.

The algorithm is straight forward: Suppose a circle packing metric l_{ij} based on (M, Φ, Γ) is known, then

1. Compute the dual graph of the mesh, each node represents a face in the mesh.

2. Compute a minimal spanning tree of all the nodes in the dual graph.

3. Suppose the root of the tree is a face f_{ijk}, then embed this face onto the plane $\mathbb{R}^2$,

$$\mathbf{p}_i = (0,0), \mathbf{p}_j = (l_{ij}, 0), \mathbf{p}_k = (l_{ik} \cos\theta_i^{jk}, l_{ik} \sin\theta_i^{jk}).$$

4. Use breadth-first-searching method to traverse the tree, once a node f_{ijk} is accessed, embed f_{ijk} on to the plane. Suppose v_i and v_j has been embedded in $\mathbb{R}^2$ already, $\mathbf{p}_k$ can be computed easily,

$$\mathbf{p}_k = \frac{l_{ik} e^{\sqrt{-1}\theta_i^{jk}}}{l_{ij}} (\mathbf{p}_j - \mathbf{p}_i) + \mathbf{p}_i,$$

   where all the vertex planar parameters $\mathbf{p}_i, \mathbf{p}_j, \mathbf{p}_k$ are treated as complex numbers.

Figure **??** demonstrates a planar layout of a flat metric of a genus one closed mesh. If we shift the planar parameter domain, the left boundary will fit to the right boundary, the top boundary will fit the bottom boundary consistently. Namely, we can *periodically* flatten the mesh with the flat metric.

6 Global Parameterizations Examples

Our implementation is based on a generic half edge mesh library [6], using attributes to represent edge lengths, vertex radii and curvature. In the following, we demonstrate the experimental results.

6.1 Data Structure

```
class Mesh : public PosGraph::Modeling::OMTriMeshBase<MeshTraits>
{
  // custom property

  // vertex radius
  OpenMesh::VPropHandleT<double>  VPropRadius;
  // vertex target curvature
  OpenMesh::VPropHandleT<double>  VPropTargetCurvature;
  // vertex current curvature
  OpenMesh::VPropHandleT<double>  VPropCurvature;
  // edge length
  OpenMesh::EPropHandleT<double>  EPropLength;
  // edge weight
  OpenMesh::EPropHandleT<double>  EPropWeight;
  // corner angle
  OpenMesh::HPropHandleT<double>  HPropAngle;


public:
  typedef PosGraph::Modeling::OMTriMeshBase<MeshTraits>       Base;

public:
  Mesh(void);
  ~Mesh(void);


  virtual void SetEdgeWeight();
  virtual void SetVertexRadius();
  virtual void SetVertexTargetCurvature();
```

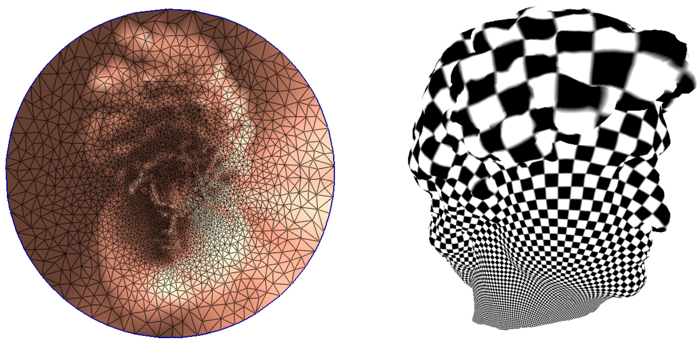
```
  virtual void RicciFlow( double step_length = 1e-4, double curvature_

protected:

  //compute edge lengths
  void  calcEdgeLengths();
  //compute corner angles
  void  calcCornerAngles();
  //compute vertex curvature
  void  calcVertexCurvature();
  //compute vetex radius
  void  calcVertexRadius( double step_length = 1e-4 );

  //compute vertex curvature error
  double calcVertexCurvatureErr();
};
```

6.2 Methods

```
void Mesh::calcEdgeLengths() {
    EdgeIter e_it;

    for ( e_it=edges_begin(); e_it!=edges_end(); ++e_it)
    {
      double a = property(VPropRadius, to_vertex_handle( halfedge_hand
      double b = property(VPropRadius, to_vertex_handle( halfedge_hand
      double C = property(EPropWeight, e_it );

      property( EPropLength,e_it) = sqrt( a * a + b * b + 2 * a * b *
  }
} void Mesh::calcCornerAngles() {
    HalfedgeIter h_it;
    for ( h_it = halfedges_begin(); h_it != halfedges_end(); ++h_it)
    {
        HalfedgeHandle phe = prev_halfedge_handle( h_it );
        HalfedgeHandle nhe = next_halfedge_handle( h_it );
```

```
        double a = property( EPropLength, edge_handle( h_it ) );
        double b = property( EPropLength, edge_handle( phe ) );
        double c = property( EPropLength, edge_handle( phe ) );
        double angle = acos(( a * a + b *b - c * c )/(2 * a * b ));
        property( HPropAngle, h_it ) = angle;
    }
}

void Mesh::calcVertexCurvature() {
    VertexIter v_it;
    for ( v_it = vertices_begin(); v_it != vertices_end(); ++v_it)
    {
        VertexIHalfedgeIter vh_it;

        double curvature = 2 * 3.1415926535;

        for( vh_it = vih_iter( v_it ); vh_it ; ++ vh_it )
        {
            curvature -= property( HPropAngle, vh_it );
        }

        property( VPropCurvature, v_it ) = curvature;
    }
}

void Mesh::calcVertexRadius( double step_length ) {
    VertexIter v_it;
    for ( v_it = vertices_begin(); v_it != vertices_end(); ++v_it)
    {
        double K  = property( VPropCurvature, v_it );
        double TK = property( VPropTargetCurvature, v_it );
        double r  = property( VPropRadius, v_it );

        double dr = 2.0 * ( TK - K ) * r * step_length;
        r += dr;

        property( VPropRadius, v_it ) = r;
```

```
    }
}

void Mesh::RicciFlow( double step_length, double
curvature_error_threshold ) {
    double curvature_err;

    while( true )
    {
        calcEdgeLengths();
        calcCornerAngles();
        calcVertexCurvature();
        calcVertexRadius();

        curvature_err = calcVertexCurvatureErr();
        if(  curvature_err < curvature_error_threshold )
            break;
    }
}

double Mesh::calcVertexCurvatureErr() {
    VertexIter v_it;

    double max_error = -1;
    for ( v_it = vertices_begin(); v_it != vertices_end(); ++v_it)
    {
        double current_curvature  = property( VPropCurvature, v_it );
        double target_curvature   = property( VPropTargetCurvature, v_
        double error = fabs( target_curvature - current_curvature );
        max_error = ( error > max_error )? error : max_error;
    }
    return max_error;
}
```



(a) (b)

Figure 9: **David Head model with $15k$ faces, with boundary singularities, each with curvature $\frac{2\pi}{m}$, where m is the number of boundary vertices.** The area distortion is 0.960747.

7 Acknowledgement

The first author would like to thank Ralph Martin for his valuable advices and mathematician Wei Luo for the inspiring discussion. All the examples are computed by PhD candidate Miao Jin and David X. Gu.

References

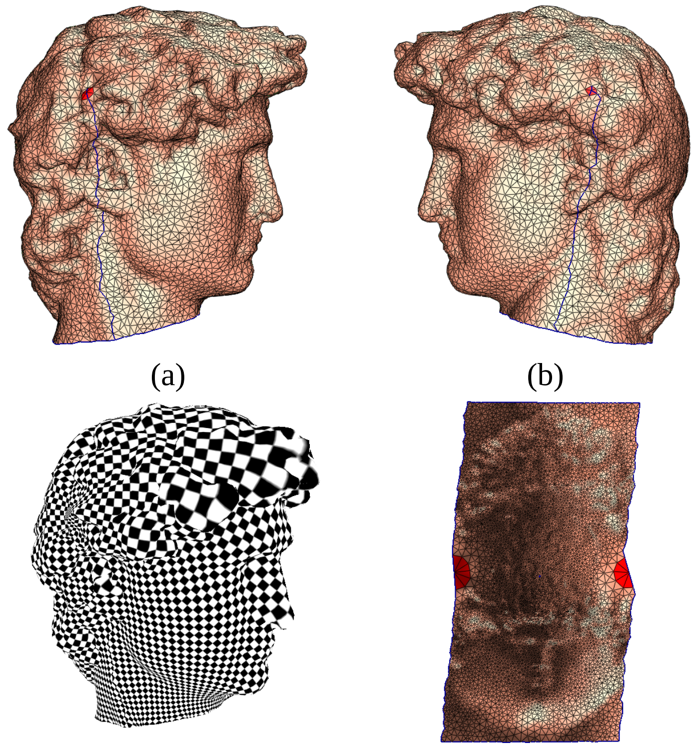
[1] Bennett Chow and Feng Luo. Combinatorial Ricci flows on surfaces. *J. Differential Geom.*, 63(1):97–129, 2003.

[2] Xianfeng Gu, Steven J. Gortler, and Hugues Hoppe. Geometry images. In *SIGGRAPH*, pages 355–361, 2002.

[3] Richard S. Hamilton. The Ricci flow on surfaces. In *Mathematics and general relativity (Santa Cruz, CA, 1986)*, volume 71 of *Contemp. Math.*, pages 237–262. Amer. Math. Soc., Providence, RI, 1988.

[4] Michael Henle. *A Combinatorial Intorduction to Topology*. W.H. Freeman and Co, 1979.

[5] Hugues Hoppe. Progressive meshes. In *SIGGRAPH*, pages 99–108, 1996.



(a) (b) (c) (d)

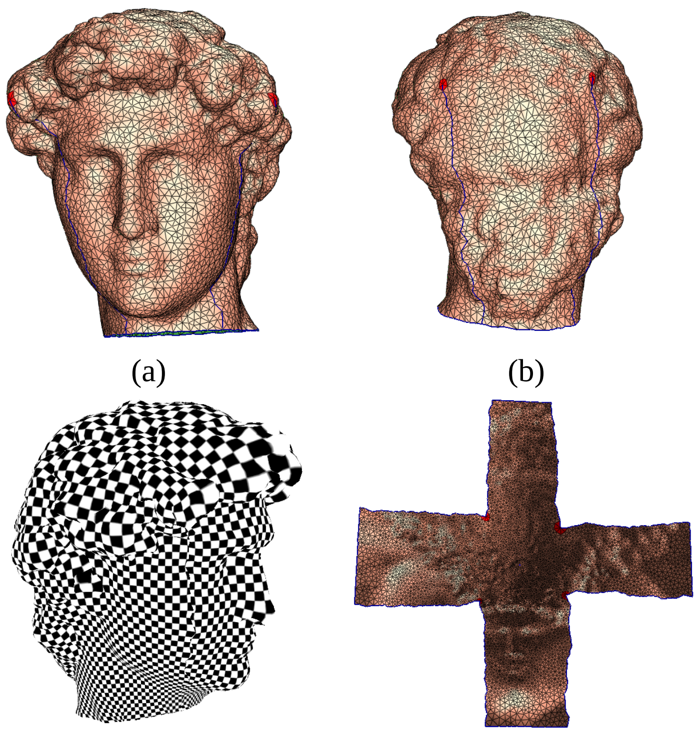
Figure 10: **David Head model with $15k$ faces, with 2 singularities, each with curvature π.** The center of the red regions are singularities. The blue curves are the cut graph. The area distortion is 0.351826.

[6] Hugues Hoppe. Efficient implementation of progressive meshes. *Computers & Graphics*, 22(1):27–36, 1998.

[7] Grisha Perelman. The entropy formula for the ricci flow and its geometric application. Nov 2002.

[8] Burt Rodin and Dennis Sullivan. The convergence of circle packings to the Riemann mapping. *J. Differential Geom.*, 26(2):349–360, 1987.

[9] W. Thurston. Hyperbolic geometry and 3-manifolds. In *Low-dimensional topology (Bangor, 1979)*, volume 48 of *London Math. Soc. Lecture Note Ser.*, pages 9–25. Cambridge Univ. Press, Cambridge, 1982.



(a) (b) (c) (d)

Figure 11: **David Head model with $15k$ faces, with 4 singularities, each with curvature $\frac{\pi}{2}$.** The center of the red regions are singularities. The blue curves are the cut graph. The area distortion is 0.330786.

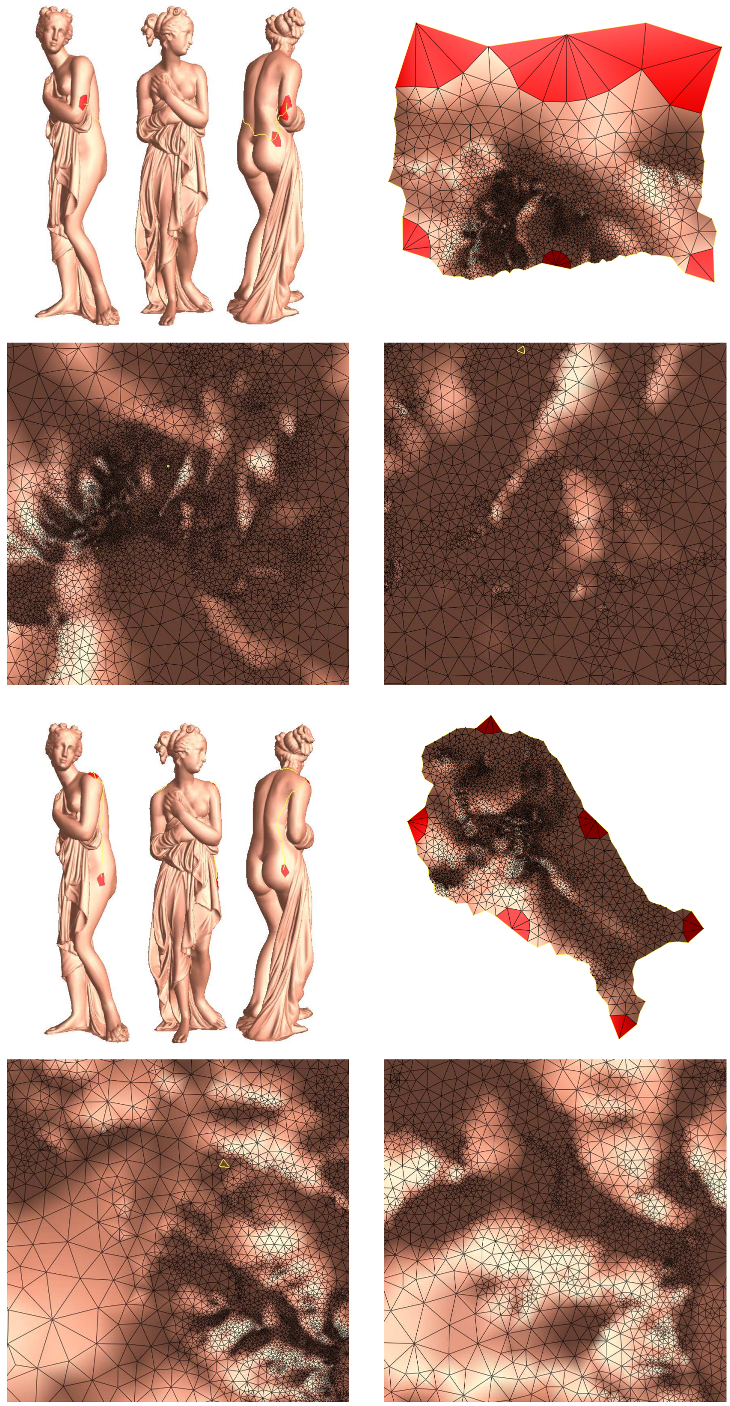


Figure 12: **Ipheginia model with** $30k$ **faces,** 4 **singularities, each with** π **curvature, area distortions are** 1.075546 **and** 0.740751 respectively.

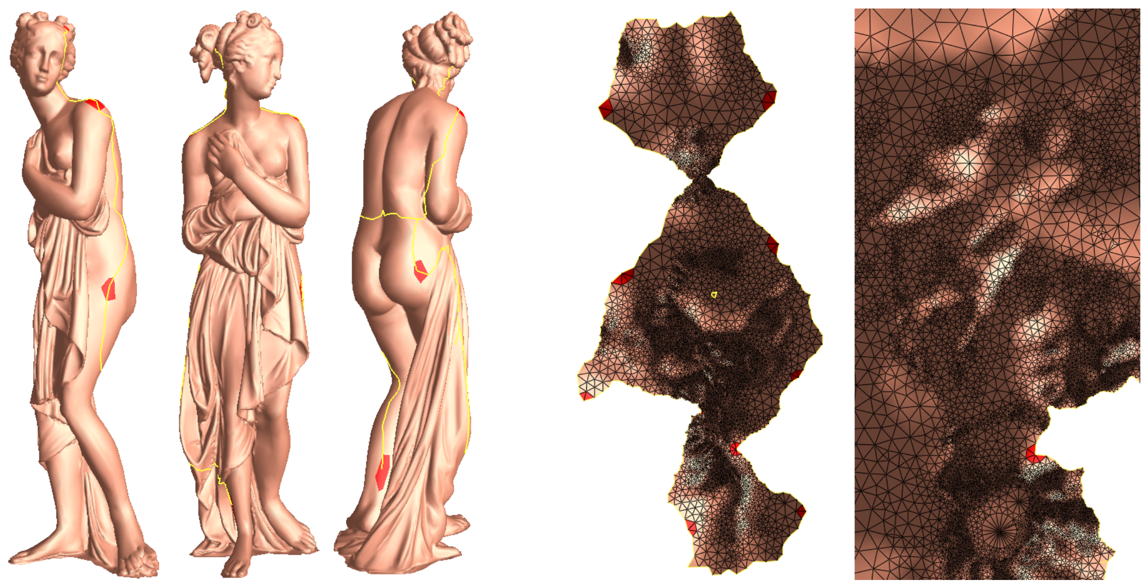
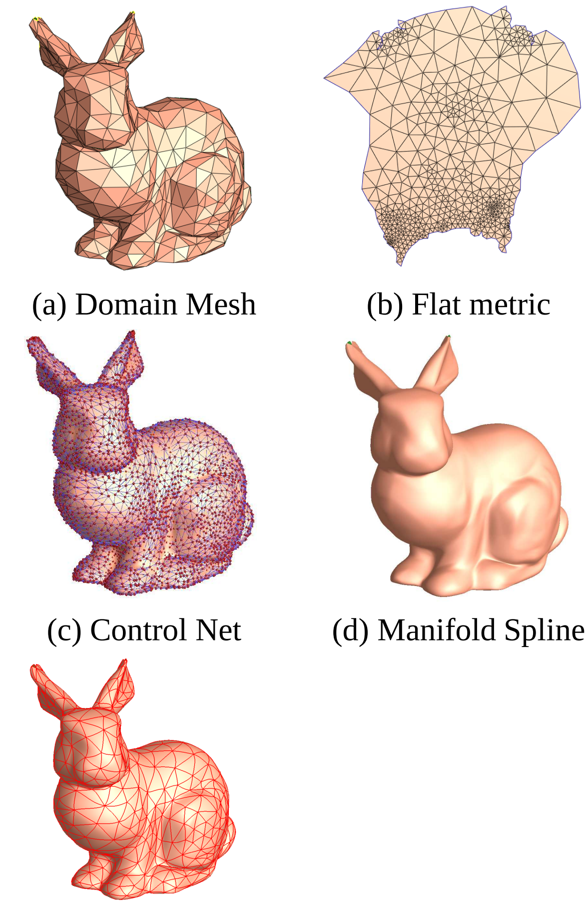
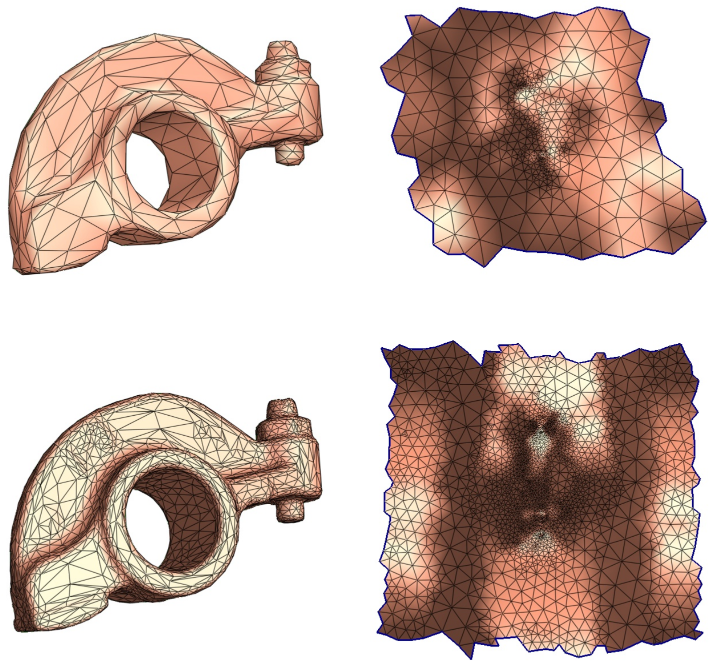


Figure 13: **Ipheginia model with $30k$ faces, 8 singularities, each with $\frac{\pi}{2}$ curvature, area distortions is 0.571903.**



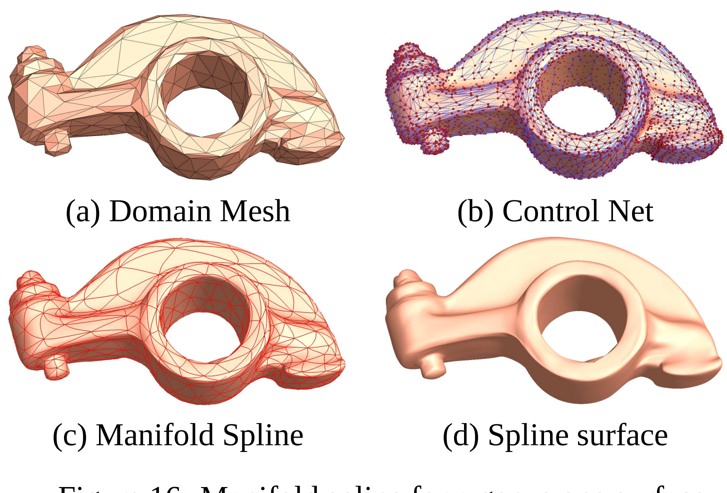
(a) Domain Mesh
(b) Flat metric
(c) Control Net
(d) Manifold Spline
(e) Spline surface with domain triangles

Figure 14: **Manifold spline for an open surface.** The bunny mesh has three boundaries, two are at the ear tips, one is at the bottle. The affine atlas is computed using Ricci Flow under free boundary condition.



(a) Rocker Arm Model (b) planar layout

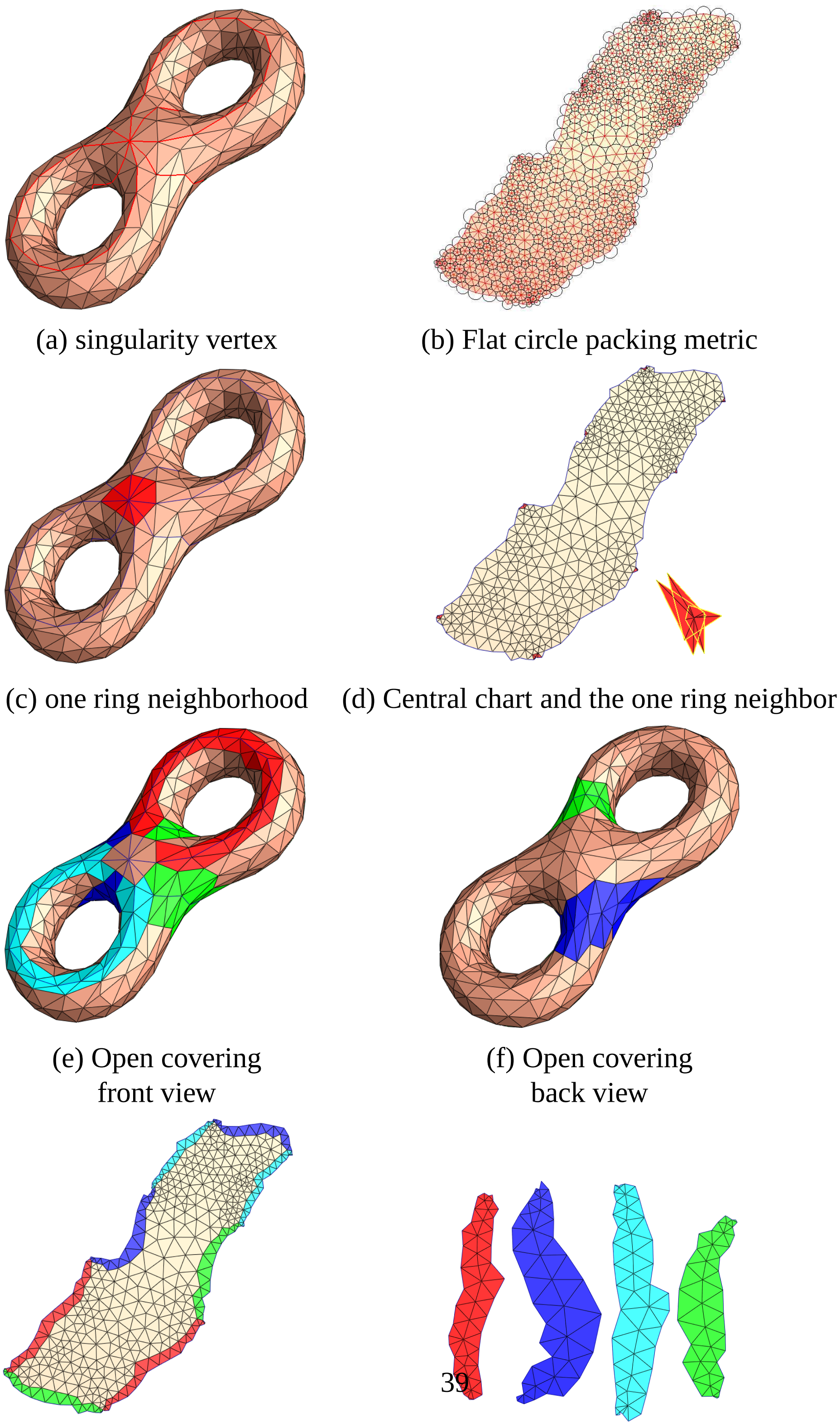
Figure 15: **Rocker arm model with $1k$ faces and $5k$ faces, no singularities. The area distortions are 0.554323,0.576723 respectively.** Planar layout of a flat metric on a genus one closed mesh.



(a) Domain Mesh (b) Control Net

(c) Manifold Spline (d) Spline surface

Figure 16: Manifold spline for a genus one surface, rocker arm model.



(a) singularity vertex

(b) Flat circle packing metric

(c) one ring neighborhood

(d) Central chart and the one ring neighbor

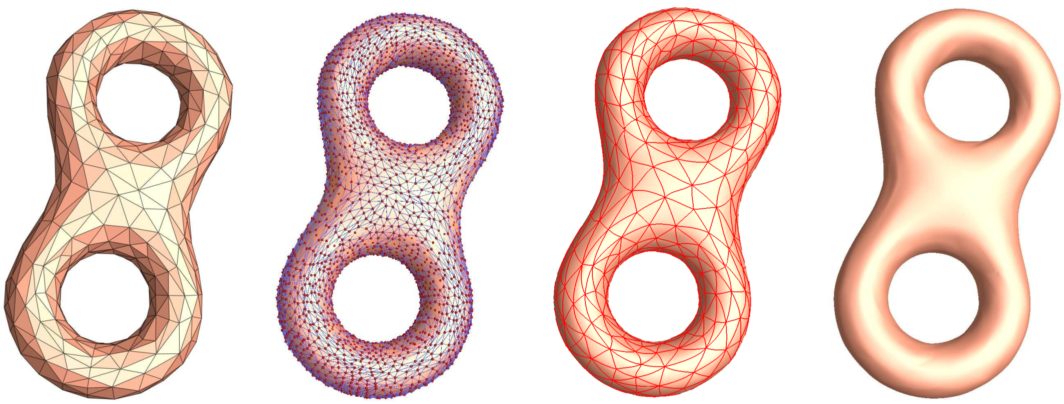
(e) Open covering front view

(f) Open covering back view

(g) Central chart

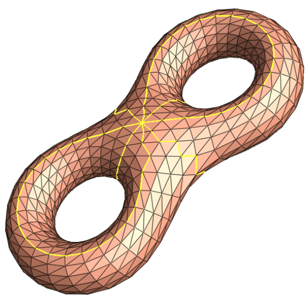
(h) boundary charts

Figure 17: **Affine atlas using Ricci Flow.**

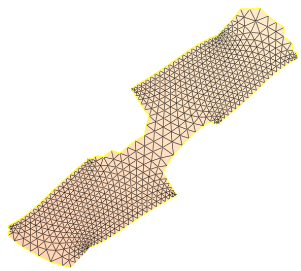


(a) Domain Mesh (b) Control Net (c) Manifold Spline (d) Spline surface

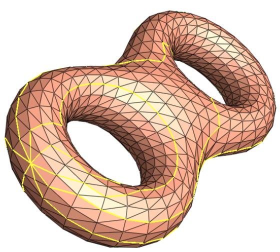
Figure 18: Manifold spline for a genus two surface.



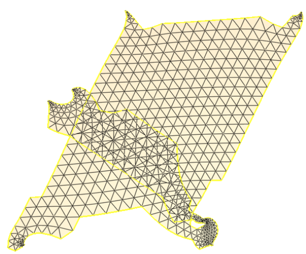
(a) Center singular Vertex



(b) flat metric of (a)

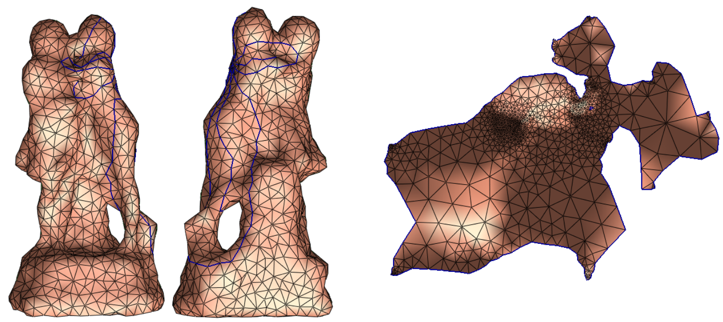


(c)Singularity on side



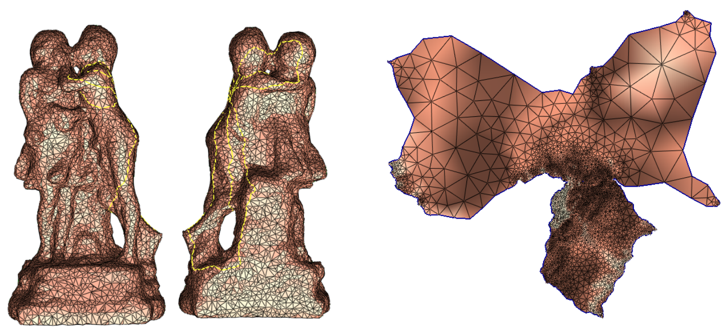
(d) flat metric of(c)

Figure 19: **The position of a singular vertex will affect the flat metric drastically.** (a) and (b) shows the flat metric when the singular vertex is selected in the center region, the metric is very uniform. (c) and (d) show the flat metric when the singular vertex is selected on the side of the mesh, the metric is highly nonuniform. Also (d) shows that the flat metric induces an immersion (locally embedding) but not an global embedding.

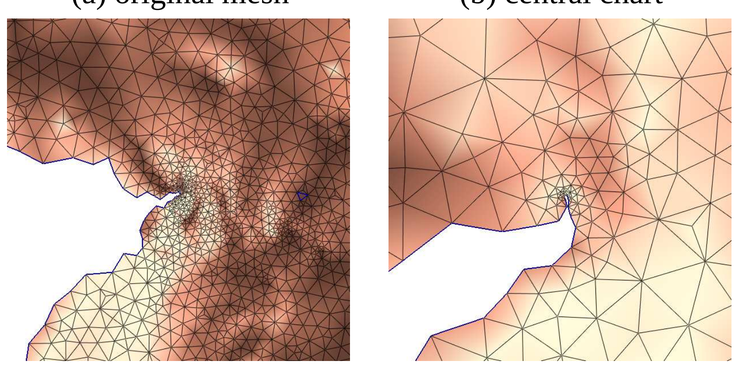


(a) Sculpture Model (b) planar layout

Figure 20: **Sculpture model with $2k$ faces, one singularity with curvature -8π, the area distortion is 0.658119.**

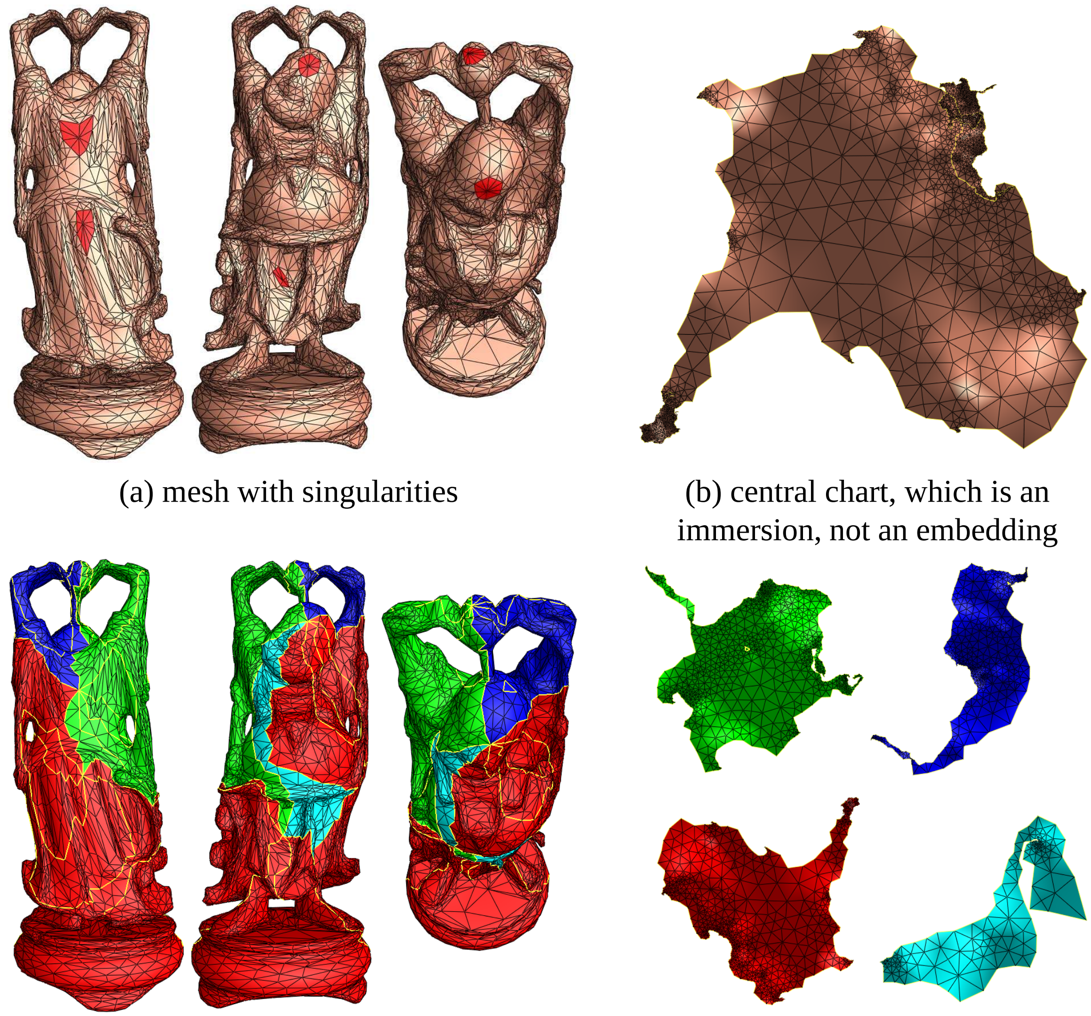


(a) original mesh (b) central chart



(c) zoomed in of (b) (d) further zoomed in of (b)

Figure 21: **Sculpture model with $10k$ faces, one singularity with curvature -8π, area distortion is 1.000959.**



(a) mesh with singularities

(b) central chart, which is an immersion, not an embedding

(c) segmentation and cut graph

(d) charts

Figure 22: **Genus six buddaha model with $10k$ faces, five singularity with curvature -4π, area distortion is 1.467170.**