USTC_CG HW5 ARAP&ASAP

张继耀,PB20000204

2023 年 6 月 23 日

目录

1 问题介绍

1.1 主要目的

- 实现ASAP和ARAP两种参数化方法，并对各种参数化进行比较
- 进一步熟悉三角网格的数据结构和编程
- 学习和实现矩阵的SVD分解
- 巩固使用Eigen库求解稀疏线性方程组

1.2 实验内容

- 模仿 Paramaterize.h 和 Paramaterize.cpp 新建文件 ASAP.h，ASAP.cpp，ARAP.h，ARAP.cpp 等。在 Attribute.cpp 中模仿已有示例给 ASAP 方法和 ARAP 方法各添加一个按钮。
- 在UEngine中添加功能，主要有
  - 求给定边界的极小曲面
  - 非封闭网格曲面的参数化(圆形边界和正方形边界,两种权重的选取)
  - 显示纹理映射

2 算法设计

2.1 能量函数

对于3D网格中的每个三角形$x_t = \{x_t^0, x_t^1, x_t^2\}$，记参数化后的三角形顶点坐标为$u_t = \{u_t^0, u_t^1, u_t^2\}$。那么在$x_t \rightarrow u_t$之间存在唯一的线性映射，记这个映射的Jacobian矩阵为$J_t(u)$，对每个三角形t事实上是常值。我们希望对应的这个线性映射尽可能接近特定的线性变换L_t，例如仿射、旋转等。若记参数化的集合$u = \{u_1, ..., u_t\}$，对应的线性变换集合$L = \{L_1, ..., L_T\}$。我们可以定义能量函数:

$$E(u, L) = \sum_{t=1}^{T} A_t \|J_t(u) - L_t\|_F^2$$

于是这个问题就变成了优化问题:求参数(u,L)使得(u,L)=$argmin_{(u,L)} E(u, L), s.t L_t \in M$.我们最终需要的只是$u$，在求解时，通过选择不同的$L$可求得不同的结果。也就是下面的ASAP和ARAP。

2.2 ASAP算法

ASAP即AS Similar As Possible。这种方法中参数族M具有形式

$$M = \begin{bmatrix} a & b \\ -b & a \end{bmatrix}, a, b \in \mathbb{R}$$

因此上面的能量函数转化为

$$E(u, L) = \frac{1}{2} \sum_{t=1}^{T} \sum_{i=0}^{2} cot(\theta_t^i) \|(u_t^i - u_t^{i+1}) - \begin{bmatrix} a & b \\ -b & a \end{bmatrix} (x_t^i - x_t^{i+1})\|_F^2$$

我们可以参考论文中的方法来求解:寻找a和b，使得下式取得极小值

$$E(a, b) = \sum_{i=0}^{2} \omega_i \|\nabla e^i\|^2 + \lambda(a^2 + b^2 - 1)^2$$

其中有:$\nabla e^i = u^i - u^{i+1} - \begin{bmatrix} a & b \\ -b & a \end{bmatrix} (v^i - v^{i+1})$，其他量均为常数。记$u^i - u^{i+1} = \begin{pmatrix} \nabla u_x^i \\ \nabla u_y^i \end{pmatrix}$,$v^i - v^{i+1} = \begin{pmatrix} \nabla v_x^i \\ \nabla v_y^i \end{pmatrix}$。让$E$分别对$a$和$b$求偏导，我们有下面的式子:

$$C_1 a + 2\lambda a(a^2 + b^2 - 1) = C_2$$

$$C_1 b + 2\lambda b(a^2 + b^2 - 1) = C_3$$

其中:

$$C_1 = \sum_{i=0}^{2} \omega_i [(\nabla v_x^i)^2 + (\nabla v_y^i)^2],$$

$$C_2 = \sum_{i=0}^{2} \omega_i [\nabla u_x^i \nabla v_x^i + \nabla u_y^i \nabla v_y^i],$$

$$C_3 = \sum_{i=0}^{2} \omega_i [\nabla u_x^i \nabla v_x^i - \nabla u_y^i \nabla v_y^i],$$

取$\lambda = 0$，我们可以解得$a = \frac{C_2}{C_1}$，$b = \frac{C_3}{C_1}$.

根据上式构造线性方程组，还是利用Eigen解方程即可。

2.3 ARAP算法

ASAP即AS Rigid As Possible。这种方法中参数族M具有形式

$$M = \begin{bmatrix} cos\theta & sin\theta \\ -sin\theta & cos\theta \end{bmatrix}, \theta \in [0, 2\pi]$$

此时为非线性方程，上面的方法不再适用。采用Local/Global方法求解，主要有以下步骤:

- 参数化坐标初始化
- Local阶段:固定u，求L_t

  对下式进行SVD分解:

  $$J_t(u) = \sum_{i=0}^{2} cot(\theta_t^i)(u_t^i - u_t^{i+1})(x_t^i - x_t^{i+1})^T = U \sum V^T$$

  取

  $$L_t = UV^T$$

- Global阶段:固定L_t，求u

  求解稀疏方程组:

  $$\sum_{j \in N(i)} [cot(\theta)_{ij} + cot(\theta)_{ji}](u_i - u_j) = \sum_{j \in N(i)} [cot(\theta)_{ij} L_{t(i,j)} + cot(\theta)_{ji} L_{t(j,i)}](x_i - x_j)$$

- 重复以上步骤，直到收敛到误差范围内或者迭代了指定步数。

3 结果展示

4 总结与讨论

从测试结果可以看出迭代次数会略有影响，但影响不大，基本上肉眼难以察觉。一般只需迭代1至2次就能得到很好的结果，迭代次数过多反而浪费性能，不划算。

从图中可以看出ASAP和ARAP两种算法都是优于HW4的参数化方法的。它们生成的网格更均匀、光滑一些。而ARAP更是优于ASAP的，生成的网格基本上十分均匀，没有不规则的地方。综上，ARAP的性能是十分优良的。

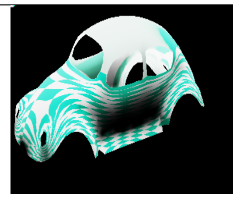
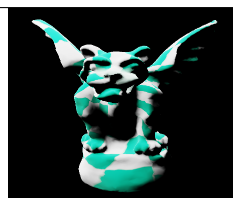
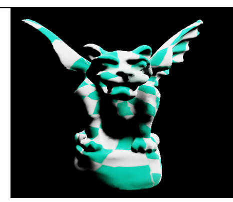
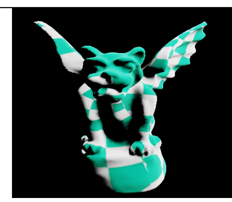
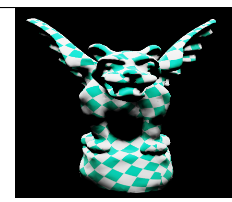
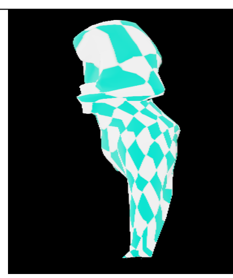
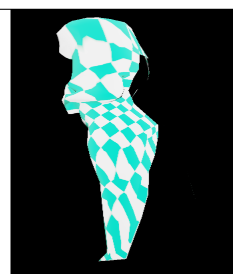
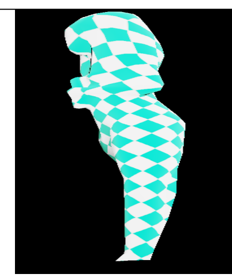
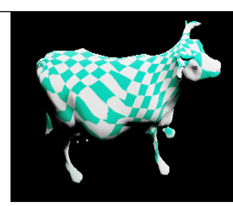
使用方法 测试例子	**Uni参数化**	**Cot参数化**	**ASAP**	**ARAP(10次)**
Beetle				
Gar				
Isis				
Cow				

表 1: 主要结果

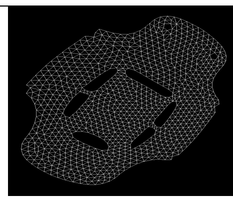
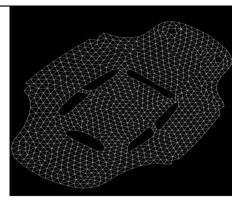
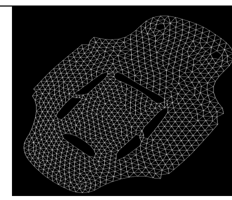
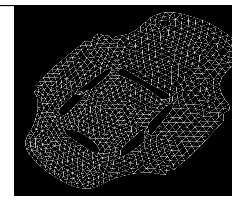
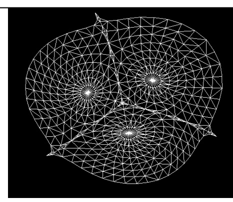
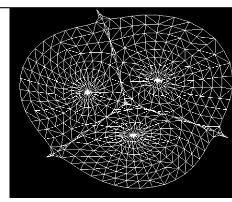
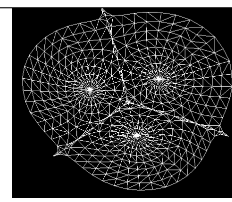
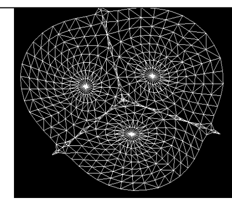
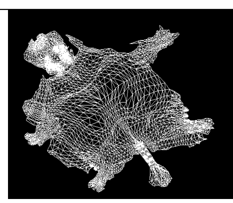
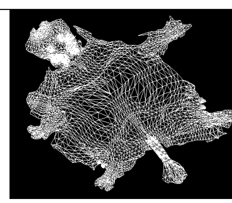
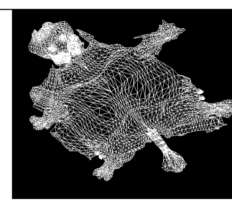
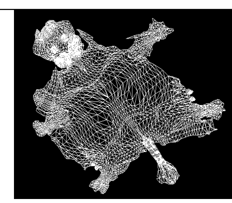
迭代次数 测试例子	**1次**	**2次**	**5次**	**10次**
Beetle				
Balls				
Cow				

表 2: ARAP不同迭代次数对结果的影响