Laplacian Surface Editing

22121053 翁罗轩

背景介绍

表面编辑操作需要尽可能地保留表面的几何细节，传统的网格自由变形方法通常会导致表面细节的严重失真。这篇04年的文章通过拉普拉斯坐标来对网格进行编辑，从而能在保留细节和结构的前提下对表面进行变形。这一算法主要有三个方面的应用：

- 网格编辑（Mesh Editing）：在保留细节的同时对网格进行编辑。
- 涂层迁移（Coating Transfer）：将一个网格的表面细节迁移到另一个网格上。
- 曲面块移植（Transplanting）：将两个网格嫁接到一起。

算法

作者提出的算法基本思路是：

1. 将网格坐标（绝对坐标）转换为拉普拉斯坐标（相对坐标）。
2. 在拉普拉斯坐标中对网格进行处理。
3. 将拉普拉斯坐标转换回网格坐标。

顶点的拉普拉斯坐标定义如下：

$$\mathscr{L}(\mathbf{v}_i) = \mathbf{v}_i - \frac{1}{d_i} \sum_{j \in \mathscr{N}_i} \mathbf{v}_j$$

简单地说，就是将一个顶点的坐标表示为其绝对坐标和其邻接点平均坐标的差值。用矩阵运算来表示一张图中所有顶点的拉普拉斯坐标，如下：

$$\Delta = LV$$

其中，L是图的拉普拉斯矩阵（文章中使用的是一种归一化形式，称为Random Walk Laplacian），定义为L = I - (D^-1)A，D和A分别为这张图的度数矩阵和邻接矩阵。可以发现，L的秩是n - 1，意味着通过直接求解线性方程V = (L^-1)Δ来还原顶点坐标是不可行的。解决方法就是给这个线性方程组增加约束，固定住其中的一些点来求解其他的点，这些被固定的点称为控制点（handle point）。

文章指出，如果控制点的目标位置满足线性方程组的最小二乘解而不是精确相等的话，可以得到更好的效果。误差公式如下：

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

因此，顶点坐标的求解转化为最小化E(V')。该公式的前半部分表示V'的相对坐标与原网格的相对坐标Δ尽可能保持一致，后半部分表示控制点的坐标在变形后尽可能处于既定的目标位置。

由于拉普拉斯坐标仅具有平移不变性，但对旋转和缩放敏感。为了解决这一问题，文章提出的方法是为每个顶点计算一个基于新坐标V'的仿射变换矩阵T'，它记录了每个顶点及其邻域在变换过程中发生的旋转和缩放。至此，原来的误差公式转换为下式：

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

这里Ti和V'都是未知的，但由于Ti是关于V'的线性函数，因此求解出V'自然可以推出Ti：

$$\min_{T_i} \left(\left\| T_i \mathbf{v}_i - \mathbf{v}_i' \right\|^2 + \sum_{j \in \mathcal{N}_i} \left\| T_i \mathbf{v}_j - \mathbf{v}_j' \right\|^2 \right)$$

上式的前半部分和后半部分分别表示原顶点及其邻接点在经过Ti的变换后能接近于新求解出来的坐标。将Ti表示成下式：

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

其中，s，h和t分别是表示缩放、旋转和平移变换的向量，文章通过应用这个矩阵来解决平移、旋转和缩放的不变性问题。因此，上面的约束可以转化为最小化以下式子：

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

这里Ai是原顶点及其邻接点坐标的矩阵，bi是新顶点及其邻接点坐标的矩阵，如下所示：

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

$$\mathbf{b_i} = \begin{pmatrix} v'_{k_x} \\ v'_{k_y} \\ v'_{k_z} \\ \vdots \end{pmatrix}, k \in \{i\} \cup \mathcal{N}_i$$

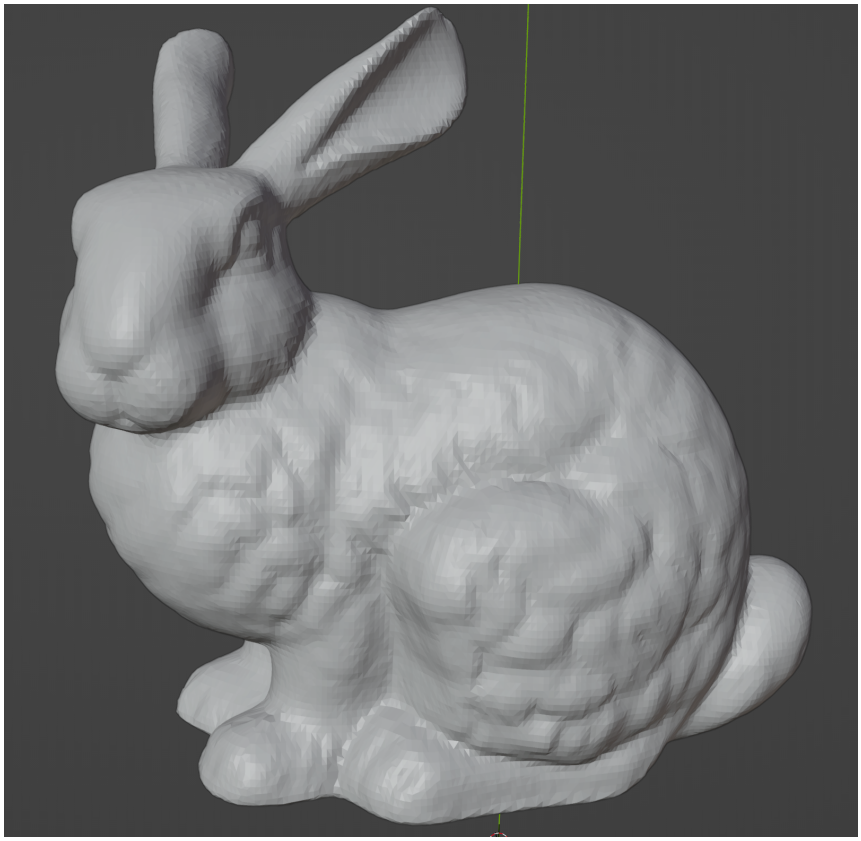
那么，使用最小二乘法求解这个线性方程组就能得到s，h和t的值，进而计算出Ti及V'的值。

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

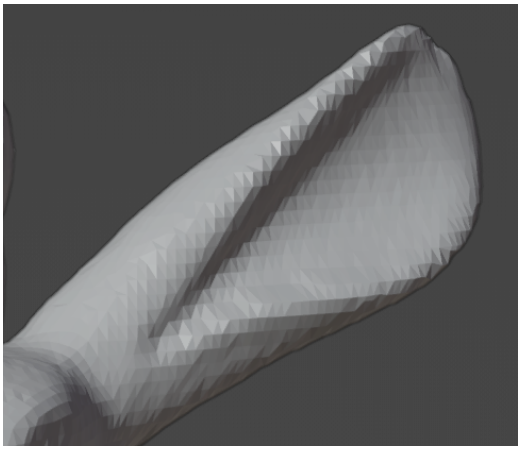
实现

有了上述的理论基础，使用Python对这一算法进行了实现。运行结果如下：

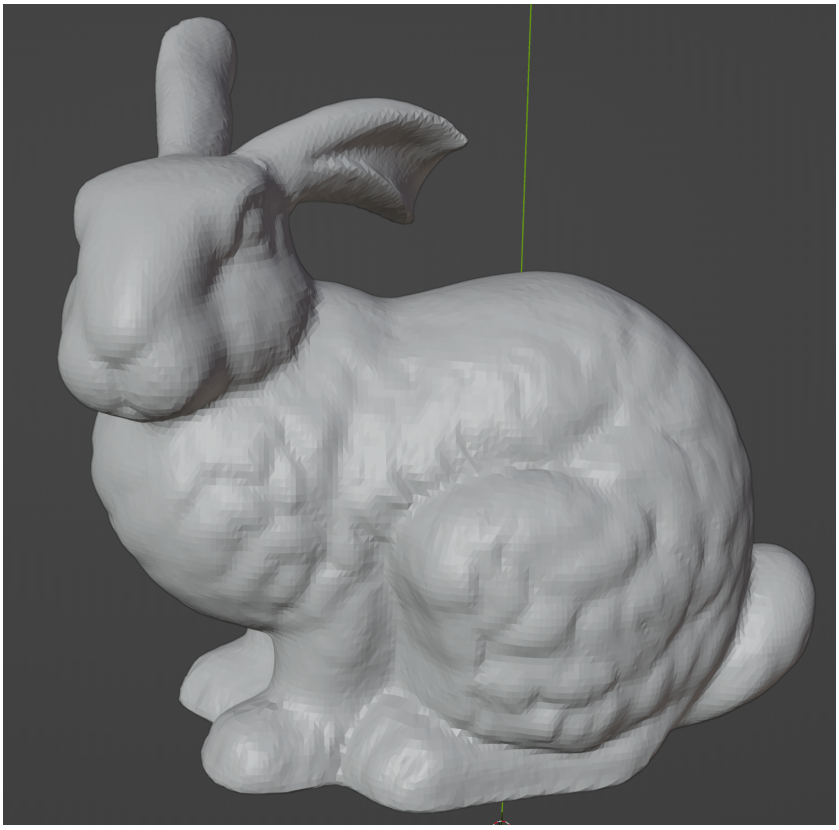
原网格



选定编辑区域



形变后的网格



可见，兔子的耳朵在尽可能保留几何细节的情况下正确地进行了形变。

运行说明

- 原网格保存在laplacian_surface_editing/meshs目录，形变后的网格保存在laplacian_surface_editing/edit_meshs目录。
- 修改laplacian_surface_editing/main.py中的ply_data以更改待形变的网格路径；修改handle_vertices和ROI_vertices以更改控制点和边界点。这里的顶点序号是通过将网格导入三维建模软件（例如Blender）来获取的。

参考资料

- 论文链接：https://people.eecs.berkeley.edu/~jrs/meshpapers/SCOLARS.pdf
- 网格数据来源：http://graphics.stanford.edu/data/3Dscanrep/