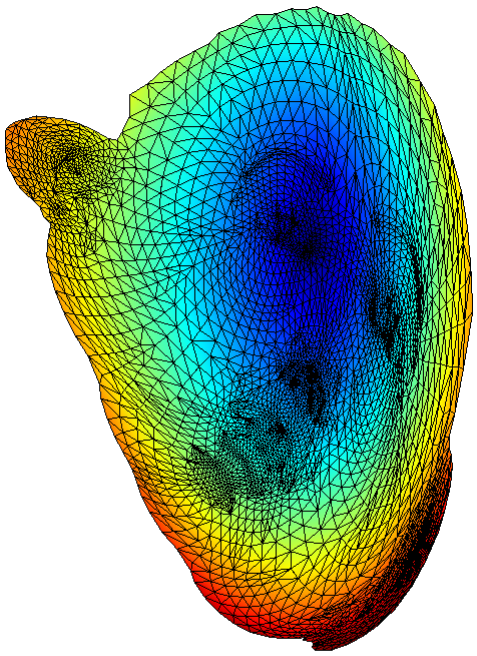
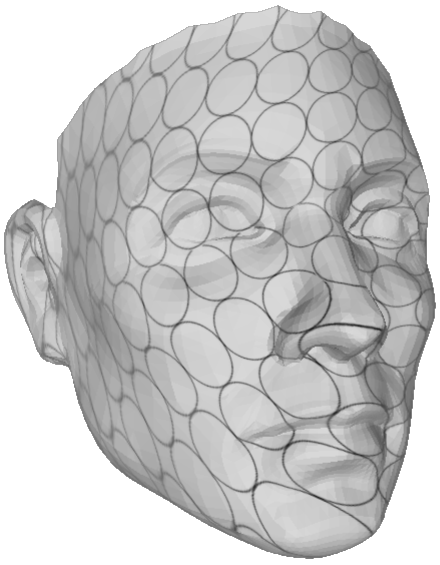
Quasi-Conformal Surface Mapping Project

Computer Graphics project by Lee Susman, supervised by Miri Ben-Chen





Introduction

Surface mapping plays a key role in digital geometry processing. In general, surface maps introduce area and angular distortions. If no angular distortion is introduced, the surface mapping is called a conformal mapping. But natural surface maps, such as the change of the surface of a face while smiling, do induce distortion. Therefore it is interesting to study Quasi-Conformal surface maps (QC maps), which are maps that induce bounded angular distortion. The amount of distortion is given by a complex valued function called a Beltrami coefficient. While the continuous theory for QC maps is well developed, the discrete one is still in the making. Paper [1] attempts to define a discrete analogue for the existing theory. Using this new definition, we get an algorithm for computing a QC map associated with a given Beltrami Coefficient.

A bit of theory

Smooth Quasiconformal Theory

The theory of QC mappings is a relatively new theory in complex analysis, introduced in 1928. Intuitively, an orientation preserving mapping from $\mathbb{C}$ to $\mathbb{C}$ is called **quasiconformal** if it is a homeomorphism which takes small circles to small ellipses of bounded eccentricity. More formally, let $f : \mathbb{C} \to \mathbb{C}$ be a complex function and let $\mu : \mathbb{C} \to \mathbb{C}$ be a complex valued, Lebesgue measurable function with $sup\,|\mu| < 1$. f is said to be QC associated with μ if it is orientation-preserving and satisfies

$$\frac{\partial f}{\partial \bar{z}} = \mu(z)\,\frac{\partial f}{\partial z}$$

known as the Beltrami equation. In particular, note that if $\mu(z) \equiv 0$ we have $\frac{\partial f}{\partial \bar{z}} = 0$, and by the Cauchy-Riemann equations we get that f is holomorphic (assuming it is differentiable). In this case f is Conformal, meaning it induces zero angular distortion.

The function μ holds in it all the information about the conformality of f: The eccentricity of the ellipse mapped by f is given by

$$e = \frac{1 + |\mu|}{1 - |\mu|}$$

and the orientation is given by

$$\theta = \frac{1}{2} arg\mu$$

We can also generalize this notion to mappings between embedded surfaces:

Suppose S_1 and S_2 are two Riemann surfaces with global conformal parametrization r_1 and r_2 respectively (equipped with the standard Euclidean metric). An orientation preserving homeomorphism $f : S_1 \to S_2$ is called QC associated with μ if the mapping $\widetilde{f} := r_2 \circ f \circ r_1^{-1}$ is QC associated with μ.

Discrete Quasiconformal Theory

In order to build a discrete analog for the smooth QC theory, which deals with triangulation meshes rather than smooth surfaces, there is a need to define a discrete metric on meshes:

A **discrete metric** on a mesh $M = (V, E, F)$ is a function $l : E \to \mathbb{R}^+$, such that on each triangle $[v_i, v_j, v_k]$, the triangle inequality holds,

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

where l_{mn} is the length of the edge $[v_m, v_n]$. Now for the definition (by [1]) of a discrete QC map between two triangulation meshes:

Let M_1 and M_2 be two triangulation meshes with discrete metrics l_1 and l_2 respectively, and let $\mu : \mathbb{C} \to \mathbb{C}$ be a given Beltrami coefficient. Also, let $z : V \to \mathbb{C}$ be a conformal parametrization of M_1. A mapping $f : (M_1, l_1) \to (M_2, l_2)$ is a **discrete quasiconformal mapping**, if with respect to a new metric $\widetilde{l}$ on M_1, the mapping $f : \left(M_1, \widetilde{l}\right) \to (M_2, l_2)$ is discrete conformal, where

$$\widetilde{l}_{ij} := l_{ij} \frac{|dz_{ij} + \mu_{ij} d\bar{z}_{ij}|}{|dz_{ij}|}$$

The discrete Beltrami coefficient on $[v_i, v_j]$ is $\mu_{ij} = \frac{\mu_i + \mu_j}{2}$ and $dz_{ij} = z(v_j) - z(v_i)$ is the per-edge derivative. $\widetilde{l}$ is called the **discrete auxiliary metric** associated with μ.

Finally, we introduce a theorem which will be used for computing the discrete QC mappings:

Theorem: Suppose (M_1, l_1) and (M_2, l_2) are two metric triangular meshes, $f : M_1 \to M_2$ is a QC mapping with Beltrami coefficient μ. Under the auxiliary metric $\widetilde{l}$ associated with μ, the mapping $f : \left(M_1, \widetilde{l}\right) \to (M_2, l_2)$ is discrete conformal.

About the Project

In this project I implemented the algorithm given in [1] for computing the discrete QC map using the auxiliary metric associated with a prescribed Beltrami coefficient on an input triangular mesh. Once the map is calculated, a texture can be added to the mesh in order to represent the angular distortion induced by the QC map.

Technical Details

The implementation

The algorithm was implemented in Matlab. Input is an .obj file representing the mesh and a $\mathbb{C} \to \mathbb{C}$ function representing the Beltrami coefficient. Output is an .obj file representing the flattened mesh in texture coordinates. Matlab also prints to the screen the original mesh, the conformally flattened mesh, and the QC flattened mesh with vertices colored according to the per-vertex Beltrami coefficient. Onto the output obj we can then add texture and present it in Meshlab to see the distortion introduced by the QC map. The implementation includes the following steps:

- Read the input mesh coordinates and connectivity from the .obj file and store them in Matlab data structures
- Compute discrete metric (standard Euclidean metric) on the input mesh, and use dcflatten.m code to compute the conformal parametrization under this metric
- Compute the discrete auxiliary metric associated with the Beltrami coefficient
- Under the discrete auxiliary metric, dc-flatten the mesh to obtain the QC map

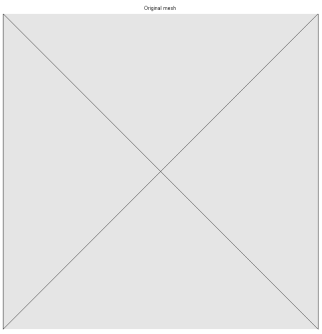
Usage

For using the QC surface mapper, you should have Matlab installed. Then,

- Place all of the Matlab source files in some directory, lets refer to it as the working directory
- Under the working directory, place a directory named 'input_objs' in which you should place the input meshes as .obj files
- In the working directory place a texture .png file and name it 'out.png'
- In the working directory place a .mtl file named 'out.mtl'
- In Matlab, run the gui.fig file
- In the window that opens, input a Beltrami coefficient (with z as the variable), choose an input mesh and hit 'Calculate'
- The output mesh is saved to 'out.obj'
- You can see the texture mapping on the original mesh by opening 'out.obj' in Meshlab

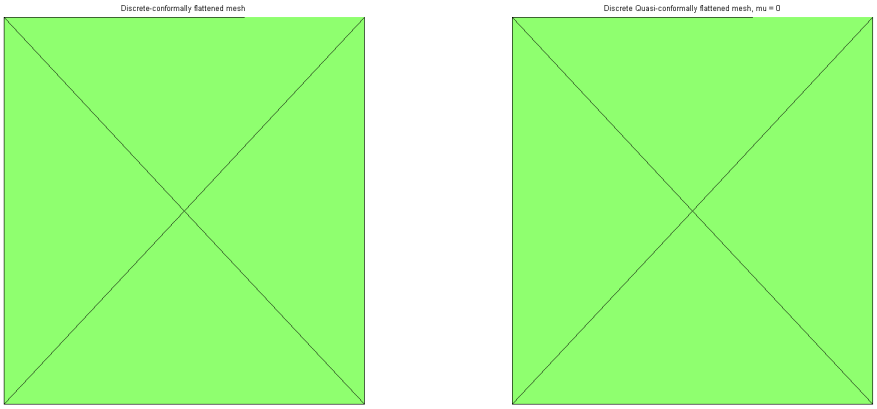
Examples

Lets start off with a flat mesh, i.e. a mesh that is contained in the xy coordinate plane:



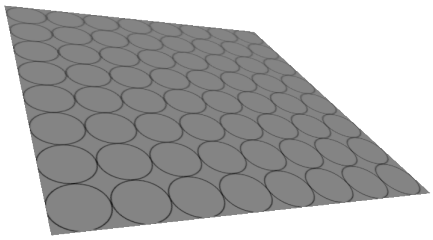
original, flat square viewed from above

Since it is already flat, after discrete conformaly flattening the mesh we should still get the same mesh:



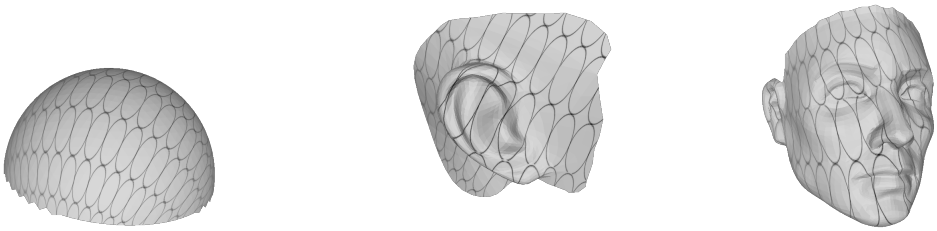
flattened squares viewed from above, faces colored by interpolating μ values on the vertices, $\mu(z) \equiv 0$

In this example I chose the Beltrami coefficient to be $\mu(z) \equiv 0$, therefore the discrete QC flattened mesh is also the same mesh. Clearly, for the texture coordinates we get a perfect circle texture:



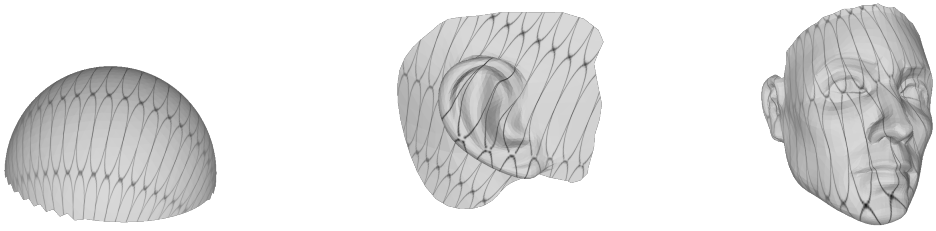
square with circle texture viewed in Meshlab

Now lets examine constant but nonzero Beltrami coefficients. First, if we choose a real and positive coefficient, the result is a mapping between circles and ellipses which are horizontally oriented since the argument is 0.



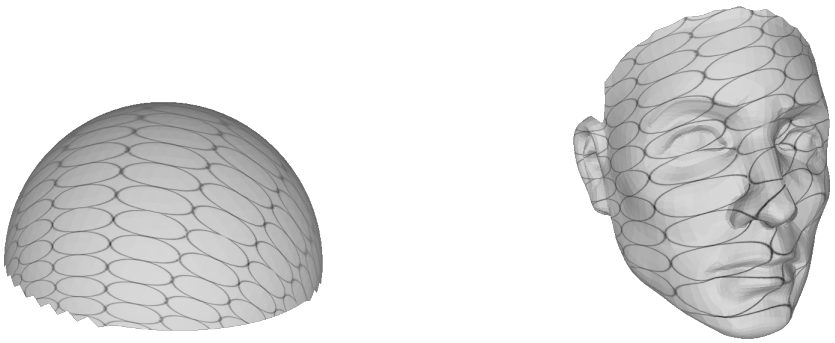
hemisphere, ear and face, $\mu(z) \equiv 0.5$

Still with a positive constant, increasing the absolute value of the coefficient, we get ellipses with greater eccentricity oriented in the same direction:



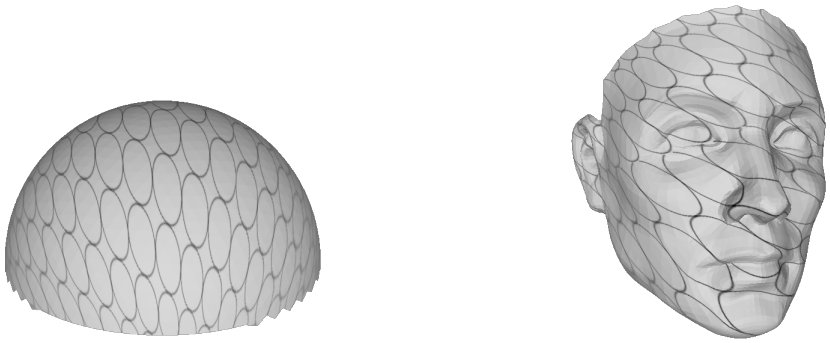
hemisphere, ear and face, $\mu(z) \equiv 0.7$

Changing μ to be negative introduces a $\frac{\pi}{2}$ counter-clockwise rotation to the ellipse's orientation since $\theta = \frac{1}{2} arg\mu = \frac{1}{2} arg(-1) = \frac{\pi}{2}$:



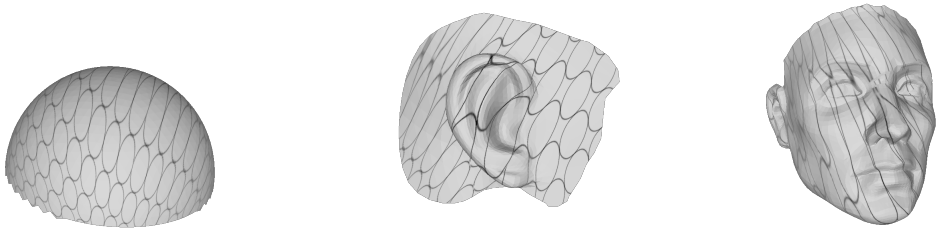
hemisphere and face, $\mu(z) \equiv -0.5$

Letting μ be a pure imaginary number introduces a $\frac{\pi}{4} = \frac{1}{2} arg\,(i)$ counter-clockwise orientation:



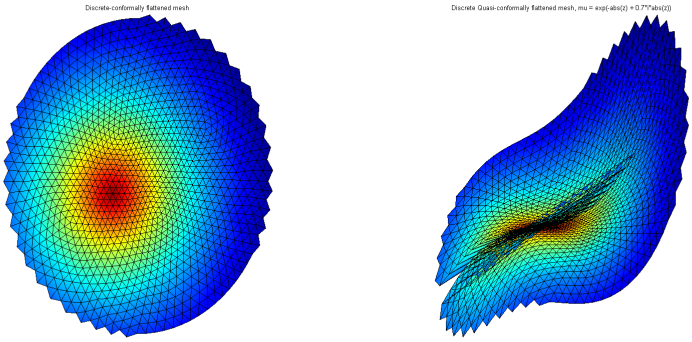
hemisphere and face, $\mu\,(z) \equiv 0.5i$

Finally, non-constant Beltrami coefficients introduce a less trivial setting of eccentricities and orientations. For example:

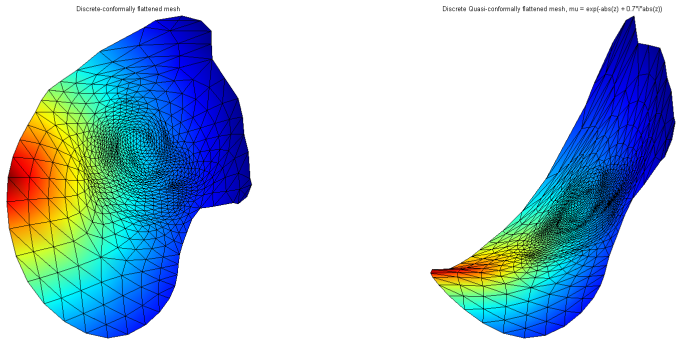


hemisphere, ear and face with $\mu\,(z) = e^{-|z|+0.7|z|i}$

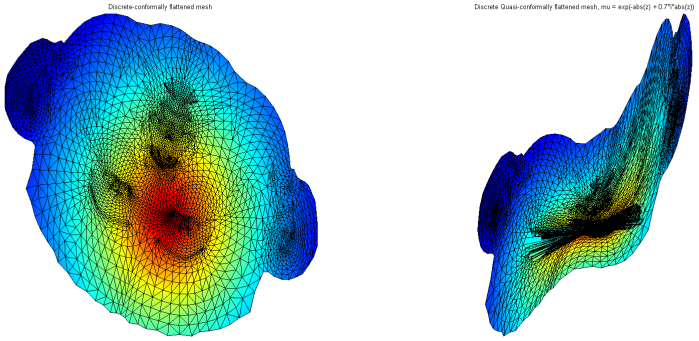
As for the flattened meshes:



dc-flat (left) and dqc flat hemisphere with $\mu(z) = e^{-|z|+0.7|z|i}$

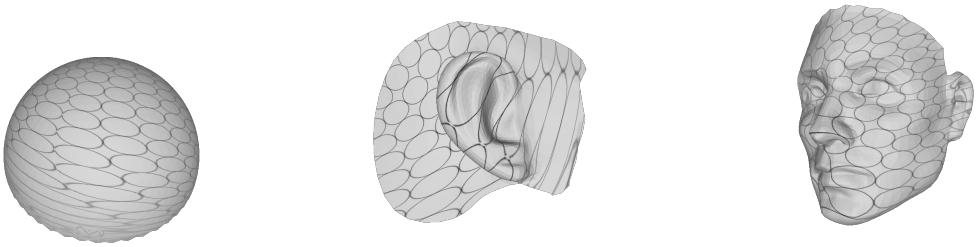


dc-flat (left) and dqc flat ear with $\mu(z) = e^{-|z|+0.7|z|i}$

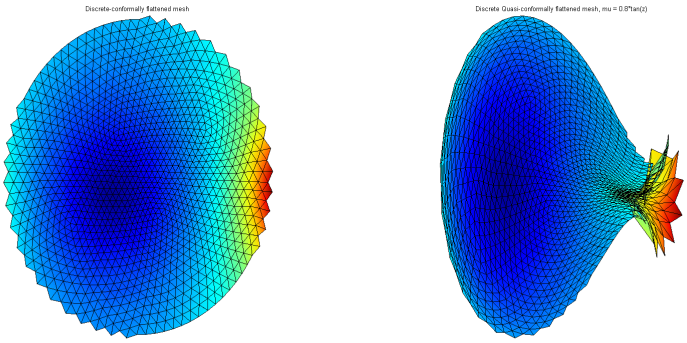


dc-flat (left)and dqc flat face with $\mu(z) = e^{-|z|+0.7|z|i}$

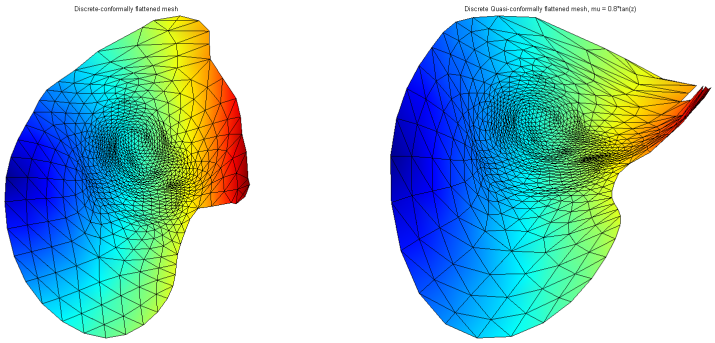
And to finish off, lets check out another function:



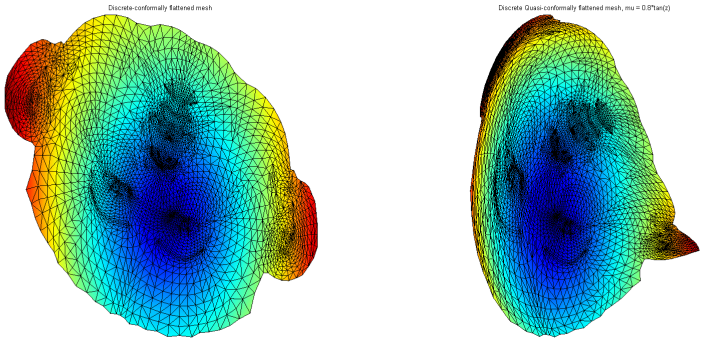
hemisphere, ear and face with $\mu(z) = 0.8 \cdot tan(z)$



dc-flat (left) and dqc flat hemisphere with $\mu(z) = 0.8 \cdot tan(z)$



dc-flat (left) and dqc flat ear with $\mu(z) = 0.8 \cdot tan(z)$



dc-flat (left) and dqc flat face with $\mu(z) = 0.8 \cdot tan(z)$

Problems that arose during the project

Computation time: Most interesting meshes are constituted of at least thousands of faces. Matlab is a renown CPU bound and memory bound software, so for many meshes we get a very long computation time (such as the face mesh exhibited here). The face for example takes A solution for this problem could be implementation in some low-level programming language (such as C).

Dcflattening: Finding isothermal coordinates for a given mesh is done in [1] using a discrete curvature flow method. These methods were not covered in the paper (are covered in [2]), and implementing them was out of the scope of this project. Therefore I used dcflatten.m (which was implemented by the authors of [2] and modified by Renjie Chen to be intrinsic) to compute the isothermal coordinates.

Downloadables

All of the source code, example meshes, mtl and texture files can be downloaded here: ..... TBD.....

References:

1. Computing quasiconformal maps using an auxiliary metric and discrete curvature flow (Wei Zeng · Lok Ming Lui · Feng Luo · Tony Fan-Cheong Chan · Shing-Tung Yau · David Xianfeng Gu)

2. Conformal Equivalence of Triangle Meshes (Boris Springborn, Peter Schröder, Ulrich Pinkall)