Homework7 Program Report

张景浩 PB20010399

2023.5.7

1 问题描述

参考文献 [1] 中的 As-Rigid-As-Possible 形状插值算法，实现二维平面中网格的插值形变，并参考文献 [2] 解决旋转一致性。

2 算法原理

2.1 As-Rigid-As-Possible 形状插值算法

设源网格中的一个三角面片为 $P_0 = \{\vec{p_{1,0}}, \vec{p_{2,0}}, \vec{p_{3,0}}\}$，目标网格中的三角面片为 $P_1 = \{\vec{p_{1,1}}, \vec{p_{2,1}}, \vec{p_{3,1}}\}$。在平面上从 P_0 到 P_1 有一个变换

$$A\vec{p_{i,0}} + \vec{l} = \begin{pmatrix} a_1 & a_2 \\ a_3 & a_4 \end{pmatrix} \vec{p_{i,0}} + \begin{pmatrix} l_1 \\ l_2 \end{pmatrix} = \vec{p_{i,1}}, \quad i = 1, 2, 3$$

记

$$\delta_0^1 = \vec{p_{1,0}} - \vec{p_{2,0}},\ \delta_0^2 = \vec{p_{2,0}} - \vec{p_{3,0}}$$

$$\delta_1^1 = \vec{p_{1,1}} - \vec{p_{2,1}},\ \delta_1^2 = \vec{p_{2,1}} - \vec{p_{3,1}}$$

所以

$$\begin{pmatrix} \delta_1^1 & \delta_1^2 \end{pmatrix} = A \begin{pmatrix} \delta_0^1 & \delta_0^2 \end{pmatrix}$$

对变换矩阵 A 做 svd 分解

$$A = U\Sigma V$$

记

$$R = UV^T \qquad S = V^T \Sigma V$$

R 为从 P 到 Q 的旋转矩阵，S 为从 P 到 Q 的伸缩矩阵，特别的

$$R = R(\theta) = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix}$$

在变换的 t 时刻 ($t \in [0,1]$)，有变换矩阵

$$A_t = R_t S_t$$

其中

$$R_t = R(t\theta), \quad S_t = (1-t)I + tS$$

所以此时刻的三角面片变换中间态 $P_t = \{\vec{p_{1,t}}, \vec{p_{2,t}}, \vec{p_{3,t}}\}$，记

$$\delta_t^1 = \vec{p_{1,t}} - \vec{p_{2,t}},\ \delta_t^2 = \vec{p_{2,t}} - \vec{p_{3,t}}$$

满足

$$\begin{pmatrix} \delta_t^1 & \delta_t^2 \end{pmatrix} = A_t \begin{pmatrix} \delta_0^1 & \delta_0^2 \end{pmatrix}$$

对网格中的每一个三角面片进行上述操作，我们就得到了 t 时刻每一个三角面片的相对坐标 (δ_t^1, δ_t^2)，由这个相对坐标确定的三角面片在刚体变换下是唯一的，根据方程

$$\begin{pmatrix} 1 & -1 & 0 \\ 0 & 1 & -1 \end{pmatrix} \begin{pmatrix} p_{1,t} \\ p_{2,t} \\ p_{3,t} \end{pmatrix} = \begin{pmatrix} \delta_t^1 \\ \delta_t^2 \end{pmatrix}$$

我们可以得到 t 时刻网格中每一个点的坐标与相对坐标的关系。因为在网格中一个三角面片就能确定一组相对坐标，而大多数点存在于不止一个三角面片中，这就导致 t 时刻网格全局坐标和全局相对坐标构成的线性方程组是超定的，我们可以通过最小二乘解得当前时刻的网格坐标。

2.2 旋转一致性

在求解旋转矩阵 $R(\theta)$ 时，通过反三角函数我们只能得到 $[-\pi, \pi]$ 的旋转角，这就会造成超过这个范围的旋转角数值解与真实值相差 $k\pi$，会造成在不同三角面片中的同一条边的旋转角相差过大，导致三角面片的旋转不连续。为了解决这个问题，寻找三角面片旋转角中不连续的部分，并对其进行修复直到其满足连续性。

在具体的实现中，我们对每个三角面片的旋转角，使用源网格和目标网格对应的三角面片的面积进行加权平均，得到的这个平均旋转角记为 α，与 α 最近的 $k\pi$ 可以看作是源网格到目标网格变换的整体旋转趋势。再对每个旋转角到平均旋转角的距离按同样的权重进行加权平

均，得到每个旋转角与平均旋转角的平均差异。与 α 正负号相异且距离大于平均差异的旋转角 θ 一定是不连续的，对其进行修正，得到 $\theta = \theta + sgn(\alpha)\pi$。在经过这样的修正后，得到的旋转角基本可以看作是连续的。

3 代码实现

本次实验使用 matlab 进行编译，代码框架由 Agreement7 给出，算法具体的实现详见 ARAP_pretreatment.m 和 ARAP_interp.m 文件。

4 测试结果

由于测试结果本身是一个动态过程，所以在报告中无法做具体的展示，请运行 morph_main.m 文件进行查看。下面给出网格变换部分时刻的中间态

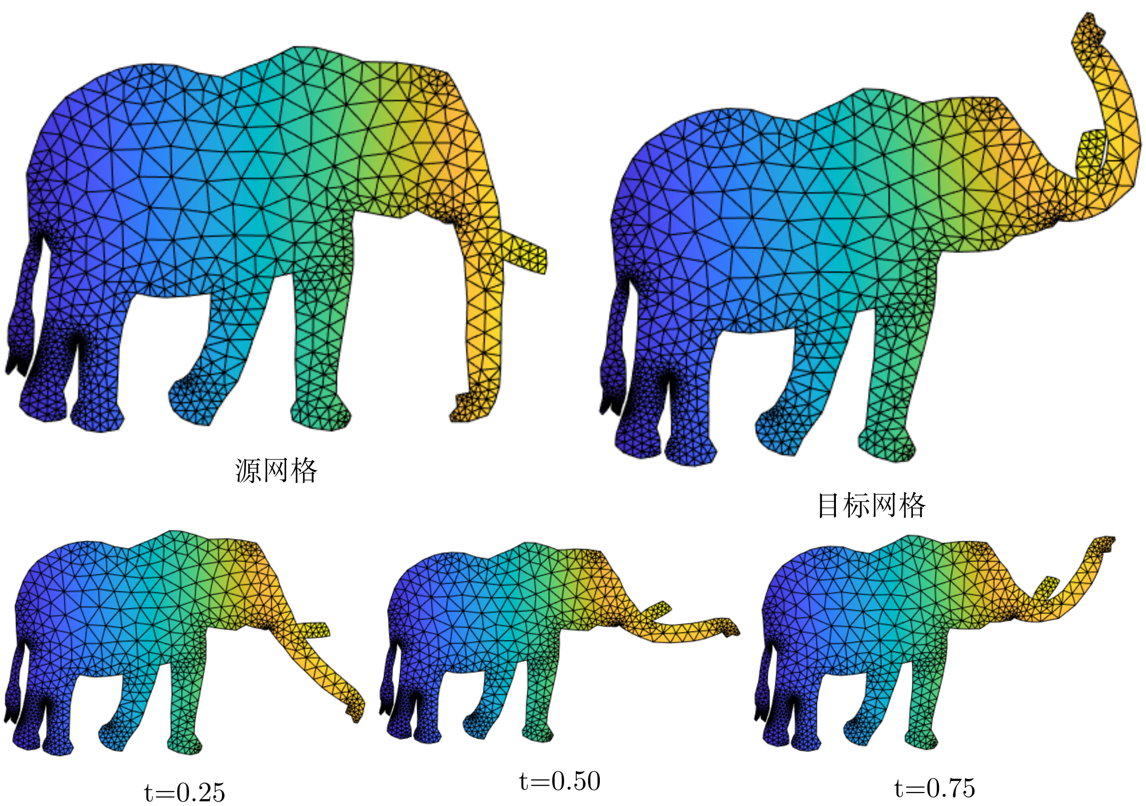

图 1: t=0.25,0.50,0.75 时的中间态

5 总结

观察变换过程我们发现，在没有解决旋转一致性问题时，大象鼻子前半部分和后半部分的旋转是相反的，与我们的预期效果不同。这是因为前半部分的旋转角在逆时针方向上超出 $[-\pi, \pi]$ 的范围，在计算时变成了顺时针方向，导致与后半部分旋转不一致；而在按照 2.2 中方法处理之后，大象前半部分和后半部分的旋转方向就一致了。在整个旋转过程中，每一个三角面片都没有产生太过突兀的形变，都是以一种十分均匀自然的变换完成了由源图像到目标图像的过渡，可以看出 ARAP 算法对于变换保形的优秀性质。

参考文献

[1] M. Alexa et al. As-Rigid-As-Possible Shape Interpolation. SIGGRAPH 2000

[2] Baxter et al. Rigid shape interpolation using normal equations. Symposium on Non-photorealistic animation and rendering 2008