中国科学技术大学

学士学位论文



高效无翻转球面参数化

作者姓名： 陈焕宇

学科专业： 数学与应用数学

导师姓名： 刘利刚教授

完成时间： 二〇二一年五月二十四日

University of Science and Technology of China

A dissertation for bachelor's degree



Efficient Bijective Spherical Parameterizations

Author: Huanyu Chen

Speciality: Mathematics and Applied Mathematics

Supervisor: Prof. Ligang Liu

Finished Time: May 24, 2021

致谢

经过不断地探究与摸索，为时一年多的课题终于有了不错的成果。本课题是刘利刚教授在 2020 春讲授计算机图形学这门课的时候布置给我的。我在刘教授的指导下阅读了近百篇文献，实现了数十种算法，提出了大量新的想法然后不断试错找到正确的方向才得以完成这个课题。在这里，我首先想要表达我对刘教授最真诚的感谢，感谢刘教授带领我走进计算机图形学的大门，走上科研之路，并不断给予我指导与启发。同时，我还要感谢傅孝明教授和叶春阳师兄在这个过程中不断地解答我的各种疑惑。

四年的大学时光转瞬即逝，在这四年里我收获了很多。感谢学校开设的各类课程夯实了我的数理基础；感谢学校举办的各种学术讲座开拓了我的视野；感谢学校提供的活动设施丰富了我的课余生活。在此，我要感谢班主任张轶老师经常鼓励我，让我不忘初心坚持科研的梦想；感谢教学秘书齐莹老师与张伟老师解答我在学习规划上的疑问；感谢宿管为我提供生活上的帮助。感谢同学和室友们与我一起努力，共同进步。

在升学之路上，我要感谢刘利刚教授，张举勇教授，曹洋教授的推荐，让我获得了多所海外名校的全奖直博录取。同时，我还要感谢陈仁杰教授与陈杲学长帮我做课程学业规划；感谢梁永祺教授与傅孝明教授在选校上给我的指导与建议。

最后，我要好好感谢一下父母。感谢父母的无私付出与养育之恩；感谢父母对我的精心呵护，让我受到良好的教育，成为一个博学多才的好学生。感谢父母培养我独立思考，独立生活的能力；感谢父母给予我自由的选择，成长的空间；感谢父母塑造我的三观，让我成为一个正义感强且乐于助人的人。未来，我将投身于科研事业，做一个红专并进，理实交融的学者来回报父母，学校，国家对我的培养。

目　录

中文内容摘要 3
Abstract 4
第一章　绪论 5
第一节　光滑曲面 5
一、　基本定义 5
二、　曲面的局部理论 6
三、　曲面的整体理论 10
第二节　三角网格曲面 11
一、　实际应用中的曲面表示 11
二、　离散微分几何 14
第三节　网格参数化问题 16
一、　问题概述 16
二、　扭曲能量 18
三、　低扭曲参数化的问题表述 21
第四节　几何优化方法简介 22
一、　一般非线性优化 22
二、　几何优化 24
第二章　平面参数化的几何优化方法 27
第一节　Tutte 方法及其变种 27
第二节　迭代求解器 29
一、　预处理器 29
二、　二阶求解器 34
三、　直线搜索 38
四、　终止条件 39
第三节　加速方法简介 40
一、　更改参考三角形 40
二、　更改搜索方向 41
三、　Nesterov 加速 41
四、　Anderson 加速 42

第三章　低扭曲无翻转球面参数化 43
第一节　球面参数化简介 43
一、　基于球极投影的球面参数化 43
二、　基于球心投影的球面参数化 44
三、　基于共形几何的球面参数化 45
四、　Colin de Verdiere 方程 46
五、　基于多分辨率的球面参数化 47
六、　球面参数化的几何优化 47
第二节　球面三角网格 49
一、　球面三角形的无翻转条件 49
二、　严格的球面参数化映射 49
三、　球面参数化映射的扭曲能量 51
四、　球面参数化几何优化问题的表述 54
第三节　球面参数化的几何优化 55
一、　球面初始化 55
二、　优化求解器 55
三、　球面约束的处理 59
四、　直线搜索 60
第四节　实验结果 62
一、　实现细节 62
二、　结果展示 62
三、　结论 66
第四章　总结与思考 67
参考文献 69

中文内容摘要

在计算机图形学中，曲面常用三角网格表示。网格的参数化问题一直是数字几何处理中的基本问题之一，它在纹理贴图，网格再划分，曲面间映射，几何图像，网格变形等一系列图形学应用中起着重要作用。实际应用中的网格通常来自物体表面的 3D 扫描，因此常为闭网格。要将这样的网格参数化，我们可以先将其切成开曲面然后映射到平面区域，或者也可直接将整个映到某个闭曲面。对于亏格为 0 的闭曲面网格，最自然的参数域便是单位球面。平面参数化问题已被广泛研究，因此求解方法多种多样。与此相反，球面参数化由于其球面约束带来的困难因此研究热度较低。特别地，在平面参数化中有一个通用的几何优化流水线可以保证得到无翻转的参数化映射，而这一流水线直接应用于球面参数化时效果并不好。本文的目的便是修改这一流水线使得它在求解球面参数化问题时也能顺利使用。

本文首先引入了低扭曲无翻转参数化问题，然后详细阐述了平面参数化中的几何优化流程以及各种求解器，之后调查了现有的球面参数化方法，最后提出了球面参数化的几何优化流程。实验结果表明本文提出的几何优化流程能够很好地求解球面参数化问题，并得到低扭曲的球面参数化映射。

关键词：几何处理，网格参数化，几何优化

Abstract

In computer graphics, surfaces are often represented as triangle meshes. Mesh parameterization is a fundamental problem in digital geometry processing, and plays an important role in a wide range of graphics applications including texture mapping, remeshing, surface mapping, geometry image, morphing, etc. In real applications, meshes usually come from 3D scans of object surfaces, and thus are often closed. To parameterize such a mesh, we can first cut it open and then map it to a planar region, or we can directly map it to some closed surface. For a closed surface mesh of genus 0, the most natural parameter region is the unit sphere. Planar parameterizations have been intensively studied, thus a variety of solvers are available. On the contrary, spherical parameterizations receive much less research focus due to the difficulties brought by the spherical constraint. In particular, there is a universal geometric optimization pipeline for planar parameterizations that guarantees a fold-over-free result, but such pipeline does not work well when directly applied to spherical parameterizations. This thesis aims to modify this pipeline so that it works well for spherical parameterizations.

We first introduce the problem of computing a low-distortion parameterization map with no flips, then elaborate the geometric optimization pipeline and various solvers for planar parameterization, then conduct a survey on existing spherical parameterization methods, and finally proposed our spherical parameterization pipeline. Experiment results show that our pipeline works well for spherical parameterizations and produces spherical mappings with low distortion.

Keywords: Geometry Processing, Mesh Parameterizations, Geometric Optimization

第一章　绪论

第一节　光滑曲面

一、　基本定义[45]

一个 n 维**拓扑流形**(topological manifold)是指局部同胚于 n 维欧氏空间且第二可数的 Hausdorff 空间。这代表对于该空间中的每一点p, 都存在开邻域U_p和同胚映射

$$\phi_p : U_p \simeq V_p \subseteq \mathbb{R}^n,$$

其逆映射ϕ_p^{-1}可视为流形在该点附近的一个参数表示。因此这对(U_p, ϕ_p)称为一个**图表**(chart)，U_p称为点p的一个**坐标邻域**，ϕ_p称为U_p上的**坐标映射**。若两个图表$(U, \phi), (V, \psi)$之间的过渡映射光滑，即

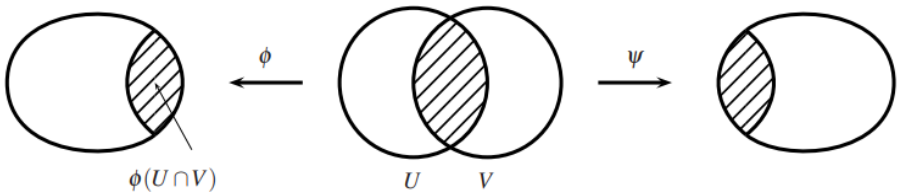


图 1.1 两个图表间的过渡映射[45]

$$\phi \circ \psi^{-1} : \psi(U \cap V) \to \phi(U \cap V), \qquad \psi \circ \phi^{-1} : \phi(U \cap V) \to \psi(U \cap V)$$

都是光滑映射，则称这两个图表**相容**(compatible)。流形M上的**图集**(atlas)是指一族两两相容且能覆盖M的图表$\mathfrak{U} = \{(U_\alpha, \phi_\alpha)\}$，即有$M = \bigcup_\alpha U_\alpha$。若流形$M$上存在这样的图集，则称其为**光滑流形**(smooth manifold)。

在微分几何中，**曲面**(surface)一般指二维光滑流形，记为S。因此，曲面上每一点都存在一个可以用平面区域参数表示的坐标邻域。在这里，我们假设曲面可嵌入三维欧氏空间，即存在嵌入映射$f : S \hookrightarrow \mathbb{R}^3$。若将曲面上的每一点与其像等价，则亦可直接认为该曲面是三维欧氏空间的子集。这样一来，曲面上的每一点p附近都有一个光滑的参数表示$\boldsymbol{f_p} : \Omega_p \to \mathbb{R}^3$，其中$\Omega_p$是一个平面区域，它在$\boldsymbol{f_p}$下的像是$p$的一个坐标邻域。

二、 曲面的局部理论[1]

1. 基本形式与曲率

为了方便研究曲面的局部性质，我们不妨假设这个曲面就是$\boldsymbol{f}:\Omega\to\mathbb{R}^3$，并把曲面上的点$p$等同于其参数坐标$(u,v)$。点$p$处由

$$\boldsymbol{f}_u(p)=\frac{\partial \boldsymbol{f}}{\partial u}(p),\qquad \boldsymbol{f}_v(p)=\frac{\partial \boldsymbol{f}}{\partial v}(p)$$

生成的平面称为该点处的**切平面**(tangent plane)，记为T_pS。

$$\boldsymbol{n}(p)=\frac{\boldsymbol{f}_u(p)\times\boldsymbol{f}_v(p)}{\|\boldsymbol{f}_u(p)\times\boldsymbol{f}_v(p)\|}$$

称为该点处的**单位法向量**(unit normal)，它在二维球面S^2上。映射

$$G:S\to S^2,\qquad p\mapsto\boldsymbol{n}(p)$$

称为该曲面的**高斯映射**(Gauss map)。

曲面的**第一基本形式**(first fundamental form)定义为：

$$I=d\boldsymbol{f}\cdot d\boldsymbol{f}=\begin{pmatrix}du & dv\end{pmatrix}\begin{pmatrix}E & F\\ F & G\end{pmatrix}\begin{pmatrix}du\\ dv\end{pmatrix},$$

它是曲面上一点处切平面上由三维欧氏空间诱导的度量，刻画了曲面局部的度量性质。由定义可知

$$E=\boldsymbol{f}_u\cdot\boldsymbol{f}_u,\qquad F=\boldsymbol{f}_u\cdot\boldsymbol{f}_v,\qquad G=\boldsymbol{f}_v\cdot\boldsymbol{f}_v.$$

曲面的**第二基本形式**(second fundamental form)定义为

$$II=-d\boldsymbol{f}\cdot d\boldsymbol{n}=\begin{pmatrix}du & dv\end{pmatrix}\begin{pmatrix}L & M\\ M & N\end{pmatrix}\begin{pmatrix}du\\ dv\end{pmatrix},$$

它刻画了曲面在一点处附近的形状。由定义可知

$$L=\boldsymbol{f}_{uu}\cdot\boldsymbol{n},\qquad M=\boldsymbol{f}_{uv}\cdot\boldsymbol{n},\qquad N=\boldsymbol{f}_{vv}\cdot\boldsymbol{n}.$$

第一基本形式和第二基本形式都与参数的选取无关，且在$\mathbb{R}^3$的正交变换下不变。

给定一点处的单位切向量$\boldsymbol{t}$，曲面在该点处沿$\boldsymbol{t}$的曲率向量沿法向量的分量称为**法曲率**(normal curvature)，它可以由

$$k_n(\boldsymbol{t})=\frac{II(\boldsymbol{t},\boldsymbol{t})}{I(\boldsymbol{t},\boldsymbol{t})}$$

计算。各方向法曲率中的极大值和极小值称为**主曲率**(principal curvatures)，记为k_1, k_2，它们是矩阵$\begin{pmatrix} L & M \\ M & N \end{pmatrix}\begin{pmatrix} E & F \\ F & G \end{pmatrix}^{-1}$的特征值。特别地，$H = (k_1 + k_2)/2$称为该点处的**平均曲率**(average curvature)，$G = k_1 k_2$称为该点处的**高斯曲率**(Gaussian curvature)。高斯曲率是高斯映射的变积系数。

2. 曲面上的微分形式

设$\{\boldsymbol{e}_1, \boldsymbol{e}_2\}$是曲面$S$上一点$p$处切平面的一组正交基，

$$\boldsymbol{e}_3 = \boldsymbol{e}_1 \times \boldsymbol{e}_2 = \boldsymbol{n}$$

是单位法向量，则

$$\omega_1 = d\boldsymbol{f} \cdot \boldsymbol{e}_1, \qquad \omega_2 = d\boldsymbol{f} \cdot \boldsymbol{e}_2$$

是其对应的一阶微分形式。再令

$$\omega_{ij} = d\boldsymbol{e}_i \cdot \boldsymbol{e}_j, \qquad i, j = 1, 2, 3,$$

则有

$$\omega_{ij} + \omega_{ji} = d(\boldsymbol{e}_i \cdot \boldsymbol{e}_j) = d\delta_{ij} = 0, \qquad i, j = 1, 2, 3.$$

曲面的结构方程表明，

$$\begin{cases} d\omega_1 = \omega_{12} \wedge \omega_2 \\ d\omega_2 = \omega_1 \wedge \omega_{12} \end{cases}, \qquad \begin{cases} d\omega_{12} = \omega_{13} \wedge \omega_{32} \\ d\omega_{13} = \omega_{12} \wedge \omega_{23}. \\ d\omega_{23} = \omega_{21} \wedge \omega_{13} \end{cases}$$

另外，曲面的几何量都可以用这些一阶微分形式表示：

$$\begin{cases} I = d\boldsymbol{f} \cdot d\boldsymbol{f} = \omega_1\omega_1 + \omega_2\omega_2 \\ II = -d\boldsymbol{f} \cdot d\boldsymbol{n} = \omega_1\omega_{13} + \omega_2\omega_{23} \\ III = d\boldsymbol{n} \cdot d\boldsymbol{n} = \omega_{13}\omega_{13} + \omega_{23}\omega_{23} \end{cases}, \qquad \begin{cases} dA = \omega_1 \wedge \omega_2 \\ d\sigma = \omega_{13} \wedge \omega_{23} \end{cases},$$

其中$d\sigma = Kdu \wedge dv = KdA$是高斯曲率的面积元。特别地，

$$d\omega_{12} = \omega_{13} \wedge \omega_{32} = -d\sigma = -KdA = -K\omega_1 \wedge \omega_2$$

称为**联络 1-形式**(connection 1-form)。上述方程还可写为：

$$K = -\frac{d\omega_{12}}{\omega_1 \wedge \omega_2},$$

称为曲面的**高斯方程**。由该方程可以得到著名的**高斯绝妙定理**(Gauss' Theorema

Egregium)：曲面的高斯曲率只与其第一基本形式有关。

3. 曲面间的映射

设$\boldsymbol{f}_1:\Omega_1\to S_1\subseteq\mathbb{R}^3,\boldsymbol{f}_2:\Omega_2\to S_2\subseteq\mathbb{R}^3$是两张光滑曲面。曲面间的映射$\boldsymbol{\phi}:S_1\to S_2$诱导切空间之间的映射

$$d\boldsymbol{\phi}:T_pS_1\to T_{\boldsymbol{\phi}(p)}S_2,\qquad \frac{\partial\boldsymbol{f}_1}{\partial u}\mapsto\frac{\partial(\boldsymbol{\phi}\circ\boldsymbol{f}_1)}{\partial u},\qquad \frac{\partial\boldsymbol{f}_1}{\partial v}\mapsto\frac{\partial(\boldsymbol{\phi}\circ\boldsymbol{f}_1)}{\partial v}$$

称为对应点处的**切映射**(tangent map)。若$\boldsymbol{\phi}$满足

$$I_{S_1}(\boldsymbol{t}_1,\boldsymbol{t}_2)=kI_{S_2}\big(d\boldsymbol{\phi}(\boldsymbol{t}_1),d\boldsymbol{\phi}(\boldsymbol{t}_2)\big),\qquad\forall\boldsymbol{t}_1,\boldsymbol{t}_2\in T_pS_1,$$

其中k只依赖于p而不依赖于$\boldsymbol{t}_1,\boldsymbol{t}_2$，则称其为**共形映射**(conformal map) 。特别地，若$k\equiv1$，则称其为**等距映射**(isometric map)。由高斯绝妙定理，曲面的高斯曲率在等距映射下保持不变。因此，只有高斯曲率相等的曲面之间才可能存在等距映射。若将参数域看成一个曲面，将参数映射看成曲面间的映射，便可以讨论参数映射本身的共形性与等距性。因此，只有高斯曲率恒为 0 的曲面才可能有等距参数化。

高斯曲率恒为 0 的直纹面称为**可展曲面**(developable surface)，例如柱面，锥面，切线面。顾名思义，这些曲面可以展开（即等距参数化）到平面。对于一般的任意曲面，可以证明其上每一点都存在一个可以共形映射到平面区域的坐标邻域。这样的参数称为**等温坐标**(isothermal coordinates)。

4. 测地线与 Gauss-Bonnet 公式

给定曲面上的一条曲线，它的曲率向量在切平面上的投影称为**测地曲率向量**(geodesic curvature vector), 其值称为**测地曲率**，记为k_g，它只与第一基本形式有关。曲面上测地曲率恒为 0 的曲线称为**测地线**(geodesic)。由其定义易知，若曲面上存在直线，则它必为测地线。测地线在等距映射下的像也是测地线。另外，曲面上两点之间的测地线长度是两点间所有曲线长度的极值。对于球面，两点间的测地线就是连接两个点的大圆弧（劣弧和优弧都是）。

若D是曲面S上的单连通区域，其边界是封闭的分段光滑曲线，则有 **Gauss-Bonnet 公式**：

$$\iint_D KdA + \int_{\partial D} k_g\, ds + \sum \alpha_i = 2\pi,$$

其中K是高斯曲率，α_i是D的边界上非光滑点的外角。特别地，对于曲面上的测地三角形，

$$\iint_D KdA = \beta_1 + \beta_2 + \beta_3 - \pi,$$

其中$\beta_1, \beta_2, \beta_3$是三个内角。对于球面，由于其高斯曲率恒为 1，上述公式表明球面测地线（大圆弧）三角形的面积就是$\beta_1 + \beta_2 + \beta_3 - \pi$。

5. 曲面上的函数及拉普拉斯算子

给定曲面S上的正交坐标系$\{\boldsymbol{e}_1, \boldsymbol{e}_2\}$和切向量场$\boldsymbol{v}$，其微分在切平面上的投影称为其**协变微分**(covariant differential)：

$$D\boldsymbol{v} = (d\boldsymbol{v} \cdot \boldsymbol{e}_1)\boldsymbol{e}_1 + (d\boldsymbol{v} \cdot \boldsymbol{e}_2)\boldsymbol{e}_2.$$

协变微分是线性微分算子。

给定曲面S上的函数ϕ，其微分可表示为S上一阶微分形式的组合：

$$d\phi = \phi_1\omega_1 + \phi_2\omega_2.$$

函数ϕ的**梯度**(gradient)定义为

$$\nabla\phi = \phi_1\boldsymbol{e}_1 + \phi_2\boldsymbol{e}_2,$$

它与坐标系的选取无关。其协变微分为

$$D(\nabla\phi) = (d\phi_1 + \phi_2\omega_{21})\boldsymbol{e}_1 + (d\phi_2 + \phi_1\omega_{12})\boldsymbol{e}_2,$$

记为$D\phi_1\boldsymbol{e}_1 + D\phi_2\boldsymbol{e}_2$。注意到$D\phi_1, D\phi_2$也是一阶微分形式，因此也可写成$\omega_1, \omega_2$的组合：

$$D\phi_1 = d\phi_1 + \phi_2\omega_{21} = \phi_{11}\omega_1 + \phi_{12}\omega_2,$$

$$D\phi_2 = d\phi_2 + \phi_1\omega_{12} = \phi_{21}\omega_1 + \phi_{22}\omega_2,$$

由$\phi_{11}, \phi_{12}, \phi_{21}, \phi_{22}$组成的二阶矩阵称为$\phi$关于$\{\boldsymbol{e}_1, \boldsymbol{e}_2\}$的 **Hessian 阵**。特别地，该

矩阵的迹$\phi_{11}+\phi_{22}$与坐标系的选取无关，记为$\Delta_S\phi$。算子Δ_S称为曲面的 **Laplace-Beltrami 算子**，简称拉普拉斯算子。特别地，可以证明，

$$\Delta_S \boldsymbol{f} = -2H\boldsymbol{n}.$$

对于一般的参数，曲面的拉普拉斯算子可由如下公式计算：

$$\Delta_S\phi = \frac{1}{\sqrt{g}}\frac{\partial}{\partial u^\beta}\left(\sqrt{g}g^{\alpha\beta}\frac{\partial\phi}{\partial u^\alpha}\right),$$

其中(u^1,u^2)是参数，$g^{\alpha\beta}$是曲面第一基本形式的矩阵元素，g是矩阵的行列式。上述公式使用了爱因斯坦求和约定。

三、 曲面的整体理论[1]

1. 可定向性

设S是一个曲面，$\mathfrak{U}=\{(U_\alpha,\phi_\alpha)\}$是它的一个图集。对于每一个图表$(U_\alpha,\phi_\alpha)$，选取一个单位法向量$\boldsymbol{n}_\alpha$作为其定向。若存在一种选法使得

$$\boldsymbol{n}_\alpha=\boldsymbol{n}_\beta,\qquad \forall U_\alpha\cap U_\beta\neq\emptyset,$$

则称该曲面是**可定向的**(orientable)。可定向曲面有整体良定的单位法向量，从而有整体良定的第二基本形式。常见的不可定向曲面有莫比乌斯环，克莱因瓶等。

2. 曲面的三角剖分

设S是一个曲面，$\mathfrak{U}=\{(U_\alpha,\phi_\alpha)\}$是它的一个图集。曲面上的**三角形**(triangle)T是指某个U_α内由曲面上三条光滑曲线所围的单连通区域。曲面的**三角剖分**(triangulation)是指曲面上的一族三角形$\{T_\lambda\}$，满足

1）曲面上的任意点p必属于某一三角形。若p是某个三角形的内点，则该三角形唯一；

2）每条边属于至多两个三角形；两个三角形至多共一条边。若两个三角形共边，则它们也共边的两个端点；

3）每个顶点属于至多有限个三角形。

可以证明，曲面上具有分段光滑边界的有限区域必有三角剖分。

3. 欧拉示性数

设S是一个曲面，$\{T_\lambda\}$是它的一个三角剖分。将其中的顶点数，边数，三角形数分别记为V, E, F，则欧拉公式表明$V-E+F$是拓扑不变量，记为$\chi(S)$，称为曲面的**欧拉示性数**(Euler characteristic)。它等于$2-2g$，其中g是曲面的**亏格**(genus)，它刻画了曲面上洞的个数。

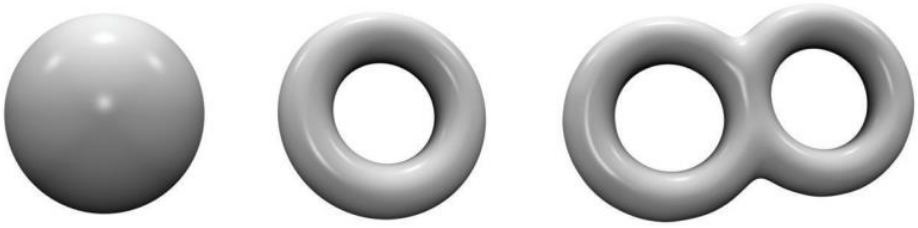


图 1.2 不同亏格的曲面[3]

设曲面S是可定向的无边紧致曲面，则有整体 Gauss-Bonnet 公式：

$$\int_S KdA = 2\pi\chi(S).$$

曲面分类定理表明，若曲面的亏格为 0，则同胚于球面；若曲面的亏格不为 0，则同胚于同亏格数的环面。因此，闭曲面无法整体参数化到平面。

第二节　三角网格曲面

一、　实际应用中的曲面表示

在数学里，常被研究的曲面主要有三种：

1）显式曲面：$z = f(x,y)$.

2）参数曲面：$(x,y,z) = \big(x(u,v), y(u,v), z(u,v)\big), (u,v) \in \Omega \subseteq \mathbb{R}^2$.

3）隐式曲面：$\phi(x,y,z) = 0$.

其中显式曲面可以看成特殊的参数曲面。这些曲面的共同特点就是表达非常简洁直观，且其中涉及的函数往往可以假设一定的光滑性。因此，常用的分析工具可以很好的研究这些曲面的几何性质，从而推出各种美妙的理论。

然而，生活中的曲面复杂多样，往往很难用直观的函数来表示。而由于计算机的存储方式是离散的，对于一般曲面不存在能够完整存储其所有信息的数据结构，因此只能用有限的采样信息来近似表示。三角网格是一种对一般曲面简单而

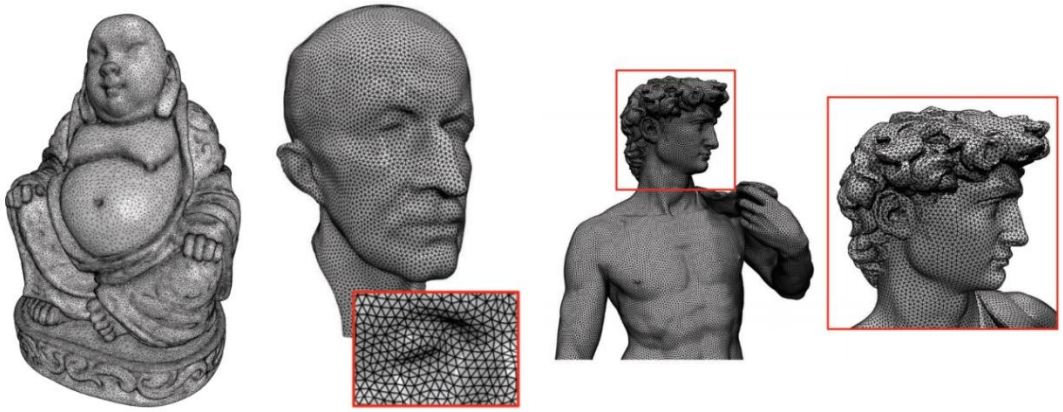


图 1.3 三角网格曲面[3]

有效的线性近似。更具体地，一个**三角网格**(triangle mesh)M由一个图$G=(V,E,F)$和一个坐标映射$X:V\to\mathbb{R}^3$组成，其中V是一组顶点，E是连接这些顶点的一组边，F是由这些顶点和边组成的一组三角形，它们反映了网格内的拓扑关系。为了能构成一般可定向曲面的一个三角剖分，它们需要满足以下条件：

1) 每个顶点应属于至少三条边，两个顶点间至多有一条边相连；

2) 每条边应属于至少一个三角形，至多两个三角形；

3) 两个三角形间至多共一条边，至多共两个顶点；

4) 所有三角形之间应存在一种相容的定向。

坐标映射X将每个顶点v映为它在空间中的位置$\boldsymbol{x}_v$，反映了网格的几何信息。于是，以v_1,v_2,v_3为顶点的三角形$f\in F$对应三维欧氏空间中的一个三角形面：

$$(v_1,v_2,v_3)\leftrightarrow\left\{\lambda_1\boldsymbol{x}_{v_1}+\lambda_2\boldsymbol{x}_{v_2}+\lambda_3\boldsymbol{x}_{v_3}\mid\lambda_1,\lambda_2,\lambda_3\geq 0,\lambda_1+\lambda_2+\lambda_3=1\right\}.$$

这些三角形用以线性近似相应部分的曲面。为了使得这些三角形能拼成一个流形曲面，这个坐标映射应该满足：

1）每个顶点的 1-邻域(即其属于的所有三角形的并集)应同胚于单位闭圆盘；

2）任意两个三角形除了点和边以外不能有其他交点。

最终，将这些三角形拼起来，我们得到一个分片线性映射作为曲面的近似。可以证明，通过增加网格的顶点数，我们可以对一般可定向流形曲面进行任意精度的近似。

实际应用中的曲面通常来自两种渠道：计算机辅助设计(CAD)与 3D 扫描。在计算机辅助设计中，工程师或艺术家们用各种 3D 建模软件（如 Auto CAD, 3ds Max, Maya, Cinema 4D, Blender, Houdini 等）来创造物体模型。这些模型经常由一些基本的元素（如长方体，球体，圆柱体，三角形，贝塞尔曲线等）组成，因此导出的模型通常以样条曲面或多边形网格存储。在 3D 扫描中，扫描器通过扫描物体的表面得到密集的采样点。但这些采样点不足以反映曲面的任何整体信息，因此需要构建点与点间的连接关系，这一过程称为**网格建立**(meshing)。通过网格建立，采样点就变为了三角形网格。

在几何处理算法中，我们往往需要访问一条边的两个端点或两个邻接三角形，遍历一个顶点所连的所有边，三角形，或顶点，因此使用半边数据结构来高效进行这些操作。在半边数据结构中，每条边被分为两个方向相反的有向半边，分别属于两个邻接的三角形。每个三角形内的半边取向为逆时针。由于所近似的曲面是可定向的，这样的分法总是存在。在这里，我们使用 OpenMesh[4]提供的三角网格类作为网格的基本数据结构，并用它提供的 API 进行三角网格中的各种遍历操作。

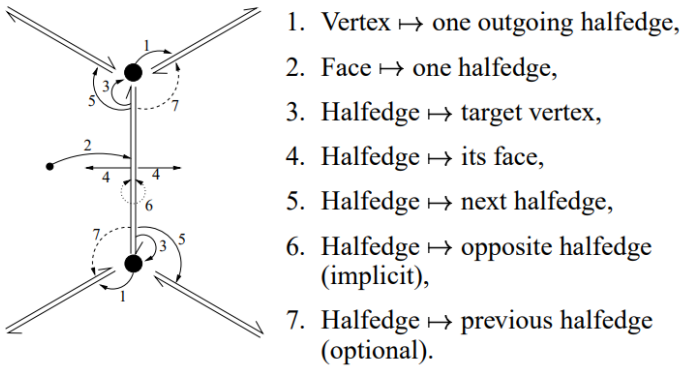


图 1.4 半边数据结构[4]

二、 离散微分几何

有了三角网格的数据结构作为光滑曲面的近似，我们便可以计算一些几何量。然而，由于微分几何中的几何量都是在假设曲面充分光滑的前提下定义的，经常涉及到对曲面映射的高阶导数。而三角网格曲面是分片线性的，每个三角形内的一阶导数为常数，高阶导数则均为 0，显然与原曲面的几何信息不符，因此无法直接将光滑设定下的概念用于实际计算，而需要做一定的离散化处理。**离散微分几何**(Discrete Differential Geometry)正是为此而设计的工具，它旨在用简单的方法在三角网格曲面上计算诸如切平面，法向量，面积元，曲率，测地线等各种几何量，使得能对原曲面相应的几何量做很好的估计。在这里，我们只介绍一些与课题相关的几何量的计算方法，其余可以参考[6][3][22]。

1. 基于三角形的几何量：

   设三角形$f=\{f_0,f_1,f_2\}$的三个顶点位置按逆时针分别为$\boldsymbol{x}_{f_0},\boldsymbol{x}_{f_1},\boldsymbol{x}_{f_2}$。

   1) 面积：$|f|=\frac{1}{2}\left\|\left(\boldsymbol{x}_{f_1}-\boldsymbol{x}_{f_0}\right)\times\left(\boldsymbol{x}_{f_2}-\boldsymbol{x}_{f_0}\right)\right\|$；

   2) 法向量：$\boldsymbol{n}_f=\frac{(\boldsymbol{x}_{f_1}-\boldsymbol{x}_{f_0})\times(\boldsymbol{x}_{f_2}-\boldsymbol{x}_{f_0})}{\|(\boldsymbol{x}_{f_1}-\boldsymbol{x}_{f_0})\times(\boldsymbol{x}_{f_2}-\boldsymbol{x}_{f_0})\|}$；

   3) 重心：$\boldsymbol{x}_f=\frac{1}{3}\left(\boldsymbol{x}_{f_0}+\boldsymbol{x}_{f_1}+\boldsymbol{x}_{f_2}\right)$。

2. 基于顶点的几何量：

   1) 面积元：$\sigma_i=\frac{1}{3}\sum_{i\in f}|f|$；

   2) 顶点法向量：$\boldsymbol{n}_i=\frac{\sum_{i\in f}|f|\boldsymbol{n}_f}{\left\|\sum_{i\in f}|f|\boldsymbol{n}_f\right\|}$。

3. 整体几何量：

   1) 欧拉示性数：$\chi(M)=|V|-|E|+|F|$；

   2) 亏格：$g(M)=\frac{1}{2}\left(2-\chi(M)\right)$；

   3) 整体面积：$A(M)=\sum_{f\in F}|f|$；

   4) 整体重心：$\boldsymbol{x}_c(M)=\frac{1}{A(M)}\sum_{f\in F}|f|\boldsymbol{x}_f$。

4. 网格上的函数

一个三角网格上的函数由各顶点处的值给定，其在每个三角形内的值由三个顶点处的值线性插值得到。对于一个在网格顶点上定义的函数f，设它在顶点$i \in V$处的值为$f_i \in \mathbb{R}$。

1) 函数的积分：$\int_M f d\sigma = \sum_{i \in V} \sigma_i f_i$；

2) 函数的拉普拉斯：$\Delta f_i = \frac{1}{\sigma_i} \sum_{(i,j) \in E} \omega_{ij} (f_j - f_i)$，其中

$$\omega_{ij} = \frac{1}{2} (\cot \alpha_{ij} + \cot \beta_{ij})$$

称为**余切权重**(cotangent weight)。具体的推导可以参考[3][6]。由余切权

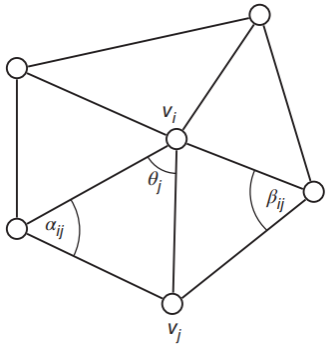


图 1.5 余切权重[22]

重组成的矩阵

$$L(M) = (L_{ij})_{i,j \in V}, \qquad L_{ij} = \begin{cases} \sum_{(i,k) \in E} \omega_{ik}, & i = j \\ -\omega_{ij}, & (i,j) \in E \\ 0, & \text{otherwise} \end{cases}$$

称为曲面的**拉普拉斯矩阵**。它是一个半正定的矩阵：

$$\sum_{i,j \in V} L_{ij} f_i f_j = -\sum_{i \in v} \sigma_i f_i \Delta f_i = -\int_M f \Delta f d\sigma = \int_M |\nabla f|^2 d\sigma \geq 0.$$

曲面的拉普拉斯矩阵在几何处理中非常有用，后面几章即将介绍的几何优化中也将反复用到这个矩阵。

第三节　网格参数化问题

一、　问题概述

对于光滑可定向曲面，其上每个点的邻域都可以用平面区域通过一个参数映射表示。因此，光滑曲面的每一个局部区域都是参数曲面。反过来，给定一个局部区域寻找一个合适的参数域和参数映射便是**参数化**(parameterization)问题。更具体地，对于曲面S上的一个区域U，参数化问题希望找到一个合适的参数域Ω和一个同胚映射$\phi: U \simeq \Omega$。若U只是局部区域，则参数域可选为平面区域；若U就

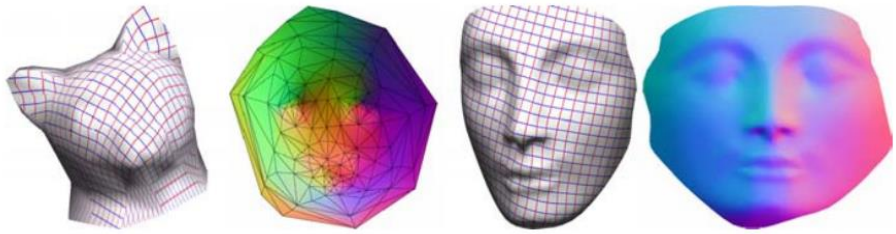


图 1.6 网格的平面参数化[39]

是整个曲面S，则参数域需要与曲面同胚，其选取需要考虑曲面的整体性质和一些拓扑量，比如亏格数与开闭性，相应的参数化问题称为**整体参数化**(global parameterization)问题。特别地，对于开曲面，其整体参数域仍可选为平面区域；而对于 0 亏格的封闭曲面，其整体参数域通常选为单位球面。

在计算机图形学中，网格参数化是最基本的几何处理问题之一[39][10]。各种应用，包括纹理贴图，网格再划分，网格变形，曲面间映射，几何图像等都需要网格参数化作为基本工具。由于在实际应用中网格通常来自对物体表面的扫描，因此通常是封闭的。一种办法是把它分割成一片或多片开网格，然后把这些网格



图 1.9 闭曲面网格的切割[34]

分别参数化到平面区域。其优势在于有大量的高质量平面参数化算法可供选择，其劣势在于网格分割导致被分割的区域在割线处参数的不连续，从而会在原网格上产生可见的**缝**(seam)。由于某些应用对这种不连续性较为敏感，因此为了满足其实际需求则需要添加一些边界处的**无缝约束**(seamless constraints)，这些约束难以处理。与此相对，另一种方法是对网格进行整体参数化，这样就不会出现上述不连续性。其劣势在于可供选择的整体参数化算法比较少。

对于三角网格曲面上的函数，若其值域是线性空间，则一般可用顶点处的函数值通过重心坐标线性插值得到每个三角形边界和内部的点处的函数值的近似：

$$\phi\left(\lambda_0 \boldsymbol{x}_{f_0} + \lambda_1 \boldsymbol{x}_{f_1} + \lambda_2 \boldsymbol{x}_{f_2}\right) \approx \lambda_0 \phi\left(\boldsymbol{x}_{f_0}\right) + \lambda_1 \phi\left(\boldsymbol{x}_{f_1}\right) + \lambda_2 \phi\left(\boldsymbol{x}_{f_2}\right).$$

因此，求解一个平面参数化问题只需求解各个顶点处的参数坐标就足够了。在这个意义下，参数化映射可看成是网格到平面的一个**形变**(deformation)，这个形变是同胚的充要条件是：

1) 局部双射性：映射在任意一个局部区域（即任意两个共边的三角形）内是双射。这等价于任意一对相邻三角形形变后的定向是相同的，也等价于无三角形翻转。
2) 整体双射性：映射在整个待参数化区域上是双射。这等价于待参数化区域的边界形变后是不自交的。

满足这两个条件的参数化映射才能称为**合法的**(valid)。网格参数化问题的基本要求便是计算这样一个合法的参数化映射。平面参数域的优势在于其线性性，这使得每个顶点与周围顶点的参数关系都是线性的，由此诞生了一大类线性方法。若

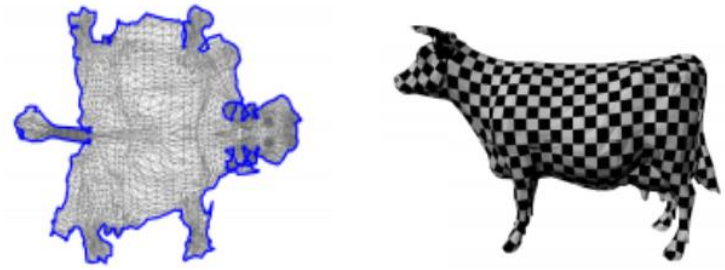


图 1.7 平面参数化与纹理贴图[43]

从形变的角度来看，可以把整个参数化映射看成一个$2|V|$维的向量，每个顶点的参数坐标都是它的分量。于是，平面的线性性也使得各顶点的两个参数坐标分量可以线性变化，只受局部双射性的约束，由此也诞生了一大类基于直线搜索的迭代方法。基于以上原因，平面参数化得到了广泛研究。

对于 0 亏格封闭三角网格的整体参数化，由于其参数域为单位球面，因此无法通过顶点处的参数线性插值得到三角形内的参数，需要提前指定一种从平面三角形到球面三角形的同胚映射。这类映射称为**球面三角形映射**(spherical triangle map)，会在第三章详细介绍。一旦给定了球面三角形映射，问题便也化为计算各顶点处的参数坐标。由于这时的网格没有边界，因此参数化映射只需要满足局部双射性便足够了，即无球面三角形翻转。相比于平面参数域，球面带来的非线性约束导致球面参数化的计算更加困难：平面的线性方法不再适用，相应的迭代方法也难以照搬。因此球面参数化的研究热度比平面参数化低许多。



图 1.8 球面参数化与纹理贴图[19]

二、 扭曲能量

为了定量描述参数化映射在整个三角网格上的扭曲，我们首先需要定义其在每个三角形上的扭曲。给定三角网格$M = (V, E, F)$和一个平面参数化映射$\boldsymbol{x}$，记顶点$v \in V$的几何位置为$\bar{\boldsymbol{x}}_v \in \mathbb{R}^3$，记其参数坐标为$\boldsymbol{x}_v \in \mathbb{R}^2$。给定三角形$f = (f_0, f_1, f_2) \in F$，我们通过刚体变换将以$\bar{\boldsymbol{x}}_{f_0}, \bar{\boldsymbol{x}}_{f_1}, \bar{\boldsymbol{x}}_{f_2}$为顶点的三角形变换到平面，使得$\bar{\boldsymbol{x}}_{f_0}$被映为原点，$\bar{\boldsymbol{x}}_{f_1}$被映到$x$轴正半轴，$\bar{\boldsymbol{x}}_{f_2}$被映到上半平面：

$$\left(\bar{\boldsymbol{x}}_{f_0}, \bar{\boldsymbol{x}}_{f_1}, \bar{\boldsymbol{x}}_{f_2}\right) \mapsto \left(\boldsymbol{u}_{f_0}, \boldsymbol{u}_{f_1}, \boldsymbol{u}_{f_2}\right) = \begin{pmatrix} 0 & u_{f_1}^x & u_{f_2}^x \\ 0 & 0 & u_{f_2}^y \end{pmatrix},$$

其中

$$\begin{cases} u_{f_1}^x = \|\overline{\boldsymbol{x}}_{f_1} - \overline{\boldsymbol{x}}_{f_0}\| \\ u_{f_2}^x = (\overline{\boldsymbol{x}}_{f_1} - \overline{\boldsymbol{x}}_{f_0}) \cdot (\overline{\boldsymbol{x}}_{f_2} - \overline{\boldsymbol{x}}_{f_0}) / u_{f_1}^x \\ u_{f_2}^y = \|(\overline{\boldsymbol{x}}_{f_1} - \overline{\boldsymbol{x}}_{f_0}) \times (\overline{\boldsymbol{x}}_{f_2} - \overline{\boldsymbol{x}}_{f_0})\| / u_{f_1}^x \end{cases}.$$

于是原三角形与由新的坐标$\boldsymbol{u}_{f_0}, \boldsymbol{u}_{f_1}, \boldsymbol{u}_{f_2}$组成的三角形全等。这样做的目的是为了把原网格中的三角形所处的空间从三维降到二维，从而方便之后计算每个三角形的形变。我们把摊平后的三角形称为**源三角形**(source triangle)。由于我们采用半

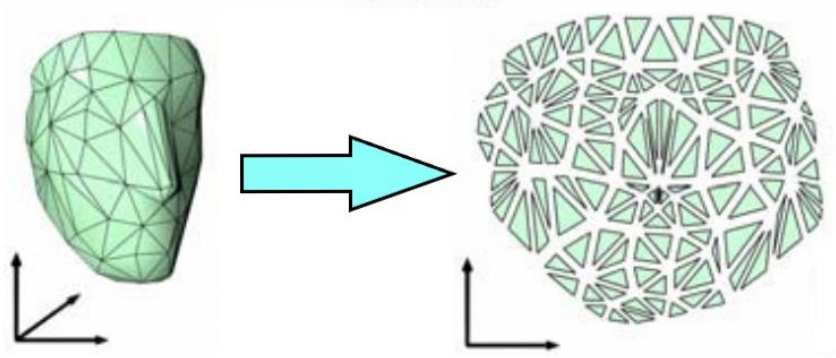


图 1.10 将网格中的三角形摊平得到源三角形[24]

边数据结构，各源三角形的定向与网格本身的定向一致。从源三角形到三角形的参数化像存在唯一一个仿射变换：

$$\boldsymbol{x}_{f_i} = J_f \boldsymbol{u}_{f_i} + \boldsymbol{x}_{f_0}, \qquad i = 1, 2, 3,$$

其中

$$J_f = (\boldsymbol{x}_{f_1} - \boldsymbol{x}_{f_0}, \boldsymbol{x}_{f_2} - \boldsymbol{x}_{f_0})(\boldsymbol{u}_{f_1}, \boldsymbol{u}_{f_2})^{-1}$$

是这个映射的梯度，称为三角形f的**形变梯度**(deformation gradient)，它刻画了该三角形的形变程度。当形变梯度为正交阵时，像三角形与源三角形全等，此时三角形f无扭曲。当形变梯度的行列式为负时，像三角形的定向与源三角形不一致，此时三角形f翻转；当形变梯度的行列式为 0 时，像三角形退化成线段。因此三角形f不翻转的充要条件是：

$$\det J_f > 0,$$

这等价于

$$\det(\boldsymbol{x}_{f_1} - \boldsymbol{x}_{f_0}, \boldsymbol{x}_{f_2} - \boldsymbol{x}_{f_0}) > 0.$$

设形变梯度的奇异值分解为：

$$J_f = U_f \operatorname{diag}\left(\sigma_{1,f}, \sigma_{2,f}\right) V_f^T,$$

其中U_f, V_f是正交阵，$\sigma_{1,f}, \sigma_{2,f}$是形变梯度的奇异值，它们刻画了这个仿射变换各方向伸缩系数的极值。记

$$R_f = U_f V_f^T, \qquad S_f = V_f \operatorname{diag}\left(\sigma_{1,f}, \sigma_{2,f}\right) V_f^T,$$

则$J_f = R_f S_f$是形变梯度的极分解。由上述定义可以看出，S_f是对称阵，称为三角形f的**伸缩张量**(stretch tensor)，它的几何意义是沿两个互相正交的主方向的伸缩。R_f是正交阵，它的几何意义是刚体变换，对三角形的形变没有贡献。可以证明，R_f是所有正交阵中与J_f的 Frobenius 距离最小的一个。换一种说法，R_f是J_f在正交空间中的投影。于是我们可以自然地用J_f到R_f的 Frobenius 距离来描述三角形f的扭曲，称为 ARAP(As-Rigid-As-Possible)扭曲函数[24]：

$$D_{ARAP}(J_f) = \frac{1}{2}\left\|J_f - R_f\right\|_F^2.$$

注意到，这个函数也可以由形变梯度的两个特征值表示：

$$D_{ARAP}(J_f) = \frac{1}{2}\left(\sigma_{1,f} - 1\right)^2 + \frac{1}{2}\left(\sigma_{2,f} - 1\right)^2.$$

显然，当形变梯度为正交阵时，该扭曲函数取到最小值 0。将各三角形的 ARAP 扭曲通过面积加权，我们得到整个网格的 ARAP 扭曲能量[24]：

$$E_{ARAP}(M, \boldsymbol{x}) = \frac{1}{A(M)} \sum_{f \in F} |f| D_{ARAP}(J_f).$$

类似地，我们还可以通过其他方式定义每个三角形的扭曲，然后通过面积加权得到整个网格的扭曲能量。常用的扭曲函数有：

1) 对称狄利克雷(symmetric Dirichlet)扭曲[43]：

$$D_{symDir}(J_f) = \frac{1}{2}\left(\left\|J_f\right\|_F^2 + \left\|J_f^{-1}\right\|_F^2\right) = \frac{1}{2}\left(\sigma_{1,f}^2 + \sigma_{2,f}^2 + \frac{1}{\sigma_{1,f}^2} + \frac{1}{\sigma_{2,f}^2}\right),$$

该函数当J_f是正交阵时取得最小值 2。由于这个扭曲函数形式简单，效果

较好，因此一般是参数化问题中的首选目标函数。后文中也主要将对称狄利克雷扭曲作为目标函数。

2) 保角扭曲(MIPS 扭曲[18])：

$$D_{MIPS}(J_f)=\frac{1}{2}\frac{\|J_f\|_F^2}{\det J_f}=\frac{1}{2}\left(\frac{\sigma_{1,f}}{\sigma_{2,f}}+\frac{\sigma_{2,f}}{\sigma_{1,f}}\right),$$

该函数当$\sigma_{1,f}=\sigma_{2,f}$时取得最小值 1。通过优化这个扭曲可以得到保角的参数化映射。

3) 保面积扭曲：

$$D_{area}(J_f)=\frac{1}{2}\left(\det J_f+\frac{1}{\det J_f}\right)=\frac{1}{2}\left(\sigma_{1,f}\sigma_{2,f}+\frac{1}{\sigma_{1,f}\sigma_{2,f}}\right),$$

该函数当$\det J_f=1$时取得最小值 1。通过优化这个扭曲可以得到保面积的参数化映射。

4) 等距扭曲[11]：

$$D_{iso}(J_f)=\alpha D_{MIPS}(J_f)+(1-\alpha)D_{area}(J_f),\qquad 0<\alpha<1.$$

该函数是保角扭曲与保面积扭曲的组合，当J_f是正交阵时取得最小值 1，且可由参数α调整对面积或角的惩罚力度。

5) AMIPS(Advanced MIPS)扭曲[11]：

$$D_{iso}^{*}(J_f)=\exp\left(sD_{MIPS}(J_f)+sD_{area}(J_f)\right).$$

该函数可以有效惩罚扭曲较大的三角形。

三、 低扭曲参数化的问题表述

最终，我们将低扭曲参数化问题表述为一个以参数化映射为变量，以其双射性为约束，以其在整个曲面上的扭曲能量为目标函数的优化问题：

$$\boldsymbol{x}^{*}=\underset{\boldsymbol{x}}{\operatorname{argmin}}\,E(M,\boldsymbol{x}),$$

$$\text{s.t.}\det(\boldsymbol{x}_{f_1}-\boldsymbol{x}_{f_0},\boldsymbol{x}_{f_2}-\boldsymbol{x}_{f_0})>0,\forall f,$$

其中

$$E(M,\boldsymbol{x}) = \frac{1}{A(M)}\sum_{f\in F}|f|D(J_f).$$

这是一个开集约束下的非线性优化问题。注意到，这里我们并没有加入整体双射性约束，因为这种约束可以合并到能量函数中，详情可参考[43][44]。另外，对于低扭曲球面参数化问题，其能量和约束条件都略有不同，具体在第三章中给出。由于这类非线性优化问题的目标函数和约束形式复杂，但却有着独特的几何结构，因此人们结合几何知识对常用的优化算法进行修改，得到了可以有效降低这类函数且结果严格满足约束的算法，称为**几何优化**(geometric optimization)方法。

第四节　几何优化方法简介

一、　一般非线性优化[26][27]

在非线性优化理论里，一般的非线性优化问题往往通过从某个初始点开始经过多次迭代一步一步收敛到极值点来求解。对于无约束的凸优化问题，常用的优化方法包括梯度下降法和牛顿法。

梯度下降法(gradient descent)将目标函数展开到一阶项，计算其下降最快的方向（即梯度的负方向），因此也称为最速下降法。但由于其只利用了目标函数的一阶导数信息，属于一阶方法，故其收敛速度是线性的。常用的加速方法有预处理法，Nesterov 加速[25]和 Anderson 加速[2]。**预处理法**(preconditioning)通过一个**预处理器**(preconditioner)在进行迭代前对梯度做一个线性变换来得到更好的下降方向。其本质是在解空间里定义一种新的黎曼度量来替代原本的欧氏度量，在这个新的度量下目标函数的负梯度方向就是预处理后下降方向。Nesterov 加速通过定义动量把迭代途中计算过的所有负梯度进行累积，以此作为下降方向。其本质是在模拟物理运动过程，把目标函数解读成一个势能，于是其负梯度相当于保守力，动量是力随时间的累积，而动量则决定了当前的运动方向。Anderson 加速结

合前面若干个迭代点的信息对新的迭代点进行改进，其本质是把每次的新迭代点看成前一个迭代点的函数，从而根据前面若干个迭代点的信息对该函数的不动点进行估计。另外，选取合适的步长可以使每次迭代尽可能地降低目标函数，也可对整个迭代过程起到加速作用。对于规模较大的问题，还可以使用**随机梯度下降**(stochastic gradient descent)或者**批量梯度下降**(batch gradient descent)以避免计算整个梯度。这些加速方法虽然都能减少收敛所需的时间，但改变不了线性收敛速度的本质。

牛顿法(newton's method)将目标函数展开到二阶项，计算这个二阶近似项的最小值点作为下一个迭代点。由于它充分利用了目标函数的二阶导数信息，属于二阶方法，故可以达到超线性的收敛速度。牛顿法也可以看成是以目标函数的 Hessian 矩阵为预处理器的预处理法。牛顿法的缺点在于每次计算新迭代点耗时较多。尽管凸函数的 Hessian 阵是正定对称阵，可以用 LLT 分解等特殊方法对下降方向的求解进行加速，但计算 Hessian 阵和其二次规划问题终究是一件耗时的事。常用的加速方法是避免计算 Hessian 阵而利用梯度信息直接对 Hessian 阵的逆矩阵进行估计，这类方法称为**拟牛顿**(Quasi-Newton)方法。

对于带约束的优化问题，我们可以根据约束的类型对上述方法进行修改，其整体思路是从一个满足约束的起始点出发，在迭代过程中对约束进行维护。线性的等式约束可以写成目标函数变量的线性方程组，只要将每次的下降方向投影到系数矩阵的零空间便可以维护这个约束。对于一般的流形约束，可以采用流形上的参数坐标，或者先将梯度投影到流形的切空间上，然后用流形上的曲线搜索来代替直线搜索。对于不等式约束，若解空间中满足该约束的集合是闭集，则可通过某种方式将每次的迭代点投影到这个闭集上；若是开集，则一般采用**内点法**(interior point method)，即每次计算一个不破坏约束的最大步长，然后利用回溯搜索决定一个合适的下降步长。

对于非凸优化问题，梯度下降法仍然适用，但牛顿法却可能导致目标函数增加或者迭代点发散，这是因为目标函数的 Hessian 阵是不定的。常用的方法是用一个正定或半正定的矩阵来代替 Hessian 阵，类似预处理法。典型的代替方法包括在 Hessian 阵上加上一个正数λ倍的单位阵，其本质是在目标函数上加上一个正则项，其结果是在 Hessian 阵的所有特征值上加了λ，因此λ至少要取为 Hessian 阵的最小负特征值的绝对值。这种方法的缺点在于收敛所需的迭代数会大大增加，且λ越大增加得越多。另一种方法是找到与 Hessian 阵最接近的半正定阵，称为投影牛顿法(projected Newton)，其具体做法是将 Hessian 阵的所有负特征值都修改为 0，因此需要计算 Hessian 阵的特征值分解。这种方法的缺点在于对于大规模问题，Hessian 阵太大，计算其特征值分解非常耗时，若提前知道特征值和特征向量的解析表达式则可带来很好的加速。一些拟牛顿方法可以保证即使对非凸函数也能产生半正定的求解器，于是就避免了求特征值分解和计算之后的二次规划问题，可以大大降低单次迭代所需要的时间。但是这种方法得到的半正定近似往往不如投影牛顿法，因此需要更多的迭代数达到收敛。

二、 几何优化

在上一节中，我们已经将低扭曲无翻转参数化问题表述成了一个带约束的优化问题。其目标函数是一个高度非凸函数，其约束是非线性的非凸开集约束，这导致求解优化问题变得困难。然而，目标函数的特点在于它是由局部的扭曲函数在各个三角形上的值按照面积加权求和得到的，约束也是对每个三角形分别约束的，而局部问题的规模较小，且蕴含着很多几何信息，于是基于这些特点可以设计出一类特殊的求解方法，称为几何优化方法。由于这样的目标函数与约束广泛存在于形变，曲面间映射等各种图形学问题中，因此几何优化是图形学最重要的工具之一。下面我们将简单描述求解上一节提出的几何优化问题的大致流程。

首先，我们需要找到一个满足约束的初始化。这意味着我们要计算一个双射

的参数化映射。幸运的是，对于平面参数化，我们可以用线性方法，通过求解一个稀疏线性方程组来得到这样的初始化。在迭代过程中，由于目标函数是非凸函数，我们可以使用预处理法，投影牛顿法或者拟牛顿法来计算下降方向。我们先对每个三角形计算求解器，然后将他们集成为整体的求解器。这样的求解器很好地保留了问题的稀疏特征，使得求解相应的二次规划问题难度降低。为了维护无翻转约束，我们需要计算一个最大步长，然后用回溯直线搜索来获得一个合适的步长。由于无翻转约束是各个三角形的局部约束的并集，而局部约束的边界是三角形参数坐标的二次多项式，因此可以通过求解二次方程得到每个三角形的最大步长，然后取各三角形最大步长的最小值作为整体的最大步长。这样一来，每次迭代都可以保证没有三角形翻转，得到的收敛结果也必然是无翻转的。

然而，实际测试中有时会在迭代的过程中卡住，无论用什么求解器都无法有效地降低目标函数。这往往是因为某些三角形接近翻转，导致整体的最大步长非常小。如果目标函数选取得不合适，便很容易导致这种现象发生。比如上一节介绍的 ARAP 扭曲函数，它在伸缩系数（即形变梯度的奇异值）较大的三角形处的值很大，能有效抑制这类三角形；但在伸缩系数接近 0 的三角形处的值较小，导致三角形容易接近翻转，这时就会出现最大步长过小的现象。为了避免这种情况，需要局部扭曲函数对伸缩系数接近 0 的三角形也要有足够大的抑制。上一节提到的对称狄利克雷扭曲和 AMIPS 扭曲满足这个要求，因此逐渐替代了 ARAP 扭曲。另外，一些特殊的技巧也可以用来避免最大步长较小的情况，比如在源三角形与像三角形之间插值一个中间三角形代替源三角形作为参考三角形，或者利用约束条件的形式以原本的搜索方向作为参考构造一个最大步长尽可能大的搜索方向。

由此可见，平面参数化的几何优化方法已被广泛研究，已经有了比较完善的流程，且有许多用来加速的办法。与此相反，球面参数化的几何优化方法却很少被研究，就连最基本的无翻转初始化都很难获得。因此，本课题旨在将平面参数

化的几何优化流程推广到球面参数化问题中，以高效获得低扭曲的无翻转球面参数化映射。为此，我们首先要理解并实现平面参数化的各种方法，研究现有球面参数化方法中出现的问题，然后想办法解决这些问题。后面的章节内容大致如下：第二章将详细讲述平面参数化几何优化方法的具体细节，第三章将调查现有的球面参数化方法，并用球心投影作为桥梁将平面中的几何优化方法应用于球面参数化问题的求解。最后，第四章将进行总结并指出未来的可研究方向。

第二章　平面参数化的几何优化方法

第一节　Tutte 方法及其变种

1963 年，W. T. Tutte 在一篇图论的文章[46]中提到如何将一个 2-联通图嵌入到平面中，其大致流程是先将图的边界映射到一个平面区域的边界上，然后通过求解一个线性方程组将其余顶点映到区域的内部。Tutte 证明了只要平面区域的边界是凸的，这种方法可以保证嵌入后的图里边与边两两不相交。之后 Michael S. Floater 在[8]中将这种方法应用到了有边三角网格的平面参数化中，成功得到了无翻转的平面参数化映射。Floater 还将这种方法进行了推广，提出一类线性的无翻转平面参数化方法[9]。下面我们将介绍这类方法。

给定一个三角形，其内部的每一个点都可以用重心坐标表示：

$$\boldsymbol{x} = \lambda_1 \boldsymbol{x}_1 + \lambda_2 \boldsymbol{x}_2 + \lambda_3 \boldsymbol{x}_3, \qquad \lambda_1, \lambda_2, \lambda_3 \geq 0, \lambda_1 + \lambda_2 + \lambda_3 = 1.$$

特别地，某点在三角形内部当且仅当$\lambda_1, \lambda_2, \lambda_3 > 0$。对于平面凸多边形，也有类似的结果：设平面凸$n$边形的$n$个顶点位置分别为$\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_n$。若某点$\boldsymbol{x}$可表示为这些顶点的凸组合，即

$$\boldsymbol{x} = \lambda_1 \boldsymbol{x}_1 + \lambda_2 \boldsymbol{x}_2 + \cdots + \lambda_n \boldsymbol{x}_n, \qquad \lambda_1, \lambda_2, \cdots, \lambda_n > 0, \lambda_1 + \lambda_2 + \cdots + \lambda_n = 1,$$

则该点一定在这个凸n边形内部。这个结论可以从三角形开始对凸多边形的边数归纳证明。于是，一个自然的想法是，如果一个平面三角网格的边界是凸多边形，且其内部每个点都可以表示成其 1-邻域顶点的凸组合，那这个三角网格的边是否两两不相交？Tutte 给出了肯定的答案，并提出了这样的计算方法：设网格M的一个边界的顶点集合为$B = \{v_1, \cdots, v_b\} \subseteq V$，按顺时针排列。任取一个平面凸$b$边形，设其各顶点位置按顺时针排列为$\boldsymbol{\xi}_1, \cdots, \boldsymbol{\xi}_b$。于是网格$M$各顶点的平面参数坐标$\{\boldsymbol{x}_i\}_{i \in V}$可通过求解以下方程组得到：

$$\begin{cases} \boldsymbol{x}_{v_k} = \boldsymbol{\xi}_k, & k = 1, \cdots, b \\ \sum_{(i,j) \in E} \boldsymbol{x}_i - \boldsymbol{x}_j = 0, & i \notin B \end{cases},$$

这是关于各平面参数坐标的稀疏线性方程组。由 Tutte 的结论[46]，解出来的参数化映射必然是双射。由于此时每个顶点的参数坐标都在其 1-邻域顶点的参数坐标的重心，因此这样的参数化映射也称**重心映射**(barycentric map)。

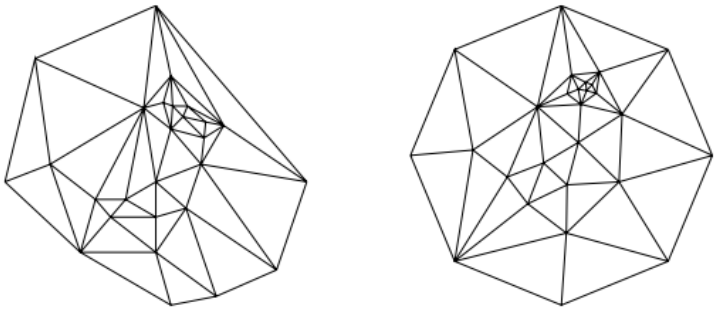


图 2.1 由 Tutte 方法得到的重心映射[8]

上述方法中，每个顶点关于其 1-邻域顶点的权重是均匀的。Floater 将其推广[8]，得到了一般权重的线性方法：对于每条边$(i,j)\in E$，指定一个权重$\omega_{ij}>0$。于是通过求解以下带权重的方程组可以得到各顶点的参数坐标：

$$\begin{cases} \boldsymbol{x}_{v_k}=\boldsymbol{\xi}_k, & k=1,\cdots,b \\ \sum_{(i,j)\in E}\omega_{ij}(\boldsymbol{x}_i-\boldsymbol{x}_j)=0, & i\notin B \end{cases},$$

Floater 证明了[9]这类方程组的解都是双射的参数化映射，并且可以通过ω_{ij}的选取来控制映射的性质。几种特殊的权重取法包括：

1) 边长的逆：$\omega_{ij}=\frac{1}{\|\bar{\boldsymbol{x}}_i-\bar{\boldsymbol{x}}_j\|}$

2) 余切权重：$\omega_{ij}=\frac{1}{2}(\cot\alpha_{ij}+\cot\beta_{ij})$

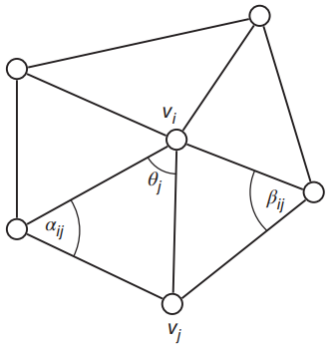


图 2.2 余切权重

Floater 在[8]中指出，由余切权重得到的参数化映射有比较好的**保形**(shape

preserving)性质。然而，若网格中存在相邻且对角都是锐角的三角形，则它们所共边的余切权重是负数，此时计算出参数化映射不一定是双射。常用的解决方法是把负的权重改成 0 或者非常小的正数。

值得注意的是，上述方法中求解的线性方程组正好等价于以下优化问题[8]：

$$\boldsymbol{x}^* = \underset{\boldsymbol{x}_i}{\operatorname{argmin}} \frac{1}{2} \sum_{(i,j)\in E} \omega_{ij} \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2,$$

$$\text{s.t.}\, \boldsymbol{x}_{v_k} = \boldsymbol{\xi}_k, \qquad k = 1, \cdots, b.$$

因此其本质也是几何优化问题。这个问题中的目标函数相当于每条边是一个弹性系数为ω_{ij}原长为 0 的弹簧系统的弹性势能，因此也称为**弦能量**(string energy)。这意味着，如果将这样一个弹簧系统的边界固定在提前给定的凸多边形上，系统最终达到稳态时各顶点的位置就是参数化坐标。

综上所述，Tutte 方法是一种简单的计算双射参数化映射的线性方法，且具有特殊的几何意义与物理意义。然而，这种方法产生的参数化映射往往扭曲较大，有些顶点的参数坐标会密集分布于凸多边形的重心或边界附近，导致周围产生特别小或者瘦长的三角形。尽管如此，我们仍可把它作为优化扭曲能量的初始化，因为我们只需要它满足最基本的无翻转条件。下一节我们将详细讲述优化扭曲能量的迭代过程。

第二节　迭代求解器

有了双射的初始化，就该进入迭代过程来降低扭曲能量了。由于扭曲能量是非凸函数，常用的方法包括预处理法，投影牛顿法和拟牛顿法。其中第一种一般是一阶方法，后两种则是二阶方法。

一、　预处理器

1. Sobolev 预处理器[24][21]

我们先来看看如何降低 ARAP 扭曲能量。首先回顾一下 ARAP 能量的定义：

$$E_{ARAP}(M,\boldsymbol{x})=\frac{1}{A(M)}\sum_{f\in F}|f|D_{ARAP}(J_f),\qquad D_{ARAP}(J_f)=\frac{1}{2}\|J_f-R_f\|_F^2,$$

其中

$$J_f=(\boldsymbol{x}_{f_1}-\boldsymbol{x}_{f_0},\boldsymbol{x}_{f_2}-\boldsymbol{x}_{f_0})(\boldsymbol{u}_{f_1},\boldsymbol{u}_{f_2})^{-1}=U_f\,\mathrm{diag}(\sigma_{1,f},\sigma_{2,f})\,V_f^T,\qquad R_f=U_fV_f^T.$$

要优化扭曲能量，首先要求其梯度。由于局部扭曲函数是关于J_f的，而J_f关于$\boldsymbol{x}$是线性的，因此只需要对J_f求梯度。然而，求R_f关于J_f的梯度并不是一件容易的事，但我们可以把R_f看成辅助变量，通过来回更新R_f与J_f来进行迭代：对当前的$\boldsymbol{x}$，可求出每个J_f，从而得到R_f；然后固定所有R_f，更新$\boldsymbol{x}$，从而得到新的J_f，然后重复这一过程直到收敛。由于第一步是局部更新，第二步是整体更新，所以这种方法也称为**局部-整体法**(local-global method)[24]。这种方法运用于优化 ARAP 能量非常有效，因为若固定所有R_f，这个能量便是关于J_f（从而关于$\boldsymbol{x}$）的二次函数，于是第二步就变成了一个二次规划问题。其二次型矩阵为

$$\Delta_{\boldsymbol{x}}E_{ARAP}(M,\boldsymbol{x})=\frac{1}{A(M)}\sum_{f\in F}|f|I_2\otimes P_fP_f^T,$$

其中

$$P_f=\begin{pmatrix}-1 & -1\\ 1 & 0\\ 0 & 1\end{pmatrix}(\boldsymbol{u}_{f_1},\boldsymbol{u}_{f_2})^{-1}.$$

经过计算，我们可以推出$\Delta_{\boldsymbol{x}}E_{ARAP}(M,\boldsymbol{x})$就是网格$M$的拉普拉斯矩阵$L(M)$。于是，相应的二次规划问题的解就是

$$\boldsymbol{x}^*=\boldsymbol{x}-L(M)^{-1}\nabla_{\boldsymbol{x}}E_{ARAP}(M,\boldsymbol{x}).$$

当然，为了保证双射性，我们应该将下降方向记为

$$\boldsymbol{d}=-L(M)^{-1}\nabla_{\boldsymbol{x}}E_{ARAP}(M,\boldsymbol{x}),$$

然后在这个方向上计算一个能有效降低能量且无三角形翻转的步长，这部分的细节会在后文中讨论。

后来，Kovalsky 等人在[21]中指出，用拉普拉斯矩阵的逆对其他扭曲能量的梯度进行变换都能得到不错的下降方向，即对任意的扭曲能量，在迭代时都取

$$\boldsymbol{d} = -L(M)^{-1}\nabla_{\boldsymbol{x}}E(M,\boldsymbol{x}),$$

其本质相当于在局部用 ARAP 能量来近似目标函数。于是，这种预处理器称为 Sobelev 预处理器，相应的预处理法称为 Sobolev 梯度下降法。值得注意的是，这种预处理器不依赖于迭代点，即在整个迭代过程中始终保持不变，因此可以预分解，从而大大减少每次迭代所需时间。另外，这种处理器与目标函数无关，一般的几何能量都可以用这种预处理器代替普通的梯度下降法来进行加速。Kovalsky 等人在[21]中用 Nesterov 方法进一步加速，提出了他们的 AQP 方法。这部分的细节会在第三节详细说明。

2. SLIM(Scalable Locally Injective Mappings)预处理器[32]

ARAP 扭曲能量虽然简单，但如同绪论中所说，由于它对接近翻转的三角形的抑制不够大，因此由它得到的下降方向上的最大步长往往非常小，导致迭代过程容易卡住。对于其他的能量，虽然 Sobolev 预处理器是通用的，但我们希望对每种能量都分别定制一种类似这样的预处理器，以更好地降低所选的目标能量。于是，Rabinovich 等人在[32]中按这样的思路提出了一类新的预处理器，称之为 SLIM 预处理器。Rabinovich 等人指出[32]，对于一般的几何能量，可以在局部用比 ARAP 函数更好的二次函数来近似。他们通过在 ARAP 扭曲函数中乘上一个权重矩阵来做到这一点：

$$P_W(J_f) = \left\|W_f(J_f - R_f)\right\|_F^2, \qquad \text{s.t.}\,\nabla_{J_f}P_W(J_f) = \nabla_{J_f}D(J_f).$$

在这个近似的二次函数中，我们只把J_f视为变量，把W_f, R_f都视为常量。计算这个近似函数的梯度，我们得到：

$$.\nabla_{J_f}P_W(J_f) = 2W_f^T W_f(J_f - R_f) = \nabla_{J_f}D(J_f),$$

$$\Rightarrow W_f = \left(\frac{1}{2}\nabla_{J_f}D(J_f)(J_f - R_f)^{-1}\right)^{1/2}.$$

特别地，对于对称狄利克雷扭曲函数，

$$\nabla_{J_f} D_{symDir}(J_f) = U_f \operatorname{diag}\left(2\sigma_{1,f} - \frac{2}{\sigma_{1,f}^3}, 2\sigma_{2,f} - \frac{2}{\sigma_{2,f}^3}\right) V_f^T,$$

$$\Rightarrow W_f = U_f \operatorname{diag}\left(\sqrt{\frac{(\sigma_{1,f}+1)(\sigma_{1,f}^2+1)}{\sigma_{1,f}^3}}, \sqrt{\frac{(\sigma_{2,f}+1)(\sigma_{2,f}^2+1)}{\sigma_{2,f}^3}}\right) U_f^T.$$

于是，SLIM 局部预处理器矩阵$H_{SLIM,f}$为

$$H_{SLIM,f} = W_f^T W_f \otimes P_f P_f^T.$$

整体预处理器矩阵

$$H_{SLIM} = \frac{1}{A(M)} \sum_{f \in F} |f| H_{SLIM,f},$$

从而迭代点$\boldsymbol{x}$处的下降方向为

$$\boldsymbol{d} = -H_{SLIM}^{-1} \nabla_{\boldsymbol{x}} E(M, \boldsymbol{x}).$$

这种预处理器比 Sobolev 预处理器的效果要好很多，因为它的权重矩阵是为每个扭曲能量专门计算的。另外，虽然这个预处理器会在迭代的过程中变化，但它的非零系数分布是不变的，因此可以将这个稀疏分布图案提前预分解。Rabinovich 等人还用实验指出[32]，这个预处理器是可扩展的(scalable)，也就是说，对于同一个模型不同规模的细分，收敛所需的迭代数是差不多的。

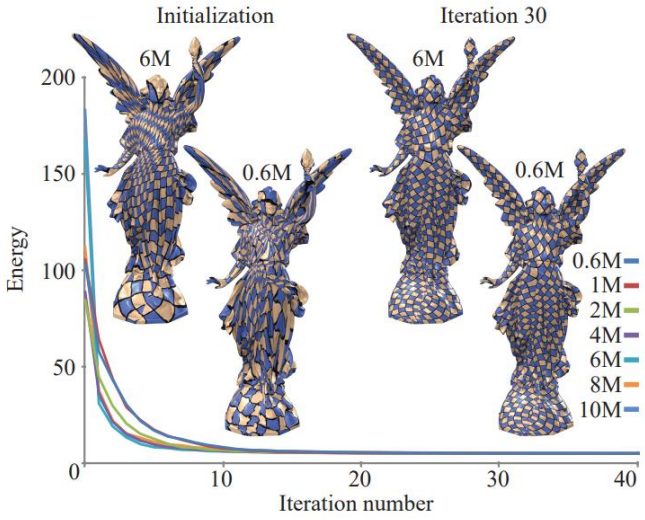


图 2.3 SLIM 预处理器的可扩展性[32]

3. AKVF(Approximated Killing Vector Field)预处理器[5]

绪论中提到过，预处理器的本质是解空间的一个黎曼度量。那么，单纯从解空间的角度考虑，设计合适的黎曼度量，也可以构造出预处理器。Claici 等人在[5]中提到，相差整体刚性变换的两个参数化映射扭曲能量是一样的，因此应该视为平等。而平面的刚性变换有三个维度——两个平移，一个旋转。于是，在原本的$2|V|$维欧氏空间中模掉刚性变换带来的等价关系之后，实际解空间应该是一个$2|V|-3$维的流形。在这个流形上定义黎曼度量就是要在其上每个点的切空间内定义一个内积。但是，直接在这样一个流形上定义黎曼度量是困难的，而且即使定义出来了，之后计算下降方向也很麻烦。因此，我们可以先在原来的$2|V|$维欧氏空间上定义一个伪黎曼度量（即半正定的黎曼度量），然后由商空间映射诱导来获得这个流形上的黎曼度量。这个伪黎曼度量需要以保持对称性的三个切方向为 Killing 向量，即它们需要是伪黎曼度量特征值为 0 的特征向量。更具体地，一个参数化映射$\boldsymbol{x}$处的切方向$\boldsymbol{d}$本质上是这个平面嵌入网格的一个切向量场，即对于每个顶点$i \in V$，$\boldsymbol{d}_i$是$\boldsymbol{x}_i$处的切向量。所有保持对称性的切向量场形如

$$\boldsymbol{d}_i = \begin{pmatrix} 0 & \omega \\ -\omega & 0 \end{pmatrix} \boldsymbol{x}_i + \boldsymbol{\xi}_i,$$

其中第一项是旋转部分，第二项是平移部分。原$2|V|$维解空间中的一个伪黎曼度量是指每个参数化映射$\boldsymbol{x}$处的一个对称半正定矩阵$H(\boldsymbol{x})$。它能在简并后的$2|V|-3$维流形上诱导一个良定的黎曼度量当且仅当其零空间就是所有保持对称性的切向量场。Claici 等人指出[5]，这样的切向量场有个共同特征，就是他们关于参数坐标的 Jacobi 矩阵都是反对称阵。然而，如果只知道单个顶点处的参数坐标与切向量，并不能唯一确定这个 Jacobi 矩阵，因此需要在三角形中考虑。对于网格M中的三角形$f = (f_1, f_2, f_3) \in F$，给定参数化映射$\boldsymbol{x}$与切向量场$\boldsymbol{d}$，由方程组

$$\boldsymbol{d}_{f_k} = A_f \boldsymbol{x}_{f_k} + \boldsymbol{\xi}_f, \quad k = 1, 2, 3,$$

可以解出唯一的A_f：

$$A_f = \left(\boldsymbol{d}_{f_1} - \boldsymbol{d}_{f_0}, \boldsymbol{d}_{f_2} - \boldsymbol{d}_{f_0}\right)\left(\boldsymbol{x}_{f_1} - \boldsymbol{x}_{f_0}, \boldsymbol{x}_{f_2} - \boldsymbol{x}_{f_0}\right)^{-1}.$$

注意到，A_f是反对称阵当且仅当$A_f + A_f^T = 0$，从而等价于$\left\|A_f + A_f^T\right\|_F^2 = 0$。而$\|X + X^T\|_F^2$是一般线性空间$\mathrm{GL}_2(\mathbb{R}) = \mathbb{R}^{2\times 2}$中的一个半范数（即半正定二次型），可以扩展成一个半内积：

$$\langle X, Y\rangle = \operatorname{tr}(X + X^T)(Y + Y^T).$$

在实际过程中，切向量场$\boldsymbol{d}$是未知的，因此要对每个单位切向量场单独求 Jacobi 并求出每个三角形 6 个 Jacobi 矩阵两两之间的半内积。把他们按顺序填入一个6×6的矩阵$H_{AKVF,f}(\boldsymbol{x})$，然后按面积加权就得到了原欧氏空间上的伪黎曼度量$H_{AKVF}(\boldsymbol{x})$。把这个伪黎曼度量作为预处理器，就是所谓的 AKVF 预处理器。此时每次迭代的搜索方向可以通过

$$\boldsymbol{d} = -H_{AKVF}(\boldsymbol{x})^{-1}\nabla_{\boldsymbol{x}}E(M, \boldsymbol{x})$$

计算。这种预处理器的特点在于它不依赖于扭曲能量，也不依赖于原网格的几何，只依赖于网格的连接关系，因此在原网格接近可展曲面时效果较好。

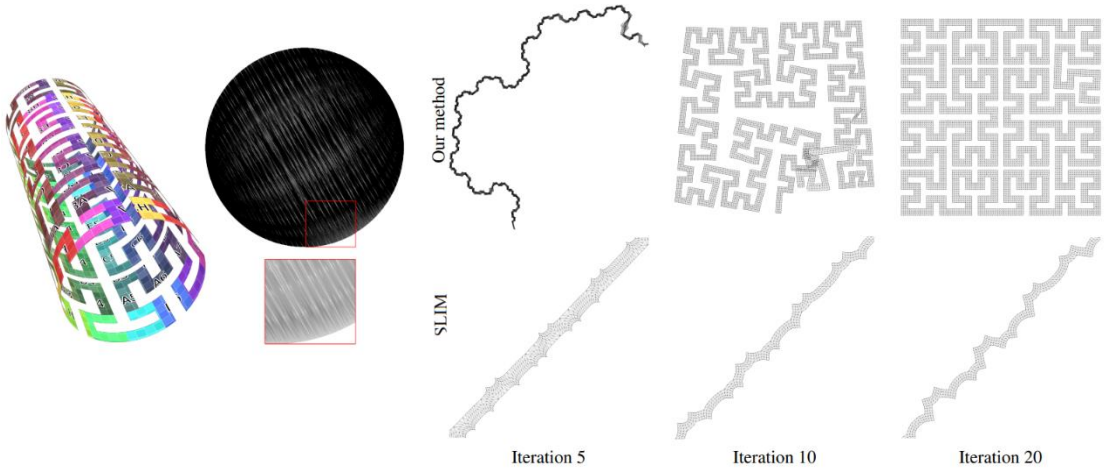


图 2.4 AKVF 与 SLIM 预处理器的比较[5]

二、 二阶求解器

1. CM(Composite Majorization)求解器[41]

既然目标函数是非凸函数，那么一个自然的想法就是用凸函数近似。前面提到的预处理器基本上都是用一个半正定二次型来近似，且只利用了目标函数的一

阶导数信息，最终的收敛速度都是线性的。若要做到超线性收敛，则必须充分利用二阶导数信息。Shtengel 等人指出[41]，对于常用的几何能量，可以将其分为一个凸函数和一个凹函数（即 Hessian 矩阵负定）之和，并以前者作为目标函数的近似。在这里，我们详细描述对称狄利克雷能量的凸近似。

首先回顾一下对称狄利克雷能量的奇异值表示：

$$E_{symDir}(M,\boldsymbol{x})=\frac{1}{A(M)}\sum_{f\in F}\frac{1}{2}|f|\left(\|J_f\|_F^2+\|J_f^{-1}\|_F^2\right)=\frac{1}{A(M)}\sum_{f\in F}h_f(\sigma_{1,f},\sigma_{2,f}),$$

其中

$$J_f=\left(\boldsymbol{x}_{f_1}-\boldsymbol{x}_{f_0},\boldsymbol{x}_{f_2}-\boldsymbol{x}_{f_0}\right)\left(\boldsymbol{u}_{f_1},\boldsymbol{u}_{f_2}\right)^{-1}=\begin{pmatrix}j_{00,f} & j_{01,f}\\ j_{10,f} & j_{11,f}\end{pmatrix}=U_f\,\mathrm{diag}\left(\sigma_{1,f},\sigma_{2,f}\right)V_f^T,$$

$$h_f(u,v)=\frac{1}{2}|f|\left(u^2+v^2+\frac{1}{u^2}+\frac{1}{v^2}\right),$$

$$\sigma_{1,f}=\frac{1}{2}\left(\|\alpha_f\|+\|\beta_f\|\right),\qquad \sigma_{2,f}=\frac{1}{2}\left(\|\alpha_f\|-\|\beta_f\|\right),$$

$$\alpha_f=\begin{pmatrix}j_{00,f}+j_{11,f}\\ j_{10,f}-j_{01,f}\end{pmatrix},\qquad \beta_f=\begin{pmatrix}j_{00,f}-j_{11,f}\\ j_{10,f}+j_{01,f}\end{pmatrix}.$$

因此，对称狄利克雷能量的 Hessian 阵为

$$\frac{\partial^2 E}{\partial \boldsymbol{x}^2}=\frac{1}{A(M)}\sum_{f\in F}\left(\left(\frac{\partial\sigma_{1,f}}{\partial\boldsymbol{x}},\frac{\partial\sigma_{2,f}}{\partial\boldsymbol{x}}\right)^T\Delta h_f(\sigma_{1,f},\sigma_{2,f})\left(\frac{\partial\sigma_{1,f}}{\partial\boldsymbol{x}},\frac{\partial\sigma_{2,f}}{\partial\boldsymbol{x}}\right)+\frac{\partial h_f}{\partial\sigma_{1,f}}\Delta\sigma_{1,f}+\frac{\partial h_f}{\partial\sigma_{2,f}}\Delta\sigma_{2,f}\right),$$

其中

$$\Delta h_f(u,v)=|f|\,\mathrm{diag}\left(1+\frac{3}{\sigma_{1,f}^4},1+\frac{3}{\sigma_{2,f}^4}\right),$$

$$\frac{\partial h_f}{\partial\sigma_{1,f}}=|f|\left(\sigma_{1,f}-\frac{1}{\sigma_{1,f}^3}\right),\qquad \frac{\partial h_f}{\partial\sigma_{2,f}}=|f|\left(\sigma_{2,f}-\frac{1}{\sigma_{2,f}^3}\right),$$

$$\Delta\sigma_{1,f}=\frac{1}{2}\left(\Delta\|\alpha_f\|+\Delta\|\beta_f\|\right),\qquad \Delta\sigma_{2,f}=\frac{1}{2}\left(\Delta\|\alpha_f\|-\Delta\|\beta_f\|\right).$$

由于$h_f(u,v)$是凸函数，因此 Hessian 阵求和内第一项永远是半正定的。又因为$\|\alpha_f\|,\|\beta_f\|$都是$\boldsymbol{x}$的线性映射与范数的复合，而由三角不等式，范数都是凸函数，因此$\Delta\|\alpha_f\|,\Delta\|\beta_f\|$也是半正定的。Hessian 阵求和内的后两项可以改写为：

$$\frac{\partial h_f}{\partial \sigma_{1,f}}\Delta\sigma_{1,f}+\frac{\partial h_f}{\partial \sigma_{2,f}}\Delta\sigma_{2,f}=\frac{1}{2}\left(\frac{\partial h_f}{\partial \sigma_{1,f}}+\frac{\partial h_f}{\partial \sigma_{2,f}}\right)\Delta\|\alpha_f\|+\frac{1}{2}\left(\frac{\partial h_f}{\partial \sigma_{1,f}}-\frac{\partial h_f}{\partial \sigma_{2,f}}\right)\Delta\|\beta_f\|.$$

由于

$$\sigma_{1,f}>\sigma_{2,f}\ \Rightarrow\ \frac{\partial h_f}{\partial \sigma_{1,f}}=|f|\left(\sigma_{1,f}-\frac{1}{\sigma_{1,f}^3}\right)>|f|\left(\sigma_{2,f}-\frac{1}{\sigma_{2,f}^3}\right)=\frac{\partial h_f}{\partial \sigma_{2,f}},$$

因此$\Delta\|\beta_f\|$前的系数永远是正的。于是只要取

$$H_{CM,symDir}=\frac{1}{A(M)}\sum_{f\in F}\left(\left(\frac{\partial \sigma_{1,f}}{\partial \boldsymbol{x}},\frac{\partial \sigma_{2,f}}{\partial \boldsymbol{x}}\right)^T\Delta h_f(\sigma_{1,f},\sigma_{2,f})\left(\frac{\partial \sigma_{1,f}}{\partial \boldsymbol{x}},\frac{\partial \sigma_{2,f}}{\partial \boldsymbol{x}}\right)\right.$$
$$\left.+\max\left(\frac{1}{2}\left(\frac{\partial h_f}{\partial \sigma_{1,f}}+\frac{\partial h_f}{\partial \sigma_{2,f}}\right),0\right)\Delta\|\alpha_f\|+\frac{1}{2}\left(\frac{\partial h_f}{\partial \sigma_{1,f}}-\frac{\partial h_f}{\partial \sigma_{2,f}}\right)\Delta\|\beta_f\|\right),$$

则它一定是半正定的，且对于原能量的 Hessian 阵是很好的近似。把它作为求解器，即用

$$\boldsymbol{d}=-H_{CM,symDir}^{-1}\nabla_{\boldsymbol{x}}E_{symDir}(M,\boldsymbol{x})$$

来计算下降方向，便是一个二阶的求解器，可以达到超线性的收敛速度。然而这种方法的缺点在于它需要利用目标能量的特定形式。这里我们只用对称狄利克雷能量举了简单例子，但一般的几何能量可能存在凹凸分解但并不直观。而且，若原能量的 Hessian 矩阵原本就是半正定的，这种方法不一定能保持这个矩阵。另外，它也利用了二阶矩阵奇异值的解析表达式，而这种简单的表达式对于三阶或更高阶的矩阵是不存在的，因此只适合二维问题而不适合更高阶的问题。下面我们要介绍的解析投影牛顿求解器可以很好地解决这些问题。

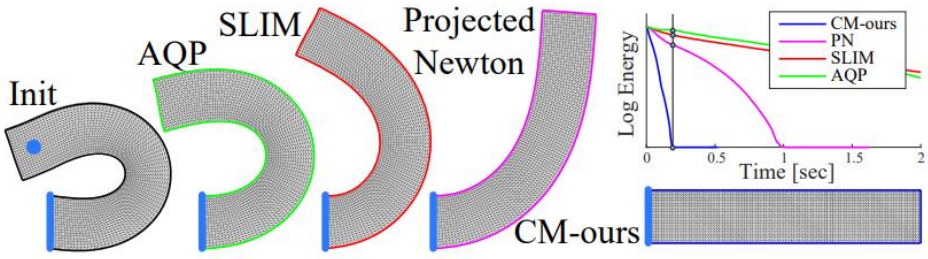


图 2.5 CM 与其他求解器的比较[41]

2. 解析投影牛顿求解器[42]

如绪论所说，求解非凸优化问题的一种常用二阶方法是投影牛顿法，其具体做法是将目标函数的Hessian阵进行特征值分解并将其中的负特征值直接改为0。若不知道特征值分解的解析表达式，则只能通过数值方法计算，这对于大规模问题来说近乎于死路。虽然我们的几何能量都可以写成局部三角形能量的加权求和，也就是说可以对每个局部能量分别使用投影牛顿然后把这些半正定矩阵加权求和作为求解器，但如果用数值方法的话每次迭代用时还是会很长。因此若能知道目标函数 Hessian 阵的解析表达式则可大大降低单次迭代用时。Smith 等人在[42]中指出，由于一般各向同性的几何能量在局部都可以表示成形变梯度的伸缩张量不变量的函数，而这些不变量也是位置坐标的函数。因此只要研究清楚这些不变量的 Hessian 阵的特征值分解，那么每个局部能量的特征值分解也就清楚了。更具体地，网格中三角形$f=(f_1,f_2,f_3)\in F$的形变梯度的极分解为：

$$J_f = U_f\,\mathrm{diag}(\sigma_{1,f},\sigma_{2,f})\,V_f^T = (U_fV_f^T)(V_f\,\mathrm{diag}(\sigma_{1,f},\sigma_{2,f})\,V_f^T) = R_fS_f,$$

其中S_f称为伸缩张量，其不变量分别为：

$$I_f = \mathrm{tr}(S_f) = \sigma_{1,f}+\sigma_{2,f},\qquad II_f = \det(S_f) = \sigma_{1,f}\sigma_{2,f}.$$

三角形f的局部对称狄利克雷扭曲可以写成：

$$D_{symDir}(J_f) = \frac{1}{2}\left(\sigma_{1,f}^2+\sigma_{2,f}^2+\frac{1}{\sigma_{1,f}^2}+\frac{1}{\sigma_{2,f}^2}\right) = \frac{1}{2}(I_f^2-2II_f)\left(1+\frac{1}{II_f^2}\right),$$

我们可以直接写出其关于形变梯度的 Hessian 阵的特征值分解[42]：

$$\Delta_{J_f}D_{symDir}(J_f) = \lambda_{1,f}\boldsymbol{d}_{1,f}\boldsymbol{d}_{1,f}^T+\lambda_2\boldsymbol{d}_2\boldsymbol{d}_2^T+\lambda_3\boldsymbol{l}\boldsymbol{l}^T+\lambda_4\boldsymbol{t}\boldsymbol{t}^T,$$

其中

$$\lambda_{1,f} = 1+\frac{3}{\sigma_{1,f}^4},\qquad \boldsymbol{d}_{1,f} = \mathrm{vec}\left(U_f\begin{pmatrix}1&0\\0&0\end{pmatrix}V_f^T\right),$$

$$\lambda_{2,f} = 1+\frac{3}{\sigma_{2,f}^4},\qquad \boldsymbol{d}_{2,f} = \mathrm{vec}\left(U_f\begin{pmatrix}0&0\\0&1\end{pmatrix}V_f^T\right),$$

$$\lambda_{3,f} = 1 + \frac{1}{II_f^2} + \frac{I_f^2 - 2II_f}{II_f^3}, \qquad \boldsymbol{l}_f = \mathrm{vec}\left(\frac{1}{\sqrt{2}} U_f \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} V_f^T\right),$$

$$\lambda_{4,f} = 1 + \frac{1}{II_f^2} - \frac{I_f^2 - 2II_f}{II_f^3}, \qquad \boldsymbol{t}_f = \mathrm{vec}\left(\frac{1}{\sqrt{2}} U_f \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} V_f^T\right),$$

其中 vec 是任意一种将2×2矩阵展开成 4 维向量的方式，只要这些特征向量的展开方式跟形变梯度各分量的展开方式一致就行。注意到，前三个特征值永远是正的，只有第四个特征值可能为负。因此 Hessian 阵的半正定部分为

$$\left(\Delta_{J_f} D_{symDir}(J_f)\right)^+ = \lambda_{1,f} \boldsymbol{d}_{1,f} \boldsymbol{d}_{1,f}^T + \lambda_2 \boldsymbol{d}_2 \boldsymbol{d}_2^T + \lambda_3 \boldsymbol{l}\boldsymbol{l}^T + \max(\lambda_4, 0)\, \boldsymbol{t}\boldsymbol{t}^T.$$

再令

$$H_{APN} = \frac{1}{A(M)} \sum_{f \in F} |f| \left(\Delta_{J_f} D_{symDir}(J_f)\right)^+,$$

我们就得到了解析投影牛顿求解器(Analytic Projected Newton, APN)。类似地，其他很多几何能量的局部 Hessian 阵也可以用这种方式解析地进行特征值展开，然后取其半正定部分加权求和作为求解器。注意到，这种方法依赖的是伸缩张量的不变量，因此很容易推广到高维。

三、 直线搜索[43]

通过以上任意一种求解器，我们可以得到一个下降方向，那么下一步就是在这个下降方向上找到一个合适的步长。更具体地，我们要求解以下问题：

$$\alpha^* = \underset{\alpha}{\mathrm{argmin}}\, E(\boldsymbol{x} + \alpha \boldsymbol{d}),$$

$$\text{s.t.}\ \det\left(\boldsymbol{x}_{f_1} - \boldsymbol{x}_{f_0} + \alpha(\boldsymbol{d}_{f_1} - \boldsymbol{d}_{f_0}), \boldsymbol{x}_{f_2} - \boldsymbol{x}_{f_0} + \alpha(\boldsymbol{d}_{f_2} - \boldsymbol{d}_{f_0})\right) > 0.$$

其中约束条件是为了保证进行这一步迭代之后没有三角形翻转，它是步长α的二次不等式组。由于初始化是无翻转的，因此我们可以假设每一次迭代前是无翻转的。所有无翻转步长构成$\mathbb{R}$中的一个开集，因此可以写成若干开区间的无交并。我们关心的是 0 所在的开区间的上界，它称为网格M在$\boldsymbol{x}$位置沿$\boldsymbol{d}$方向的整体最大步长。这个整体的最大步长可以通过计算各个三角形最大步长的最小值来得到，

而每个三角形的最大步长则是该三角形退化条件方程的最小正根，即

$$\det\left(\boldsymbol{x}_{f_1}-\boldsymbol{x}_{f_0}+\alpha\left(\boldsymbol{d}_{f_1}-\boldsymbol{d}_{f_0}\right),\boldsymbol{x}_{f_2}-\boldsymbol{x}_{f_0}+\alpha\left(\boldsymbol{d}_{f_2}-\boldsymbol{d}_{f_0}\right)\right)=0$$

的最小正根α_f[43]。于是整体最大步长$\alpha_{max}=\min\{\alpha_f|f\in F\}$。只要实际步长在这个最大步长内，则必不会有三角形翻转。具体取什么步长，则需要用直线搜索。我们并不需要精确得到能量下降最多的步长，因为这需要反复计算能量，比较费时间；我们只需要得到一个能使扭曲能量下降足够多的步长就够了。由于已经有了最大步长，我们使用 Armijo 准则的回溯直线搜索[43]，即每次将步长乘上一个小于 1 的系数直到找到合适的步长，并用 Wolfe 条件作为终止条件。这样，我们可以避免反复计算能量，并在数次迭代之内很快地找到一个合适步长。然后，我们便可以通过$\boldsymbol{x}\leftarrow\boldsymbol{x}+\alpha\boldsymbol{d}$来更新网格的参数坐标。

四、 终止条件

通过前面提到的求解器计算一个下降方向，再在下降方向上找到一个合适的步长并更新各顶点参数坐标，便是一个完整的迭代过程。然而，决定这个迭代过程何时终止并不是一件容易的事情。我们不希望迭代次数太多，但也不希望迭代在还没收敛的时候就停止了。我们希望对于不同能量函数以及不同规模的网格能有一个统一的迭代终止条件，这样我们便不需要为不同的情况专门调整参数。一般基于迭代的优化算法往往使用以下几种终止条件：

1) 基于目标函数梯度：$\|\nabla_{\boldsymbol{x}}E(M,\boldsymbol{x})\|<\epsilon$；

2) 基于迭代点的变化：$\|\Delta\boldsymbol{x}\|<\epsilon$ 或 $\|\Delta\boldsymbol{x}\|/\|\boldsymbol{x}\|<\epsilon$；

3) 基于目标函数的变化：$|\Delta E|<\epsilon$ 或 $|\Delta E|/|E|<\epsilon$。

由于我们的几何能量是恒正的且有下界，因此我们选择在能量相对降低足够小的时候终止。这个终止条件对于不同的能量函数，不同规模的网格是相对公平的。Zhu 等人在[51]中对几何优化方法的终止条件进行了更细致的研究。

最后，我们将上述所有过程拼在一起，整理成一个完整的迭代算法：

算法 1. 平面参数化几何优化迭代算法[41]	
1	$\boldsymbol{x} = \text{Tutte}(\,)$
2	while $\Delta E/E > \epsilon$
3	$\boldsymbol{d} = -H^{-1}\nabla_{\boldsymbol{x}}E(M,\boldsymbol{x})$
4	$\alpha_{max} = \text{max_step}(\boldsymbol{x},\boldsymbol{d})$
5	$\alpha = \text{backtracking_line_search}(\boldsymbol{x},\boldsymbol{d},\alpha_{max})$
6	$\boldsymbol{x} \leftarrow \boldsymbol{x} + \alpha\boldsymbol{d}$
7	end

这个框架适合各种几何能量和各种求解器，其中H取决于求解器的选择。

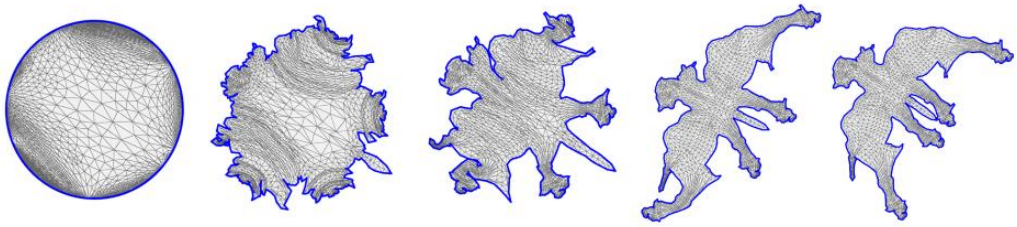


图 2.6 平面参数化几何优化流程[43]

第三节 加速方法简介

上一节介绍了各种迭代求解器，并在最后给出了一个统一的几何优化迭代框架。这一节我们简单介绍一些特殊的加速方法，包括更改参考三角形，更改搜索方向，Nesterov 加速和 Anderson 加速。其中前两种属于几何加速方法，利用了参数化问题的几何特性；后两种属于更为一般的加速方法，适用于一般的非线性优化问题。

一、 更改参考三角形[23]

形状较为复杂的网格的初始化往往会含有局部扭曲特别大的三角形，这些三角形往往会导致计算出来的下降方向上最大步长特别小，使得迭代无法顺利进行。Liu 等人发现[23]，若各三角形的局部扭曲都在一定的范围内，那迭代就可以顺利进行。于是，他们提出，可以插值一些中间三角形来代替源三角形作为参考三角形，使得那些以源三角形为参考时扭曲较大的三角形在改变参考之后扭曲较小，

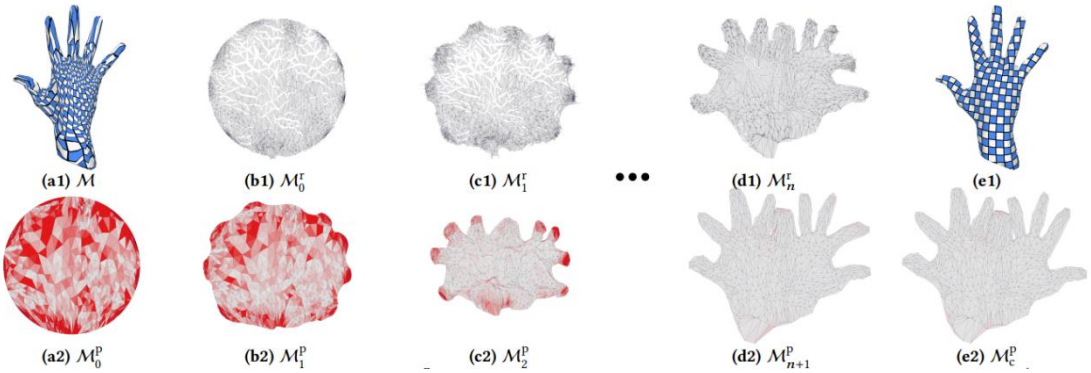


图 2.7 逐渐收敛到原网格的参考三角形[23]

这样从像三角形到新的参考三角形的扭曲能量优化会变得容易。然后让这些参考三角形在迭代的过程中逐渐收敛到源三角形，就可以使得最终仍然以源三角形为参考并成功优化扭曲能量。换句话说，这种方法给扭曲较大的三角形指了一条路，使得原本无法一步到位的三角形可以按着这条路一步一步降低自己的扭曲，不至于一开始由于方向不合适而卡住。一些具体的细节可以参考[23]。

二、 更改搜索方向[51]

对于最大步长太小的搜索方向，Zhu 等人在[51]中提出了另外一种解决方法，就是找一个离已计算的下降方向最近且最大步长尽量大的搜索方向。他们将其表述为一个新的优化问题，并用衰减投影 Jacobi 法(damped projected Jacobi)在固定迭代次数内得到一个近似解，作为调整过的搜索方向。他们将这一策略称为矫正直线搜索(Curing Line Search)[51]。

三、 Nesterov 加速[25][21]

从物理的角度来看，一个优化问题的目标函数可以看成势能，从而其解便是最低能量所对应的稳态。于是，一个迭代过程的本质便是解空间中的一个点在势能产生的力的作用下运动。假设阻力足够大，那么这个点每次只能朝着力的方向（也就是目标函数的负梯度方向）移动一小步，这就是梯度下降法。假设解空间的度量不再是欧氏度量而是某种特殊的黎曼度量，那么力的方向便是黎曼流形上目标函数的负梯度，这就是预处理器法。那么，若减小阻力，使得点的速度得以

累积，那它的运动就会变得更加顺畅，这便是 Nesterov 加速[25]的核心思想。这个加速方法引入一个动量项用以累积目标函数的梯度，然后由动量来决定点的移动方向。Kovalsky 等人[21]正是利用了这种方法对 Sobolev 梯度下降法进行了加速，得到了他们的 AQP(Accelerated Quadratic Proxy)方法。

四、 Anderson 加速[2][28]

Anderson 在[2]中将迭代过程看成一个对迭代点不停调用的函数，即

$$\boldsymbol{x}^{(n+1)} = f\left(\boldsymbol{x}^{(n)}\right), \qquad n = 1, 2, \cdots.$$

于是，这个迭代过程最终将收敛于这个函数的不动点。因此，可以通过迭代过程中的几个连续的迭代点来估计这个不动点作为下一个迭代点，这样往往比直接调用函数进行迭代更有效。由于这种方法是在迭代过程中采样这个函数的信息，因此在迭代刚开始进行时不稳定（因为刚开始的几个迭代点相差太多导致无法对这个函数进行很好的估计），而在迭代点接近收敛时效果较好。Peng 等人[28]将 Anderson 加速加以修改引入几何优化，并用实验[28]说明这种加速方法能有效加快 SLIM 预处理器的收敛。

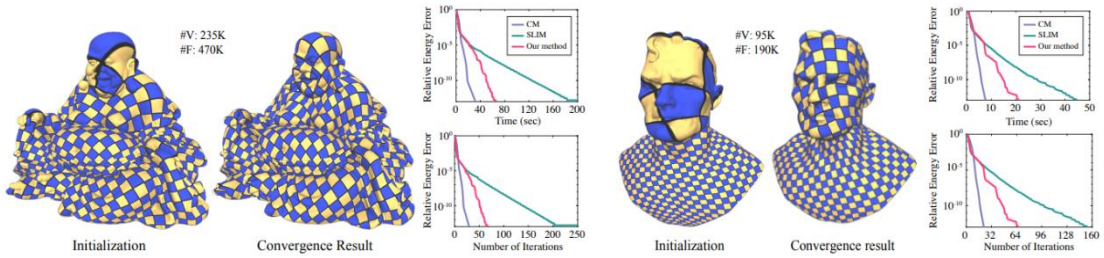


图 2.8 带 Anderson 加速的 SLIM 方法[28]

第三章　低扭曲无翻转球面参数化

第一节　球面参数化简介

实际应用中许多网格来自物体表面的 3D 扫描，因此是封闭的。相比于切开再展开到平面，很多应用中这些网格更适合直接在球面上展开。然而，由于球面约束，计算球面参数化映射比平面参数化要复杂不少。上一章讲述了平面参数化几何优化方法的完整过程，然而将这些方法推广到球面并不是一件平凡的事。首先，一个从闭网格到球面的无翻转映射并不容易获得。提到曲面到球面的映射，最容易想到的肯定是高斯映射，即把每一点映到该点处的法向量，然而对于一般网格这显然不是双射。另外一种容易想到的便是把网格做某个平移然而把每个点进行单位化，而这种方法只有在网格包裹的空间是星形区域时才是双射。鉴于平面中的双射参数化可以很容易通过线性方法获得，一种自然的想法便是先将原网格映到平面，然后再通过某种平面到球面的映射映回球面。这便是基于球极投影的球面参数化。

一、　基于球极投影的球面参数化

由于光滑情形下的球极投影是双射，并且还有保角性质，因此很容易想到用这个映射将平面参数化坐标转移到球面。Haker 等人在[17]提出可以在原网格上挖出一个三角形，将剩余部分用类似 Tutte 的方法展开到平面，然后将平面参数坐标通过球极投影投回球面，以此得到球面映射。Isenberg 等人在[20]中将原网格分割成两块，分别用 Tutte 方法参数化到单位圆盘中，然后用球极投影分别将这两个圆盘投到单位球面的上半球与下半球上。然而，Sheffer 等人在[39]中指出，这些方法并不能保证得到无翻转的球面参数化，因为离散情形下的球极投影是可能引入翻转的。他们用了下面的图举例说明：

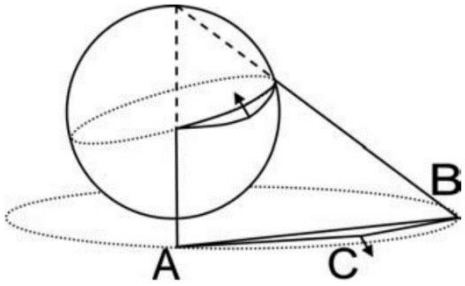


图 3.1 离散球极投影不再具有双射性[39]

图中，平面上 A, B 两点经过球极投影在球面上的像之间的测地线是球面大圆弧，这个大圆弧再经过球极投影映回平面时是一个经过 A, B 的圆弧。考虑一个平面中夹在连接 A, B 的直线与上述圆弧之间的点 C，这个点在穿过上述圆弧时，平面三角形 ABC 是不翻转的，而相应的球面三角形会翻转。特别地，当 C 正好在上述圆弧上时，平面三角形非退化，但相应的球面三角形退化成了一段大圆弧。这个例子很好地说明了离散情形下的球极投影是可能引入翻转的。

之所以会发生这种现象，是因为球极投影无法在平面与球面的测地线间构成对应：若上述 A, B 两点在球面上的像点之间的测地线再映回平面正好是 A, B 之间的直线，也就不会出现上述翻转现象了。因此，用球极投影来将平面参数化坐标转移到球面是不合适的。球极投影在光滑情形下是双射且有很好的性质，但在离散情形下却会引入翻转。这是因为在光滑情形下的映射是点到点的映射，而在离散情形下的映射是三角形到三角形的映射，因此我们除了点之间的对应外还要考虑边之间的对应。由于我们所指的三角形都是测地线三角形，即平面上的直线段三角形和球面上的大圆弧三角形，因此我们需要一种可以在平面直线与球面大圆弧间构成对应的映射来将平面中的参数化坐标转移到球面。

二、 基于球心投影的球面参数化

球面大圆弧的一种很有用的刻画，就是空间中过原点的平面与球面的交线。球面两点间的测地线便是这两点与原点形成的平面与球面的交线。于是，对于一个不包含原点的空间三角形，它生成的椎体跟球面的交集便是一个球面三角形。

这个过程相当于把这个空间三角形上的每个点进行单位化，因此称为球心投影。由上述陈述容易看出，假设我们有一个包含原点的凸多面体，它的面由三角形组成，那么将它的面通过球心投影映到球面，便能得到一个其表面到球面的无翻转映射。于是，找一个从任意闭网格到球面的无翻转映射的问题变成了先找一个从该网格到某个凸多面体表面的无翻转映射，再将这个凸多面体表面球心投影到球

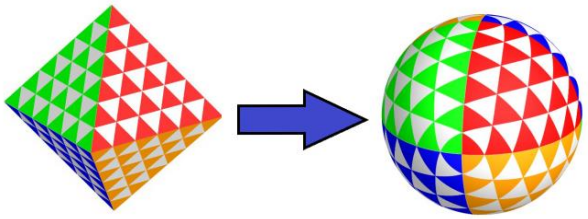


图 3.2 凸多面体的球心投影[31]

面就行了。而这个从闭网格到凸多面体表面的无翻转映射是很容易获得的：我们可以将闭网格分割成 4 块，使得每块与其他 3 块都相连且块与块之间有且仅有一条连续边界。我们可用**分层聚类法**(hierarchical clustering)[34][36][37]或者利用网格上的 Voronoi 图[7]来实现这种分割。另外，我们还要求没有三角形的三个顶点在同一条边界上，这可以通过将边界用最短路径拉直来满足[34]。然后，我们用 Tutte 方法[46]将每块中的顶点映到四面体的某个面上，之后再用球极投影将这四个面映到球面，我们就能得到一个无翻转的球面参数化。这样得到的参数化容易计算且其双射性有理论保证，但其缺点是得到的球面映射扭曲极大，而且对于大规模的网格，由于分出来的每个块的顶点数过多，在用 Tutte 方法时容易出现由数值精度导致的翻转。

三、 基于共形几何的球面参数化

由于光滑曲面理论已经很完善了，一些早期的球面参数化工作[14][15][16]从共形几何理论出发，将计算球面共形参数化的问题转化为一个微分方程，然后以高斯映射为初始化用 Gauss-Seidel 迭代法来求解。然而，离散情形下的映射双射性与连续情形的截然不同。因此即使上述微分方程的解在连续情形中是一个双射

的共形参数化映射，在离散情形中也可能会出现翻转。另外，后来[33]也证明了这个迭代过程在接近一个双射的解时会变得不稳定，然后收敛到一个退化解。

四、 Colin de Verdiere 方程

平面 Tutte 参数化能保证双射的一个重要原因是每个顶点的参数坐标都能表示成周围顶点参数坐标的凸组合。随着对球面参数化的深入研究，一些与平面类似的结果也得以被证明。Gotsman 等人在[12]中引入图论中的 Colin de Verdiere 理论，证明了一个离散的球面参数化是无翻转的当且仅当每个顶点都能表示成周围顶点参数化的凸组合的球心投影，如图：

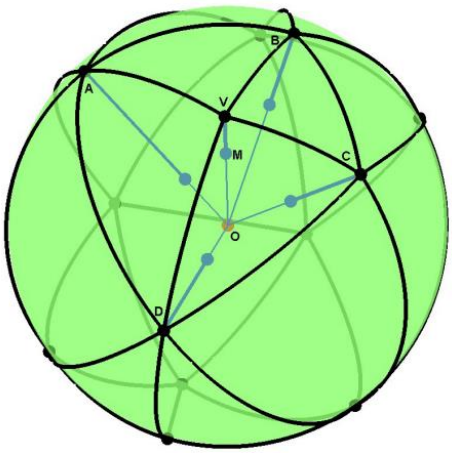


图 3.3 顶点v是周围顶点A,B,C,D的凸组合的球心投影[12]

他们还给出了一个通用的球面参数化方程组[12]：

$$\sum_{(i,j)\in E} \omega_{ij}\boldsymbol{x}_j = \alpha_i \boldsymbol{x}_i, \qquad \forall i \in V$$

$$\|\boldsymbol{x}_i\| = 1, \qquad \forall i \in V$$

这个方程组类似于平面参数化的 Tutte 方程组，是对离散的球面参数化最本质的刻画，称为 Colin de Verdiere 方程组。然而，由于这是一个非线性方程组，因此直接求解极为困难。他们试着用一些数值工具来求解这些方程，但即使对于较小规模的网格都需要很长时间。之后，Saba 等人在[33]中试着提出一种更高效的解法，但他们并不能理论保证得到无翻转的参数化结果。Colin de Verdiere 方程组虽然是球面参数化最本质的方程，但至今还没有高效且鲁棒的求解器。

五、 基于多分辨率的球面参数化

既然直接利用原网格的几何信息很难设计出理论保证无翻转的球面参数化，那么不妨从拓扑的角度出发。从拓扑上来说，若网格的一个球面映射满足每个顶点的像都在其邻域顶点的像所围成的球面区域内部，那么这个映射便是无翻转的。因此，只要找到一种原网格顶点的填充顺序，按照这个顺序填充到球面上，便能得到一个无翻转的球面映射，这个技术称为多分辨率技术。于是，Shapiro 和 Tal 在[38]中首次运用了这个技术并给出了理论证明。他们的方法分为网格简化与网格加密两个阶段。在网格简化阶段，他们每次在原网格上挖掉一个顶点，记录这个顶点的信息，将其邻域重新三角化，然后重复这些步骤直至原网格简化为一个四面体。在网格加密阶段，他们按照挖去顶点的逆序将顶点一个一个填到球面上去，直至所有顶点都被还原。Praun 和 Hoppe 在[31]中也用了这个技术，并在网格加密的过程中试着去优化扭曲能量。后来，[47][19]也进一步延续了这种技术的研究。这种技术的优点在于理论上能保证无翻转且实际应用中对数值精度的忍受程度比球心投影法要好，但其缺点在于耗时较长，不适合规模特别大的网格。

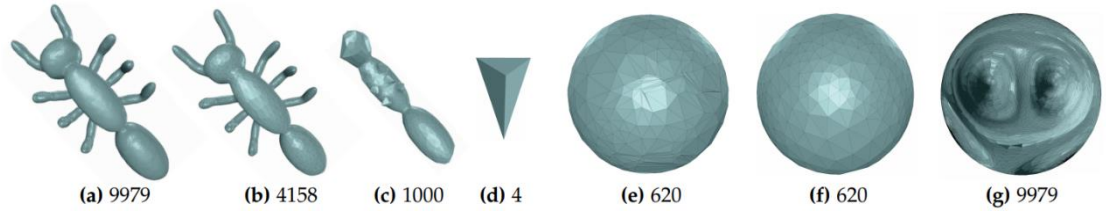


图 3.4 多分辨率技术流程[19]

六、 球面参数化的几何优化

平面参数化问题的几何优化方法已被广泛研究，然而球面参数化问题几何优化方法的相关文献却很少。这里我们知道的只有[49]，[48]和[19]。其中[49]将平面的 ARAP 扭曲能量推广到球面并进行优化。然而与平面 ARAP 方法的缺点一样，球面 ARAP 能量完全无法抑制球面三角形的翻转。[48]将平面的 AMIPS 能量推广到球面，并假设已有一个无翻转的球面映射作为初始化，在这个基础之上

来优化球面 AMIPS 能量。他们使用的优化策略是**块坐标下降法**(block coordinate descent)，是单点松弛法的推广，使用的求解器是普通的梯度下降法，因为当时还没有很好的几何优化求解器。他们的方法虽然能保证迭代过程中无翻转，但容易卡在扭曲较高的地方，这是因为他们的能量是用分片线性映射定义的，因此处理的仍然是平面三角形而不是球面三角形；而球面三角形与相应的平面三角形的退化行为并不一致，因此这样定义的能量无法对接近翻转的球面三角形起到很好的抑制作用。另外，他们对球面约束只做了简单的处理，在直线搜索过程中也没有计算最大步长，而是先设定一个足够大的步长然后用一个给定的比例系数不断缩减步长直到得到一个无球面三角形翻转且扭曲能量充分下降的步长。[19]使用的是多分辨率法，在将顶点逐个填充到球面上的时候优化一个扭曲能量。他们的几何优化是逐点的，而非对整个网格进行优化，因此耗时也较长。由此可见，至今还没有充分利用平面中常用几何优化方法的球面参数化算法。本文的主要贡献正是这一工作。

第二节　球面三角网格

一、　球面三角形的无翻转条件

既然球面三角形是指球面上的测地线三角形，那么一个球面三角形在翻转的时候必然会经过退化阶段，即球面三角形退化成一条球面大圆弧的时候。显然原本的平面三角形的退化判别条件不再适用，因为这时候三个球面顶点共处一条大圆弧，因此相应的平面三角形大部分时候是非退化的，除非刚好有两个或以上的顶点重合。根据上一节提到的球面大圆弧的刻画，若三个球面顶点共处一个球面大圆弧，那这三个顶点必然与原点共面。反之，若球面上的三个点不与原点共面，那么这三个点必不可能共球面大圆弧，从而相应的球面三角形也是非退化的。于是，我们可以用由三个球面顶点与原点组成的四面体的有向体积来判断一个球面三角形是否翻转或者退化：设网格M中三角形$f = (f_0, f_1, f_2) \in F$各顶点的球面参数坐标分别为$\boldsymbol{x}_{f_0}, \boldsymbol{x}_{f_1}, \boldsymbol{x}_{f_2} \in S^2$，则这个三角形在球面上不翻转当且仅当

$$\det\left(\boldsymbol{x}_{f_0}, \boldsymbol{x}_{f_1}, \boldsymbol{x}_{f_2}\right) > 0.$$

与平面情形相比，我们发现，球面三角形的本质不再是三角形，而是三个球面顶点与原点形成的椎体。球面三角形的本质是三维的而不是二维的，因此在定义相关参数化问题时应该更多地从三维角度进行考虑，而不是简单当作三角网格间的分片线性映射。

二、　严格的球面参数化映射

为了更好地定义球面参数化问题，我们首先要弄清楚网格到球面映射的本质。很多球面参数化方法的结果并不是一个严格的球面映射，而是一个从网格到球内接凸多面体的分片线性映射——他们仍然将边映到直线边，将三角形映到平面三角形，只不过这些边与三角形的顶点限制在球面上而已。然而，对于一个严格的球面映射，不仅网格的顶点要映到球面上，网格中的边要映到球面上的大圆弧，网格中的三角形也要映到球面三角形。给定三角形三个顶点在球面上的像点，可

以唯一确定一个球面大圆弧三角形，但从原网格的三角形内部到这个球面三角形内部的映射并不是唯一的。任意一种从平面三角形到相应的球面三角形的映射都称为**球面三角形映射**(spherical triangle map)。Praun 和 Hoppe 在[31]中充分地研究了这类映射，列举出了 8 种球面三角形映射并分析了各种映射的性质：

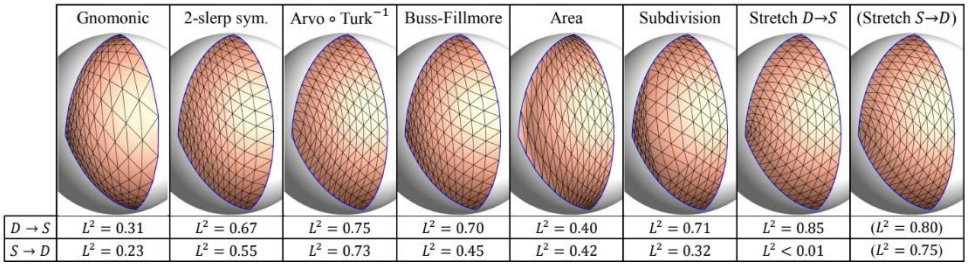


图 3.5 各种球面三角形映射扭曲对比[31]

Triangle map	Boundaries		Smooth interior	Analytic		Stretch	
	C^0	arc-len.		$D\to S$	$S\to D$	$D\to S$	$S\to D$
Gnomonic	yes	**no**	yes	yes	yes	poor	poor
2-slerp-sym.	yes	yes	yes	yes	**no**	ok	ok
Arvo ∘ Turk^{-1}	yes	**no**	yes	yes	**no**	good	good
Buss-Fillmore	yes	yes	yes	**no**	yes	good	poor
Area	**no**	**no**	yes	yes	yes	poor	poor
Subdivision	yes	yes	**no**	**no**	**no**	good	poor
Stretch optim.	**no**	**no**	visually	**no**	**no**	best	best

图 3.6 各种球面三角形映射性质对比[31]

其中最简单的一种称为**球心映射**(gnomonic map)，其本质就是平面三角形之间的线性映射复合上球心投影：给定网格M中的一个三角形$f = (f_0, f_1, f_2) \in F$，其源三角形三个顶点的位置分别为$\boldsymbol{u}_{f_0}, \boldsymbol{u}_{f_1}, \boldsymbol{u}_{f_2} \in \mathbb{R}^2$，这三个顶点在球面上的像点分别为$\boldsymbol{x}_{f_0}, \boldsymbol{x}_{f_1}, \boldsymbol{x}_{f_2} \in S^2$，于是这个三角形内部各点到相应的球面三角形内部的点球心映射为：

$$\lambda_0 \boldsymbol{u}_{f_0} + \lambda_1 \boldsymbol{u}_{f_1} + \lambda_2 \boldsymbol{u}_{f_2} \mapsto \frac{\lambda_0 \boldsymbol{x}_{f_0} + \lambda_1 \boldsymbol{x}_{f_1} + \lambda_2 \boldsymbol{x}_{f_2}}{\left\| \lambda_0 \boldsymbol{x}_{f_0} + \lambda_1 \boldsymbol{x}_{f_1} + \lambda_2 \boldsymbol{x}_{f_2} \right\|},$$

其中$\lambda_0, \lambda_1, \lambda_2 \geq 0, \lambda_0 + \lambda_1 + \lambda_2 = 1$是三角形内的重心坐标。这个映射的优势在于它很简单，它的逆映射可以显式表达，且它在三角形内部也满足测地线间的对应。其他有一些球面三角映射有的在边界光滑性上比球心映射要好，但这些映射一般

都较为复杂，且无法保证三角形内测地线间的对应。因此，我们就选用球心映射作为我们的球面三角映射。于是，给定网格中各个顶点在球面上的参数位置，我们把相应的球面参数化定义为将顶点映到相应的球面参数坐标，将网格中的三角形通过球心映射映到相应的球面三角形的**分片球心映射**(piecewise gnomonic map)。只有这样定义才能严格地给出一个真正意义上的球面映射。根据这个定义，只要知道各个顶点的球面参数坐标，就能唯一确定一个球面参数化映射。

三、 球面参数化映射的扭曲能量

有了严格的球面参数化映射，我们现在来定义这个映射的扭曲能量。虽然我们可以直接模仿平面参数化扭曲能量的定义方式将其定义成三角网格间分片线性映射的相应扭曲能量，但我们发现这样定义的扭曲能量是难以优化的，这是因为在球面三角形接近退化时，相应的平面三角形并不会退化；而球面三角形翻转

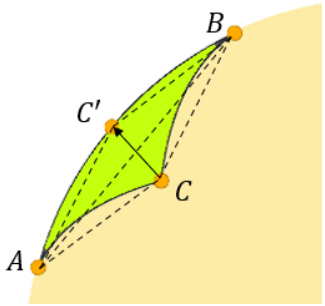


图 3.7 球面三角形的退化

时相应的平面三角形也不会翻转。因此这样定义的能量对接近退化的球面三角形无法起到很好的抑制作用。

一种可以考虑的替代方案是用平面能量的连续形式：以对称狄利克雷能量为例，其平面离散形式为

$$E_{symDir}(M,\boldsymbol{x}) = \frac{1}{A(M)}\sum_{f\in F}\frac{1}{2}|f|\left(\|J_f\|_F^2 + \|J_f^{-1}\|_F^2\right).$$

相应的连续形式应该在局部写为积分：

$$E_{symDir}(M,\boldsymbol{x}) = \frac{1}{A(M)}\sum_{f\in F}\int_f \frac{1}{2}\left(\|J_f(\lambda_0,\lambda_1,\lambda_2)\|_F^2 + \|J_f^{-1}(\lambda_0,\lambda_1,\lambda_2)\|_F^2\right)d|f|,$$

其中$(\lambda_0,\lambda_1,\lambda_2)$是三角形$f$中的点的重心坐标。这两种形式是相容的，因为平面中的参数化映射是分片线性映射，每个三角形上的形变梯度恒为常量，因此相应的积分形式积出来就等于离散求和的形式。然而如前面所说，球面参数化映射不再是分片线性映射，而是分片球心映射，因此不应该再使用第一种离散求和的形式，而应该使用第二种分片积分求和的形式。在这个形式中形变梯度在三角形内并不是常量，于是局部扭曲函数不能直接对三角形定义而需要在三角形内的每个点处定义然后对三角形积分。这种定义方式严格地反映了球面参数化的信息，并且具有普适性。可以对所有几何能量都做类似的处理。然而，这个定义中的积分对于大部分能量是无法显式地积出来的，在实际算法中只能用数值方法精确计算。这是非常耗时间的，尤其是在几何优化中，除了计算能量本身还要计算梯度，Hessian 阵等，精确计算需要的时间不可想象。

我们可以试着去估算这个积分能量。给定网格M各顶点的球面参数坐标$\boldsymbol{x}=\{\boldsymbol{x}_v\in S^2\subseteq\mathbb{R}^3\}_{v\in V}$，对于三角形$f=(f_0,f_1,f_2)\in F$，相应的源三角形到球面三角形的球心映射为：

$$\lambda_0\boldsymbol{u}_{f_0}+\lambda_1\boldsymbol{u}_{f_1}+\lambda_2\boldsymbol{u}_{f_2}\mapsto\frac{\lambda_0\boldsymbol{x}_{f_0}+\lambda_1\boldsymbol{x}_{f_1}+\lambda_2\boldsymbol{x}_{f_2}}{\left\|\lambda_0\boldsymbol{x}_{f_0}+\lambda_1\boldsymbol{x}_{f_1}+\lambda_2\boldsymbol{x}_{f_2}\right\|},$$

其中$\lambda_0,\lambda_1,\lambda_2\geq 0,\lambda_0+\lambda_1+\lambda_2=1$是三角形内的重心坐标。我们记

$$\boldsymbol{p}(\lambda_0,\lambda_1,\lambda_2)=\lambda_0\boldsymbol{x}_{f_0}+\lambda_1\boldsymbol{x}_{f_1}+\lambda_2\boldsymbol{x}_{f_2},$$

于是球心映射在$(\lambda_0,\lambda_1,\lambda_2)$处的形变梯度为

$$J_f(\lambda_0,\lambda_1,\lambda_2)=\frac{1}{\|\boldsymbol{p}\|}\left(I-\frac{\boldsymbol{p}\boldsymbol{p}^T}{\|\boldsymbol{p}\|^2}\right)(\boldsymbol{x}_{f_1}-\boldsymbol{x}_{f_0},\boldsymbol{x}_{f_2}-\boldsymbol{x}_{f_0})(\boldsymbol{u}_{f_1}-\boldsymbol{u}_{f_0},\boldsymbol{u}_{f_2}-\boldsymbol{u}_{f_0})^{-1},$$

此时的形变梯度不再是2×2的矩阵，而是3×2的矩阵，因此它的逆不是良定的。尽管如此，我们可以用它的伸缩张量来代替：

$$S_f(\lambda_0,\lambda_1,\lambda_2)=\left(J_f^T(\lambda_0,\lambda_1,\lambda_2)J_f(\lambda_0,\lambda_1,\lambda_2)\right)^{1/2}.$$

于是三角形f处的局部积分能量可以改写为

$$\int_f \frac{1}{2}\left\|S_f(\lambda_0,\lambda_1,\lambda_2)\right\|_F^2\left(1+\frac{1}{\det\left(S_f(\lambda_0,\lambda_1,\lambda_2)\right)^2}\right)d|f|.$$

计算伸缩张量的行列式，我们发现：

$$\det S_f(\lambda_0,\lambda_1,\lambda_2)=\frac{\det\left(\boldsymbol{x}_{f_0},\boldsymbol{x}_{f_1},\boldsymbol{x}_{f_2}\right)}{\|\boldsymbol{p}\|^3\det\left(\boldsymbol{u}_{f_1}-\boldsymbol{u}_{f_0},\boldsymbol{u}_{f_2}-\boldsymbol{u}_{f_0}\right)},$$

这正好解释了积分形式的扭曲能量可以有效抑制球面三角形翻转的根本原因：积分能量中含有伸缩张量的倒数项，而伸缩张量的行列式正比于球面三角形三个顶点与原点形成的四面体体积。因此当球面三角形接近退化时，这个体积趋于 0，从而积分能量趋于正无穷。于是只要能量中含有这个四面体体积的倒数项，都能有效抑制球面三角形翻转。

既然积分形式的扭曲能量计算起来较为复杂，那么我们不妨从令一个角度出发。注意到，球面三角形的退化条件是用四面体体积表述的，因此我们不妨从四面体的角度入手。对于原网格中的三角形，我们不再把它摊平到平面，而是将它对应到一个由球面上三个顶点与原点形成的四面体[49]，使得球面三个点组成的三角形面与网格中的这个三角形全等，且定向也一致。我们把这样一个四面体称为**源四面体**(source tetrahedron)。对于一个三角形$f=(f_0,f_1,f_2)\in F$，记其源四面

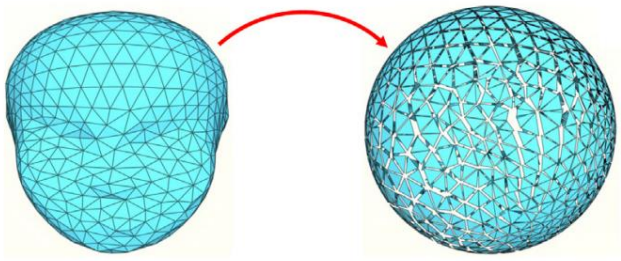


图 3.8 三角形对应到源四面体[49]

体中的三个球面顶点分别为$\boldsymbol{u}_{f_0},\boldsymbol{u}_{f_1},\boldsymbol{u}_{f_2}$，则它到像四面体的形变梯度为

$$J_{f,tet}=\left(\boldsymbol{x}_{f_0},\boldsymbol{x}_{f_1},\boldsymbol{x}_{f_2}\right)\left(\boldsymbol{u}_{f_0},\boldsymbol{u}_{f_1},\boldsymbol{u}_{f_2}\right)^{-1},$$

这是一个3×3的矩阵。于是我们可以将球面参数化的对称狄利克雷能量定义为

$$E_{symDir}^{sphere}(M,\boldsymbol{x})=\frac{1}{A(M)}\sum_{f\in F}\frac{1}{2}|f|\left(\left\|J_{f,tet}\right\|_F^2+\left\|J_{f,tet}^{-1}\right\|_F^2\right).$$

这个能量能有效抑制球面三角形的翻转，因为它含有形变梯度行列式的倒数项，而形变梯度的行列式正比于球面参数坐标与原点形成的四面体体积。用同样的方式，我们也可以将其他几何扭曲能量中的三角形形变梯度换成相应的四面体形变梯度来得到相应的球面参数化扭曲能量。这种定义方式的优点在于其形式比前面提到的积分形式要简单得多，从而计算其梯度和 Hessian 阵会变得容易，相应的平面优化方法也得以容易地应用于球面参数化。另外，这种通过四面体的定义方式对前面提到的球面三角形是很好的近似，因为它们的退化行为是一致的，且当网格较大的时候每个三角形都很小，从而球面三角形与原点形成的椎体可以近似为相应的四面体。

四、 球面参数化几何优化问题的表述

最后，我们将低扭曲球面参数化问题表述为一个在球面约束和无球面三角形翻转约束下对球面扭曲能量优化的问题：

$$\boldsymbol{x}^*=\operatorname*{argmin}_{\boldsymbol{x}}E^{sphere}(M,\boldsymbol{x}),$$

$$\text{s.t.}\ \|\boldsymbol{x}_v\|=1,\qquad \forall v\in V,$$

$$\det\left(\boldsymbol{x}_{f_0},\boldsymbol{x}_{f_1},\boldsymbol{x}_{f_2}\right)>0,\qquad \forall f\in F,$$

其中

$$E^{sphere}(M,\boldsymbol{x})=\frac{1}{A(M)}\sum_{f\in F}|f|D^{sphere}\left(J_{f,tet}\right).$$

注意到，由于球面无边，因此满足局部无球面三角形翻转的球面参数化映射就是双射，所以无需再添加整体双射性条件。上述问题是一个在流形约束与非线性开集约束下的非凸优化问题，因此在用平面参数化的优化方法之外还需对这个球面约束进行良好的处理，才能保证扭曲能量顺利降低。下一节我们将详细讲述各种平面几何优化求解器在这个球面参数化问题上的应用以及如何处理球面约束。

第三节　球面参数化的几何优化

在这一节中，我们要将无翻转平面参数化的几何优化流程推广到球面参数化。首先，让我们来回顾一下平面参数化的几何优化流程[43]：我们通过 Tutte 方法获得一个无翻转的平面参数化映射作为初始化，然后用一个迭代过程来降低扭曲能量直至收敛。在每个迭代过程中，我们先计算扭曲能量的梯度，然后通过一个半正定的求解器得到扭曲能量的一个下降方向，然后再在这个下降方向上计算能保证不出现三角形翻转的最大步长，然后从最大步长出发在这个下降方向上做回溯直线搜索直至找到一个合适的下降步长，最后更新各顶点的参数化坐标。我们现在分别将每个步骤推广到球面参数化。

一、　球面初始化

首先，我们需要一个无翻转的球面参数化作为初始化。在第一节中我们介绍了两种可以保证得到无翻转球面参数化的方法：球心投影法和多分辨率法。其中球心投影法有效地将平面参数化中的 Tutte 方法[46]产生的参数坐标转移到球面，因此非常快，但其缺点在于容易出现数值精度导致的翻转。多分辨率法通过拓扑的手段将顶点按照一定的顺序插入球面，因此对数值精度的忍耐较好，但其缺点在于耗时较长。不管选用哪种方法，只要能得到一个无翻转的球面参数化映射，就能作为初始化，然后用几何优化方法来降低扭曲能量。

二、　优化求解器

在第二章中，我们介绍了平面几何优化的各种求解器，主要分为预处理器和二阶求解器两类。其中 CM 求解器利用了二维奇异值分解的特殊形式从而无法推广到球面，而其他的求解器都有三维形式从而可以在球面参数化问题中使用。在这里，我们介绍这些求解器的球面形式。

1. Sobolev 预处理器[21]

Sobolev 预处理器直接用网格的拉普拉斯矩阵作为处理器，不依赖于目标能

量，也不依赖于具体的参数坐标，甚至不依赖于问题的维数。因此，这个预处理器可以不加修改地应用于球面参数化的扭曲能量优化。它的优点在于可以提前预分解，所以单次迭代用时很少，但缺点是收敛速度较慢。

2. SLIM 预处理器[32]

SLIM 预处理器在形式上也很容易推广到球面。其基本思想是在 Sobolev 预处理器的基础之上对不同的目标能量做一定的修正，其具体做法是通过一类加了权的二次函数来局部近似目标能量。我们用球面狄利克雷能量举例来演示这个过程：对于网格M中的三角形$f = (f_0, f_1, f_2) \in F$，其加权近似为

$$P_W^{sphere}(J_f) = \|W_f(J_f - R_f)\|_F^2, \qquad \text{s.t.}\, \nabla_{J_f} P_W^{sphere}(J_f) = \nabla_{J_f} D_{symDir}^{sphere}(J_f),$$

这里我们默认J_f是四面体间的形变梯度。计算这个近似函数的梯度，我们得到：

$$.\nabla_{J_f} P_W^{sphere}(J_f) = 2W_f^T W_f(J_f - R_f) = \nabla_{J_f} D_{symDir}^{sphere}(J_f),$$

$$\Rightarrow W_f = \left(\frac{1}{2}\nabla_{J_f} D_{symDir}^{sphere}(J_f)(J_f - R_f)^{-1}\right)^{1/2}.$$

将其中的每一项写成奇异值分解的形式，有：

$$J_f = U_f \,\mathrm{diag}(\sigma_{1,f}, \sigma_{2,f}, \sigma_{3,f})\, V_f^T, \qquad R_f = U_f V_f^T,$$

$$\nabla_{J_f} D_{symDir}^{sphere}(J_f) = U_f \,\mathrm{diag}\left(2\sigma_{1,f} - \frac{2}{\sigma_{1,f}^3}, 2\sigma_{2,f} - \frac{2}{\sigma_{2,f}^3}, 2\sigma_{3,f} - \frac{2}{\sigma_{3,f}^3}\right) V_f^T,$$

$$\Rightarrow W_f = U_f \,\mathrm{diag}\left(\sqrt{\frac{(\sigma_{1,f}+1)(\sigma_{1,f}^2+1)}{\sigma_{1,f}^3}}, \sqrt{\frac{(\sigma_{2,f}+1)(\sigma_{2,f}^2+1)}{\sigma_{2,f}^3}}, \sqrt{\frac{(\sigma_{3,f}+1)(\sigma_{3,f}^2+1)}{\sigma_{3,f}^3}}\right) U_f^T.$$

因此，球面的 SLIM 局部预处理器$H_{SLIM,f}^{sphere}$为

$$H_{SLIM,f}^{sphere} = W_f^T W_f \otimes \left(\boldsymbol{u}_{f_0}, \boldsymbol{u}_{f_1}, \boldsymbol{u}_{f_2}\right)^{-1} \left(\boldsymbol{u}_{f_0}, \boldsymbol{u}_{f_1}, \boldsymbol{u}_{f_2}\right)^{-1^T}.$$

将局部预处理器用三角形面积加权，便得到整体的 SLIM 预处理器：

$$H_{SLIM}^{sphere} = \frac{1}{A(M)} \sum_{f \in F} |f| H_{SLIM,f}^{sphere}.$$

这个预处理器的球面形式之所以较为简单是因为我们用四面体来定义对称狄利

克雷能量，从而它可以写为形变梯度三个奇异值的函数，而且是可分的：

$$D_{symDir}^{sphere}(J_f)=\frac{1}{2}\sum_{k=1}^{3}\left(\sigma_{k,f}^2+\frac{1}{\sigma_{k,f}^2}\right).$$

对于其他几何能量，他们的四面体形式的 SLIM 预处理器也可以类似地计算。值得注意的是，若用积分形式来定义扭曲能量，那局部的能量就没有这种简洁的形式，从而需要用梯度相等条件来计算每个点的加权矩阵。但即使计算出了加权矩阵，此时形变梯度在三角形内不是常量，于是需要在三角形内对每个点计算出的局部预处理器进行积分，这部分计算量是不可想象的。

3. AKVF 预处理器[5]

AKVF 预处理器的本质是将解空间中相差刚性变换的参数化映射视为平等，并在由这些等价类形成的流形上定义黎曼度量。为了方便计算，这个黎曼度量会写成原解空间中半正定矩阵的形式作为预处理器。对于球面参数化，由于每个顶点的参数坐标写成三维形式但限制在球面上运动，其解空间是$3|V|$维欧氏空间中的$2|V|$维流形。球面上的刚性变换有三个维度——北极点的位置占两个维度，绕北极点的旋转占一个维度。模掉这三个维度之后，实际的解空间还是一个$2|V|-3$维的流形，这与平面参数化映射的实际解空间维数是一致的。由于该流形上的黎曼度量不方便定义，我们仍然选择在原欧氏空间中定义一个伪黎曼度量使得每点处的 Killing 向量就是所有与流形在该点处切空间垂直的向量。这些向量的本质就是在当前状态下保持球面参数化映射对称性的向量场，它们均形如

$$\boldsymbol{d}_v=A\boldsymbol{x}_v,\qquad A+A^T=0.$$

对于网格M中的三角形$f=(f_0,f_1,f_2)\in F$，由方程组

$$\boldsymbol{d}_{f_k}=A_f\boldsymbol{x}_{f_k},\qquad k=1,2,3$$

可以解出唯一一个A_f，通过衡量其反对称性便可知道它离一个刚性变换有多远，即用$\|X+X^T\|_F^2$作为三阶一般线性空间$GL_3(\mathbb{R})=\mathbb{R}^{3\times3}$中的一个半范数。与平面情形类似，这个半范数可以扩展成一个半内积：$\langle X,Y\rangle=\mathrm{tr}(X+X^T)(Y+Y^T)$，也

就是$GL_3(\mathbb{R})$中的一个半正定二次型。由于在迭代过程中向量场$\boldsymbol{d}$是未知的，因此局部 AKVF 预处理器取为这个半正定二次型在单位向量场下的表示，即对每个单位切向量场单独求A_f并求出每个三角形 9 个A_f矩阵两两之间的半内积，然后把他们按顺序填入一个9×9的矩阵$H^{spere}_{AKVF,f}(\boldsymbol{x})$。最后我们将这些矩阵然后按面积集成得到原欧氏空间上的伪黎曼度量$H^{sphere}_{AKVF}(\boldsymbol{x})$，这个就是整体的球面 AKVF 预处理器。可见，球面 AKVF 预处理器的形式与平面情形的计算非常相似，只有 Killing 向量场的形式有很小的区别。球面的 AKVF 预处理器也只依赖于网格的连接关系，与网格的几何信息无关，也与目标能量的选取无关。

4. 解析投影牛顿求解器[42]

上面介绍的三种预处理器要么完全不依赖于目标能量的选取，要么只依赖于其一阶导数信息，因此都是一阶求解器，收敛速度较慢。现在我们来介绍一种二阶求解器。与平面情形类似，我们希望用解析的方式表达球面几何能量 Hessian 阵的特征值分解，从而可以通过将其中的负特征值修改为 0 来得到一个半正定的求解器。很幸运的是，[42]中给出了常用三维各向同性几何能量 Hessian 阵的特征值分解。我们在这里用球面对称狄利克雷能量举例：

$$E^{sphere}_{symDir}(M,\boldsymbol{x})=\frac{1}{A(M)}\sum_{f\in F}|f|D^{sphere}_{symDir}(J_f),$$

$$D^{sphere}_{symDir}(J_f)=\frac{1}{2}II_f+\frac{1}{8}\left(\frac{I_f^2-II_f}{III_f}\right)^2-\frac{I_f}{III_f},$$

其中

$$I_f=\sigma_{1,f}+\sigma_{2,f}+\sigma_{3,f},\qquad II_f=\sigma_{1,f}^2+\sigma_{2,f}^2+\sigma_{3,f}^2,\qquad III_f=\sigma_{1,f}\sigma_{2,f}\sigma_{3,f}$$

是三角形f的四面体形变梯度的伸缩张量的三个不变量。于是局部狄利克雷能量关于形变梯度的 Hessian 阵的特征值分解可写为[42]：

$$\nabla^2_{J_f}D^{sphere}_{symDir}(J_f)=\sum_{k=1}^{3}\left(\lambda_{\boldsymbol{d}_{k,f}}\boldsymbol{d}_{k,f}\boldsymbol{d}_{k,f}^T+\lambda_{\boldsymbol{l}_{k,f}}\boldsymbol{l}_{k,f}\boldsymbol{l}_{k,f}^T+\lambda_{\boldsymbol{t}_{k,f}}\boldsymbol{t}_{k,f}\boldsymbol{t}_{k,f}^T\right),$$

其中

$$\lambda_{\boldsymbol{d}_{k,f}} = 1 + \frac{3}{\sigma_{k,f}^4}, \qquad \boldsymbol{d}_{k,f} = \mathrm{vec}\big(U_f E_{kk} V_f^T\big),$$

$$\lambda_{\boldsymbol{l}_{1,f}} = 1 + \frac{\sigma_{1,f}^2}{III_f^2} + \big(II_f - \sigma_{1,f}^2\big)\frac{\sigma_{1,f}^3}{III_f^3}, \qquad \boldsymbol{l}_{1,f} = \mathrm{vec}\left(\frac{1}{\sqrt{2}} U_f \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} V_f^T\right),$$

$$\lambda_{\boldsymbol{t}_{1,f}} = 1 + \frac{\sigma_{1,f}^2}{III_f^2} - \big(II_f - \sigma_{1,f}^2\big)\frac{\sigma_{1,f}^3}{III_f^3}, \qquad \boldsymbol{t}_{1,f} = \mathrm{vec}\left(\frac{1}{\sqrt{2}} U_f \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{pmatrix} V_f^T\right),$$

其中第二组和第三组特征值与特征向量我们都只分别列举了一对，因为另外两对可以对下标进行轮换得到。注意到，前两组特征值是恒正的，只有第三组可能出现负特征值。因此我们的解析投影牛顿求解器为：

$$H_{APN}^{sphere} = \frac{1}{A(M)} \sum_{f \in F} |f| \sum_{k=1}^{3} \left(\lambda_{\boldsymbol{d}_{k,f}} \boldsymbol{d}_{k,f} \boldsymbol{d}_{k,f}^T + \lambda_{\boldsymbol{l}_{k,f}} \boldsymbol{l}_{k,f} \boldsymbol{l}_{k,f}^T + \max\left(\lambda_{\boldsymbol{t}_{k,f}}, 0\right) \boldsymbol{t}_{k,f} \boldsymbol{t}_{k,f}^T\right).$$

三、 球面约束的处理

如果直接用上述求解器计算下降方向，即通过

$$\boldsymbol{d} = -H^{-1} \nabla_{\boldsymbol{x}} E(M, \boldsymbol{x})$$

来计算下降方向，然后在这个下降方向上搜索步长，那么原本限制在球面的参数坐标是肯定会离开球面的。幸运的是，我们有一种很简单的将离开球面的点拉回球面的方式，那就是球心映射。更具体地，对于网格M的空间参数化映射$\boldsymbol{x}$（不一定在球面上），它的球心投影记为$P(\boldsymbol{x})$，具体在顶点$v \in V$处

$$P(\boldsymbol{x})_v = \frac{\boldsymbol{x}_v}{\|\boldsymbol{x}_v\|}.$$

注意到，对于原本就限制在球面上的映射，$P(\boldsymbol{x}) = \boldsymbol{x}$。于是，我们可以把对球面参数化映射的能量定义延伸到一般的空间参数化映射：

$$\tilde{E}(M, \boldsymbol{x}) \coloneqq E\big(M, P(\boldsymbol{x})\big).$$

在这个意义下，我们便无需再担心球面约束了，因为我们只需要对这个新定义的一般空间映射的能量函数求梯度，在求得下降方向$\boldsymbol{d}$和步长α后也只需要通过

$$\boldsymbol{x} \leftarrow P(\boldsymbol{x} + \alpha \boldsymbol{d})$$

来更新参数坐标，这样就可以保证每次迭代更新后的参数坐标始终在球面上。更具体地，在求梯度的时候，

$$\nabla_{\boldsymbol{x}} \tilde{E}(M, \boldsymbol{x}) = dP\big(\nabla_{\boldsymbol{x}} E(M, \boldsymbol{x})\big).$$

对于顶点$v \in V$，

$$dP\big(\nabla_{\boldsymbol{x}} E(M, \boldsymbol{x})\big)_v = (I - \boldsymbol{x}_v \boldsymbol{x}_v^T) \nabla_{\boldsymbol{x}} E(M, \boldsymbol{x})_v.$$

这代表在求解下降方向前需要将原本的梯度沿每个顶点的径向分量过滤掉，使得

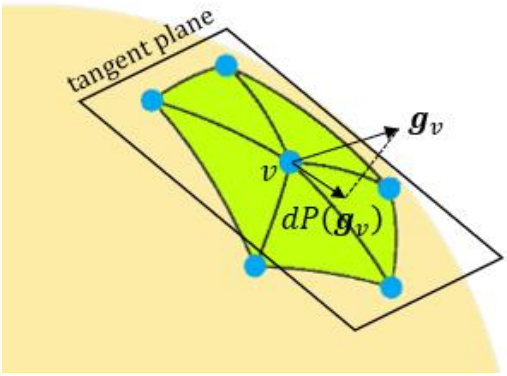


图 3.9 梯度的切投影

每个顶点处的移动方向都是沿球面切向的，然后用

$$\boldsymbol{d} = -H^{-1} dP\big(\nabla_{\boldsymbol{x}} E(M, \boldsymbol{x})\big)$$

来计算下降方向。经过试验我们发现，只有加上这个投影变换之后才能得到一个好的下降方向；若不加这个投影，得到的下降方向上最大步长往往很小，导致优化过程很容易在一开始卡住。因此这个投影变换是非常关键的。实际上，对于一般的流形约束也需要使用类似的投影变换[26][27]。

四、 直线搜索

有了下降方向，我们便需要计算一个合适的下降步长。由于我们最后是通过

$$\boldsymbol{x} \leftarrow P(\boldsymbol{x} + \alpha \boldsymbol{d})$$

来更新参数坐标的，因此这相当于在球面的大圆弧上进行搜索。在搜索步长的过程中，我们求解的问题是：

$$\alpha^* = \underset{\alpha}{\operatorname{argmin}} E\big(M, P(\boldsymbol{x} + \alpha \boldsymbol{d})\big),$$

$$\text{s.t.}\det\big(\boldsymbol{x}_{f_0} + \alpha \boldsymbol{d}_{f_0}, \boldsymbol{x}_{f_1} + \alpha \boldsymbol{d}_{f_1}, \boldsymbol{x}_{f_2} + \alpha \boldsymbol{d}_{f_2}\big) > 0, \qquad \forall f \in F.$$

我们首先要计算一个最大步长。与平面情形类似，这个最大步长是所有三角形最大步长的最小值，而在球面情形中每个三角形的最大步长则是相应的球面三角形刚好退化时的步长，也就是球面三个顶点与原点形成的四面体刚好退化时的步长，它是如下三次方程的最小正根：

$$\det\big(\boldsymbol{x}_{f_0} + \alpha \boldsymbol{d}_{f_0}, \boldsymbol{x}_{f_1} + \alpha \boldsymbol{d}_{f_1}, \boldsymbol{x}_{f_2} + \alpha \boldsymbol{d}_{f_2}\big) = 0.$$

由于三次方程的解析解形式较为复杂，我们可以用一些数值计算的库（比如 gsl）来求解上述方程。有了最大步长之后，我们便可通过回溯直线搜索得到一个合适的下降步长，这部分与平面情形差不多，就不细说了。

我们已经详细讲述了球面参数化几何优化的完整迭代过程，迭代的终止条件跟平面参数化是一样的。作为总结，我们将上述整个流程写成一个完整的算法：

算法 2. 球面参数化的几何优化迭代算法

$\boldsymbol{x} = \text{Gnomonic_Tutte}()$
while $\Delta E/E > \epsilon$
    $\boldsymbol{d} = -H^{-1}P\big(\nabla_{\boldsymbol{x}} E(M, \boldsymbol{x})\big)$
    $\alpha_{max} = \text{max_step}(\boldsymbol{x}, \boldsymbol{d})$
    $\alpha = \text{backtracking_line_search}(\boldsymbol{x}, \boldsymbol{d}, \alpha_{max})$
    $\boldsymbol{x} \leftarrow P(\boldsymbol{x} + \alpha \boldsymbol{d})$
end

其中H取决于求解器的选择，E取决于能量的选择。由此可见，我们的球面参数化几何优化框架适用于各种几何能量以及各种几何优化求解器。另外，平面几何优化中的各种加速方法也可以在略微修改后应用于球面参数化的几何优化，这里就不赘述了。

第四节　实验结果

一、　实现细节

在这里，我们将测试上一节所描述球面几何优化流程。我们在一台 8 核 i7 处理器 Windows 系统的 DELL 电脑上用 C++语言实现了上述算法，并用[52]提供的网格可视化图形用户界面来测试我们的算法。其中对于初始化，我们随机选取 4 个三角形据点并使用[7]的方法来获得网格的分割。这种分割可以保证每片都是连通的，但无法保证片与片间的连接关系与四面体一致。若违反了这一点，我们重新选取三角形据点直到找到满足上述条件的划分。我们将划分的边界通过最短路径拉直，映到正四面体的边上，然后用 Tutte 方法[46]将每片内的顶点映到正四面体的面中，再用球心投影将四面体上的参数坐标映到球面，得到球面初始化。在迭代过程中，我们使用 Eigen 进行奇异值分解和矩阵运算，在计算下降方向时用 Pardiso[29][30]求解相应的二次规划问题，在计算最大步长时用 gsl 求解相应的三次方程。

二、　结果展示

1. 各求解器之间的比较

Model Name: Horse, $|V| = 2000, |E| = 5994, |F| = 3996.$

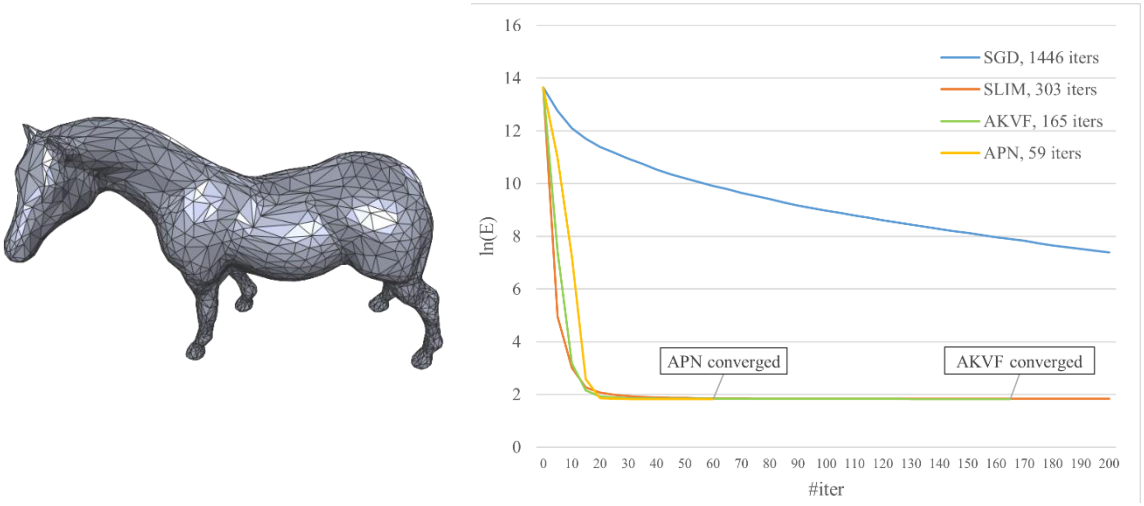


图 3.10　Horse 模型各求解器比较

Model Name: Spot, $|V| = 2930, |E| = 8784, |F| = 5856.$

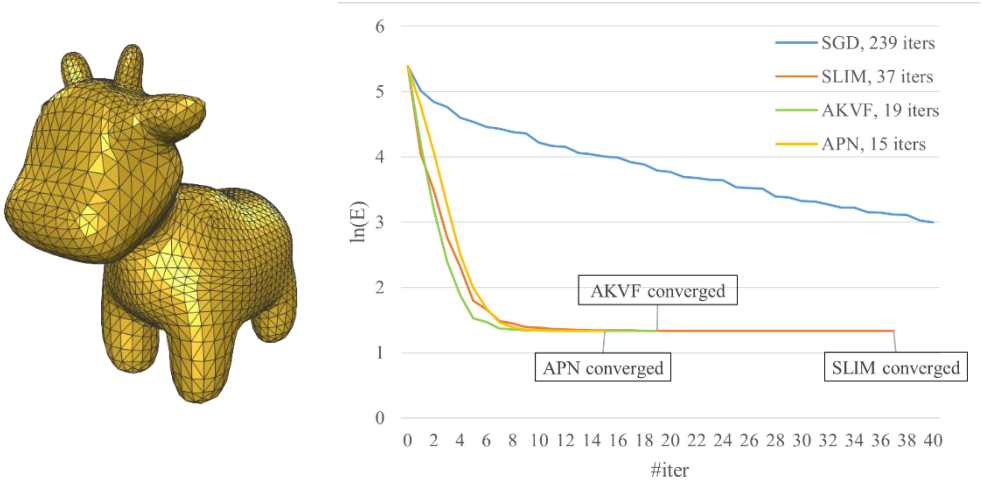


图 3.11 Spot 模型各求解器比较

Model Name: Ant, $|V| = 9501, |E| = 28497, |F| = 18998.$

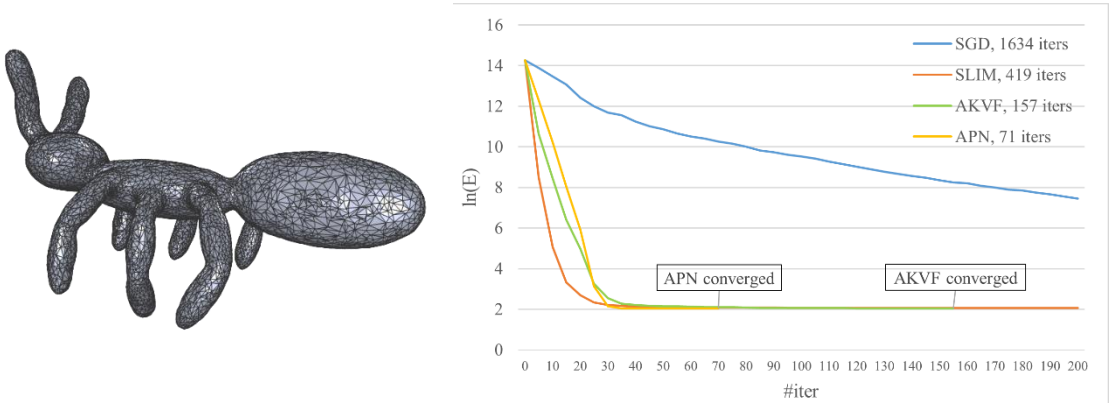


图 3.12 Ant 模型各求解器比较

Model Name: Hand, $|V| = 42068, |E| = 126198, |F| = 84132.$

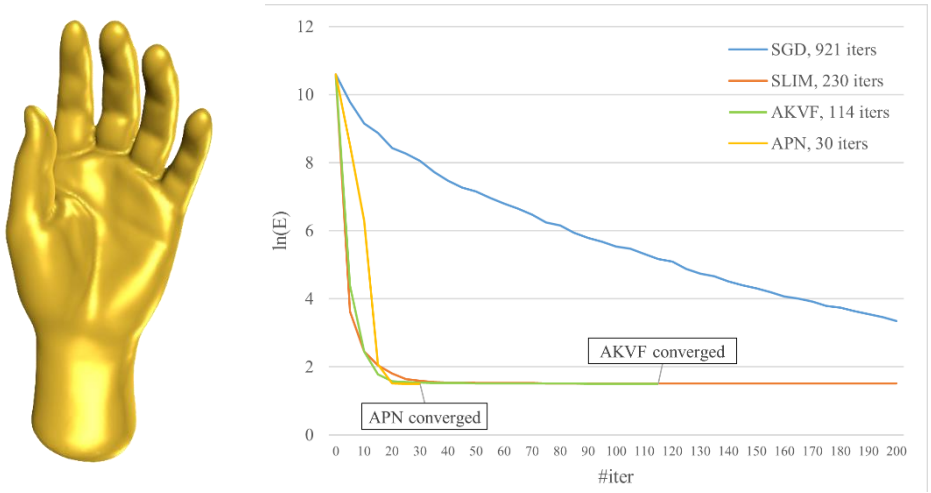


图 3.13 Hand 模型各求解器比较

2. 球面参数化映射展示

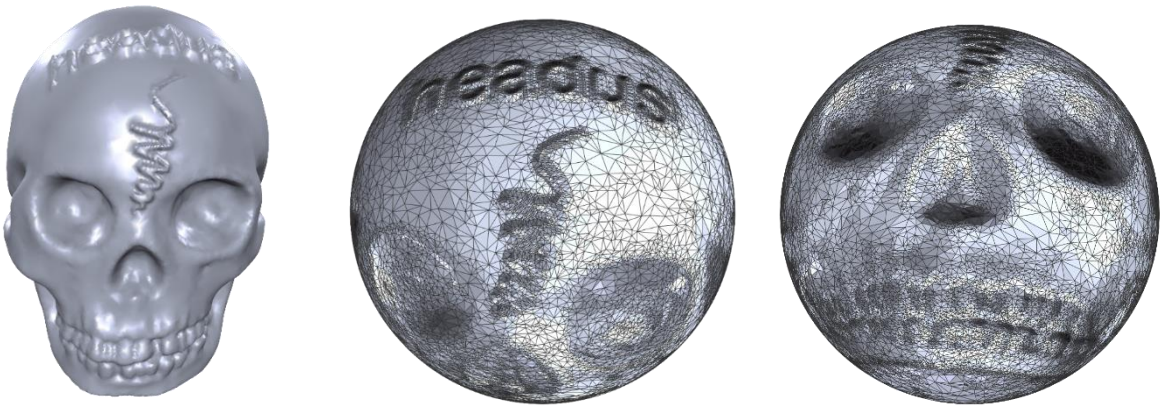


图 3.14 Skull 模型球面参数化映射



图 3.15 Buddha 模型球面参数化映射



图 3.16 Max-Planck 模型球面参数化映射



图 3.17　　Bunnyih 模型球面参数化映射



图 3.18　　Gargoyle 模型球面参数化映射

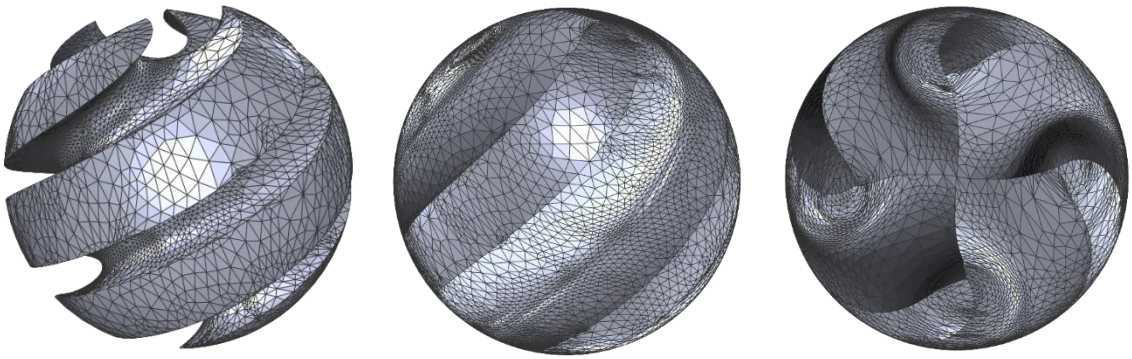


图 3.19　　Sharp-sphere 模型球面参数化映射

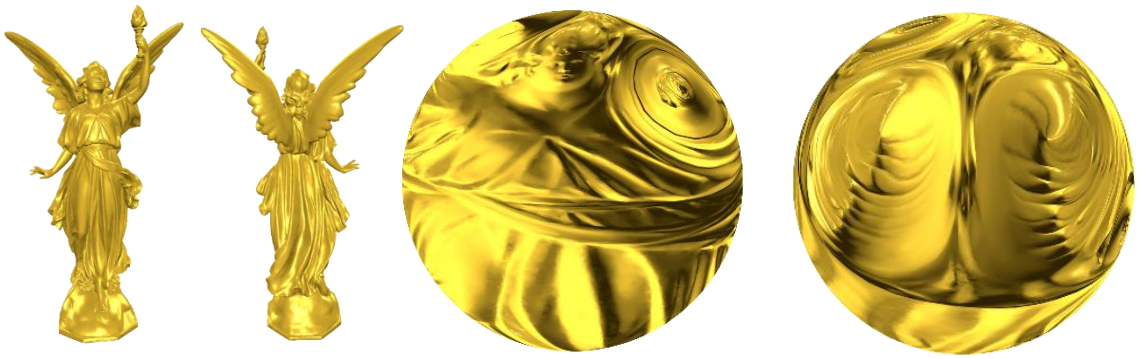


图 3.20　　Lucy 模型球面参数化映射



图 3.21　　Armadillo 模型球面参数化映射

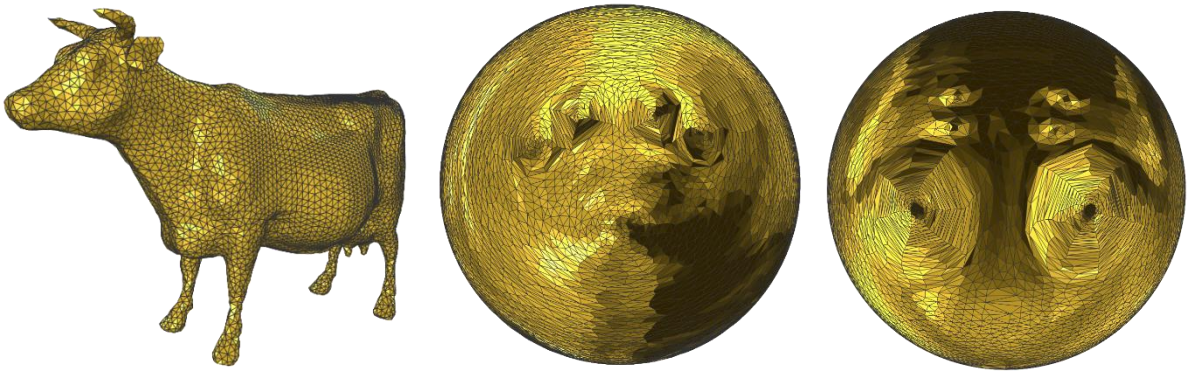


图 3.22　　Cow 模型球面参数化映射

三、　结论

经过以上实验，我们得出我们提出的几何优化流程能够很好地求解球面参数化问题，得到低扭曲无翻转的球面参数化映射。其中 Sobolev 预处理器迭代较快，但单次迭代减少的扭曲较少，导致收敛所需的迭代数远远大于其他求解器；SLIM 预处理器开始下降较快，但收敛很慢；AKVF 预处理器和解析牛顿求解器表现均较好，解析牛顿求解器收敛得最快。相比于其他方法，我们的方法总能得到扭曲较低的结果，而不会在扭曲较高的时候卡住。对于规模较大的网格，我们的球面参数化映射能清晰反映曲面上的各种细节。

第四章　总结与思考

到此为止，我们引入了网格参数化问题，详细介绍了平面参数化问题的各种几何优化方法，调研了现有的球面参数化方法，然后用自己的方式将平面中的几何优化方法应用于球面参数化问题的求解中，并用实验展示了效果。在用几何优化方法求解球面参数化问题时，我们仔细分析了球面参数化映射的特点，指出了它与平面参数化问题本质上的不同。我们还指出了球心投影的重要性，并反复利用这个映射来构建从平面参数化到球面的桥梁，从而得到一个通用的球面参数化几何优化框架，适用于各种几何能量以及各种几何优化求解器。我们对球面参数化问题的特殊处理使得我们能够高效地优化球面参数化映射的扭曲，从而从任意高扭曲的初始化出发都能很快得到一个低扭曲无翻转的球面参数化映射。目前还没有其他方法能做到这一点。

目前还有很多关于球面参数化的问题有待研究，比如 Colin de Verdiere 方程的鲁棒求解器，以及网格过大导致初始化出现精度问题时的解决方法。对于前者，我们尝试设计了一个求解器，该求解器虽然很快，但无法保证得到一个无翻转的结果，因此未在本文提及。对于后者，[40]提出了平面情形的解决方法，使用的就是多分辨率技术。然而，正如前文所说，这种技术需要将网格中的顶点逐个移除再按逆序插入回去，而对于大规模网格来说是非常费时的。在这里，我们提出这样一种解决思路：既然 Tutte 方法是对整体求解的，而多精度技术是对每个顶点单独处理的，那么我们不妨设计一个中和策略，即对网格从粗到精分层，在分层的过程中使用多分辨率技术，而在求解每层的参数化坐标时使用 Tutte 方法。更具体地，对于一个规模较大的网格[50]，我们在其上采样若干个据点，然后用[7]中的方法构建一个基网格作为原网格的近似。我们先将这个基网格进行参数化，再分别将基网格中每个三角形所涵盖的原网格中的顶点参数化到相应的像三角形内部。对于规模更大的网格，我们可以计算一个从粗到精的基网格序列，使

得每个基网格中的三角形所涵盖下个基网格中的三角形数不超过一个固定阈值（比如 100），直到最精细的网格，即原网格。于是，这种结构构成一个树，树的每一层都代表一个基网格，每个节点代表基网格中的三角形，其子节点就是它所涵盖的下个基网格中的所有三角形。构建这个树的过程要用到多分辨率技术，而树的每个节点的参数化要用到 Tutte 方法。在几何优化的过程中，我们可以将原网格的顶点按树的节点分块，然后按照这个数的深度或者广度优先遍历顺序来进行**块坐标下降**(block coordinate descent)。这样可以保证每块内都能使用各种几何优化求解器同时使得子问题的规模不会太大。以上只是一个大概的思路，这个思路适用于一般的超大规模网格处理，但很多细节值得仔细研究，因此我们将它作为下一步的工作目标。

另外，除了平面和球面，也有其他参数域上的网格参数化问题。球面参数域只适合 0 亏格闭曲面网格的整体参数化。对于亏格更高的曲面网格，可以考虑双曲平面作为参数域，相关的参数化问题也非常值得研究，我们也将这部分工作留给未来。更长远的目标则是用一个统一的框架来解决所有网格参数化问题，对于目前来说这显然是相当困难的，但值得尝试。

参 考 文 献

[1] 彭家贵，陈卿. 微分几何. 高等教育出版社.

[2] Donald G. Anderson. Iterative Procedures for Nonlinear Integral Equations. *J. ACM* 12, 4 (1965), 547-560.

[3] M. Botsch, L. Kobbelt, M. Pauly, P. Alliez, B. Levy. Polygon Mesh Processing. *A K Peters* (2010).

[4] M. Botsch, S. Steinberg, S. Bischoff, L. Kobbelt. OpenMesh – a generic and efficient polygon mesh data structure.

[5] S. Claici, M. Bessmeltsev, S. Schaefer, and J. Solomon. Isometry-Aware Preconditioning for Mesh Parameterization. *Comp. Graphics. Forum* 36, 5 (2017), 37-47.

[6] Keenan Crane. Discrete Differential Geometry: An Applied Introduction.

[7] M. Eck, T. DeRose, T. Duchamp, H. Hoppe, M. Lounsbery, and W. Stuetzle, Multiresolution Analysis of Arbitrary Meshes. *ACM SIGGRAPH* (1995), 173-182.

[8] Michael S. Floater. Parameterization and smooth approximation of surface triangulations. *Computer Aided Geometry Design* 14 (1997), 231-250.

[9] Michael S. Floater. One-to-one Piecewise Linear Mappings over Triangulations. *Math. Comput.* 72 (2003), 685-696.

[10] Michael S. Floater and Kai Hormann. Surface parameterization: a tutorial and survey. *In Advances in Multiresolution for Geometric Modelling.* Springer,157-186.

[11] Xiao-Ming Fu, Yang Liu, and Baining Guo. Computing locally injective mappings by advanced MIPS. *ACM Trans. Graph.* 34, 4 (2015), 71:1-71:12.

[12] Craig Gotsman, Xianfeng Gu, and Alla Sheffer. Fundamentals of Spherical Parameterization for 3D Meshes. *ACM Trans. Graph.* (2003), 358-364.

[13] Xianfeng Gu, Steven Gotler, and Hugues Hoppe. Geometry Images. *ACM SIGGRAPH* (2002), 356-361.

[14] X. Gu, Y. Wang, T. F. Chan, P. M. Thompson, and S.-T. Tau. Genus Zero Surface Conformal Mapping and its Application to Brain Surface Mapping. *IEEE Transaction on Medical Imaging,* 23, 7 (2004).

[15] Xianfeng Gu and Shing-Tung Yau. Computing Conformal Structures of Surfaces. *Communications in Information and Systems.* 2, 2 (2002), 121-146.

[16] Xianfeng Gu and Shing-Tung Yau. Global Conformal Surface Parameterization. *Symposium on Geometry Processing.* (2003), 127-137.

[17] S. Haker, S. Angement, S. Tannenbaum, R. Kikinis, G. Sapiro, and M. Halle. Conformal Surface Parameterization for Texture Mapping. *IEEE TVCG* 6, 2 (2000), 181-189.

[18] Kai Hormann and Gunther Greiner. MIPS: An Efficient Global Parameterization Method. *Curve and Surface Design: Saint Malo 1999.* Vanderbilt University Press (2000), 153-162.

[19] Xin Hu, Xiao-Ming Fu, and Ligang Liu. Advanced Hierarchical Spherical Parameterizations. *IEEE T. Vis. Comput. Gr.* 24, 6 (2018), 1930-1941.

[20] Martin Isenburg, Stefan Gumhold, and Craig Gotsman, Connectivity Shapes. *Proceedings of IEEE Visualization* (2001), 135-142.

[21] Shahar Z. Kovalsky, Meirav Galun, and Yaron Lipman. Accelerated Quadratic Proxy for Geometric Optimization. *ACM Trans. Graph.* 35, 4 (2016), 134:1-134:11.

[22] H. Laga, Y. Guo, H. Tabia, R. B. Fisher, M. Bennamoun. 3D Shape Analysis: Fundamentals, Theory, and Applications. *Wiley* (2019).

[23] Ligang Liu, Chunyang Ye, Ruiqi Ni, and Xiao-Ming Fu. Progressive Parameterizations. *ACM Trans. Graph.* 37, 4 (2018), 41:1-41:12.

[24] Ligang Liu, Lei Zhang, Yin Xu, Craig Gotsman, and Steven J Gotler. A Local /Global Approach to Mesh Parameterization. *Comp. Graphics. Forum (SGP)* 27, 5 (2008), 1495-1504.

[25] Yurii Nesterov. A method of solving a convex programming problem with convergence rate to (1/k2). *Soviet Mathematics Doklady,* 27 (1983), 372-376.

[26] Yurii Nesterov. Lectures on Convex Optimization. *Springer International Publishing* (2018).

[27] J. Nocedal and S. Wright. Numerical Optimization. *Springer Science & Business Media* (2006).

[28] Y. Peng, B. Deng, J. Zhang, F. Geng, W. Qin, L. Liu. Anderson Acceleration for Geometry Optimization and Physics Simulation. *ACM Trans. Graph.* 37, 4 (2018), 42:1-42:14

[29] C. G. Petra, O. Schenk, M. Anitescu. Real-time Stochastic Optimization of

Complex Energy Systems on High-performance Computers. *Computing in Science & Engineering* 16, 5 (2014), 32-42.

[30] C. G. Petra, O. Schenk, M. Lubin, K. Gartner. An augmented incomplete factorization approach for computing the Schur complement in stochastic optimization. *SIAM J. Sci. Comp.* 36, 2 (2014), C139-C162.

[31] Emil Praun and Hugues Hoppe. Spherical Parameterization and Remeshing. *ACM SIGGRAPH* (2003), 340-350.

[32] Michael Rabinovich, Roi Poranne, Daniele Panozzo, and Olga Sorkine-Hornung. Scalable Locally Injective Mappings. *ACM Trans. Graph.* 36, 2 (2017), 16:1-16:16.

[33] S. Saba, I. Yavneh, C. Gotsman, and A. Sheffer, Practical Spherical Embedding of Manifold Triangle Meshes. *Proceedings of Shape Modeling International* (2005).

[34] P. Sander, J. Snyder, S. Gotler, and H. Hoppe. Texture Mapping Progressive Meshes. *ACM SIGGRAPH* (2001) 409-416.

[35] P. Sander, Z. Wood, S. Gotler, J. Synder, and H. Hoppe, Multi-chart Geometry Images. *ACM Symposium on Geometry Processing* (2003).

[36] Ariel Shamir. A Formulation of Boundary Mesh Segmentation. *International Symposium on 3D Data Processing, Visualization and Transmission* (2004), 82-89.

[37] Ariel Shamir. A Survey on Mesh Segmentation Techniques. *Comp. Graphics. Forum* 27, 6 (2008), 1539-1556.

[38] Avner Shapiro and Ayellet Tal. Polyhedron Realization for Shape Transformation. *The Visual Computer* 14, 8 (1998), 429-444.

[39] Alla Sheffer, Emil Praun, and Kenneth Rose. Mesh parameterization methods and their applications. *Found. Trends. Comput. Graph. Vis.* 2, 2 (2006), 105-171.

[40] Hanxiao Shen, Zhongshi Jiang, Denis Zorin, and Daniele Panozzo. Progressive Embedding. *ACM Trans. Graph.* 38, 4 (2019), 32:1-32:13.

[41] Anna Shtengel, Roi Poranne, Olga Sorkine-Hornung, Shahar Kovalsky, and Yaron Lipman. Geometric Optimization via Composite Majorization. *ACM Trans. Graph.* 36, 4 (2017), 38:1-38:11.

[42] Breannan Smith, Fernando De Goes, and Theodore Kim. Analytic Eigensystems for Isotropic Distortion Energies. *ACM Trans. Graph.* 38, 1 (2019), 3:1-3:15.

[43] Jason Smith and Scott Schaefer. Bijective Parameterization with Free Boundaries.

ACM Trans. Graph. 34, 4 (2015), 70:1-70:9.

[44] Jian-Ping Su, Chunyang Ye, Ligang Liu, Xiao-Ming Fu. Efficient Bijective Parameterizations. *ACM Trans. Graph.* 39, 4 (2020).

[45] Loring W. Tu. An Introduction to Manifolds, Second Edition. *Springer-Verlag New York* (2011).

[46] W. T. Tutte. How to draw a graph. In *Proceedings of the London Mathematical Society,* Vol. 13, 747-767.

[47] S. Wan, T. Ye, M. Li, H. Zhang, X. Li. An Efficient Spherical Mapping Algorithm and its Application on Spherical Harmonics. *Sci. China Inf. Sci.* (2003), 56(9):1-10

[48] Chunxue Wang, Xin Hu, Xiao-Ming Fu, Ligang Liu. Bijective Spherical Parameterization with low distortion. *Computers & Graphics* 58, (2016), 161-171.

[49] Chunxue Wang, Zheng Liu, Ligang Liu. As-Rigid-As-Possible Spherical Parameterization. *Graph Models* (2014), 76(5):457-67.

[50] Chunyang Ye, Jian-Ping Su, Ligang Liu, Xiao-Ming Fu. Memory-Efficient Bijective Parameterizations for Very-Large-Scale Models. *Pacific Graphics* 39, 7 (2020).

[51] Yufeng Zhu, Robert Bridson, and Danny M. Kaufman. Blended Cured Quasi-newton for Distortion Optimization. *ACM Trans. Graph.* 37, 4 (2018), 40:1-40:14.

[52] Software: Surface Mesh Framework from Xiao-Ming Fu's group. Website: http://staff.ustc.edu.cn/~fuxm/code/index.html#sec_surface_framework