中文内容摘要

我们提出了一种基于边长的形状插值方法. 其在两个兼容（具有相同连接关系）的三维网格之间插值. 该方法有以下几个优点. 首先, 一般的插值边长的形状插值算法还往往需要同时插值其它几何量, 例如二面角. 而我们只使用了边长作为插值量, 因此得到的结果与期望边长的误差更小. 而其它只利用边长作为插值量的算法也因为需要先计算出二面角再进行间接重建, 导致边长误差变大, 不如我们的方法. 其次, 与一般线性插值边长的做法不同, 我们线性插值的是边长平方. 因为有文章已经证明了这样能保证插值结果是有界扭曲的. 不过该文章是应用在四面体网格上的. 因此我们将其迁移应于三维网格上. 并且我们将其结果的表面作为初始化网格可以取得快速收敛的效果. 最后, 我们使用牛顿法对能量进行优化, 并且利用多种技巧将海森矩阵快速近似为正交矩阵. 同时因为直接以顶点位置作为变量, 不需要间接重建. 因此相比于已有的只基于边长的插值方法, 该方法在速度方面也更快.

关键词：计算机图形学; 形状插值

Abstract

We propose a shape interpolation method based on edge lengths. It interpolates between two compatible(with the same connection relationship) 3D meshes. This method has the following advantages. First, the shape interpolation algorithm that interpolates the edge lengths usually needs to interpolate other geometric quantities at the same time, such as dihedral angles.Instead,we only use edge lengths as the interpolation quantity. So the error between the result and the expected edge lengths is smaller. Other algorithms that only use the edge lengths as the interpolation quantity need to calculate the dihedral angles first and then reconstruct the mesh indirectly, resulting in a larger edge lengths error.So the result is not as good as our method. Secondly, unlike the general linear interpolation method of edge lengths, we linearly interpolate the square of the edge lengths.Because there is an article that has proved that this can ensure that the interpolation result has bounded distortion.This article is applied to tetrahedral meshes. So we apply it to 3D meshes. And we take the surface of the result as the initial mesh to achieve rapid convergence. Finally, we use Newton's method to optimize the energy, and use several tricks to quickly approximate the Hessian matrix with an orthogonal matrix. At the same time, the vertex position is directly used as a variable. So there is no need for indirect reconstruction. Therefore, compared with the existing interpolation methods based only on edge lengths, this method is also faster in speed.

Key Words: computer graphics; shape interpolation

第一节　引言

形状插值问题是一个经典的图形学问题, 能将一个几何模型光滑的变换到目标模型. 其可应用于很多方面, 比如计算机动画 [14] 等.

衡量一个形状插值算法的好坏的一个重要标准是插值结果是否是有界扭曲的.Chen 等 [1] 提出了一种有界扭曲的四面体网格插值算法. 该算法的有界扭曲既是等距变换意义上的, 又是共形变换意义上的. 其主要的理论依据是有界的共形扭曲和等距扭曲所构成的回拉度量 (pullback metric) 空间是凸的 [5]. 而回拉度量可以写成边长平方的线性表示. 因此可以通过线性插值边长的平方来从理论上保证是有界扭曲的. 还有虽然线性插值边长平方后得到的四面体网格由于内边的邻二面角和一般不等于 2π, 不能直接嵌入到 R^3 空间中, 但可以证明其与 2π 的差是有界的且实验显示其很接近 2π. 因此最后得到的插值结果的边长与目标边长相差不大.

受此启发, 我们也可以线性插值边长的平方, 将其迁移应用到三维网格中. 由于其已经是一个很好的线性插值了边长平方的算法, 并且体网格表面也是一个三维网格, 因此我们将其作为初始化网格可以获得快速收敛的效果. 本方法就是用其初始化顶点坐标.

形状插值问题的相关工作非常多, 其中插值边长的方法占有相当一部分. 不过这些方法往往需要同时插值其它几何量, 如文章 [11, 12] 均是同时插值了边长与二面角, 文章 [6] 则同时插值了边长, 二面角与体积. 但由于插值量过多, 自由度不足, 因此这些方法的边长误差均较大. 而只插值边长的方法则比较少, 近些年只有文章 [9]. 该方法的一个不足是为了结果的鲁棒性分了两步进行重建, 导致边长误差在重建过程中放大. 为解决这一问题, 我们直接使用了顶点位置作为变量定义能量.

因此, 我们的方法既避免文章 [9] 的缺点, 直接使用顶点坐标作为变量, 省去了重建过程, 能得到边长误差更小的结果. 而且, 像文章 [1] 一样, 插值的是边长平方, 能够从理论上保证插值序列是有界扭曲的.

第二节　相关工作

形状插值问题研究已经有几十年历史了, 其相关工作非常多.

Sedeberg 等 [10] 早在 1993 年就研究了二维多边形插值问题. 如之前所说, 形状插值可以基于许多标准, 比如角度、仿射变换矩阵、边长.

Alexa 等 [2] 提出了 ARAP 方法, 即是基于仿射变换矩阵进行形状插值. 该方法首先将仿射变换矩阵分解成旋转部分和放缩部分, 然后分别对旋转部分和放缩部分进行插值, 最后将这两部分进行混合, 得到了目标的仿射变换矩阵.

Winkler 等 [12] 则利用了二面角和边长进行形状插值. 他们将网格分解成多个层次, 形成一棵树, 顶部根节点包含整个网格, 而底部节点则只有几个相邻的三角形面片. 然后将之从下到上一层层地进行重建, 最终得到插值结果.

Wang 等 [11] 也是利用了二面角和边长进行形状插值. 不过, 他们并没有将网格分解成多个层次, 而是在每个面上建立了局部坐标系. 然后利用局部坐标系分别建立了二面角和边长对应的能量. 最终通过求解一个二次多项式的最小值获得最终的结果.

相比于刚才两种方法都利用了二面角,Rojas 等 [9] 则提出了一种只利用边长就得到插值结果的方法. 他们先利用边长求出满足条件的二面角, 再利用二面角得到每条边的旋转矩阵, 最后再求出网格顶点的位置.

第三节　算法

一、插值的几何量

首先, 我们先说明一下为何线性插值边长的平方得到的结果有界扭曲的. 对于一个三角形面, 设其在源网格和目标网格分别为 t_1 和 t_2, 则存在一个变换矩阵 $\boldsymbol{J}$ 使得

$$t_2 = \boldsymbol{J} t_1 \tag{1}$$

对于三角形 t_2 的任意一条边长 e_2, 有 $|e_2|^2 = |\boldsymbol{J}e_1|^2 = (\boldsymbol{J}e_1)^T(\boldsymbol{J}e_1) = e_1^T \boldsymbol{J}^T \boldsymbol{J} e_1 = e_1^T \boldsymbol{M} e_1$, 其中 $\boldsymbol{M} = \boldsymbol{J}^T \boldsymbol{J}$. 因此 $|e_2|^2$ 可以展开成 $\boldsymbol{M}$ 的一次多项式. 故线性插值 $\boldsymbol{M}$ 等价于是线性插值边长的平方. 故只需证明线性插值 $\boldsymbol{M}$ 得到的结果有界扭曲的即可.

设 $\boldsymbol{M}_t = (1-t)\boldsymbol{I} + t\boldsymbol{M}$, 其中 $t \in [0,1]$ 是时间. 则可以证明存在 $\alpha \in [0,1]$ 使得 $c(\boldsymbol{M}_t) \leqslant (1-\alpha)c(\boldsymbol{I}) + \alpha \boldsymbol{C}(\boldsymbol{M}) = (1-\alpha) + \alpha \boldsymbol{C}(\boldsymbol{M})$, 其中 $\boldsymbol{C}(\boldsymbol{M}) = \frac{\lambda_{max}}{\lambda_{min}}$,$\lambda_{max}$ 和 λ_{min} 分别是 $\boldsymbol{M}$ 的最大特征值和最小特征值. 其详细证明可见文章 [4] 中的附录 A. 这可以说明线性插值边长平方可以保证共形变换的扭曲是有界的.

除此之外, 还可以证明对称狄利克雷能量关于矩阵 $\boldsymbol{M}$ 是光滑且凸的. 详细证明可见文章 [1] 的定理 2. 这可以说明线性插值边长平方可以保证等距变换的扭

曲也是有界的.

因此由于线性插值边长的平方有着保证有界扭曲的优势, 我们的方法对源网格和目标网格的对应边长的平方进行线性插值. 即令

$$l_t = \sqrt{(1-t) * l_1^2 + t * l_2^2} \tag{2}$$

其中 l_1, l_2 分别为源网格和目标网格的边长,t 为时间.

二、能量及其优化

为了避免如文章 [9] 需要间接重建的缺点, 我们直接使用了顶点位置作为变量定义能量.

设顶点 i 的坐标为 p_i, 则顶点 i 和顶点 j 连成的边 e_{ij} 的长度为 $\|p_i - p_j\|$. 我们的目标是优化能量使得最终的曲面的边长等于目标边长. 对于 e_{ij}, 定义能量

$$\begin{aligned} E_{ij} &= |\|p_i - p_j\|^2 - l_{ij}^2|^2 \\ E &= \sum_{e_{ij}} E_{ij} \end{aligned} \tag{3}$$

其中 l_{ij} 是 e_{ij} 的目标边长.

由于牛顿法具有二阶收敛速度, 因此我们打算使用牛顿法优化能量 E, 从而需要一阶梯度和二阶偏导. 设 $p_i = (p_{i1}, p_{i2}, p_{i3})$, 则通过计算可得,

$$\begin{aligned} \frac{\partial E_{ij}}{\partial p_{ik}} &= 4(\|p_i - p_j\|^2 - l_{ij}^2)(p_{ik} - p_{jk}) \\ \frac{\partial^2 E_{ij}}{\partial p_{ik}^2} &= 8(p_{ik} - p_{jk})^2 + 4(\|p_i - p_j\|^2 - l_{ij}^2) \\ \frac{\partial^2 E_{ij}}{\partial p_{ik}\partial p_{jk}} &= -8(p_{ik} - p_{jk})^2 - 4(\|p_i - p_j\|^2 - l_{ij}^2) \\ \frac{\partial^2 E_{ij}}{\partial p_{ik}\partial p_{il}} &= 8(p_{ik} - p_{jk})(p_{il} - p_{jl}) \\ \frac{\partial^2 E_{ij}}{\partial p_{ik}\partial p_{jl}} &= 8(p_{ik} - p_{jk})(p_{jl} - p_{il}) \end{aligned} \tag{4}$$

其中 $i, j, k, l = 1, 2, 3$ 且 $k \neq l$. 观察可得,

$$\begin{aligned} \frac{\partial^2 E_{ij}}{\partial p_{ik}^2} &= -\frac{\partial^2 E_{ij}}{\partial p_{ik}\partial p_{jk}} \\ \frac{\partial^2 E_{ij}}{\partial p_{ik}\partial p_{il}} &= -\frac{\partial^2 E_{ij}}{\partial p_{ik}\partial p_{jl}} \end{aligned} \tag{5}$$

设 H_{ij} 是 E_{ij} 的海森矩阵 (hessian matrix), 则 $H_{ij} \in \mathrm{R}^{6*6}$ 是一个对称矩阵. 由等式5可得 $H_{ij} = \begin{pmatrix} A_{ij} & -A_{ij} \\ -A_{ij} & A_{ij} \end{pmatrix}$, 其中 A_{ij} 是一个 3*3 的对称矩阵, 即

$$A_{ij} = 4\begin{pmatrix} d_{ij} + 2q_1^2 & -2q_1q_2 & -2q_1q_3 \\ -2q_1q_2 & d_{ij} + 2q_2^2 & -2q_3q_2 \\ -2q_1q_3 & -2q_2q_3 & d_{ij} + 2q_3^2 \end{pmatrix} \tag{6}$$

其中 $d_{ij} = \|p_i - p_j\|^2 - l_{ij}^2, q_k = p_{ik} - p_{jk}, k = 1, 2, 3.$

令 $H = \sum_{e_{ij}} H_{ij}$. 由于 H_{ij} 不一定是一个正定矩阵, 所以 H 也不一定是一个正定矩阵. 因此我们需要做适当的处理使得 H 是一个正定矩阵. 如果直接对 H 进行谱分解后正定化, 则由于 H 阶数巨大, 计算量巨大, 因此不太合适. 考虑到正定矩阵之和也是正定矩阵, 我们可以对 H_{ij} 进行正定化, 从而间接将 H 正定化, 这样计算量小且能保证 H 是一个正定矩阵.

注意到 $H_{ij} = \begin{pmatrix} A_{ij} & -A_{ij} \\ -A_{ij} & A_{ij} \end{pmatrix}$, 我们可以利用这一特殊分布进一步减少计算量. 事实上, 由舒尔补的性质 [13] 可知, 若 $H = \begin{pmatrix} A_{ij} & -A_{ij} \\ -A_{ij} & A_{ij} \end{pmatrix}$, 则 H 是半正定矩阵的充分必要条件是 A_{ij} 为半正定矩阵.

由上述事实可知我们只需令 A_{ij} 为半正定矩阵即可使得 H_{ij} 为半正定矩阵. 设

$$P_{ij} = \begin{pmatrix} q_1^2 & -q_1q_2 & -q_1q_3 \\ -q_1q_2 & q_2^2 & -q_3q_2 \\ -q_1q_3 & -q_2q_3 & q_3^2 \end{pmatrix} \tag{7}$$

计算可得 P_{ij} 的特征值分别为 0,0 和 $q_1^2 + q_2^2 + q_3^2$.

设 A_{ij} 的三个特征值分别为 $\lambda_1, \lambda_2, \lambda_3$. 注意到 $A_{ij} = 4(\|p_i - p_j\|^2 - l_{ij}^2)I + 8P_{ij}$, 其中 I 为单位矩阵, 因此有

$$\begin{aligned} \lambda_1 &= 4(\|p_i - p_j\|^2 - l_{ij}^2) \\ \lambda_2 &= 4(\|p_i - p_j\|^2 - l_{ij}^2) \\ \lambda_3 &= 12(q_1^2 + q_2^2 + q_3^2) - 4l_{ij}^2 \end{aligned} \tag{8}$$

由于 A_{ij} 是对称矩阵, 因此存在正交矩阵 Q 和对角阵 D 使得

$$A_{ij} = QDQ^T \tag{9}$$

其中 $D = \mathrm{diag}(\lambda_1, \lambda_2, \lambda_3)$. 设 $Q = (x_1, x_2, x_3)$, 则将上式展开可得

$$A_{ij} = \lambda_1 x_1 x_1^{\mathrm{T}} + \lambda_2 x_2 x_2^{\mathrm{T}} + \lambda_3 x_3 x_3^{\mathrm{T}} \tag{10}$$

由于 A_{ij} 是一个半正定矩阵等价于为 $\lambda_1, \lambda_2, \lambda_3 \geqslant 0$, 因此我们只需让 $\widetilde{\lambda_k} = \max(\lambda_k, 0)$,k=1,2,3. 然后令 $\widetilde{A_{ij}} = \widetilde{\lambda_1} x_1 x_1^{\mathrm{T}} + \widetilde{\lambda_2} x_2 x_2^{\mathrm{T}} + \widetilde{\lambda_3} x_3 x_3^{\mathrm{T}}$. 则此时 $\widetilde{A_{ij}}$ 为半正定矩阵. 由于 Q 是一个正交阵, 即 $QQ^T = I$. 因此有 $x_1 x_1^{\mathrm{T}} + x_2 x_2^{\mathrm{T}} + x_3 x_3^{\mathrm{T}} = I$. 并且由于 $\widetilde{\lambda_1} = \widetilde{\lambda_2}$, 因此

$$\widetilde{A_{ij}} = \widetilde{\lambda_1}(I - x_3 x_3^{\mathrm{T}}) + \widetilde{\lambda_3} x_3 x_3^{\mathrm{T}} \tag{11}$$

接下来我们只需计算 x_3. 经过计算可得

$$x_3 = \frac{(p_{i1} - p_{j1}, p_{i2} - p_{j2}, p_{i3} - p_{j3})}{\sqrt{(p_{i1} - p_{j1})^2 + (p_{i2} - p_{j2})^2 + (p_{i3} - p_{j3})^2}} \tag{12}$$

将其代入上式即可得到 $\widetilde{A_{ij}}$. 接下来我们用 $\widetilde{A_{ij}}$ 替代 A_{ij} 则此时有 H_{ij} 为半正定矩阵, 进而有 H 为半正定矩阵. 由于 H 为半正定矩阵, 并不能保证是一个正定矩阵. 为了确保 H 是一个正定矩阵, 在这里我们可以令 $\widetilde{H} = H + \epsilon I$, 其中 ϵ 为一很小的数, 例如 $\epsilon = 10^{-15}$. 然后使用 $\widetilde{H}$ 来替换 H.

将 H 近似为正定矩阵后我们就得到了下降方向 $d = -H^{-1}\nabla E$. 令 $E = f(x + \alpha d)$, 其中 x 是当前点的坐标,α 是步长. 由于 $f(x + \alpha d)$ 是一个关于 α 的四次函数, 因此我们可以对 $f(x + \alpha d)$ 求导, 并令 α 为满足 $f(x + \alpha d)' = 0$ 的最小正实数. 由于 $f(x + \alpha d)$ 在 0 处的导数小于 0 且在 α 足够大时有 $f(x + \alpha d)'$>0, 因此这样的 α 总是存在的.

三、正则项

使用上一节的算法插值后得到的结果如图 1 所示. 可以看到中间的图显示出在某些时刻存在插值结果不够光滑, 自然的情况.

这是因为只有边长的连续性无法保证得到自然的插值结果, 还需要二面角的连续性. 于是我们在之前的能量 E 的基础上加一项正则项, 即令

$$\widetilde{E} = E + w\|P_{t_i} - P_i\|^2 \tag{13}$$

其中 t_i 为第 i 帧对应的时间 t;P_t 为 t 时刻所有点的坐标;P_i 为初始化网格;w 为权重. 我们取 $w = 10^{-3} l_{avg}^2$, 其中 l_{avg} 为网格的平均边长. 加了正则项之后, 只要初始化的网格的二面角连续, 则可使得得到的插值序列的二面角连续.

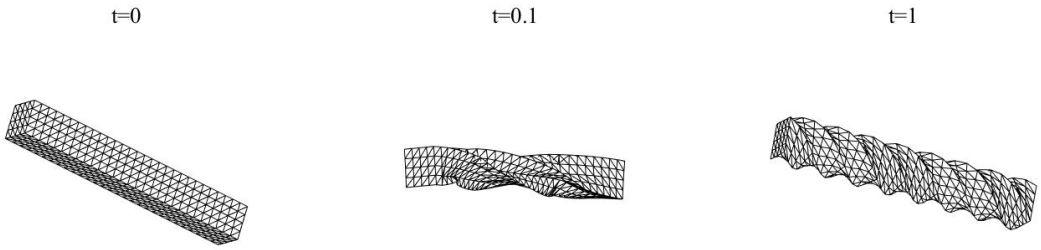

图 1：上图是两个木条之间的插值结果. 其中左图和右图分别为初始网格和目标网格. 中图是 $t = 0.1$ 时的结果.

为了得到良好的插值结果, 我们需要一个合适的初始化网格. 考虑到我们的目标是优化边长的平方, 因此我们自然觉得初始化的边长平方的误差越小, 结果越好. 于是我们准备使用 SQP[1] 这个已经对边长平方进行较好优化的方法进行初始化. 为了验证该方法确实更加优秀, 我们还将其与 FFMP[7],ARAP[2],GE[3],ABF[8] 共 4 种方式得到的网格比较. 当然, 这些方法的插值边长部分也相应改为线性插值边长的平方.

为了进行比较, 我们首先定义一个评价标准

$$L = \sqrt{\frac{1}{|E|}\sum_{e_{ij}\in E}(\frac{l_{real}}{l_{target}} - 1)^2} \tag{14}$$

其中 E 为所有边长的集合,l_{real} 是我们得到的插值结果的边长,l_{target} 是目标边长. 可以看到实际上 L 就是边长的相对误差的 l_2 范数.

如表 2.1 所示, 在这五种初始化网格中,SQP 的 L 值更小, 说明插值结果的总体误差最小. 图 2 则是多个模型在使用不同初始化网格下得到的 t=0.5 时刻的结果. 从图 2 则可以看到 SQP 得到的结果更蓝, 对应了其更小的边长误差. 并且我们可以看到 SQP 得到的结果也很合理. 因此, 利用 SQP 进行初始化得到的结果最好, 于是我们选择其作为初始化网格.

第四节　实验结果

我们使用的系统为 Windows10,CPU 为 E5-2650v2, 内存容量为 32GB. 我们选取了 LSRDF[11],ELI[9] 和 MSGI[12] 用于对比.

图 3 展示了在 $t = 0.25, 0.5, 0.75$ 时在一些模型下的插值结果. 图 4 和图 5 则进一步展示了我们的方法与选取的三种方法的插值结果的比较. 从图 4 可以看到我们的方法在这几个模型下的边长误差均是最小的, 且整体上误差相近. 从图 5

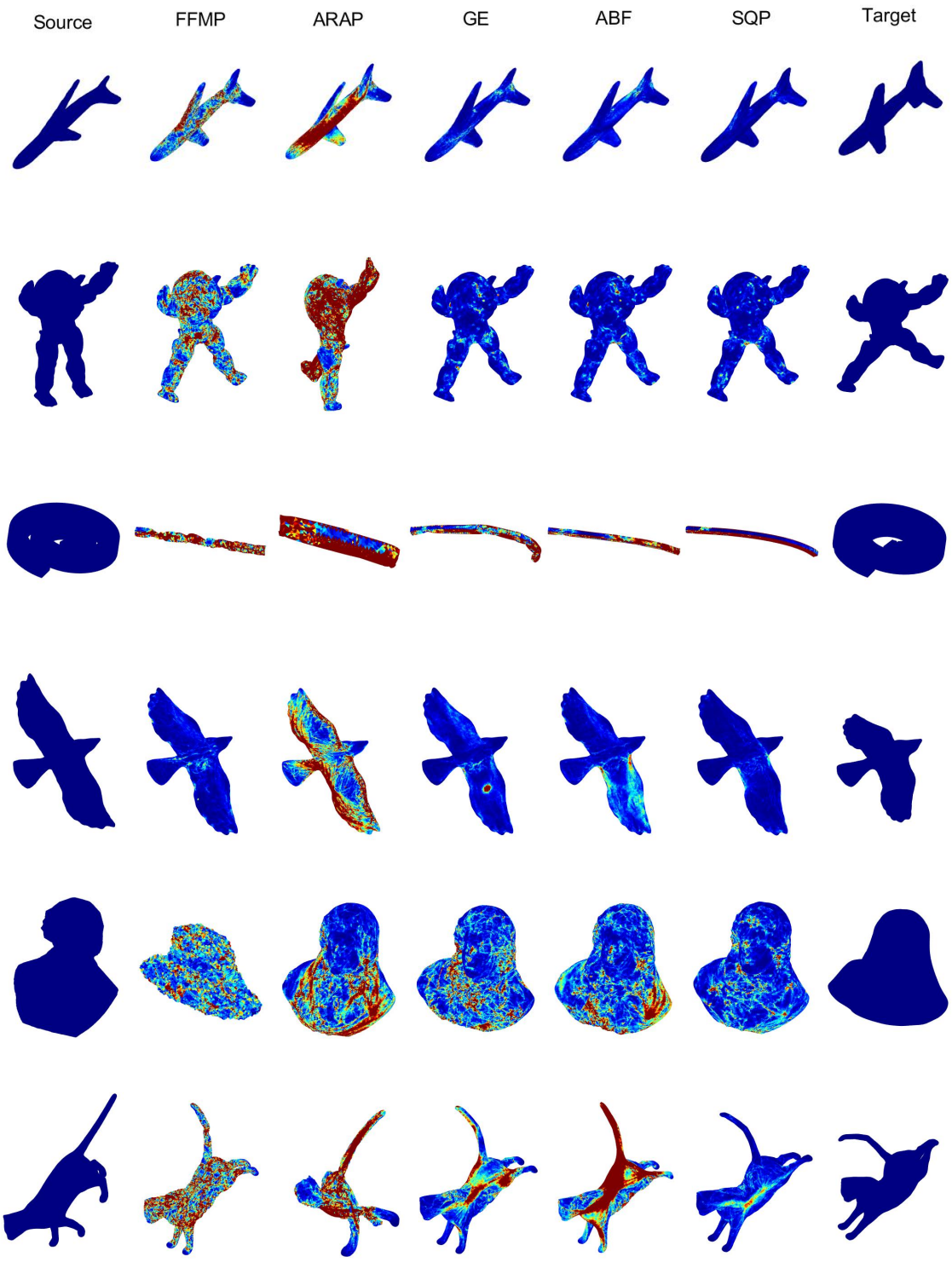

图 2: 在不同初始化网格下 t=0.5 时得到的结果. 从上往下分别是对应于表 2.1 的飞机 (airplane), 犰狳 (armadillo), 长条 (bar), 鸟 (bird), 半身像 (bust), 猫 (cat) 这 6 个模型. 颜色越偏橙红说明实际边长与目标边长的差越大, 越偏深蓝色则说明差越小.

		$L(10^{-3})$						
		airplane	armadillo	bar	bird	bust	camel	cat
FFMP	t=0.25	1.39	0.47	12.42	0.13	3.45	0.96	3.46
	t=0.5	2.50	0.86	4.59	0.18	3.44	1.45	3.85
	t=0.75	1.75	0.62	8.50	0.13	3.44	0.90	2.62
ARAP	t=0.25	2.57	3.35	11.17	1.20	1.73	2.98	6.29
	t=0.5	3.58	4.17	7.46	1.50	2.43	2.21	5.51
	t=0.75	2.36	3.88	5.03	0.96	1.76	1.61	5.55
GE	t=0.25	0.89	0.14	8.50	0.12	1.84	0.43	2.10
	t=0.5	0.65	0.16	9.62	0.16	2.21	0.48	2.09
	t=0.75	0.53	0.15	7.26	0.18	1.93	0.36	1.22
ABF	t=0.25	0.90	0.14	10.81	0.17	1.90	0.63	9.30
	t=0.5	0.59	0.18	13.80	0.23	2.70	0.74	7.50
	t=0.75	0.45	0.16	7.74	0.18	1.98	0.54	3.45
SQP	t=0.25	0.24	0.12	9.83	0.08	1.04	0.33	1.02
	t=0.5	0.29	0.15	8.74	0.10	1.24	0.38	0.81
	t=0.75	0.20	0.13	5.56	0.07	1.00	0.31	0.43

表 1　在不同初始化网格下得到的结果的 L 值

左列则可以看到我们的方法的 L 值更小, 即总体上来说我们的方法在各个形变的过程中各边长误差均是最小的. 而从图 5 右列则可以看到除了在第一个模型, 即 bar 这个模型的某些时刻外, 我们的方法在各个形变的过程中最大的相对误差均是最小的. 因此无论是从图 4, 还是从图 5 来看, 我们的方法均是最好的. 不过从图 5 也可以看出我们的方法的 L 值曲线不够光滑, 说明其边长误差不够稳定.ELI 和 MSGI 这两种算法也有这种现象的出现, 但 LSRDF 没有. 这很可能是因为 LSRDF 的能量函数是一个二次函数, 可求解出其全局最优点, 并且只需要一步即可嵌入到 R^3 空间, 因此结果更加稳定. 而其它方法均可能会陷入局部非全局最优.

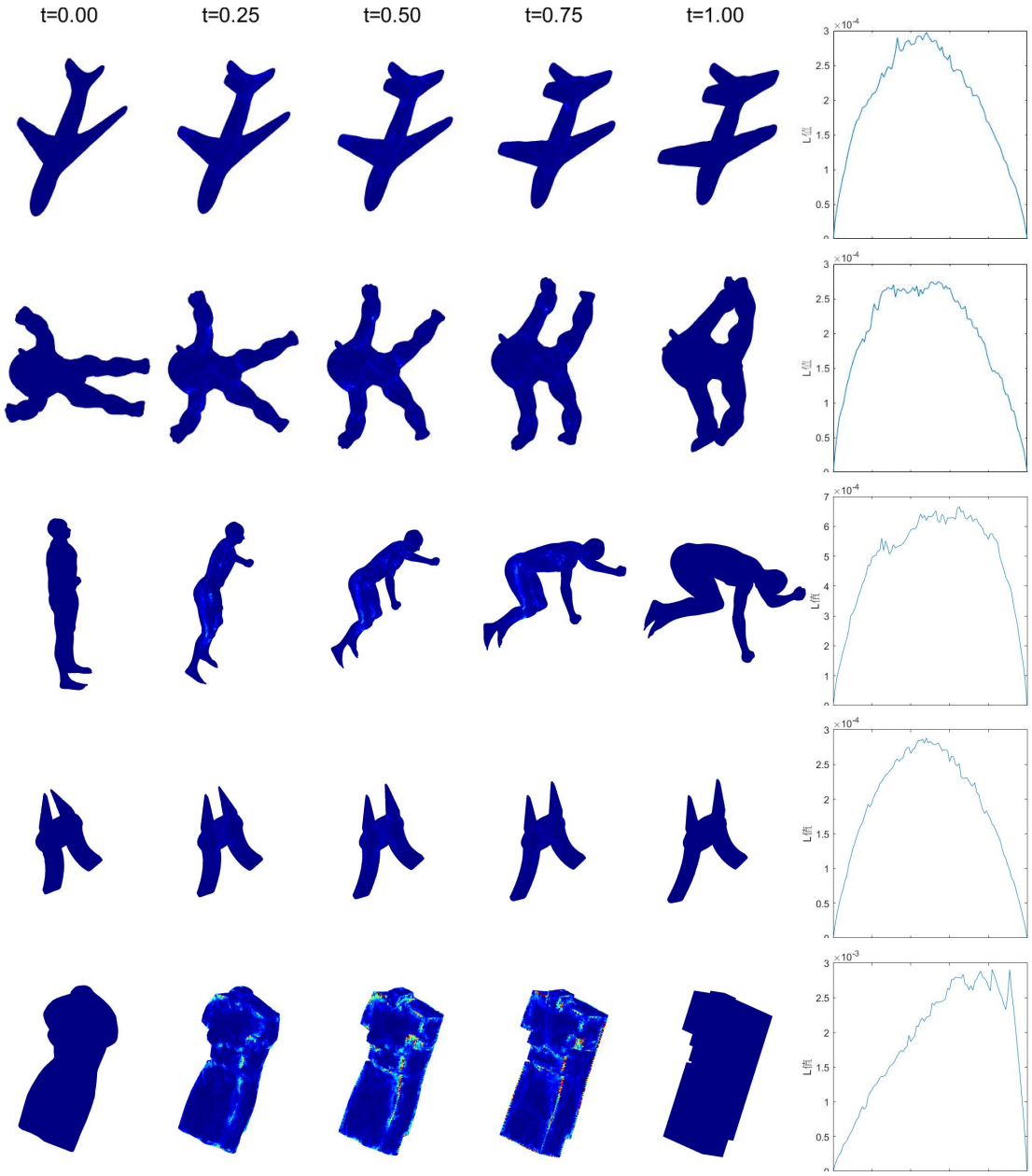

图 3: 飞机、犰狳、人、钳子和半身像这五个模型的插值结果. 其中 t=0.00 和 t=1.00 分别代表源模型和目标模型, 其它时刻则为插值得到的结果. 最右边是不同时刻的 L 值误差.

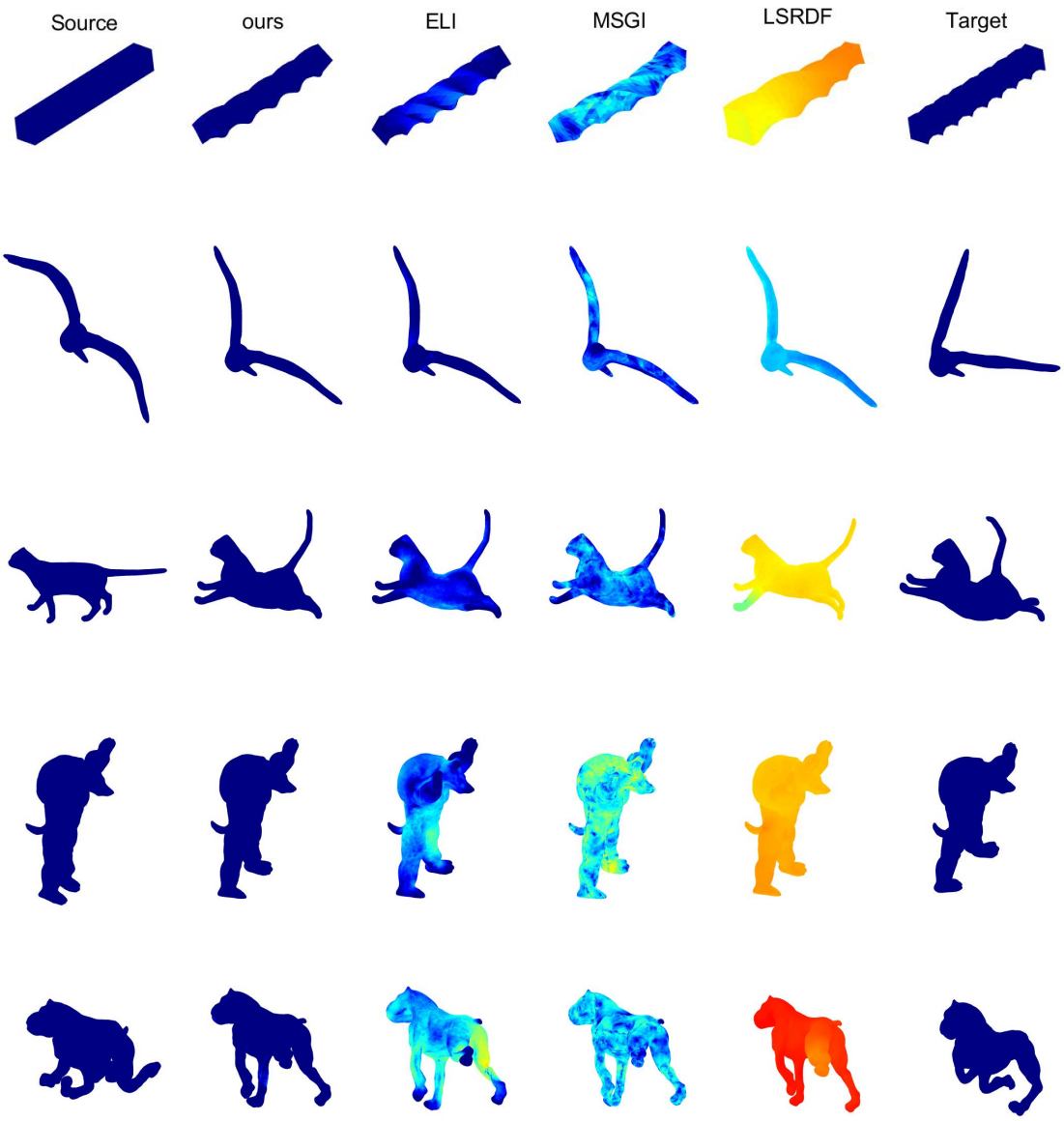

图 4: 比较四种方法在五个模型下的插值结果. 其中中间四列是四种方法在 $t = 0.5$ 时刻的结果. 颜色越偏橙红说明实际边长与目标边长的差越大, 越偏深蓝色则说明越小.

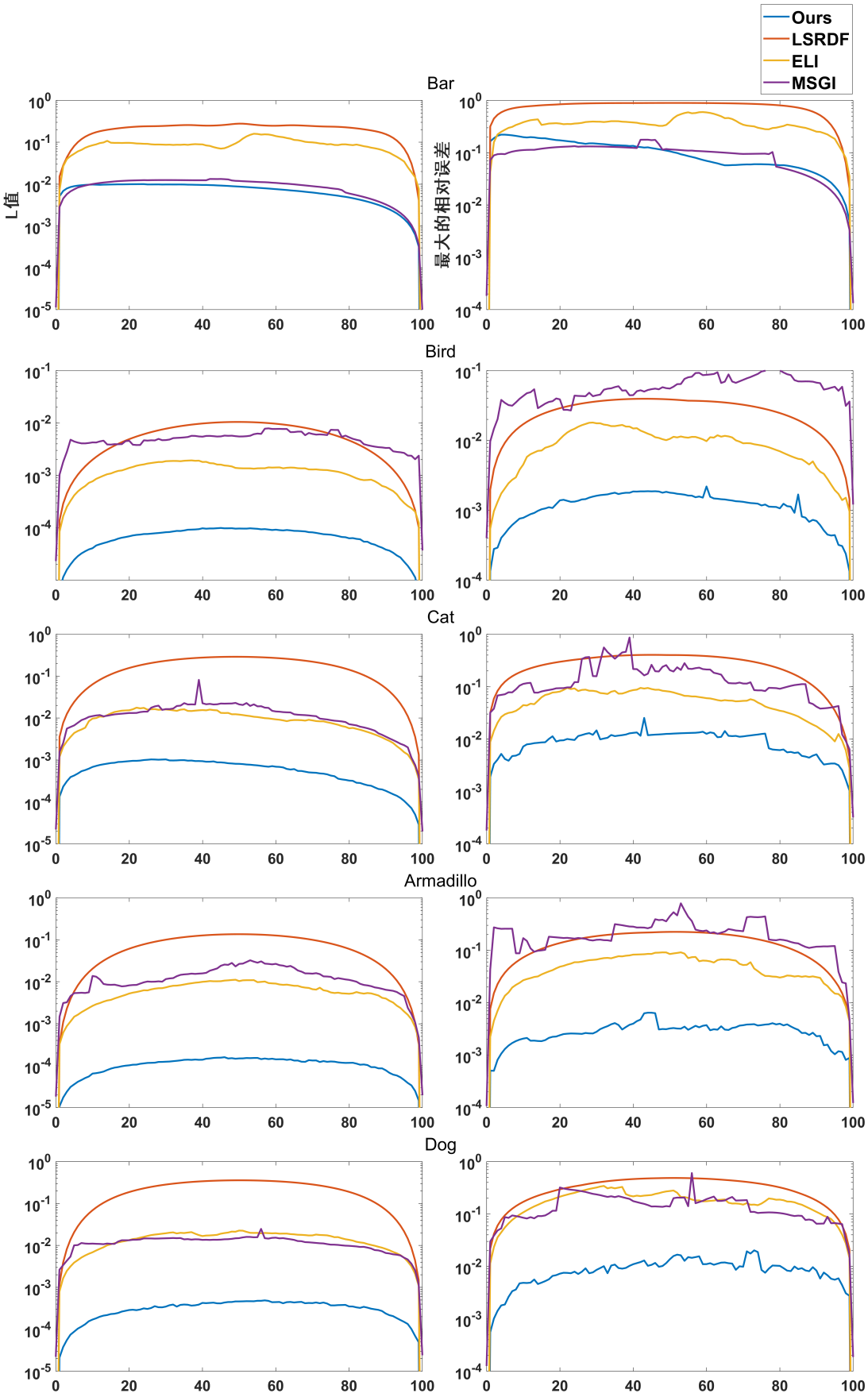

图 5: 比较四种方法在图 4 的五个模型下的误差. 横轴代表帧数, 其中第 0 帧和第 100 帧分别代表源模型和目标模型. 左列是四种方法下的 L 值, 右列则是最大的相对误差, 其中相对误差为 $|\frac{l_{real}-l_{target}}{l_{target}}|$.

第五节　总结和讨论

我们提出了一种新的基于边长的形状插值算法. 该方法应用于两个兼容的三维网格. 我们先是解释了为何线性插值边长平方能够保证插值结果是有界扭曲的. 其次, 我们使用了顶点坐标作为变量定义了能量函数, 避免了像文章 [9] 一样的间接重建. 随后, 我们介绍了本方法的优化算法. 我们使用了牛顿法作为我们的优化方法, 并且为此将海森矩阵近似为正定矩阵. 为了加快速度, 我们利用了多种技巧, 使得我们能够直接计算出近似后的正定矩阵, 无需特征值分解. 最后我们进行了实验, 与方法 [9, 11, 12] 相比较, 证明了我们的方法在边长误差上的表现更为优异.

不过该方法也有其局限性. 其局限性主要有两个. 第一个是由于正则项包含初始化网格的信息, 因此结果的好坏一定程度上依赖于初始化的好坏. 第二个则是虽然总体上的边长的相对误差相较于其它方法有了较大减少, 但是最大的边长的相对误差并不一定, 在少数情况下该指标不如其它方法. 而且我们得到的最终结果的边长依然与我们期望的长度有一定差距. 因此, 我们接下来可以尝试寻找更好的初始化方法. 其次可以考虑是否有更好的优化方法并结合更好的初始化使得最终结果的边长等于我们期望的长度.

参考文献

[1] Aharon, I., Chen, R., Zorin, D., and Weber, O. Bounded distortion tetrahedral metric interpolation. *ACM Transactions on Graphics 38*, 6 (2019), 182.1–182.17.

[2] Alexa, M. As-rigid-as-possible shape interpolatioin. In *SIGGRAPH2000 Conference Proceedings* (2000).

[3] Chao, I., Pinkall, U., Sanan, P., and Schröder, P. A simple geometric model for elastic deformations. *ACM transactions on graphics (TOG) 29*, 4 (2010), 1–6.

[4] Chen, R., Weber, O., Keren, D., and Ben-Chen, M. Planar shape interpolation with bounded distortion. *ACM Transactions on Graphics (TOG) 32*, 4 (2013), 1–12.

[5] Chien, E., Levi, Z., and Weber, O. Bounded distortion parametrization in the space of metrics. *ACM Transactions on Graphics (TOG) 35*, 6 (2016), 1–16.

[6] Fröhlich, S., and Botsch, M. Example-driven deformations based on discrete shells. In *Computer graphics forum* (2011), vol. 30, Wiley Online Library, pp. 2246–2257.

[7] Kircher, S., and Garland, M. Free-form motion processing. *Acm Transactions on Graphics 27*, 2 (2008).

[8] Paillé, G.-P., Ray, N., Poulin, P., lla Sheffer, A., and Lévy, B. Dihedral angle-based maps of tetrahedral meshes. *ACM Transactions on Graphics (SIGGRAPH 2015 Conf. Proc.) 34*, 4 (2015).

[9] Rojas, C., Tsui, A., He, S., Simons, L., Li, S., and Amenta, N. Edge length interpolation. In *ACM Symposium on Solid and Physical Modeling* (2014).

[10] Sederberg, T. W., Gao, P., Wang, G., and Hong, M. 2-d shape blending: An intrinsic solution to the vertex path problem. In *Conference on Computer Graphics & Interactive Techniques* (1993).

[11] Wang, Y., Liu, B., and Tong, Y. Linear surface reconstruction from discrete fundamental forms on triangle meshes. In *Computer Graphics Forum* (2012), vol. 31, Wiley Online Library, pp. 2277–2287.

[12] Winkler, T., Drieseberg, J., Alexa, M., and Hormann, K. Multi-scale geometry interpolation. *Computer Graphics Forum 29*, 2 (2010).

[13] ZHANG, F. *The Schur complement and its applications*, vol. 4. Springer Science & Business Media, 2006.

[14] 金小刚, AND 鲍虎军. 计算机动画技术综述. 软件学报 *8*, 4 (1997), 11.