计算机图形学第五次作业实验报告

- ID：999
- 姓名：袁保杰
- 学号：PB21111714

实验描述

本次实验要求在以下两个作业中任选其一完成：

可选作业一

- 实现论文 [A Local/Global Approach to Mesh Parameterization] 中介绍的 ARAP (As-rigid-as-possible) 网格参数化方法
- （Optional）实现论文中的另外两种参数化
  - ASAP（As-similar-as-possible）参数化算法
  - Hybrid 参数化算法
- 使用测试纹理和网格检验实验结果

可选作业二

- 实现论文 [Laplacian Surface Editing] 中介绍的 Laplace 表面编辑方法
- 实时拖动区域显示结果
- 不同权重的 Laplacian 的选取，至少实现均匀权重和余切权重
- （Optional）实现一个局部的 Laplace 表面编辑方法，即可以在局部的区域进行表面编辑

我选择了可选作业一，完成了 ARAP 参数化方法的实现，未完成 ASAP 和 Hybrid 参数化方法。

算法描述

基本思想

Tutte 参数化要求边界固定，虽然实现简单，且保证了参数化结果不发生翻转，但牺牲了参数化的效果；相比之下，边界不固定的参数化方法提高了边界点移动的自由度，可以获得更好的结果。

论文的基本思想是：先将网格上的所有三角分片各自等距变换到平面上，然后将这些平面上的三角形进行变形映射，全局整合为平面三角网格，作为参数化结果。

对于上面整合过程中三角形发生的变形，其中的线性变换部分（用 Jacobi 矩阵描述），根据实际需求，需要尽可能接近某一小类的线性变换：

- ARAP 需要接近于旋转变换（尽可能刚性）
- ASAP 需要接近于相似变换（尽可能相似）

将上面的问题形式化，将三角网格上的每个三角分片标号为 $1...T$，对于每个三角形 t：

- 对于其在平面上的变形映射，考虑其 2 x 2 的 Jacobi 矩阵 $J_t(u)$，它是该映射的最佳线性逼近
- L_t 为 t 对应的优化目标矩阵，Jacobi 矩阵需要尽可能地接近 L_t
- A_t 为三角形的面积

得到下面的能量优化问题：

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

展开 Jacobi 矩阵：

$$E(u,L)=\frac{1}{2}\sum_{t=1}^{T}\sum_{i=0}^{2}\cot(\theta_t^i)||(u_t^i-u_t^{i+1})-L_t(x_t^i-x_t^{i+1})||^2$$

其中 x_t^i 为原三角形展平到平面上之后的顶点坐标，u_t^i 为映射后三角形顶点的坐标，θ_t^i 为 (x_t^i,x_t^{i+1}) 的对角。

优化方法

上述的能量优化问题存在两个变量 L_t 和 u，采用 Local/Global 两步迭代法来求解：

- Local Phase：固定 u，对于每个三角形，求解其对应的最佳旋转变换 L_t。
  - 对于 ARAP 方法，根据 Procrustes 分析的结果，将雅可比矩阵做奇异值分解，得到 $J = U\Sigma V^T$，然后取 Σ 为单位矩阵即可（即奇异值取 1）
- Global Phase：固定 L_t，求解 u。将梯度置零，求解下面的稀疏线性方程组即可。稀疏线性方程组的系数是确定的，因此系数矩阵只需要分解一次，便于加速求解。

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

代码细节

我在 `utils` 文件夹中定义了 `ARAP` 类：

```
class ARAP
{
public:
    ARAP(std::shared_ptr<PolyMesh> mesh_3d, std::shared_ptr<PolyMesh> initial_param);

    std::shared_ptr<PolyMesh> get_param() const;

    void local_phase();
    void global_phase();
    void unitlize();

private:
    std::vector<std::array<OpenMesh::Vec2f, 3>> _x;
    std::shared_ptr<PolyMesh> _u;
    std::vector<Eigen::Matrix2f> _L;
    std::shared_ptr<PolyMesh> _mesh_3d;
    std::unordered_map<int, bool> _m;

    std::shared_ptr<Eigen::SimplicialLDLT<Eigen::SparseMatrix<float>>> _solver;
};
```

然后在 `node_arap.cpp` 中使用：

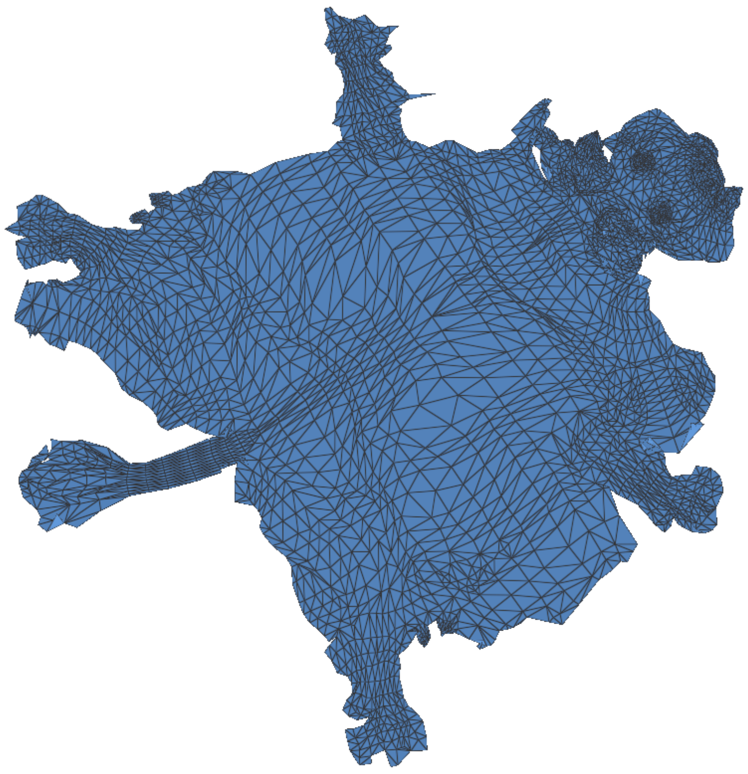
```
    // ...
    // Initial Setup
    auto arap = ARAP(halfedge_mesh, initial_param_mesh);

    // Iterations
    while (iteration_num--)
    {
        arap.local_phase();
        arap.global_phase();
    }
    arap.unitlize();

    // The result UV coordinates
    auto geometry = openmesh_to_operand(arap.get_param().get());
    // ...
```

实现上，还需要注意以下细节：

- 初始参数化：论文使用的优化方法属于迭代算法，因此需要一个初始参数化作为 u 使用，要求是：不能出现翻转、扭曲不太大、并且生成足够快，比如 Homework 4 实现的 Floater97 shape-preserving method
- 固定点：将参数化结果视作一个刚体，其存在平移和旋转的自由度，可以任意选择一条边的两个顶点钉死，防止参数化结果漂移。方法是，在解方程组时，如果想要钉死顶点 i，就在第 i 个顶点对应的方程中，在左边的第 i 项加入一个很大的系数，右边加入系数乘以点的坐标值即可。

实验结果

使用 Cow.usda 的结果如下：

材质贴图结果	参数化结果
	

使用 Gargoyle.usda 的结果如下：

材质贴图结果	参数化结果