

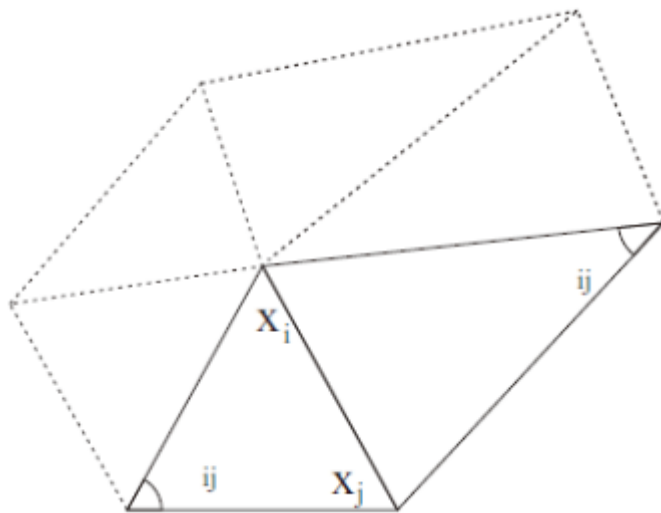
homework 6

Discrete Mean Curvature

- 由Laplace-Beltrami定理

$$K(\mathbf{x}_i) = \frac{1}{2\mathcal{A}_M} \sum_{j \in N_1(i)} (\cot \alpha_{ij} + \cot \beta_{ij}) (\mathbf{x}_i - \mathbf{x}_j)$$

对该点一邻域的点进行上述计算来得到平均曲率的估计：



Discrete Gauss Curvature

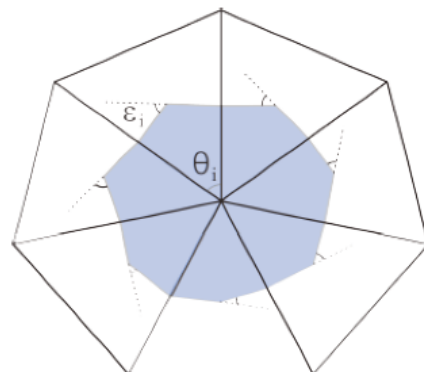
- Gauss-Bonnet定理

高斯曲率的计算就是用

$$\iint_{\mathcal{A}_M} \kappa_G dA = 2\pi - \sum_j \epsilon_j = 2\pi - \sum_{j=1}^{\#f} \theta_j$$

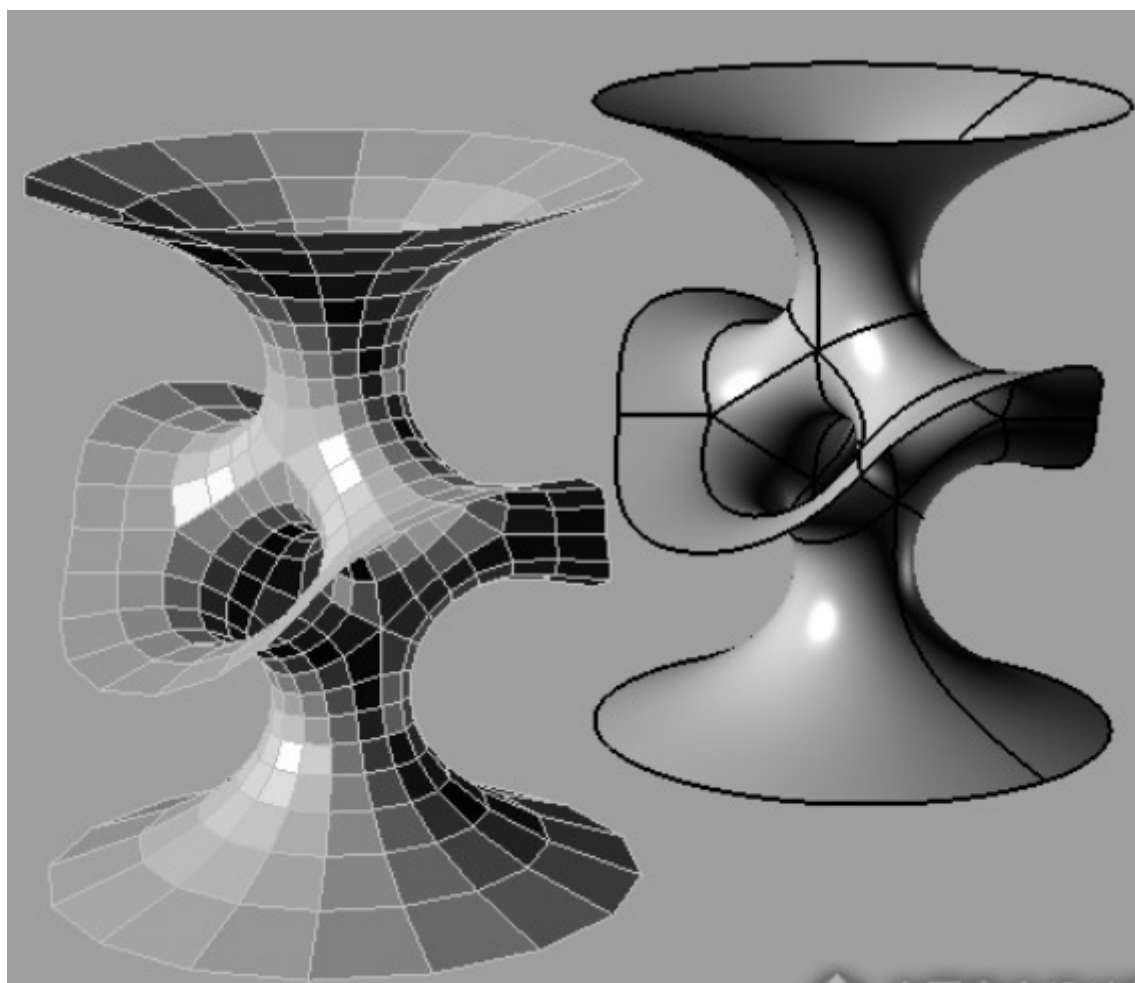


$$\kappa_G(x_i) = (2\pi - \sum_{j=1}^{\#f} \theta_j) / \mathcal{A}_M$$




Minimal Surface

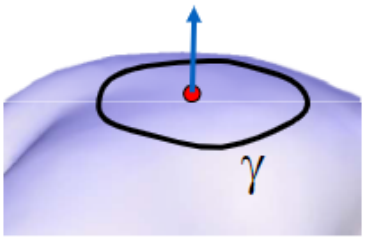
极小曲面就是平均曲率处处为0的曲面。一般来说，极小曲面上的点都是鞍点。



平均曲率流指的是一个曲面上点的运动方向，而如果该曲面为极小曲面，也就是平均曲率处处为0，那么该曲面的平均曲率流也就达到了临界值，不会改变了，可以称为收敛了。平均曲率流一般由微分方程来定义（see https://en.wikipedia.org/wiki/Mean_curvature_flow），但是对于离散的情况，平均曲率流可以分别近似为：



$$\delta_i = \frac{1}{d_i} \sum_{\mathbf{v} \in N(i)} (\mathbf{v}_i - \mathbf{v})$$



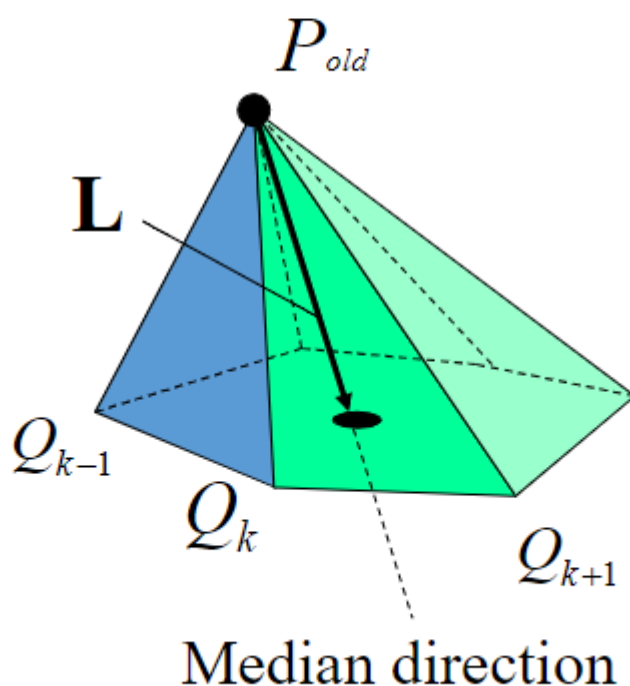
$$\frac{1}{len(\gamma)} \int_{\mathbf{v} \in \gamma} (\mathbf{v}_i - \mathbf{v}) ds$$

$$\lim_{len(\gamma) \rightarrow 0} \frac{1}{len(\gamma)} \int_{\mathbf{v} \in \gamma} (\mathbf{v}_i - \mathbf{v}) ds = H(\mathbf{v}_i) \mathbf{n}_i$$

(1) 离散与连续情况下的平均曲率流

可以看到平均曲率实际上是该点的平均曲率乘上法向量。对于离散的情况，可以直接使用Laplace运算子来得到平均曲率流的一个粗略估计：

$$L(P) = \frac{1}{n} \sum_{i=1}^n \overrightarrow{PQ_i} = \frac{1}{n} \sum_{i=1}^n Q_i - P$$

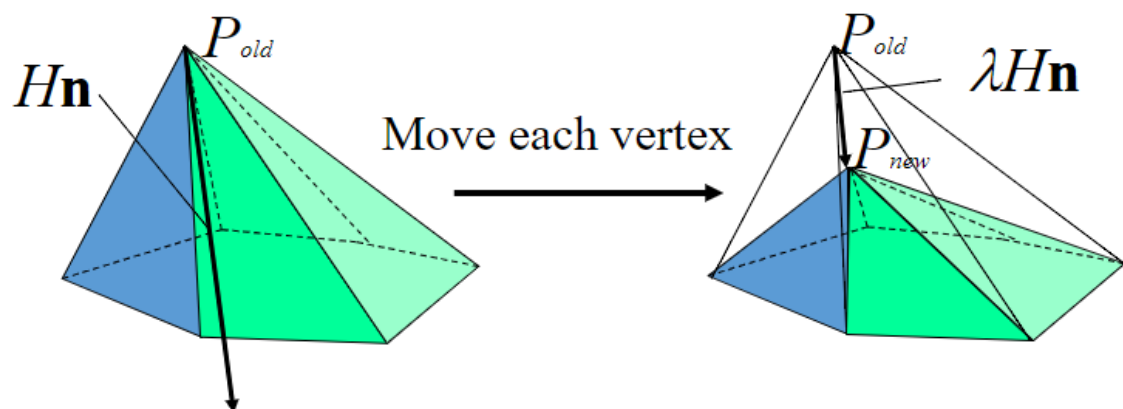


如果我们想将一个曲面变成极小曲面，只要将每个点往平均曲率流方向去迭代运动即可：

$$P_{new} \leftarrow P_{old} + \lambda \boxed{H(P_{old})} \boxed{\mathbf{n}(P_{old})}$$

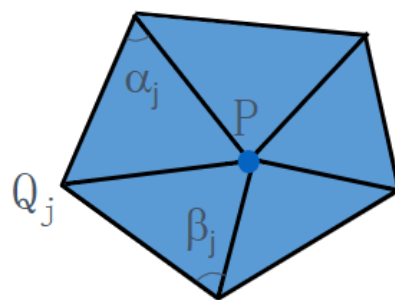
Speed = discrete mean curvature

Direction = normal



可以根据图（1）中的方法，也可以根据前面介绍的Laplace-Beltrami定理来计算平均曲率，并乘上估计的法向量，来得到平均曲率流：

$$H\mathbf{n} = \frac{\nabla_P A}{2A}$$



$$H\mathbf{n} = \frac{1}{4A} \sum_j (\cot \alpha_j + \cot \beta_j) (\mathbf{P} - \mathbf{Q}_j)$$

这样就得到了离散极小曲面的局部迭代法：

- 找到边界
- 固定边界顶点
- 对每个内部顶点
 - 找顶点1-邻域
 - 更新其坐标
- 迭代
- 更新所有顶点法向

algorithm

需要注意的是，只能对非封闭曲面操作，也就是有边界的三角网格。边界点保持不变。一般 λ 取值为0.1。

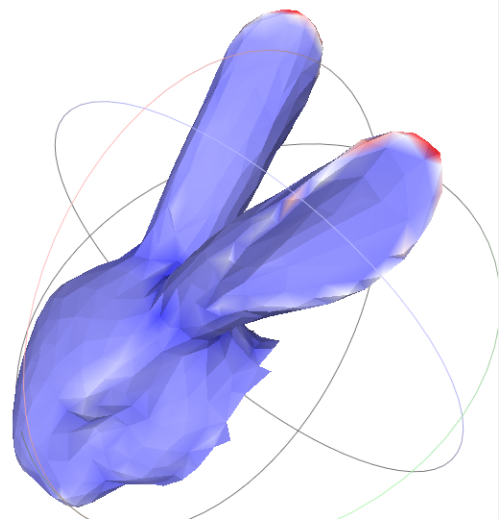
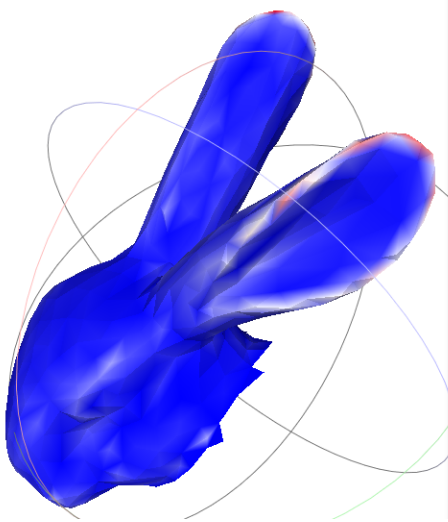
作业情况

由于我在当前的实验室暂时没有windows系统的电脑，所以我没有使用Utopia框架。为了方便文件读写，我使用的了OnePiece库中的部分代码用来读写网格。

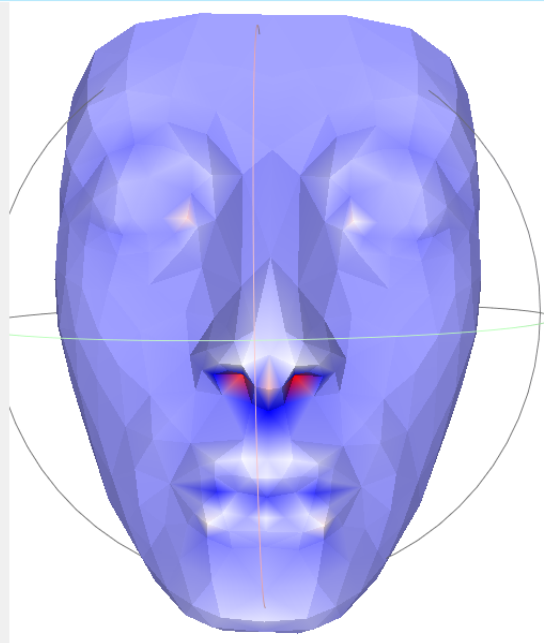
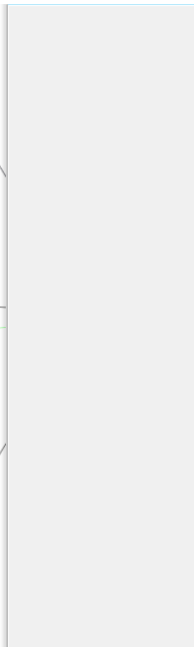
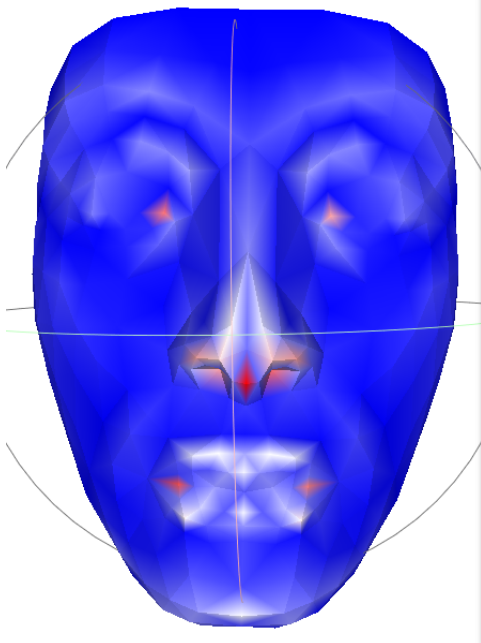
在网格的处理部分，我实现了HalfEdge数据结构，用来寻找一邻域的顶点，同时使用HalfEdge中孪生边的设定，很容易找到closed mesh的边界在哪里。

在作业中，我实现了平均曲率与高斯曲率的计算与可视化，以及基于平均曲率流来生成极小曲面的算法。也许是我的实现可能哪里有问题（求解部分包含了面积计算，三角形外心的计算，三角形种类的判断等），在处理bunny与david时，会产生NaN，有时候也会出现一些异常面片。

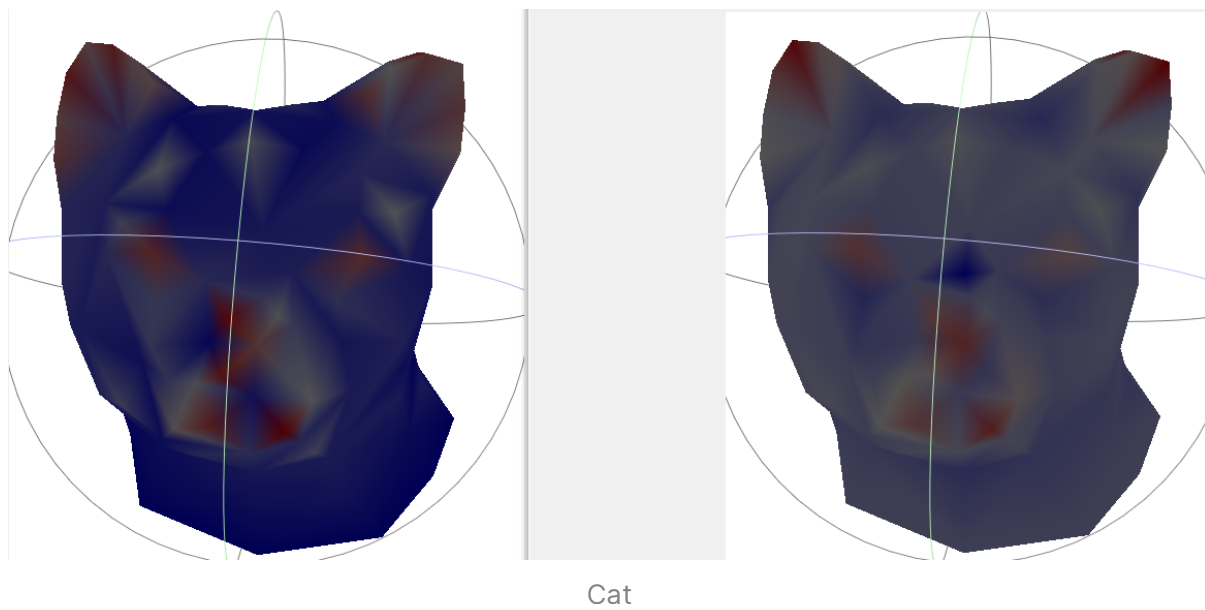
下图中展示了平均曲率与高斯曲率（红色表示曲率最大，蓝色表示曲率比较小，白色是中间的颜色，也就是曲率由小到大为蓝→白→红。有时候蓝色占了大多数，不好区分，这个可以通过直方图均衡化来解决，使得可视化效果更好）：



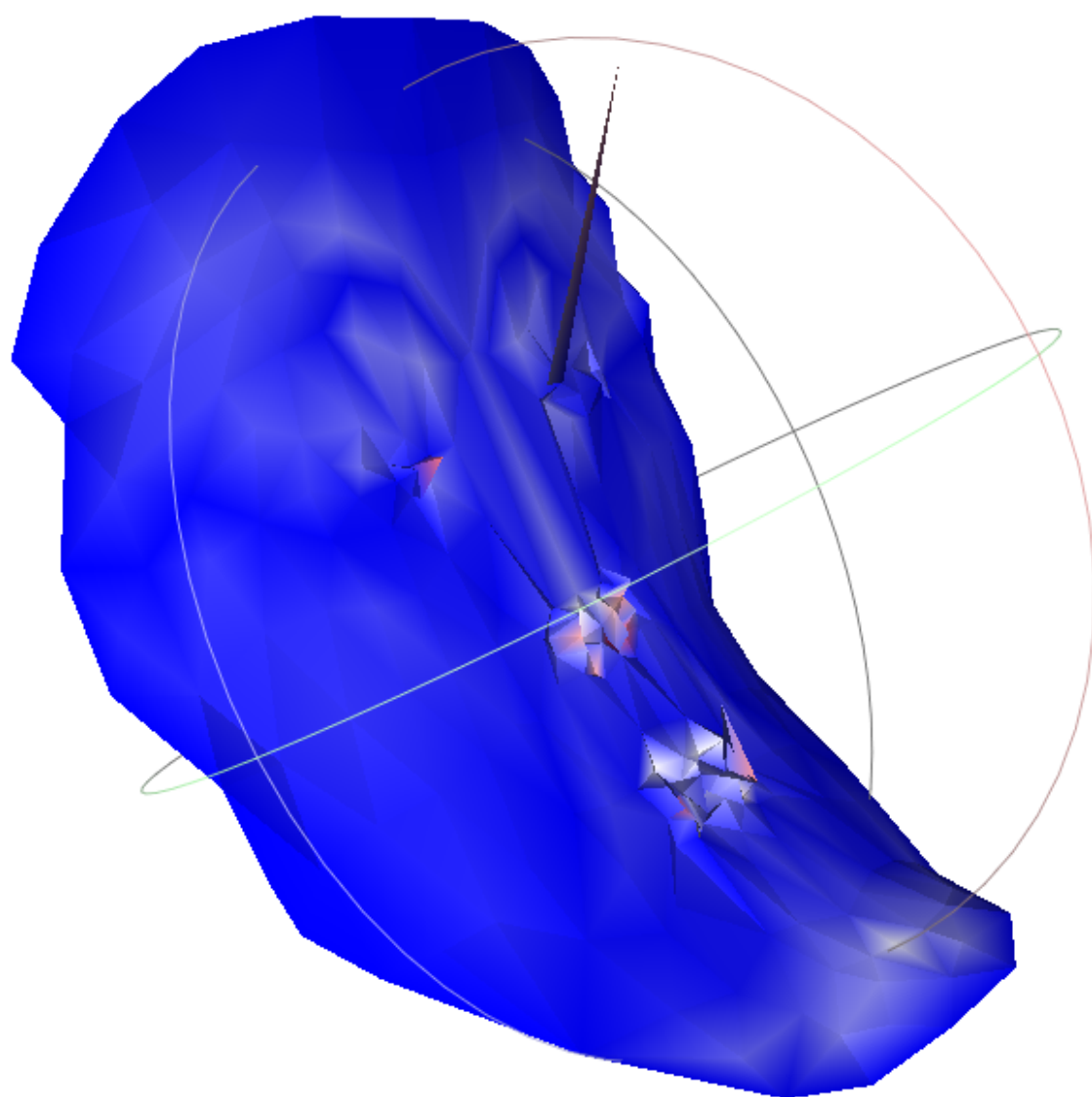
Bunny_head



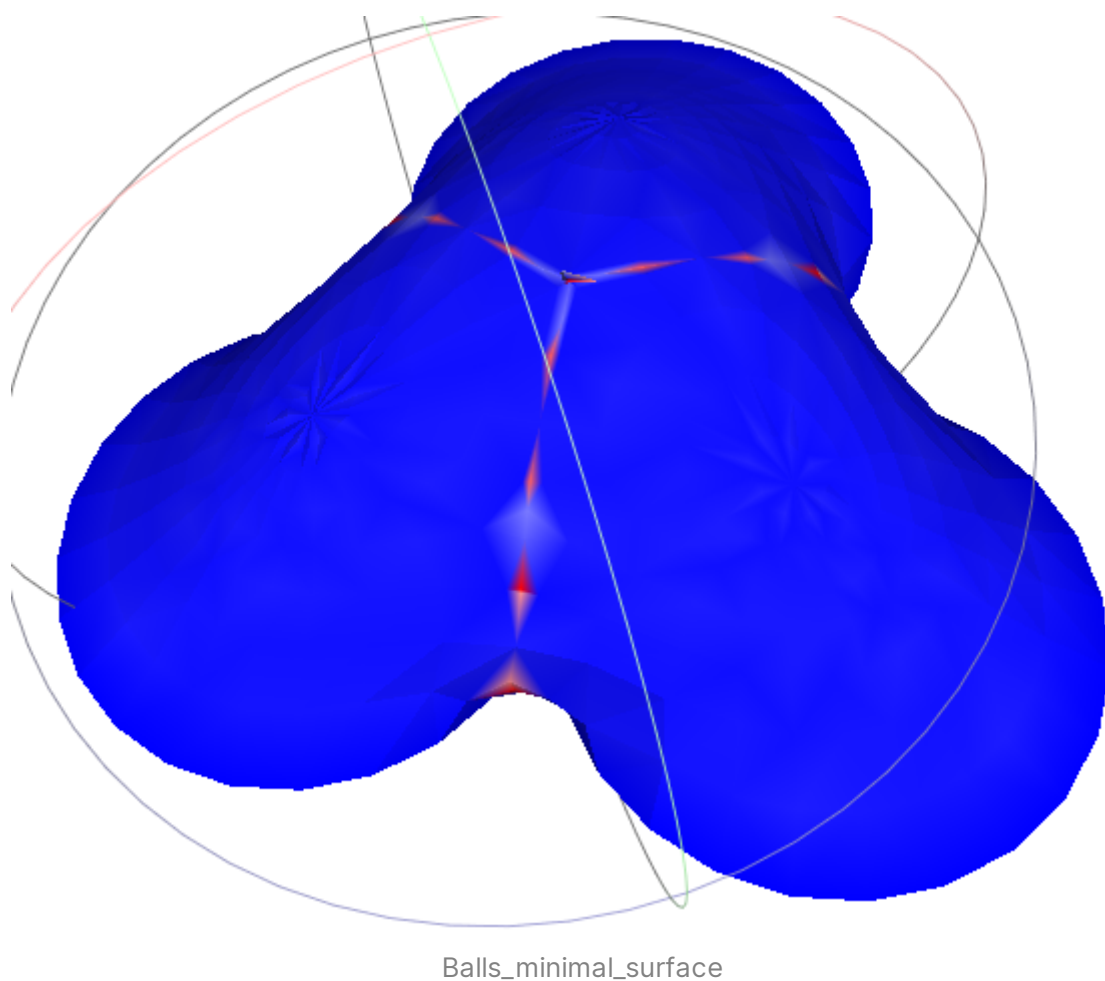
Face



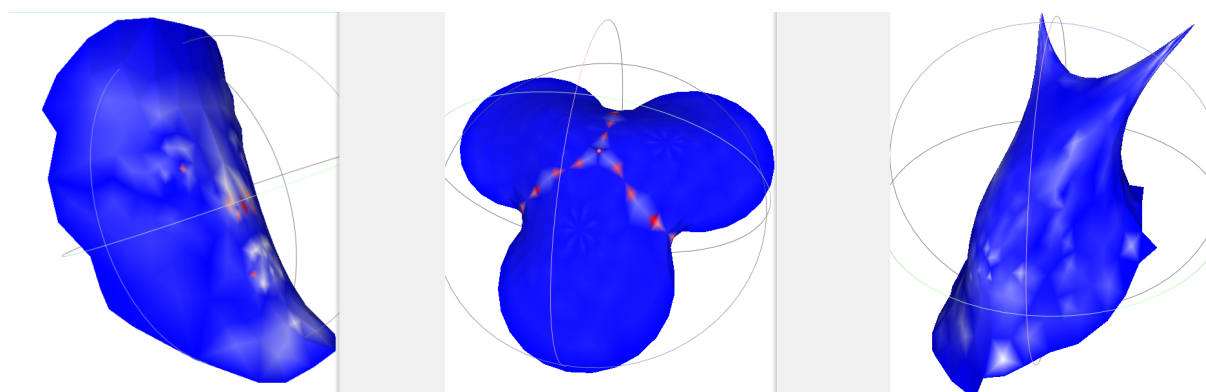
下图展示了生成的极小曲面，可以看到有部分是异常面片，没有收敛:



Face_minimal_surface



在查找资料过程中，我了解到了Laplacian平滑，有时候会产生类似于极小曲面的效果。它的做法非常简单，就是将每个点的位置更新成为邻域顶点的平均值。为了对比，我也实现了这个算法。下面是使用Laplacian平滑算法生成的曲面：



Laplacian Smoothing(face/balls/bunny)

本次作业就完成了这些部分。由于本周的事情有点多，所以我没有足够的时间找到异常面片以及NaN出现的原因。之后会继续寻找。

代码公布在

<https://github.com/MyEvolution/Dragon/tree/main/src/Geometry/TriangleMesh>,
是一个未完成的库的一部分，目前我只在ubuntu系统上进行了测试，所以可能不方便
助教评测。

谢谢老师与助教。