HIGH QUALITY QUADRILATERAL MESH GENERATION BASED ON MEROMORPHIC QUARTIC DIFFERENTIALS

Xiaopeng Zheng[1] Xin Shi[1] Chao Wang[2] Yiming Zhu[1]
Zhongxuan Luo[14] Na Lei[13*]

[1] *Dalian University of Technology, Dalian, Liaoning, China. nalei@dlut.edu.cn*
[2] *State Key Laboratory of Intelligent Manufacturing System Technology, Beijing, China*
[3] *Beijing Advanced Innovation Center for Imaging Theory and Technology, Beijing, China*
[4] *Institute of Artificial Intelligence, Guilin University of Electronic Technology, Guilin, China*

ABSTRACT

Quadrilateral meshes generated for the industrial applications have to meet some special quality demands and it remains a tricky problem. We propose a novel method for high quality quad mesh generation based on the equivalence relation between quadrilateral meshes and meromorphic quartic differentials[1]. As long as more than one meromorphic quartic differential were obtained, the linear space extended by them is also available for us according to the meromorphic quartic differentials space theory. We can search the space for better quadrilateral mesh and the high quality mesh construction problem becomes a optimization problem. This paper gives the algorithm and carries out experiments under two different mesh quality constrains including mesh cell size constrain and feature lines constrain. The experimental results demonstrate the efficiency and efficacy of the algorithm.

Keywords: Quad mesh generation, Meromorphic quartic differential, Linear space, Mesh quality constrain

1. INTRODUCTION

Quadrilateral mesh generation is an essential issue in the area of computational mechanics, geometric modeling, computer aided design, animation and digital geometry processing. There have been many excellent algorithms for quadrilateral mesh generation. However, to totally meet the various mesh quality demands in practice, there is still a long way to go. Some researchers have a high demand on the control of the mesh cell sizes because they need to balance the efficiency and efficacy. Some researchers pay much attention to the number and distribution of the singularities. In numerical simulation area, either the sharp feature preserving or the tensor structure, or other properties of the quad mesh will significantly affect the accuracy, efficiency, and stability of the experiments. High quality mesh generation remains a challenge in practice.

In recent years, a novel theory and technology for quadrilateral mesh generation has emerged. In paper [1] and [2], the authors clarified the relationship among quadrilateral mesh, Riemann metric and Meromorphic quartic differential. Each quadrilateral mesh induces a Riemann metric of the surface by treating all the quad mesh edge length as unit and a meromorphic quartic differential, where the configuration of singular vertices corresponds to the configurations of the poles and zeros (divisor) of the meromorphic differential. Inversely, if a meromorphic quartic differential of the surface is with finite trajectories, then it also induces a quad-mesh, the poles and zeros of the meromorphic differential correspond to the singular vertices of the quad-mesh.

The intrinsical insight that meromorphic differentials and quad meshes are equivalent provided us a new direction to see the quad mesh generation and improvement. Since the meromorphic differentials of one surface form a linear space, the quad meshes also form a linear space. As long as we have more than one quad mesh, other quad mesh can be obtained by linear combination. This offers the flexibility for mesh improvement.

Paper [3] also introduced a algorithm to generate quadrilateral mesh: A set of singularities should be specified in advance and check by the Abel-Jacobi condition, then compute the Riemann metric with cone singularities with Ricci flow and the metric can induce quad mesh. In fact the quality of the mesh generated by Ricci flow based methods is severely affected by the quality of the singularity configuration. The quality of the resulting mesh will not be good if we cannot offer a proper distribution of the singularities.

However there is a great chance that a high quality mesh exists in the space extended by several low quality meshes. When we have the basis of the biggest linear space including all the possible quad mesh, any kind of quad mesh we need can be constructed by linear combination of the basis. Hence we take full advantage of the existing quad meshes or meromorphic differentials by searching the linear space extended by them.

In this paper, we specify multiple sets of singularities sytisfying the Abel-Jacobi condition for the same triangular mesh and calculate the corresponding meromorphic differentials using Ricci flow. Then we have a linear space whose basis are these meromorphic differentials. By adjusting the coefficients of the linear combination, we can get different meromorophic differential and quad mesh. Mesh improvement under special constrains is all about optimizing the coefficients.

We also provided the algorithm to compute meromorphic differential from a given quad mesh. This eliminates the dependence of our method on Ricci flow. As long as there are more than one quad mesh which have the the same conformal structure with the original surface, no matter how they were constructed, the improve method in this paper can work through.

The structure of this article is as follows. We review some methods of quadrilateral meshes generation in section 2. In section 4, we show how to complete the Riemann metric calculated by the RicciFlow method [4] and describe the details of the linear combination of meromorphic differentials. We present how to add constraints in the process of linear combination to improve the quality of the result mesh in section 5. In section 5.1, we add constrains on the size of the mesh unit. In section 5.2, we require the edges of the quadrilateral mesh to be aligned with the feature lines of the surface. We summarize this paper and propose some possible ways to further optimize our work in the future in section 6.

2. RELATED WORKS

In this section, we briefly review the quad mesh generation method.

Triangle to quad method. This type of method directly converts the triangular meshes to quad meshes. The general idea is to pair triangles to quads. There are many different ways to implement this process e.g. an advancing front algorithm [5] a global optimal solution [6] and a greedy algorithm [7]. The quality of the mesh generated by these methods depends on the quality of the initial triangular mesh. And since the global information of the surface is not taken into account, these methods cannot achieve smoothness and regularity globally.

Cross-field based method. The directional field can be used to guide the growth of mesh lines. This type of method first calculates a directional field for the surface, usually a four-way rotationally symmetric orientation field [8] [9] or cross field. The cross field is usually calculated by optimizing a nonlinear energy function [10] [11]. Finally, quadrilateral meshes are generated by using streamline tracing techniques [12] or parametric methods. A problem with these methods is that the optimization of the energy function tends to be trapped in the local minimum, which results in too many singularities and poor mesh quality.

Parameterization based method. Parameterization based method computes the quadrilateral tesselation in the parameter domain or finds the skeleton from intrinsic geometric functions or differentials. The spectral surface quadrangulation method [13] [14] produces the skeleton structure from the Morse-Smale complex of an eigenfunction of the Laplacian operator on the input mesh. [15] [16] introduces seamless parameterization, which means the transition functions across the cuts of the parameterization are not arbitrary but of a very restricted class: rigid transformations with a rotation angle of some multiple of $90\circ$. Quantized Global Parametrization [17] studies quantized parameterization on this basis. Quantized parameterization means that transitions in seamless parameterization are all integral.

Metric based methods. [2] proposed a novel mesh generation algorithm based on the Riemann metric. It gave the necessary and sufficient conditions for whether a Riemann metric can induce quadrilateral mesh. That paper uses the Ricciflow method proposed in [4] to calculate the appropriate Riemann metric. However, this method has some shortcomings. One is that the singularities need to be specified manually, not automatically. Besides, it does not provide how to make the specified singularity configuration meet the holonomy condition which is essential for inducing a valid quad mesh. [18] uses the holomorphic quadratic differential to generate a quadrilateral mesh, which has a tiny number of singularities, but the holomorphic quadratic differential is inherently unable to deal with models which contain singularities of odd valances. [1] further studies the methods based on the Riemann metric and holomorphic differential. It solves some shortcomings in the methods mentioned above. On the one hand, it uses meromorphic quartic differential instead of holomorphic quadratic differential to generate meshes, which theoretically can handle meshes containing singularities of arbitrary valence. On the other hand, it gives the necessary and sufficient condition for whether a configuration of singularities satisfies the holonomy condition, that is, the Abel-Jacobi condition, which makes the metric-based method more mature.

3. THEORETIC BACKGROUND

In this section, we briefly review the basic concepts and theorems in Riemann surface theory and meromorphic quartic differential.

3.1 Basic Concepts of Riemann Surface

Definition 3.1 ***[Topological Manifold]*** *Suppose* Σ *is a topological space,* $\{U_\alpha\}$ *is a family of open sets covering the space,* $\Sigma \subset \bigcup_\alpha U_\alpha$*. For each open set* U_α*, there exists a homeomorphism* $\varphi_\alpha : U_\alpha \to \mathbb{R}^n$*, the pair* $(U_\alpha, \varphi_\alpha)$ *is called a local chart. The collection of local charts form the atlas of* M*,* $\mathcal{A} = \{(U_\alpha, \varphi_\alpha)\}$*. For any pair of open sets,* U_α *and* U_β*, if* $U_\alpha \cap U_\beta \neq \emptyset$*, the transition map is given by* $\varphi_{\alpha\beta} : \varphi_\alpha(U_\alpha \cap U_\beta) \to \varphi_\beta(U_\alpha \cap U_\beta)$*,* $\varphi_{\alpha\beta} = \varphi_\beta \circ \varphi_\alpha^{-1}$*. Then* Σ *is called a closed* n*-dimesnional manifold.*

Two dimensional manifolds are called surfaces.

Definition 3.2 ***[Holomorphic Function]*** *Suppose* $f : \mathbb{C} \to \mathbb{C}$ *is a complex function,* $(x, y) \mapsto (u(x, y), v(x, y))$*, if the function satisfies the Cauchy-Riemann equation*

$$\frac{\partial u}{\partial x} = \frac{\partial v}{\partial y}, \quad \frac{\partial u}{\partial y} = -\frac{\partial v}{\partial x},$$

then f *is called a holomorphic function. If* f *is invertible, furthermore* f^{-1} *is also holomorphic, then* f *is called biholomorphic.*

Definition 3.3 ***[Meromorphic Function]*** *Suppose* $f : \mathbb{C} \to \mathbb{C}$ *is a complex function,* $f(z) = p(z)/q(z)$*, where* $p(z)$ *and* $q(z)$ *are holomorphic functions, then* $f(z)$ *is called a meromorphic function.*

Definition 3.4 ***[Conformal Atlas]*** *Suppose* S *is a two dimensional topological manifold, equipped with an atlas* $\mathcal{A} = \{(U_\alpha, \varphi_\alpha)\}$*, every local chart are complex coordinates* $\varphi_\alpha : U_\alpha \to \mathbb{C}$*, denoted as* z_α*, and every transition map is biholomorphic,*

$$\varphi_{\alpha\beta} : \varphi_\alpha(U_\alpha \cap U_\beta) \to \varphi_\beta(U_\alpha \cap U_\beta), \quad z_\alpha \mapsto z_\beta,$$

then the atlas is called a conformal atlas.

Definition 3.5 ***[Riemann Surface]*** *A topological surface with a conformal atlas is called a Riemann surface.*

3.2 Meromorphic Differentials

The concepts of holomorphic and meromorphic functions can be generalized to Riemann surfaces.

Definition 3.6 ***[Meromorphic Function on Riemann Surface]*** *Suppose a Riemann surface* $(S, \{(U_\alpha, \varphi_\alpha)\})$ *is given. A complex function is defined on the surface* $f : S \to \mathbb{C} \cup \{\infty\}$*. If on each local chart* $(U_\alpha, \varphi_\alpha)$*, the local representation of the functions* $f \circ \varphi_\alpha^{-1} : \mathbb{C} \to \mathbb{C} \cup \{\infty\}$ *is meromorphic, then* f *is called a meromorphic function defined on* S*.*

A memromorphic function can be treated as a holomorphic map from the Riemann surface to the unit sphere.

Definition 3.7 ***[Meromorphic Differential]*** *Given a Riemann surface* $(S, \{z_\alpha\})$*,* ω *is a meromorphic differential of order* n*, if it has local representation,*

$$\omega = f_\alpha(z_\alpha)(dz_\alpha)^n,$$

where $f_\alpha(z_\alpha)$ *is a meromorphic function,* n *is an integer; if* $f_\alpha(z_\alpha)$ *is a holomorphic function, then* ω *is called a holomorphic differential of order* n*.*

A meromorphic differential of order 4 is called a *meromorphic quartic differential.*

Definition 3.8 ***[Zeros and Poles of Meromorphic Differentials]*** *Given a Riemann surface* $(S, \{z_\alpha\})$*,* ω *is a meromorphic differential with local representation,*

$$\omega = f_\alpha(z_\alpha)(dz_\alpha)^n.$$

If z_α *is a pole (or a zero) of* f_α *with order* k*, then* z_α *is called a pole (or a zero) of the meromorphic differential* ω *of order* k*.*

We use $Sing_\omega$ to denote the singularity set of ω. Locally near a regular point p, the differential $\omega = f(z)dz^n$ can be represented as the n-th power of a 1-form $h(z)dz$ where $h^n(z) = f(z)$ and thus $h(z) = \sqrt[n]{f(z)}$ coincides with one of n possible branches of the n-th root. We call this n-valued 1-form the *n-th roots* of ω, which is a globally well-defined multi-valued meromorphic 1-form on S.

Definition 3.9 ***[Trajectories of meromorphic differentials]*** *Given a meromorphic n-differential ω on S we define n distinct line fields on $S \setminus Sing_\omega$ as follows. At each non-singular point z there are exactly n distinguished directions dz at which $\omega = f(z)(dz)^n$ attains real values. Integral curves of these line fields are called trajectories of ω.*

Suppose ω is a meromorphic quadratic differential, dz is a *horizontal (vertical) direction* if $f(z)(dz)^2 > 0$ ($f(z)(dz)^2 < 0$). Integral curves of horizontal direction are called *horizontal (vertical) trajectories.*

3.3 Abel-Jacobi Condition

Given a Riemann surface $(S, \{z_\alpha\})$, ω is a well-defined meromorphic differential on $(S, \{z_\alpha\})$, $Sing_\omega$ is the singularity set of ω, then $Sing_\omega$ satisfies the Abel-Jacobi condition. There exists no well-defined meromorphic differential ω on Riemann surface whose singularity set is $Sing_\omega$ when $Sing_\omega$ does not satisfy the Abel-Jacobi condition. For more detail of Abel-Jacobi condition, refer to paper[1].

3.4 Quad-Meshes and Meromorphic Quartic Forms

Here we review the intrinsic relation between a quad-mesh and a meromorphic quartic differential.

Definition 3.10 ***[Quadrilateral Mesh]*** *Suppose Σ is a topological surface, $\mathcal{Q}$ is a cell partition of Σ, if all cells of $\mathcal{Q}$ are topological quadrilaterals, then we say $(\Sigma, \mathcal{Q})$ is a quadrilateral mesh.*

On a quad-mesh, the *topological valence* of a vertex is the number of faces adjacent to the vertex.

Definition 3.11 (Singularity) *Suppose $(S, \mathcal{Q})$ is a quadrilateral mesh. If the topological valence of an interior vertex is 4, then we call it a* regular vertex*, otherwise a* singularity*; if the topological valence of a boundary vertex is 2, then we call it a* regular boundary vertex*, otherwise a* boundary singularity. *The index of a singularity is defined as follows:*

$$Ind(v_i) = \begin{cases} 4 - val(v_i) & v_i \notin \partial(S, \mathcal{Q}) \\ 2 - val(v_i) & v_i \in \partial(S, \mathcal{Q}) \end{cases}$$

where $Ind(v_i)$ and $val(v_i)$ are the index and the topological valence of v_i.

Theorem 3.12 ***[Quad-Mesh to Meromrophic Quartic Differential]*** *Suppose $(\Sigma, \mathcal{Q})$ is a closed quadrilateral mesh, then*

1. *the quad-mesh $\mathcal{Q}$ induces a conformal atlas $\mathcal{A}$, such that $(\Sigma, \mathcal{A})$ form a Riemann surface, denoted as S_Q.*
2. *the quad-mesh $\mathcal{Q}$ induces a quartic differential ω_Q on S_Q. The valence-k singular vertices correspond to poles or zeros of order $k - 4$. Furthermore, the trajectories of ω_Q are finite.*

Theorem 3.13 ***[Quartic Differential to Quad-Mesh]*** *Suppose $(\Sigma, \mathcal{A})$ is a Riemann surface, ω is a meromorphic quartic differential with finite trajectories, then ω induces a quadrilateral mesh $\mathcal{Q}$, such that the poles or zeros with order k of ω corresponds to the singular vertices of $\mathcal{Q}$ with valence $k + 4$.*

Definition 3.14 ***[Linear combination of quartic Differential]*** *Given a Riemann surface $(S, \{z_\alpha\})$, $\omega_1, \ldots, \omega_i, \ldots, \omega_n$ are n different meromorphic quartic differentials with finite trajectories, they have the local representation,*

$$\omega_i = f_{i,\alpha}(z_\alpha)(dz_\alpha)^4,$$

the linear combination of these meromorphic quartic differentials is defined as

$$\sum_{i=1}^{n} \omega_i = (\sum_{i=1}^{n} f_{i,\alpha}(z_\alpha))(dz_\alpha)^4$$

Theorem 3.15 ***[linear space of Quartic Differential]*** *Given $n > 1$ meromorphic quartic differentials on a Riemann surface, their linear combination is also a meromorphic quartic differential, all the combination forms a linear space. All the possible meromorphic quartic differentials on the Riemann surface forms the biggest linear space.*

4. COMPUTATIONAL ALGORITHMS

The algorithm pipeline can be summarized as follows:

- Compute flat Riemannian metric with cone singularities which satisy the Abel-Jacobi condition;
- Isometric immerses the triangle surface face-wise onto the complex plane and pull back the canonical holomorphic differential to the surface to obtain the meromorphic differential and meromorphic quartic differential;

- Repeat the previous steps to get $n > 1$ meromorphic quartic differentials;
- Represent the linear space by the linear combination of the n meromorphic quartic differentials;
- Optimize the coefficients of the linear combination to improve the quality of the quad mesh.

The optimization step will be illustrated together with concrete quality constrain example in the next section.

In the following, we explain other steps of the algorithm in details. The input of the algorithm is a triangle mesh M.

4.1 Riemannian metric computation

Discrete form of Riemann metric on triangle mesh can be defined as the mesh edge length. Riemann metric computation is all about assigning the proper edge length.

The first way to obtain the Riemann metric is the RicciFlow algorithm [4]. We can set up several sets of different singularities with degrees for the same triangular mesh, and compute the corresponding Riemann metric using Ricci Flow.

It should be noted that the input singularity set here must satisfy the Abel-Jacobi condition, otherwise we can not get a valid well-defined meromorphic quartic differential.

The second way is to get the Riemann metric from existing quad meshes. The existing quad meshes used here must be conformally equivalent to the original triangle surface. Assume that we have obtained some quadrilateral meshes generated by other methods and want to obtain a higher quality mesh by linear combination method. Then we can directly use these quadrilateral meshes to calculate several sets of Riemann metrics for the triangular mesh.

We take each quadrilateral mesh as background mesh, and map the triangle mesh onto it. Figure 1 shows that a triangle T_{ABC} of the triangle mesh is mapped onto a quadrilateral background mesh. In order to calculate the metric on edge E_{AB}, we assume that these local quads are coplanar and the length of each edge of the quad mesh is 1. Then we can easily calculate the length of edge E_{AB} and use it as the Riemann metric of that edge. Then we can calculate the corresponding differential.

For a given triangular mesh, either we have $n > 1$ meromorphic quartic differentials or quadrilateral meshes, we can use the linear combination method to improve the quality of the meshes.

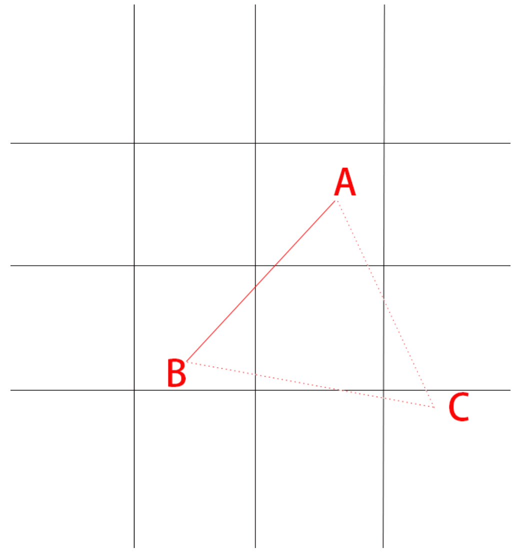


Figure 1: A triangle mapped to a background quad mesh

4.2 Isometric Immersion and Meromorphic Quartic Differential

Slice triangle mesh M to obtain a topological disk $\tilde{M}$ and make sure all the singlularities are on the disk boundary.

Then we flatten $\tilde{M}$ face by face using the metric obtained. This produces an immersion of $\varphi : \tilde{M} \to \mathbb{C}$. On the complex plane, there is a canonical differential d_z^4, the pull back $\varphi^* d_z^4$ is a meromorphic quartic differential defined on M.

The discrete form of meromorphic differential on triangle mesh is the complex numbers defined on the mesh edges.

Here we use duv to denote the meromorphic first differential defined on edges; duv^4 for the meromorphic quartic differential.

In this step, arbitrary slicing method works under our framework. Different slicing and isometric immersion way may induce different meromorphic quartic differentials, but they are all equivalent to each other with respect to a global rotation.

4.3 The process of linear combination

Assume we have n different meromorphic quartic differentials, we use the formula

$$duv^{new} = \sqrt[4]{\sum_{i=1}^{n} (duv_i{}^4 * \alpha_i)}$$

to linearly combine the differentials, where n means the number of differential bases, $\alpha_i \in F$ is the linear combination coefficient corresponding to the i-th meromorphic quartic differential bases duv_i^4.

The fourth root of a complex number are four complex numbers with the same norm and different directions. We choose either one as as the differential value for the first edge, then the differentials of other edges can be fixed under the global smooth constrain on the sliced disk. After each edge has a new differential, we can integrate differentials to obtain a parameterization. Based on the parameterization, we extract trajectories and finally obtain a quad mesh.

5. LINEAR COMBINATION WITH CONSTRAINTS

This section shows the detail of how to improve mesh quality in the linear meromorphic differential space . We consider two mesh quality constrains, the first one is the size control of the mesh cell, and the second is the alignment between edges of the generated mesh and the feature lines of the surface. The program reads in several differential bases and adjust the combination coefficients based on the performance of the combination result. In the end, a set of better coefficients is obtained, which corresponds to a higher quality quadrilateral mesh.

We use three different model to check our algorithm, and provide both texture maps visualization and quad mesh for each experiment data. In the pictures, small pink ball represents a valence 3 singularity; blue color means valence 5 singularity and green color for velence 6 singularity.

5.1 Size constraints of mesh cells

5.1.1 Size uniformation

In many applications, a common requirement for quadrilateral meshes is that the size of the mesh cells need to be as uniform as possible. Compared with a quadrilateral mesh with large fluctuations in element size, a quad mesh with cells of uniform size is more suitable for actual industrial applications .

In order to control the cell size of the target quadrilateral mesh generated by meromorphic differentials, we define a ratio for each edge using the quotient between the norm of the differential and the Euclidean length of edge. When the ratios of these edges are relatively close, the cells of the generated quadrilateral mesh are relatively uniform.

We define a vector $r \in R^{|E|}$, whose element is

$$r_i = \frac{|duv_i|}{l_i} \tag{1}$$

where $|duv_i|$ means the norm of the differential on the i-th edge and l_i is the Euclidean length of the edge. $|E|$ represents the number of mesh edges. We set a target mesh size control vector $t \in R^{|E|}$ and define

$$\alpha = \frac{||t||_1}{||r||_1} \tag{2}$$

Then we define the loss function,

$$loss = \frac{1}{|E|} \sum_{i=1}^{|E|} (r_i * \alpha - t_i)^2 \tag{3}$$

We assume that each row of the vector t is 1 in this section, which means we want to make the result mesh as uniform as possible. We use a primary optimization method to calculate the coefficients of the linear combination. First, a set of initial combination coefficients are given for linear combination. Then we calculate the initial loss value. Then we adjust each coefficient individually. Whether the current coefficient is to increase or decrease depends on the change of the loss value after modifying the coefficient. We repeatedly adjust each coefficient until we get the desired results.

Here we use model eight to check our mesh size uniformation algorithm. Figure 2 shows four different meromorphic differentials and the corresponding quad meshes. The meromophic differential norms varies a lot across the surface, hence the cell size of these meshes is quite not uniform. The mesh quality need to be improved.

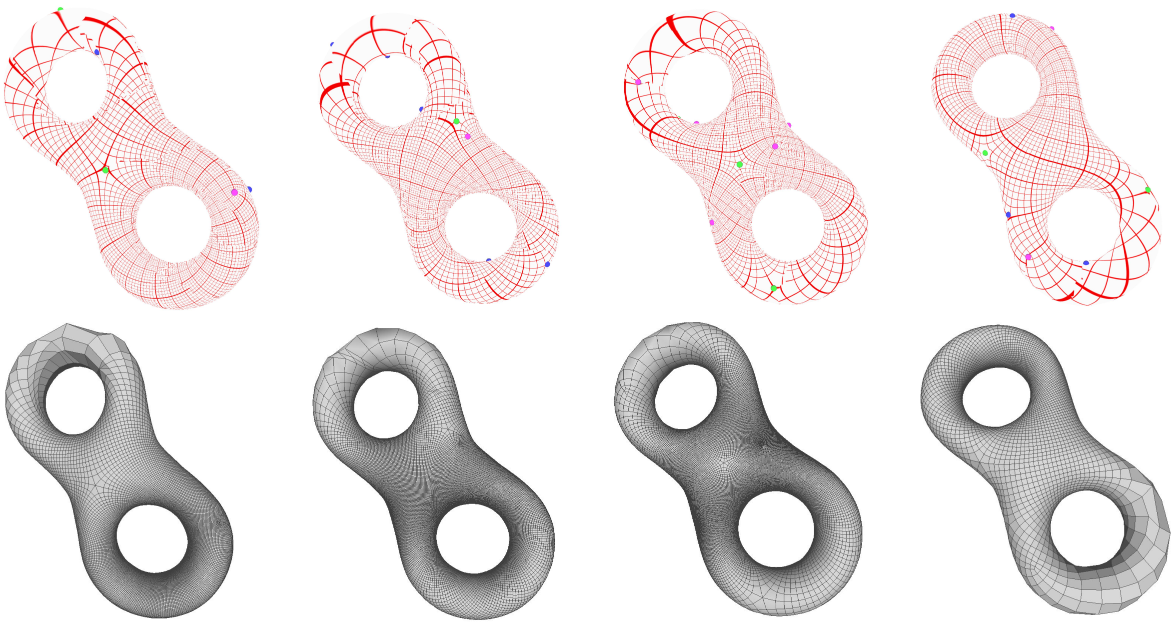
Fig.4 shows two results of mesh cell uniformation. Each model is a result of linear combination of four differential bases according to a set of optimized coefficients. It is obvious that our optimized results are more uniform. The loss value of each result is showed under the model which is quite close to zero.

5.1.2 Size control of mesh cells

Apart from pursuing the uniformity of the size of mesh cells, we can also impose other constrains on the size of the mesh cells according to the practical needs to improve mesh quality.

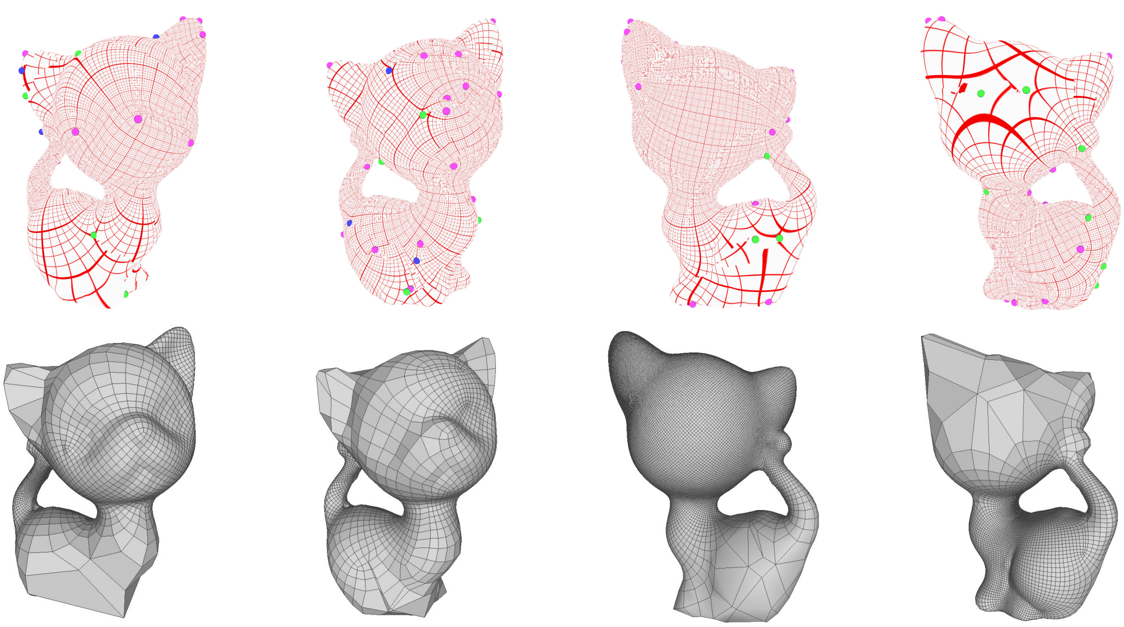
The basic idea is to set specific target ratio for each edge, Then we calculate a set of combination coefficients that meets the constrains by optimizing the loss function.

We try to control the cell size according to the curvature of the origin triangle surface. A larger curvature at a certain point indicates that the surface at that point has more details. Therefore the quad cells here need to be denser. Where the curvature is smaller, the surface here is smoother and the size of quad mesh cells can be appropriately larger. Here a curvature-aware function is set to calculate a size attribute for each



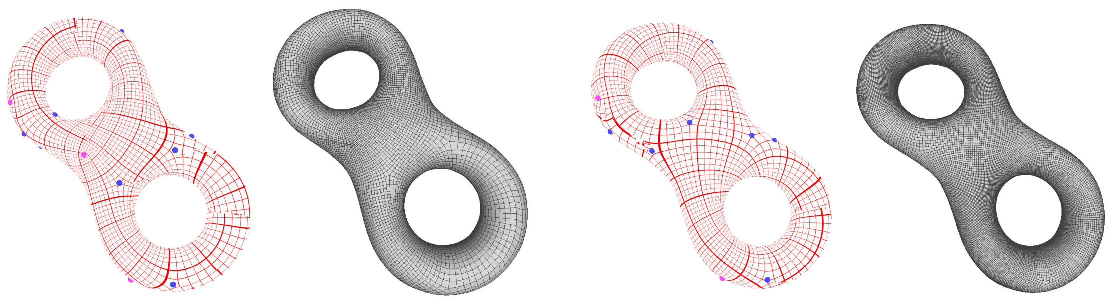
(a) basis 1 (b) basis 2 (c) basis 3 (d) basis 4

Figure 2: Four meromorphic differentials and the corresponding quad meshes of model eight



(a) basis 1 (b) basis 2 (c) basis 3 (d) basis 4

Figure 3: Four meromorphic differentials and the corresponding quad meshes of model kitten



(a) loss=0.056878

(b) loss=0.0537774

Figure 4: Size uniformation for model eight

vertex according to the Gaussian curvature. We defined the size attribute on each vertex as $s = exp(x)$, where x is the gaussian curvature of the vertex. The ratio of each edge e_i is set to be the product of the size attribute of the two adjacent vertices of edge e_i. We use this ratio of the edges to assign the elements of the target mesh size control vector $r \in R^{|E|}$.

Here we use model kitten to check the algorithm. Fig.3 shows four meromorphic bases and the corresponding quad meshes. Figure 5 shows the results of curvature-aware mesh size control. Figure 5 (a) is the mesh corresponding to the initial coefficients. Figure 5 (b) is the result for a set of optimized coefficients with a lower loss value.

5.2 Alignment of feature lines constrain

Another aspect that affects the quality of the quadrilateral mesh is the alignment with the feature lines of the surface. Models in the industry often contain many obvious features. A natural requirement is that the edges of the quadrilateral mesh we get need to be aligned with the feature lines of the model as much as possible. This section explores this constraint from the perspective of linear combination. Assuming that there is a new differential on each edge after the combination, we hope that the differential directions on the two adjacent feature edges are collinear, so as to ensure that the edges of the final generated mesh are aligned with the feature edges as much as possible.

So we define a new loss function,

$$loss\_angle = avg\left(\sum_{i,j} \langle duv_i, duv_j \rangle\right) \quad (4)$$

$$where \quad i = 1, ..., n; \quad j = 1, ..., m_i$$

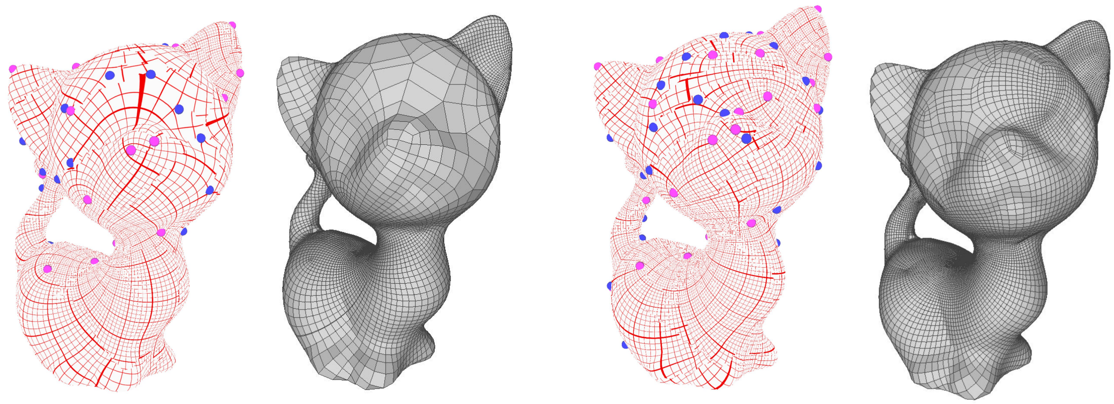
Among them, n represents the number of feature edges of the mesh, m_i represents the number of other feature edges connected to the two end points of the $i-th$ feature edge, and $< a, b >$ represents the radian value corresponding to the smallest angle of the four angles formed by vector a and vector b. $avg()$ is a function for averaging.

The subsequent algorithm pipeline is similar to the element size control method in the previous section. We first specify a set of linear combination coefficients and calculate the initial loss value. Then repeat the individual adjustment of each coefficient until the loss value no longer decreases.

Here we use model arm to take the experiment, the blue line in fig.9 is the feature line we want the quad mesh to align with. Fig.6 displays four different meromorphic differentials on the model arm surface. Fig.7 shows two results of the feature line alignment experiment. The algorithm improves the feature alignment with loss values equal to 0.3133 and 0.2831 respectively. However the mesh size is quite not uniform. Hence we try to optimize the mesh cell size and the feature alignment, Fig.8 shows the results we obtained. We achieve a balance between two constrains.

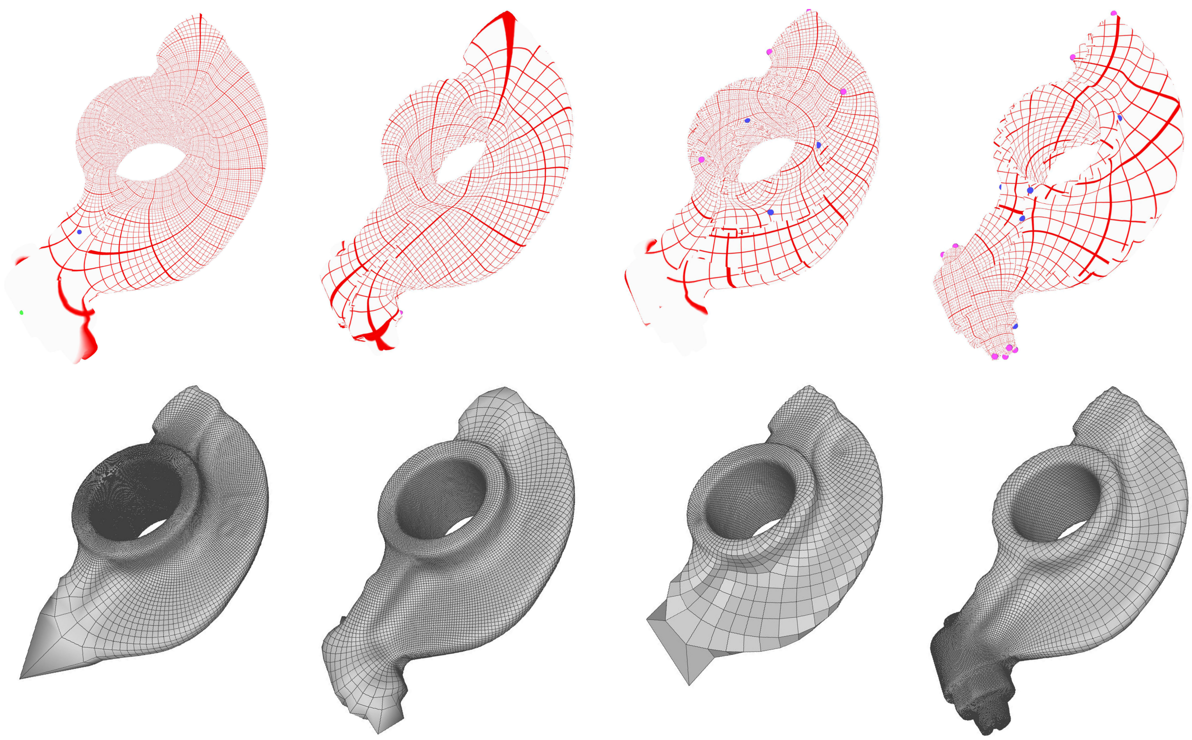
6. CONCLUSION

Based on the idea of generating quad meshes using Riemann metric and meromorphic quartic differential, this paper offers an algorithm to linearly combine different differential bases of the same surface, which can generate new meromorphic differentials and new quad meshes. This paper takes in consideration of the mesh cell size constrain and feature line alignment constrain in the process of linear combination. The programs search for a set of coefficients that meet the constrains to obtain quad meshes with better quality. Since we use multiple sets of singular points and multiple meromorphic differentials to generate a quadrilateral mesh, we reduce the dependence of mesh quality improvement on manual experience to some degree. This paper shows the feasibility and practicability of the idea that using linear combination method to gen-



(a) loss=0.201438 (b) loss=0.180988

Figure 5: Curvature aware size control for model kitten

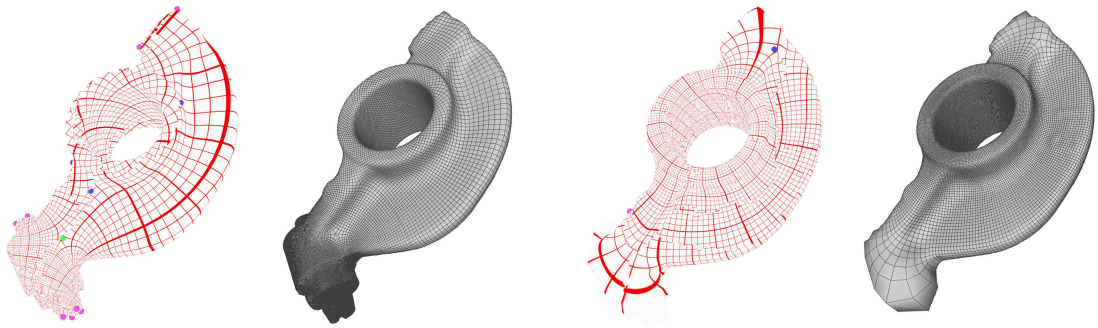


(a) basis 1 (b) basis 2 (c) basis 3 (d) basis 4

Figure 6: Four meromorphic differentials and the corresponding quad meshes of model arm

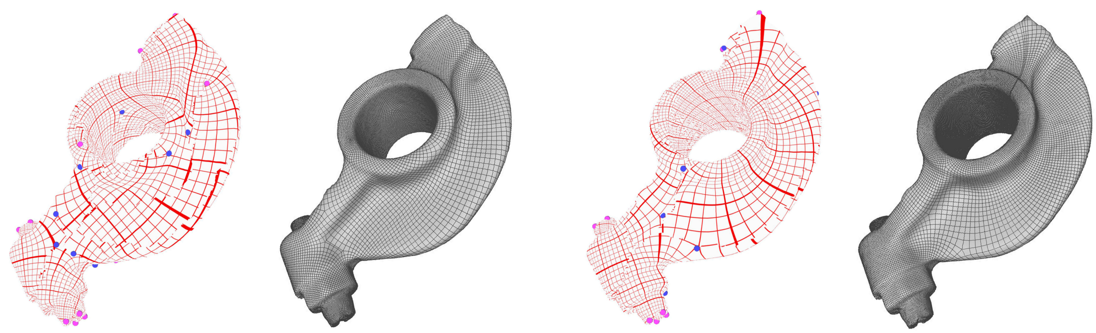
erate quad meshes and improve mesh quality. This paper offers a total new perspective to optimize the mesh quality, especially in the way of changing the mesh topology.

Our method can be improved in several ways in the future. On one hand, the differential bases we selected for each model in the experiment can not extend to the whole linear space of the meromorphic quartic differentials. This causes that the final results can not achieve the global optimum. In the future, we will choose the whole differential bases to improve the flexibility of the linear combination method. On the other hand, the optimization method we used is not so good



(a) loss=0.313383 (b) loss=0.283196

Figure 7: Feature alignment : Arm model



(a) feature loss=0.385276, size loss=0.158864 (b) feature loss=0.396364, size loss=0.181362

Figure 8: Feature alignment and size control : Arm model

Figure 9: Feature line to be aligned

enough that the result may not be the best in the current linear space. A more effective optimization method would improve our result greatly.

ACKNOWLEDGEMENT

This work is supported by Open Fund of State Key Laboratory of Intelligent Manufacturing System Technology and the National Natural Science Foundation of China under Grant No. 61907005, 61720106005, 61772105, 61936002.

References

[1] Lei N., Zheng X., Luo Z., Luo F., Gu X. "Quadrilateral mesh generation II: Meromorphic quartic differentials and Abel–Jacobi condition." *Computer Methods in Applied Mechanics and Engineering*, vol. 366, 112980, 2020

[2] Chen W., Zheng X., Ke J., Lei N., Luo Z., Gu X. "Quadrilateral mesh generation I: Metric based method." *Computer Methods in Applied Mechanics and Engineering*, vol. 356, 652–668, 2019

[3] Zheng X., Zhu Y., Lei N., Luo Z., Gu X. "Quadrilateral Mesh Generation III : Optimizing Singularity Configuration Based on Abel-Jacobi Theory." URL `https://arxiv.org/abs/2007.07334`

[4] Gu X.D., Luo F., Sun J., Wu T., et al. "A discrete uniformization theorem for polyhedral surfaces." *Journal of differential geometry*, vol. 109, no. 2, 223–256, 2018

[5] Owen S.J., Staten M.L., Canann S.A., Saigal S. "Q-Morph: an indirect approach to advancing front quad meshing." *International journal for*

numerical methods in engineering, vol. 44, no. 9, 1317–1340, 1999

[6] Remacle J.F., Lambrechts J., Seny B., Marchandise E., Johnen A., Geuzainet C. “Blossom-Quad: A non-uniform quadrilateral mesh generator using a minimum-cost perfect-matching algorithm.” *International journal for numerical methods in engineering*, vol. 89, no. 9, 1102–1119, 2012

[7] Velho L., Zorin D. “4–8 Subdivision.” *Computer Aided Geometric Design*, vol. 18, no. 5, 397–427, 2001

[8] Ray N., Vallet B., Li W.C., Lévy B. “N-symmetry direction field design.” *ACM Transactions on Graphics (TOG)*, vol. 27, no. 2, 1–13, 2008

[9] Lai Y.K., Jin M., Xie X., He Y., Palacios J., Zhang E., Hu S.M., Gu X. “Metric-driven rosy field design and remeshing.” *IEEE Transactions on Visualization and Computer Graphics*, vol. 16, no. 1, 95–108, 2009

[10] Hertzmann A., Zorin D. “Illustrating smooth surfaces.” *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pp. 517–526. 2000

[11] Ray N., Vallet B., Alonso L., Levy B. “Geometry-aware direction field processing.” *ACM Transactions on Graphics (TOG)*, vol. 29, no. 1, 1–11, 2009

[12] Ray N., Sokolov D. “Robust polylines tracing for n-symmetry direction field on triangulated surfaces.” *ACM Transactions on Graphics (TOG)*, vol. 33, no. 3, 1–11, 2014

[13] Dong S., Bremer P.T., Garland M., Pascucci V., Hart J.C. “Spectral surface quadrangulation.” *ACM SIGGRAPH 2006 Papers*, pp. 1057–1066. 2006

[14] Huang J., Zhang M., Ma J., Liu X., Kobbelt L., Bao H. “Spectral quadrangulation with orientation and alignment control.” *ACM SIGGRAPH Asia 2008 papers*, pp. 1–9. 2008

[15] Myles A., Zorin D. “Global parametrization by incremental flattening.” *ACM Transactions on Graphics (TOG)*, vol. 31, no. 4, 1–11, 2012

[16] Purnomo B., Cohen J.D., Kumar S. “Seamless texture atlases.” *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pp. 65–74. 2004

[17] Campen M., Bommes D., Kobbelt L. “Quantized global parametrization.” *Acm Transactions On Graphics (tog)*, vol. 34, no. 6, 1–12, 2015

[18] Lei N., Zheng X., Si H., Luo Z., Gu X. “Generalized regular quadrilateral mesh generation based on surface foliation.” *Procedia engineering*, vol. 203, 336–348, 2017