

第九章 几何处理

随着三维技术的发展，各种三维应用层出不穷，比如 VR、AR 以及形形色色的相关产业，使得大家对高质量的三维模型的需求越来越旺盛。其中，几何处理技术作为获得高质量三维模型的基础，更是得到了越来越多的关注。这是一个十分庞大的领域，包含了许多子方向，如从点云重建表面、去噪滤波、几何分析、形状简化以及几何建模和交互式设计等等。

本章将以应用最为广泛的三角网格为基础，介绍几何处理的一些基本概念以及一些简单的几何处理算法。我们将首先介绍离散微分几何中的一些常用概念，这是后续几何处理算法的操作基础，然后我们将分别讨论网格平滑、网格简化、网格编辑等几何处理算法。网格细分和网格参数化我们已在前面的章节中介绍过，这里不再赘述。

9.1 离散微分几何

离散微分几何 (discrete differential geometry) 是几何处理的数学基础。传统的微分几何是在连续空间中的进行几何分析的，如光滑的参数化曲面，而计算机中的一切量都是离散的，如由连续曲面离散化成的三角网格，离散微分几何就是将二者进行联系的桥梁。这里我们会简单介绍离散微分几何中几个比较基础的微分量。

在连续域的几何分析中，通常会定义一些有着独特性质和重要应用的微分量，但对于离散的三角网格而言，在边或顶点的连接处，很多微分量是没有定义的，因为由三角网格定义出的几何形状只能给出 C^0 连续性。因此，如何将它们应用在离散的情况下是离散微分几何的研究重点。这部分我们将介绍如何在三角形网格上定义以及计算一些连续微分量的离散近似，从而为后面的几何处理提供数学帮助。离散微分几何博大精深，本章无法完全覆盖，我们推荐感兴趣的同学观看 CMU 的 DDG 课程。

9.1.1 局部平均区域

首先，我们介绍局部平均区域 (*local averaging region*)。为了定义三角网格顶点上的微分量，我们考虑从积分的角度来描述它：我们计算这个微分量在顶点周围邻域内的积分值，然后假定该微分量在顶点周围邻域内变化很小，可以被近似看成一个定值，于是将积分值除以邻域面积，就可以得到该微分量。

记该顶点为 x ，顶点邻域的选取方法会直接影响到离散微分量的结果和准确度：若选取的邻域较大，那么得到的微分量在空间上会更为平滑，若选取的邻域较小，得到的微分量会更加准确，同时对于网格上的噪声也会更加敏感。图9.1展示了较为常见的三种定义邻域的方式。

1. 重心单元 (barycentric cell)，即把三角形的两边的中点和三角形的重心顺次连起来。
2. 泰森多边形单元 (Voronoi cell)，是由每条边的中垂线确定的区域，根据中垂线的性质，每个三角形内，蓝色区域中的点到 x 的距离都会小于另两个顶点。但是正如图中所示，若

顶点周围存在钝角三角形，则蓝色区域会超出三角形.

3. 混合泰森多边形单元 (mixed Voronoi cell) 是对上一种方法的改进，在出现钝角三角形时，用三条边的中点连线来框选出该三角形内的蓝色区域.

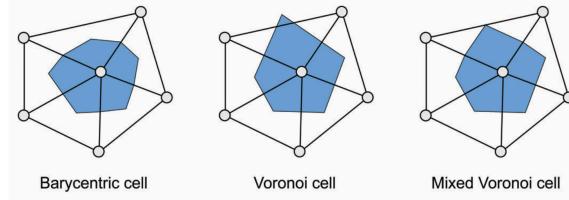


图 9.1: 计算局部平均区域时对顶点邻域的三种选取方式，所选定的邻域用蓝色标出.

9.1.2 法向量

三角形内部点的法向量可以唯一定义成三角形所在平面的法向量，而顶点的法向量则可以通过对顶点周围三角形法向量取平均得到：

$$\mathbf{n}(\mathbf{x}) = \frac{\sum_{T \in \Omega(\mathbf{x})} \alpha_T \mathbf{n}(T)}{\|\sum_{T \in \Omega(\mathbf{x})} \alpha_T \mathbf{n}(T)\|} \quad (9.1)$$

其中， $\Omega(\mathbf{x})$ 是与顶点 \mathbf{x} 邻接的三角形的集合， α_T 及 $\mathbf{n}(T)$ 分别是三角形 T 的权重和法向量. 对于权重 α_T 的选择有很多方式，常见的有以下三种：

1. $\alpha_T = 1$. 取常数权重是最容易计算的，不过这种取法没有考虑任何三角形的信息，对于不规则的网格会得到一些反常的结果.
2. $\alpha_T = \text{area}(T)$. 根据三角形面积大小进行加权比常数权重更加合理，且面积大小可以通过两条边叉乘得到，便于计算. 但遇到过于不规则的网格时这种做法也不够可靠.
3. $\alpha_T = \theta(T)$. 根据三角形对应顶点 \mathbf{x} 处的角度进行加权可以给出更好的结果，不过这种做法计算起来比较麻烦.

9.1.3 梯度

梯度是非常重要的一阶微分量，在计算拉普拉斯算子以及网格参数化、网格形变中都有着重要的作用.

对单个三角形而言，给定三个顶点 $\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k$ 上的函数值 f_i, f_j, f_k ，则在三角形内部任一点 \mathbf{x} 上的函数值 $f(\mathbf{x})$ 可以根据点 \mathbf{x} 的重心坐标 (barycentric coordinate) 进行线性插值得到：

$$f(\mathbf{x}) = \alpha(\mathbf{x})f_i + \beta(\mathbf{x})f_j + \gamma(\mathbf{x})f_k \quad (9.2)$$

其中权重 α, β, γ 是点 \mathbf{x} 关于顶点 $\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k$ 的重心坐标，且满足 $\alpha + \beta + \gamma = 1$. 如图9.2所示定义，重心坐标表征的是该点到各顶点的相对位置，一组重心坐标可以唯一地确定三角形中的一点. 其中权重 α, β, γ 是点 \mathbf{x} 关于顶点 $\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k$ 的重心坐标，且满足 $\alpha + \beta + \gamma = 1$. 如图9.2所示定义，重心坐标表征的是该点到各顶点的相对位置，一组重心坐标可以唯一地确定三角形中的一点.

那么函数 $f(\mathbf{x})$ 的导数为：

$$\nabla_{\mathbf{x}} f(\mathbf{x}) = f_i \nabla_{\mathbf{x}} \alpha + f_j \nabla_{\mathbf{x}} \beta + f_k \nabla_{\mathbf{x}} \gamma \quad (9.3)$$

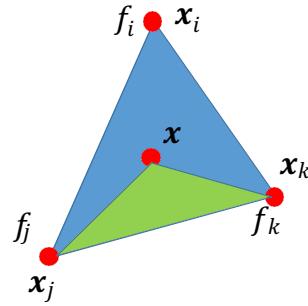


图 9.2: 重心坐标 $\alpha(\mathbf{x})$ 的值等于绿色部分三角形与整个三角形的面积之比.

代入重心坐标的表达式可以算得:

$$\nabla_{\mathbf{x}} f(\mathbf{x}) = (f_j - f_i) \frac{(\mathbf{x}_i - \mathbf{x}_k)^{\perp}}{2A_T} + (f_k - f_i) \frac{(\mathbf{x}_j - \mathbf{x}_i)^{\perp}}{2A_T} \quad (9.4)$$

其中 \perp 表示在三角形平面上旋转 90 度, A_T 是三角形的面积, 推导过程此处不再赘述. 可以看出, 经过线性插值得到的 $f(\mathbf{x})$ 的梯度在一个三角形内是定值.

9.1.4 离散拉普拉斯算子

拉普拉斯算子 (Laplace-Beltrami operator) 是一个非常重要的工具, 涉及几何处理的方方面面, 包括网格滤波、参数化、压缩、差值以及模拟等等. 拉普拉斯算子被定义为函数梯度的散度:

$$\Delta f = \operatorname{div} \nabla f \quad (9.5)$$

对于标量函数来说, 即 $\Delta f = \sum_i \frac{\partial^2 f}{\partial x_i^2} \in \mathbb{R}$. 从定义中可以看出, 拉普拉斯算子是一个二阶微分量. 拉普拉斯算子应用在标量函数上表达的是函数的趋势, 应用在几何表面上刻画的就是几何形状的走向. 例如, 我们可以通过拉普拉斯算子来定义曲面的平均曲率 H , 它刻画的是曲面某处的弯曲程度. 记函数 \mathbf{S} 是映射到三维空间中的曲面, 曲面上某点处的平均曲率为 H , 法向量为 \mathbf{n} , 则有:

$$\Delta \mathbf{S} = -2H\mathbf{n} \in \mathbb{R}^3 \quad (9.6)$$

在离散情形下, 由于三角形面内的梯度是定值, 所以拉普拉斯算子恒为 0. 因此我们通常从三角形顶点的值出发研究拉普拉斯算子. 利用二阶差分的思想, 函数 $f(\mathbf{x}_i)$ 在顶点 \mathbf{x}_i 处的拉普拉斯算子的一般形式可以写成:

$$\Delta f_i = \sum_{j \in \Omega(i)} \omega_{ij} (f_j - f_i) \quad (9.7)$$

这里, ω_{ij} 是标量权重, $\Omega(i)$ 是顶点 \mathbf{x}_i 的邻居顶点集合. 不同权重选取方式对应着不同的离散拉普拉斯算子, 这里我们介绍其中两种.

均匀拉普拉斯

均匀拉普拉斯 (uniform Laplacian) 是最简单的一种形式, 它将权重取为定值, 定义为:

$$\Delta f_i = \sum_{j \in \Omega(i)} (f_j - f_i) / N_i \quad (9.8)$$

其中 N_i 表示邻居顶点的数量. 均匀拉普拉斯算子计算的是函数 f 从 \mathbf{x}_i 到其所有周围顶点 \mathbf{x}_j 的差的平均值. 这种定义方法便于高效计算, 并且被用于滤波等几何处理. 但实际上它并不是一种恰当的定义方式, 例如, 对于平面而言, 由于曲率为 0, 按照连续情形下的公式9.6, 它的拉普拉斯的值也应该为 0, 但对平面进行不规则的离散化后, 很明显式 (9.8) 不一定满足这个要求. 也就是说, 我们不能保证一个平面经过三角网格划分后, 网格顶点 \mathbf{x}_i 一定处于它所有邻居的中心.

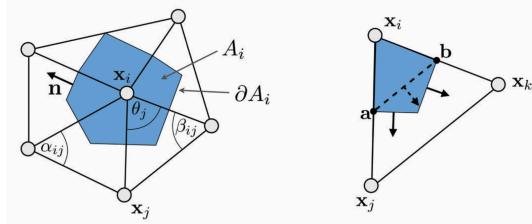


图 9.3: 计算余切拉普拉斯算子的方式.

余切拉普拉斯

一种更加合理的离散拉普拉斯算子定义方式是通过局部平均区域思想得到的余切拉普拉斯 (cotangent Laplacian), 它可以通过混合有限元 (mixed finite element) 或者有限体积 (finite volume) 方法得到. 为了简化积分的运算, 我们需要用到高斯散度定理:

$$\int_{A_i} (\Delta f) dA = \int_{A_i} \operatorname{div}(\nabla f) dA = \int_{\partial A_i} (\nabla f) \cdot \mathbf{n} ds \quad (9.9)$$

在计算的时候, 我们分别考虑每个三角形的积分, 如图9.3所示, 因为在一个三角形网格上, 梯度是不变的, 同时带入法向的定义:

$$\int_{\partial A_i \cap T} (\nabla f) \cdot \mathbf{n} ds = \frac{1}{2} (\nabla f) \cdot (\mathbf{a} - \mathbf{b})^\perp = \frac{1}{2} (\nabla f) \cdot (x_j - x_k)^\perp$$

将梯度的定义带入其中, 得到

$$\int_{\partial A_i \cap T} (\nabla f) \cdot \mathbf{n} ds = \frac{1}{2} (\cot \gamma_k (f_j - f_i) + \cot \gamma_j (f_k - f_i))$$

其中, γ_k, γ_j 是点 x_k, x_j 处的三角形内角. 因此

$$\int_{\partial A_i} (\nabla f) \cdot \mathbf{n} ds = \frac{1}{2} \sum_{j \in \Omega(i)} (\cot \alpha_{ij} + \cot \beta_{ij}) (f_j - f_i)$$

由于假定拉普拉斯算子在所选取的区域内是个定值, 因此我们可以得到它在顶点 \mathbf{x}_i 上的值为:

$$(\Delta f)_i = \frac{1}{2A_i} \sum_{j \in \Omega(i)} (\cot \alpha_{ij} + \cot \beta_{ij}) (f_j - f_i) \quad (9.10)$$

这种拉普拉斯算子会满足公式9.6的性质, 也是应用最广的一种形式. 值得注意的是, 当式中两个角度大于 π 时, $(\cot \alpha_{ij} + \cot \beta_{ij})$ 会变为负数, 这会导致一些三角形发生旋转, 具体情况可以参考关于网格参数化的书籍.

9.2 网格平滑

随着三维扫描和曲面重建技术的发展，得到实体表面的多边形网格表示已经不是难事，但所得到的表面往往包含噪声。在形状设计领域，散点拟合、形状光滑、纹理映射等应用都有对平滑曲面的极大需求。故产生了网格平滑这一个研究点。

网格平滑 (smoothing) 也称为网格降噪 (denoising)。降噪一般是指去掉噪音（高频信息）保留整体形状（低频信息）从而获得一个相对光滑的几何模型的过程，比如去掉扫描或者采集数据因为误差或者其他原因导致的噪音。平滑在统计学中指的是通过一个近似函数对输入数据进行较好的拟合，从而保留重要的模式，去掉噪音或者其他不重要的信息。如图9.4所示，我们可以对一个具有噪音的初始扫描人脸进行平滑处理。实际上，噪音在几何上是很难判定的，并不是高频的就一定是噪音，比如棱角分明的几何体的顶点。因此，不严格的对噪音的定义会给算法的设计带来许多困难。本部分我们主要介绍拉普拉斯平滑 (Laplacian smooting)。

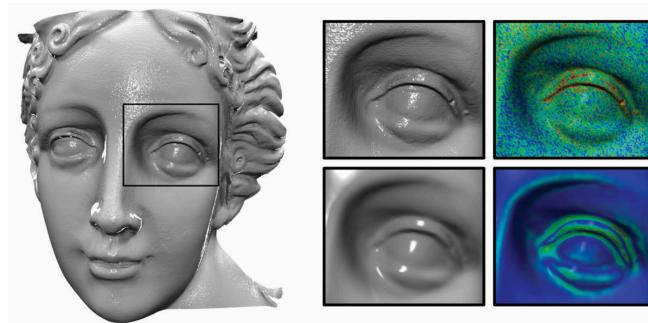


图 9.4：左边是带有噪音的三维扫描的人脸。右边上面一行表示原始的几何以及平均曲率，下面表示经过降噪后的几何和平均曲率。

我们可以借助扩散流的数学模型来帮助理解，它常被用来平滑给定的时空信号 $f(x, t)$ 。很多常见的物理现象可以用扩散流解释，比如热量从高温处逐渐往低温处扩散的过程，可以看作是一个扩散流的过程。扩散流公式写作：

$$\frac{\partial f(x, t)}{\partial t} = \lambda \Delta f(x, t) \quad (9.11)$$

扩散流的过程应用到网格平滑上，也就是将带噪音的顶点坐标理解成杂乱分布的热量。随着扩散过程的进行，热量分布趋于平衡，对应着顶点构成的表面趋于平滑。因此，我们可以通过定义在表面上的拉普拉斯算子，利用扩散流模型，对表面进行平滑。首先，我们需要对公式中的时空信息进行离散化。

对于空间上的离散化，我们依旧把函数值离散为网格顶点上的值，记 $\mathbf{f}(t) = (f(v_1, t), \dots, f(v_n, t))^T$ ，于是对每个顶点有：

$$\frac{\partial f(v_i, t)}{\partial t} = \lambda \Delta f(v_i, t), i = 1, \dots, n. \quad (9.12)$$

写成矩阵形式为 $\partial \mathbf{f}(t) / \partial t = \lambda L \mathbf{f}(t)$ 。

对于时间上的离散化，我们可以采用均匀间隔的时间步长 h 将时间划分为若干段 $(0, h, 2h, \dots, t, t+h, t+2h, \dots)$ ，于是通过有限差分可以将偏微分用差值来近似，即：

$$\frac{\partial \mathbf{f}(t)}{\partial t} = \frac{\mathbf{f}(t+h) - \mathbf{f}(t)}{h} \quad (9.13)$$

这种差分格式被称为显式欧拉，它的形式可以改写为：

$$\mathbf{f}(t+h) = \mathbf{f}(t) + h \frac{\partial \mathbf{f}(t)}{\partial t} = \mathbf{f}(t) + h \lambda L \mathbf{f}(t) \quad (9.14)$$

此式给出了经过时空离散化后函数值的迭代更新方法，每经过一个 h 时间步，用此式更新每个顶点上的函数值，直至达到设定的迭代停止条件。

在上式中代入第 k 次更新后的顶点坐标 $f(v_i, kh) = \mathbf{x}_i^{(k)}$ ，我们就可以得到拉普拉斯平滑在第 $k+1$ 步的更新公式：

$$\mathbf{x}_i^{(k+1)} \leftarrow \mathbf{x}_i^{(k)} + h \lambda \Delta \mathbf{x}_i^{(k)} \quad (9.15)$$

在两种不同的拉普拉斯算子下，可以得到不同的平滑效果。在取余切算子时，由于满足 $\Delta \mathbf{x} = -2H\mathbf{n}$ 公式，于是所有顶点将沿着平均曲率定义的方向移动，因此也称为平均曲率流 (mean curvature flow)。图9.5展示的就是对模型进行平均曲率平滑之后的效果。



图 9.5: 对左边的网格进行曲率平滑，中间是迭代 10 次的结果，右边是迭代 100 次的结果。

在取均匀算子的情况下，因为没有考虑面的信息，只考虑了顶点位置，那么平滑不会沿着曲率方向，而是会往重心方向移动，即将每个顶点往相邻顶点的平均位置移动，如图9.6所示。这个过程等价于某种能量函数最小化，平衡状态下的形状应该是所有边长度相同。这种方式定义的拉普拉斯算子不反映网格形状，可能导致局部几何特征的失真。从图9.7中可以看出，余切算子能更好地保持几何特征，而均匀算子导致了三角网格切向的松弛。

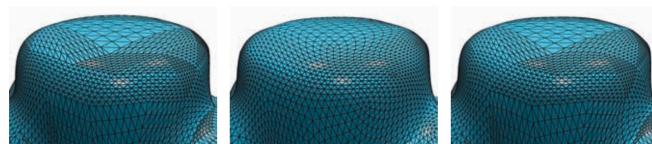


图 9.6: 使用不同算子对左边的网格平滑（十次迭代）。均匀算子会使得规则化三角形的重心（中心），而余切算子会保留三角形形状（右图）。

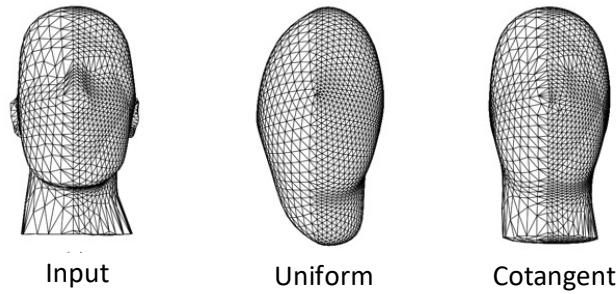


图 9.7: 不同算子在网格平滑的过程中对几何特征保持的不同程度。

9.3 网格简化

在游戏、科学可视化等计算机图形学的应用中，渲染效率与渲染质量是要兼顾的。当前大部分的计算机性能还不足以以较快的速度渲染海量三角形。经过精心制作的模型，或者3d扫描仪扫描并重构得到的模型一般都有几十万、几百万甚至更多的三角形，但不是所有时候都需要用最高精度的模型。例如，同样的模型处在远处时，所占屏幕像素数会比较小，必然无法看法，这时候再渲染超多顶点的高精度的模型就会浪费计算性能。这时候如果在远处的能用更低精度的模型渲染，那么在不太影响渲染质量的情况下可以提高效率。这就需要用网格简化算法来生成不同细节层次（Level Of Detail，简称LoD）的模型，于是可以在不同情况下选择不同细节层次的模型来渲染。



图 9.8: 网格简化。将初始形状（左上）简化到 10%（右上）、1%（左下）以及 0.1%（右下）。

网格简化（mesh simplification）的目标是用更少的面片数来表达原有模型，降低模型精度的同时保持模型的整体形状。网格简化分为静态简化和动态简化两类。静态简化预先计算好一系列不同简化率的模型，在实时运行的程序中可以按照模型离视点（view point）的距离选择不同版本的模型进行渲染。动态简化是静态简化的延伸，在运行过程中按照需要对模型进行简化，一般使用局部的几何变换来实现，从而生成具有连续的具有不同分辨率的近似模型。

简化网格一般可以通过移除顶点或坍缩边来进行，相对来说边坍缩（edge collapsing）算法是更为简单而常用的，它通过删除网格中的边来实现简化的效果。这类算法的核心是设计一种方法来选出需要被删除的边，并将这条边两端顶点合并成一个点放置在新的位置上。根据不同原则设计出的不同方法直接关系到简化后的模型的质量，不良的处理方法甚至会导致网格模型出现拓扑上的错误，如图9.9所示。

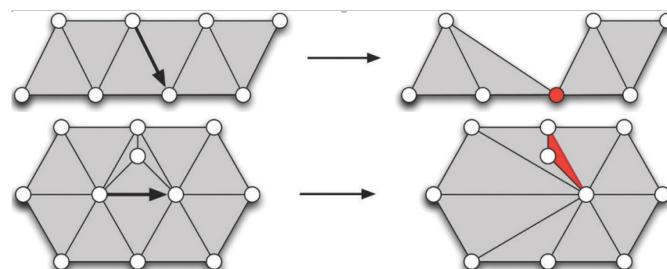


图 9.9: 边坍缩算法可能导致的拓扑问题。

为了衡量坍缩某条边对模型造成的影响，我们需要引入一种误差度量的方式，一种最常用的方式是二次误差度量（quadratic error metrics）[1]。在删除一条边后，我们需要引入一

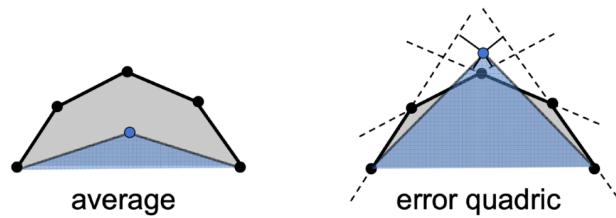


图 9.10: 根据局部邻域平均或根据取最小化二次误差度量来设置新的顶点, 黑色代表原网格, 蓝色代表经过边坍缩后的网格.

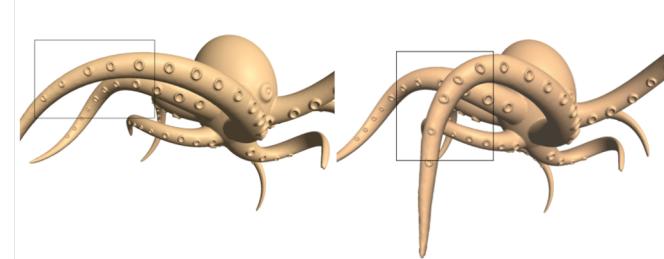


图 9.11: 拉普拉斯网格编辑. 左侧章鱼模型的腕足经过编辑后形状发生了变化, 但保持了腕足上吸盘的几何细节.

一个新的顶点, 通过最小化新的顶点与之前对应的平面的 L_2 距离, 我们可以得到删除这条边会引入的二次度量误差, 于是我们可以选取引入二次度量误差最小的边进行坍缩. 而另一方面, 使得这条边二次度量误差最小化的顶点, 也将被设置为边坍缩后的新顶点, 如图9.10所示, 这种设置方式优于取边中点或局部区域平均, 可以更好地保留模型的几何特征. 利用这种贪心的思想, 通过不断的迭代坍缩, 我们就能实现对网格的简化.

网格简化是一个复杂的问题, 涉及到多种多样的度量方案和评判准则, 目前每年仍有很多关于网格简化的文章发表, 以期达到普适、高效、鲁棒、简化率高、可交互等目标, 或在各目标中寻求平衡. 有些研究者将神经网络运用到网格简化中, 同样取得了不错的效果, 这里我们不再对这个课题更加深入, 感兴趣的同学可以自行搜索文献阅读.

9.4 网格编辑

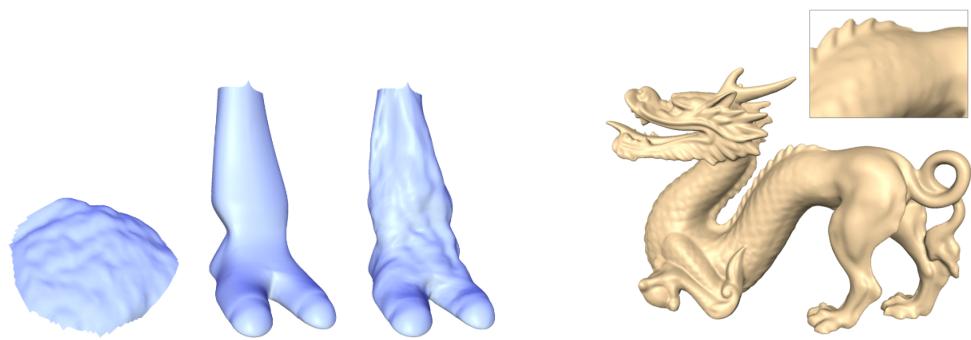
网格编辑 (mesh editing), 是操纵和修改网格表面的几何形状, 同时能够保留原始网格几何细节 (detail preserving) 的操作. 由于几何细节是对网格内在特质的描述, 因此直接编辑网格顶点的位置坐标并不是一个好的想法, 往往导致细节的失真.

我们可以首先回顾在图像编辑中的做法: 如何给图片中的特定区域注入某些给定信息? 对于定义在区域 Ω 上的目标颜色场 $g(x, y)$, 泊松图像编辑通过求解下面的优化问题来得到融入后的颜色场 $f(x, y)$:

$$\arg \min_f \int_{\Omega} \|\nabla f - \nabla g\|^2 dx dy, \quad \text{s.t.} \quad f|_{\partial\Omega} = f^*|_{\partial\Omega} \quad (9.16)$$

它在保证边界上无缝衔接的同时使得填补后区域的颜色梯度 ∇f 与给定颜色场的梯度 ∇g 尽可能接近. 拉普拉斯编辑是 $\nabla g = 0$ 的一类特殊情况, 它完成的是图像补全的任务.

类似的思想在三维的应用就是拉普拉斯网格编辑技术 [2], 它通过尽可能对齐编辑后网格上的拉普拉斯微分坐标来保留细节内容, 如图拉普拉斯微分坐标本质上就是每个网格顶



(a) 涂层迁移: 将兔子模型的几何细节 (左) 移植到动物腿的 (b) 网格移植: 将虎模型的后半部分移植
模型上 (中) 形成兔毛涂层的效果 (右). 到龙模型上.

图 9.12: 拉普拉斯网格编辑在涂层迁移和网格移植任务上的应用.

点上关于位置坐标的均匀拉普拉斯算子, 记为:

$$\mathcal{L}(\mathbf{v}_i) = \mathbf{v}_i - \frac{1}{N_i} \sum_{j \in \Omega(i)} \mathbf{v}_j \quad (9.17)$$

它是一种相对坐标, 在给定边界条件的情况下可以与世界空间的位置坐标互相转化. 拉普拉斯坐标是网格顶点坐标的线性组合, 与点的法向和曲率无关, 因此可以被用来描述局部特征.

拉普拉斯网格编辑的思路是: 在交互中指定一些顶点的目标位置后, 重建一个既满足这些顶点约束、又尽可能保持局部拉普拉斯微分坐标的网格. 具体来说, 设给出的具有 n 个顶点的网格模型的顶点坐标为 $\{\mathbf{v}_1, \dots, \mathbf{v}_n\}$, 首先计算出对应的拉普拉斯坐标: $\Delta_i = \mathcal{L}(\mathbf{v}_i)$, $i = 1, \dots, n$. 而后对需要调整的顶点给出约束: $\mathbf{v}'_j = \mathbf{u}_j$, $j \in C$, 其中 C 是受约束点的集合. 于是我们可以构造出拉普拉斯网格编辑对应的最优化问题:

$$\arg \min_{\mathbf{v}'_1, \dots, \mathbf{v}'_n} \left(\sum_{i=1}^n \|\mathcal{L}(\mathbf{v}'_i) - \Delta_i\|^2 + \sum_{j \in C} \|\mathbf{v}'_j - \mathbf{u}_j\|^2 \right) \quad (9.18)$$

利用最小二乘方法求解该优化问题即可得到生成编辑后的网格.

拉普拉斯网格编辑技术可以被用在众多应用中, 包括并不限于网格编辑、涂层迁移、表面移植等等, 如图9.12a、9.12b所示.