前沿计算研究实践（II）	**2019 年春季**

专题 4：谱变换

教授：陈宝权　　作者：郭永康、李沛卓、阮良旺、吴铭东、吴松隐、张若禹

免责声明：该笔记尚未进行正式出版物的通常审查，未经教师允许不能向课程外分发.

4.1 谱变换（Spectral Transform）基本知识及应用

4.1.1 介绍

压缩是数据处理领域中一项重要的技术。对于图像压缩，我们有各种不同的方法 (e.g. 奇异值分解、傅里叶变换等)，这部分内容往往被深入探讨。在本节中我们主要介绍一种几何压缩 (geometry compression) 的手段——谱变换。我们将首先介绍它在二维形体的应用，并进一步拓展到三维的情况。

4.1.2 拉普拉斯光滑化（Laplacian smoothing）

拉普拉斯光滑化是一个将形体（shape）平滑化的手段。我们从二维的情形出发。考虑二维平面上的一个由离散点集首尾相连形成的轮廓。我们将相邻每条边的中点两两相连形成一个新的封闭轮廓，并重复一次这个过程，这就叫做拉普拉斯光滑化。

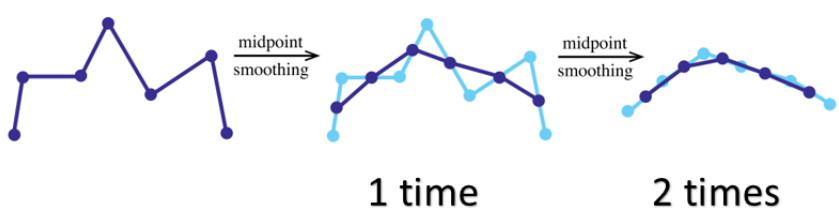

图 4.1: 拉普拉斯光滑化示意图

记 $\mathbf{v}_{i-1} = [x_{i-1}, y_{i-1}]$，$\mathbf{v}_i = [x_i, y_i]$，$\mathbf{v}_{i+1} = [x_{i+1}, y_{i+1}]$ 是相邻的三个顶点。则我们经过拉普拉斯光滑化得到的新的顶点可表示为两次平均的过程：

$$\mathbf{v}_i' = \frac{1}{2}(\frac{1}{2}(\mathbf{v}_{i-1} + \mathbf{v}_i)) + \frac{1}{2}(\frac{1}{2}(\mathbf{v}_i + \mathbf{v}_{i+1})) = \frac{1}{4}\mathbf{v}_{i-1} + \frac{1}{2}\mathbf{v}_i + \frac{1}{4}\mathbf{v}_{i+1}$$

同时我们定义 $\mathbf{v_i}$ 处的一维离散拉普拉斯算子（1D discrete Laplacian）$\delta(\mathbf{v_i})$：

$$\delta(\mathbf{v}_i) = \frac{1}{2}(\mathbf{v}_{i-1} + \mathbf{v}_{i+1}) - \mathbf{v}_i$$

注意到

$$\mathbf{v}_i' = \mathbf{v}_i + \frac{1}{2}\delta(\mathbf{v}_i)$$

4.1.3 一维信号与谱变换

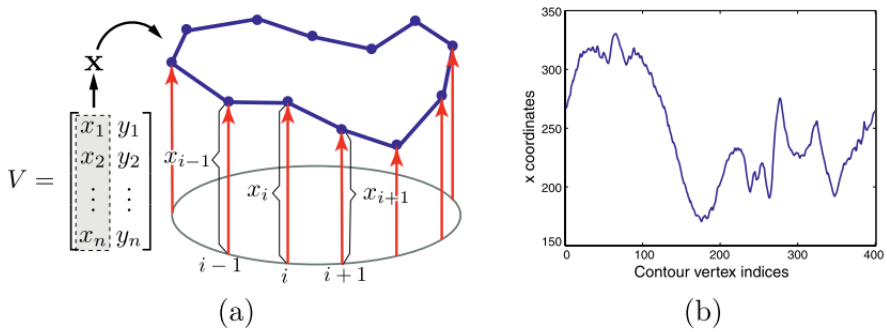

图 4.2: $\mathbf{x}$ 可以被看作在圆上均匀采样的一维信号

重新回到这个问题，现在，我们将点集的所有 x 坐标用 $\mathbf{x}$ 表示，即 $x = [x_1, x_2, \cdots, x_n]$，同理记 $\mathbf{y} = [y_1, y_2, \cdots, y_n]$。注意到所有顶点是首尾相连的，所以进一步，我们可以将 $\mathbf{x}$ 视为周期性的离散一维信号，如图所示。

利用这个记号，现在我们可以用矩阵形式表示一维离散拉普拉斯算子：

$$\delta(\mathbf{x}) = L\mathbf{x} = \begin{bmatrix} 1 & -\frac{1}{2} & 0 & \cdots & \cdots & 0 & -\frac{1}{2} \\ -\frac{1}{2} & 1 & -\frac{1}{2} & 0 & \cdots & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \\ 0 & \cdots & \cdots & 0 & -\frac{1}{2} & 1 & -\frac{1}{2} \\ -\frac{1}{2} & 0 & \cdots & \cdots & 0 & -\frac{1}{2} & 1 \end{bmatrix} \mathbf{x}$$

进一步，拉普拉斯平滑化即为对信号 $\mathbf{x}$ 施加平滑化算子，即 $\mathbf{x}' = S\mathbf{x}$，其中 $S = I - \frac{1}{2}L$。

为了分析拉普拉斯平滑化的性质，我们可以分析矩阵 L 的特征值和特征向量（注意，这些特征向量同时也是 S 的特征向量）。由于 L 是实对称矩阵，所以我们可以对它进行特征值分解，并找出一组单位正交基 $\mathbf{e}_1$，$\mathbf{e}_2$，...，$\mathbf{e}_n$。记 $E = [\mathbf{e}_1, \mathbf{e}_2, ..., \mathbf{e}_n]$，我们可以将 $\mathbf{x}$ 表示为这些特征向量的线性组合：

$$\mathbf{x} = \sum_{i=1}^{n} \widetilde{x}_i \mathbf{e}_i = E\widetilde{\mathbf{x}}$$

由于 E 是正交矩阵，所以有

$$\widetilde{\mathbf{x}} = E^\top \mathbf{x}$$

这种对原信号 $\mathbf{x}$ 进行的转换就被称为**谱变换**。

4.1.4 信号压缩与平滑化

在本节我们将利用二维形体的例子给出谱变换的两个应用。
之前我们已经分析过，随着特征值增大，相应的特征向量的“频率”变高。那么一个很自然的想法是通过截断式地舍去高频信息以达到信号压缩的目的。
只保留前 k 个参数，我们得到的重建信号为

$$\mathbf{x}^{(k)} = \sum_{i=1}^{k} \widetilde{x}_i \mathbf{e}_i$$

我们可以通过衡量 $\mathbf{x}$ 和 $\mathbf{x}^{(k)}$ 之间的欧式距离（l_2 误差）以定量描述信息的损失：

$$||\mathbf{x} - \mathbf{x}^{(k)}|| = ||\sum_{i=k+1}^{n} \widetilde{x}_i \mathbf{e}_i|| = \sqrt{\sum_{i=k+1}^{n} \tilde{x}_i^2}$$

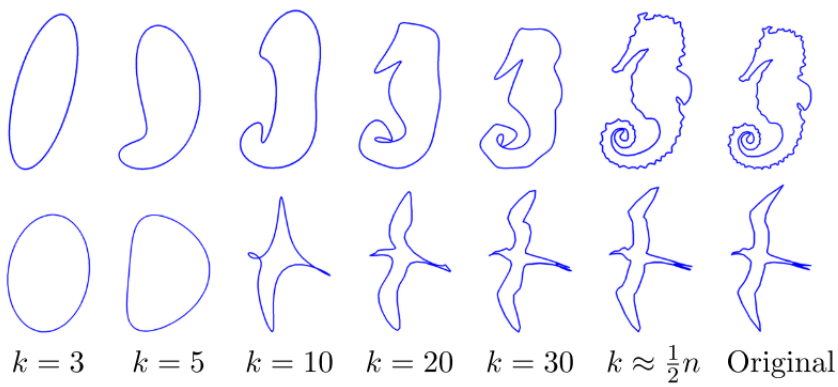

图 4.3: 利用前 k 个参数完成的形体重建

回忆一下，本章我们是从拉普拉斯光滑化进行引入的。那么，如果对一个形体重复做 m 次拉普拉斯光滑化，利用我们介绍的数学工具，我们可以定量描述这一过程：

$$\mathbf{x}^{<m>} = S^m \mathbf{x} = (I - \frac{1}{2}L)^m \mathbf{x} = \sum_{i=1}^{n} (I - \frac{1}{2}L)^m \mathbf{e}_i \tilde{x}_i = \sum_{i=1}^{n} \mathbf{e}_i (1 - \frac{1}{2}\lambda_i)^m \tilde{x}_i$$

利用这个式子，我们可以清楚地看出拉普拉斯光滑化的工作原理：随着 m 的增长，图形中的高频分量（即较大的特征值对应的特征分量）的系数以更快的速度收敛到 0。...(centroid) 现在总结一下这两种方法。我们可以察觉到它们内在的工作机制是类似的，我们可以将其抽象为滤波器（filter）：我们通过对每个频率的信号叠加上一个滤波函数来对信号进行处理，即

$$\mathbf{x}' = \sum_{i=1}^{n} f(i) \tilde{x}_i \mathbf{e}_i = E\tilde{\mathbf{x}}'$$

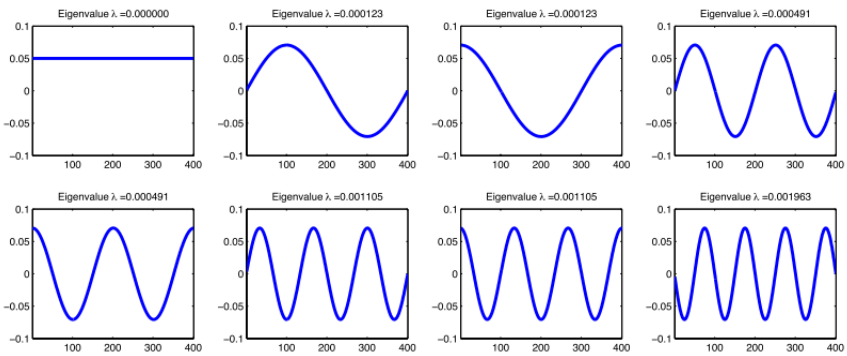

图 4.4: 拉普拉斯矩阵的特征向量

4.1.5 拉普拉斯矩阵的特征向量

令人惊奇的一点是拉普拉斯矩阵的特征向量若将其各个分量绘制成散点图的话，其形如三角函数。由此我们可以猜测它的形式为 $v_k = [1, \omega_n^k, ..., \omega_n^{(n-1)k}]^T$。接下来我们去验证这一点：

$$
\begin{aligned}
(Lv)_j &= \sum L_{ji} v_i \\
&= v_j - \frac{1}{2}(v_{j-1} + v_{j+1}) \\
&= \omega_n^{kj}[1 - \frac{1}{2}(\omega_n^k + \omega_n^{-k})] \\
&= v_j(1 - \cos\frac{2\pi k}{n})
\end{aligned}
\tag{4.1}
$$

可以发现猜想正确，并且 $\lambda_k = 1 - \cos\frac{2\pi k}{n}$。进一步，我们可以得出 $\lambda_k = \lambda_{n-k}$，因为 $v_k = \begin{bmatrix} 1 \\ \omega_n^k \\ \vdots \\ \omega_n^{(n-1)k} \end{bmatrix}$，$v_{-k} = v_{n-k} = \begin{bmatrix} 1 \\ \omega_n^{-k} \\ \vdots \\ \omega_n^{-(n-1)k} \end{bmatrix}$ 都是 λ_k 的特征向量。在实数域上，则有

$$
\alpha_k = \frac{1}{2}(v_k + v_{-k}) = \begin{bmatrix} 1, & \cos\frac{2\pi k}{n} & \dots & \cos\frac{2\pi k(n-1)}{n} \end{bmatrix}^T,
$$

$$
\beta_k = \frac{1}{2i}(v_k - v_{-k}) = \begin{bmatrix} 1, & \sin\frac{2\pi k}{n} & \dots & \sin\frac{2\pi k(n-1)}{n} \end{bmatrix}^T.
$$

这就解释了前文所述问题。

4.2 谱方法在三角网格中的应用

4.2.1 介绍

在本节中我们将谱方法扩展到三角网格中并加以运用。

定义 4.1 三角网格（*triangle mesh*） 三角网格是由三角形表面构成的三维封闭形体。

一个 n 个顶点的三角网格可以被表示为 $\mathcal{M} = (\mathcal{G}, \mathbf{P})$，$\mathcal{G} = (\mathcal{V}, \mathcal{E})$，其中 $\mathcal{V}$ 为顶点集，$\mathcal{E} \in \mathcal{V} \times \mathcal{V}$ 为边集，$\mathbf{P} \in \mathbb{R}^{n\times 3}$ 是一个矩阵，代表了顶点的三维坐标，其中第 i 行 $\mathbf{P}_i = [x_i, y_i, z_i]$ 代表了第 i 个顶点的三维坐标。

4.2.2 网格上的平滑化与信号表示

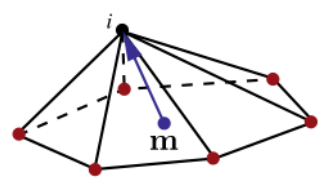

图 4.5: 拉普拉斯算子在网格上的向量表示

类似于二维形体的引入，我们将首先介绍网格上的拉普拉斯光滑化方法。注意到在二维的情形中我们的拉普拉斯算子代表了相邻点的中点到顶点的向量；因此，在这里我们用一个顶点所有相邻点的重心到顶点的向量作为拉普拉斯算子。
我们先关注 x 坐标，并用 $\mathbf{x}$ 代表 $\mathbf{P}$ 的第一列。从而网格上的拉普拉斯算子对 x 分量的作用可以用一个 n 维方阵表示。记 W 为图 $\mathcal{G}$ 的相邻矩阵，即

$$W_{i,j} = \begin{cases} 1 & \text{if } (i,j) \in \mathcal{E}, \\ 0 & \text{otherwise.} \end{cases}$$

并用 D 表示图 $\mathcal{G}$ 每个顶点的相邻点数，即

$$D_{i,j} = \begin{cases} d_i = |\mathcal{N}(i)| & \text{if } i = j, \\ 0 & \text{otherwise,} \end{cases}$$

其中 $\mathcal{N}(i)$ 是顶点 i 的相邻点集。由此我们定义拉普拉斯矩阵 T 为

$$T = I - D^{-1}W.$$

注意到这样定义后，将 T 作用于 $\mathbf{x}$，可以得到

$$\delta(x_i) = x_i - \frac{1}{|\mathcal{N}(i)|}_{j\in\mathcal{N}(i)} x_j,$$

故符合我们的定义。

4.2.3 网格上的谱变换

在定义了网格上的信号表示与拉普拉斯算子之后，这一部分就与二维情形基本一样了。记

$$E = [\mathbf{e}_1, \mathbf{e}_2, ..., \mathbf{e}_n]$$

为特征值 $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$ 对应的单位特征向量构成的矩阵，则经过谱变换后有 $\widetilde{\mathbf{x}} = E^\top \mathbf{x}$。

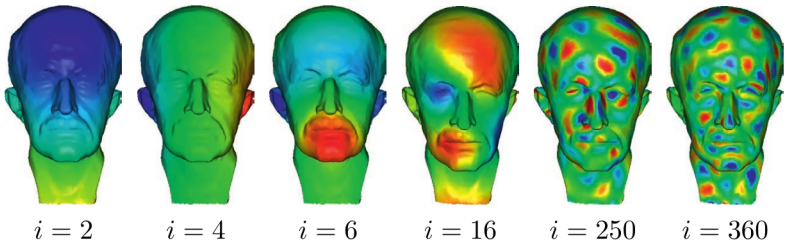

图 4.6: 第 i 小的特征值对应的特征向量彩图

利用舍弃高频信息的手段，我们可以进行去噪和信息压缩，如图所示。

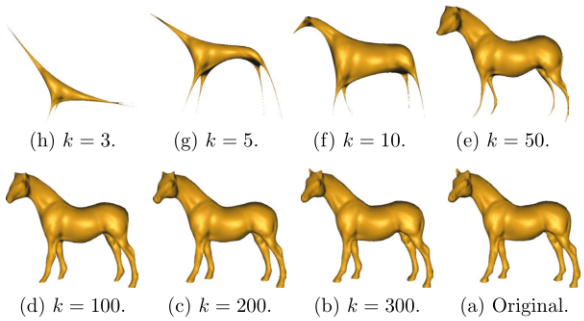

图 4.7: 取前 k 个特征向量时的三维重建

4.3 Laplacian surface editing

4.3.1 拉普拉斯坐标

4.3.1.1 概念引入

由于绝对坐标的不便之处，我们引入拉普拉斯坐标来维持变换时的形状（shape），它是由一个差分集合 $\Delta = \delta_i$ 组成的，其中

$$\delta_i = L(\boldsymbol{v_i})$$

$$L(\boldsymbol{v_i}) = \boldsymbol{v_i} - \frac{1}{d_i}\sum_{j\in N_i}\boldsymbol{v_j}$$

而用拉普拉斯坐标来表示网格（mesh）有着更优越的地方。对于一个变换，我们可以只确定某些顶点，而希望用 Δ 解出其他顶点的坐标，尽量保持形状不变，决定固定哪些顶点的方式是最小化下面的损失函数：

$$E(V^{'}) = \sum_{i=1}^{n}\left\|\delta_i - \mathcal{L}(\mathbf{v}_i^{'})\right\|^2 + \sum_{i=m}^{n}\left\|\mathbf{v}_i^{'} - \mathbf{u}_i\right\|^2$$

4.3.1.2 一些提升

上面的优化问题可以很好的处理平移，但是对伸缩和旋转表现不佳。一个办法是基于变换后的坐标引入一个合适的变换函数 T, 求解下面的最小化问题：

$$E(V^{'}) = \sum_{i=1}^{n}\left\|T_i(V^{'})\delta_i - \mathcal{L}(\mathbf{v}_i^{'})\right\|^2 + \sum_{i=m}^{n}\left\|\mathbf{v}_i^{'} - \mathbf{u}_i\right\|^2$$

在这个问题中 T_i 和 V' 都是未知的，但是如果 T_i 是 V' 的线性函数，问题会简化许多。一个简单的想法是既然我们不想改变形状，T 在改变 v_i 的同时也应该改变其邻域内的点。因此我们定义:

$$T_i = \arg\min_{T_i}\left(\left\|T_i v_i - v_i^{'}\right\|^2 + \sum_{j\in\mathcal{N}_i}\left\|T_i v_j - v_j^{'}\right\|^2\right)$$

4.3.1.3 计算 T_i

但是这样还不够，我们还需要对 T_i 增加一些限制，否则会失去许多几何上的细节，对于旋转和同向的伸缩变换，可以写成 $T = s\exp(H)$，H 是一个反对称矩阵，通过推导，它也可以表示成

$$s\exp(H) = s(\alpha I + \beta H + \gamma\mathbf{h}^T\mathbf{h})$$

$\mathbf{h}^T\mathbf{h}$ 是二次的，我们直接删除它，最后就得到一个矩阵

$$T_i = \begin{pmatrix} s & -h_3 & h_2 & t_x \\ h_3 & s & -h_1 & t_y \\ -h_2 & h_1 & s & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

这是一个对小角度旋转很好的逼近，假定 $(s_i, \mathbf{h}_i, \mathbf{t}_i)^T$ 是未知量，我们希望最小化

$$\left\| A_i(s_i, \mathbf{h}_i, \mathbf{t}_i)^T - \mathbf{b}_i \right\|^2$$

其中 A_i 包含 v_i 和其邻点的位置，$\mathbf{b}_i$ 包含 $v_i^{'}$ 和其邻点的位置

$$A_i = \begin{pmatrix} v_{k_x} & 0 & v_{k_z} & -v_{k_y} & 1 & 0 & 0 \\ v_{k_y} & -v_{k_z} & 0 & v_{k_x} & 0 & 1 & 0 \\ v_{k_z} & v_{k_y} & -v_{k_x} & 0 & 0 & 0 & 1 \\ \vdots & & & & & & \end{pmatrix},\ k \in \{i\} \cup \mathcal{N}_i$$

$$\mathbf{b}_i = \begin{pmatrix} v_{k_x}^{'} \\ v_{k_y}^{'} \\ v_{k_z}^{'} \\ \vdots \end{pmatrix},\ k \in \{i\} \cup \mathcal{N}_i$$

用最小二乘法得出解

$$(s_i, \mathbf{h}_i, \mathbf{t}_i)^T = (A_i^T A_i)^{-1} A_i^T \mathbf{b}_i$$

，最终我们得到了 T_i

4.3.2 网格编辑（mesh editing）

下面我们准备运用拉普拉斯坐标来进行一些实际的操作。普通的变换一般分为几步：a. 确定需要变换的区域（ROI）和固定点（anchor），b. 使用者在 ROI 内确定把手（handle），c. 对 handle 进行变换，其他点就用之前的方法重建，下面是一个效果图

4.3.3 纹理转移（coating transfer)）

4.3.3.1 简单情形

现在我们希望将某个源表面（surface）的纹理（coat）覆盖到目标的表面上去。为此，我们定义纹理是一些高频的细节，也就是原表面和其低频化结果的差异。具体而言，设 S' 是 S 光滑化后的结果，δ_i 和 δ_i' 是对应顶点 i 的拉普拉斯坐标，$\xi_i = \delta_i - \delta_i'$ 就表示纹理，利用 $S = L^{-1}(\delta' + \xi$ 就可以重建出 S，而

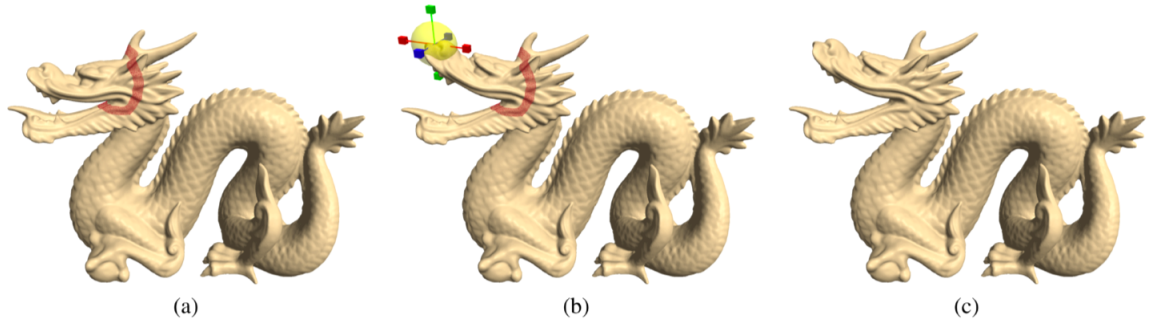

Figure 1: *The editing process. (a) The user selects the region of interest – the upper lip of the dragon, bounded by the belt of stationary anchors (in red). (b) The chosen handle (enclosed by the yellow sphere) is manipulated by the user: translated and rotated. (c) The editing result.*

图 4.8: 龙抬头

这也是我们解决问题的灵感。首先我们假设源表面 S 和目标表面 U 连通性相同，接下来我们计算出各个点的法向量 $\boldsymbol{n_s}$ 和 $\boldsymbol{n_u}$，通过旋转将法向量对应起来，得到 $\xi_i' = R_i(\xi_i)$,$U' = L{-}1(\Delta + \xi_i')$，$\Delta$ 是 U' 的拉普拉斯坐标，这样我们就成功的改变了高频信息，达到了转移纹理的效果。

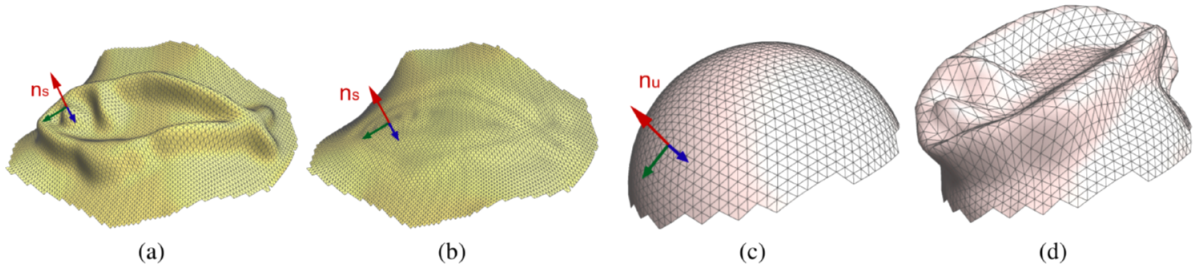

Figure 6: *Coating transfer. The orientation of a coating detail (a) is defined by the local frame at the corresponding vertex in the low frequency surface in (b). The transferred coating vector needs to be rotated to match the orientation of the corresponding point in (c) to reconstruct (d).*

图 4.9: 奇怪的表面

4.3.3.2 映射与重采样（mapping and resampling）

假如 S 和 U 表面连通性不时，上面的方法就失效了，为此我们采取的办法是，将两个表面都参数化后映射到同一个域中，此处我们假定它们都同构于一个圆盘。为了保证效果，这样的映射应该能够将对应的表面细节联系起来，例如脸部的五官。

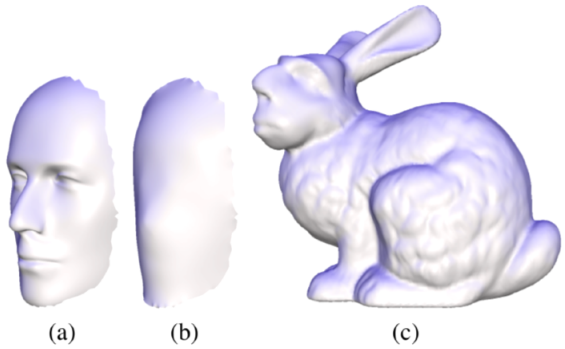

Figure 7: *Transferring the coating of the Mannequin onto the face of the Bunny. (a) The source surface S. It is significantly smoothed to peel the coating. (b) The smoothed surface $\tilde{S}$. (c) The result of coating transfer onto the Bunny.*

图 4.10: 兔身人面兽

4.3.4 混合细节（mixing details）

接下来我们想把两个表面混合在一起，构成第三个表面。只需要对重叠的区域的拉普拉斯坐标进行线性插值。

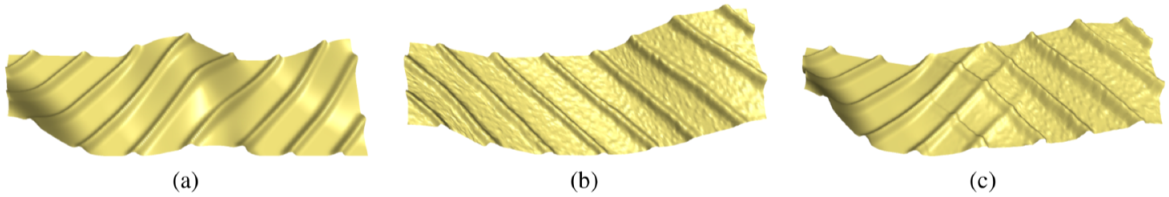

Figure 8: *Mixing details using Laplacian coordinates. The Laplacian coordinates of surfaces in (a) and (b) are linearly blended in the middle to yield the shape in (c).*

图 4.11: 融合

4.3.5 移植表面（transplant surface patches）

最后的任务是把一个表面截取出来移植到另一个表面中。这也并不困难，只需要选取移植的区域，并保证对应的连接处一致，这也可以通过将连接处映射到相同域上（如单位正方形参数域）实现。最后在连接处使用上面的混合细节法即可。

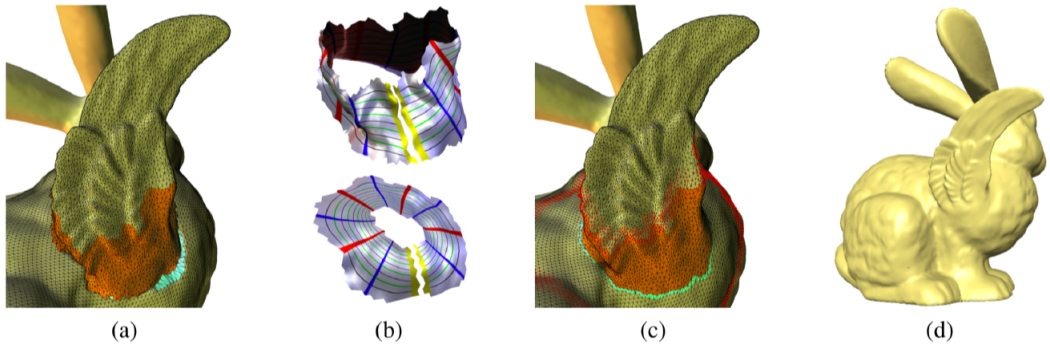

Figure 10: *Transplanting of Feline's wings onto the Bunny (see also the color section). (a) After cutting the parts and fixing the desired pose, the zipping (in green) defines the target connectivity D. The transitional region D' is marked red. (b) D' is sampled over the respective regions $U' \subset U_\circ$ ($U_\circ$ is the cut part of the Bunny's back) and S' (the bottom of the wing). The texture with uv-isolines visualizes the mapping over the unit square. The cut (in yellow) aligns the two maps. (c) The result of reconstruction. The reconstructed submesh is padded by a belt of anchors (red dots). Note the change of the zipping seam triangles (in green) and the gradual change and preservation details within the transition region (red). (d) The flying Bunny.*

图 4.12: 移植

参考文献

[1] Daniel Cohen-Or,Chen Greif,Tao Lu,Niloy J.Mitra,Ariel Shamir,Olga Sorkine-Hornung and Hao (Richard) Zhang A Sampler of Useful Computational Tools for Applied Geometry, Computer Graphics, and Image Processing

[2] O. Sorkine,D. Cohen-Or and Y. Lipman "Laplacian surface editing." Eurographics/acm Siggraph Symposium on Geometry Processing ACM, 2004.