Laplace surface editing

> 李晗驰 PB19020583

问题描述

计算机图形的表面大多使用一个全局的坐标系表示，其中外侧一般用以绝对欧式坐标表示的点集来刻画，而内侧则用在欧氏空间中的一系列约束函数描述。在渲染、变形等操作时，全局的坐标系是一个自然朴素的选择，不过对于表面模型，我们更希望它可以展示出自己的形态，而非仅仅使用在欧氏空间中的绝对位置。

为了能够减少网格的形态特征在进行表面编辑时的丢失，我们希望使用拉普拉斯坐标来描述网格，并为此实现一种编辑方法，使得我们进行网格编辑时，能够在保型的同时，满足我们的编辑约束需求。

算法内容

拉普拉斯坐标拟合变换

给定网格$\mathcal{M}$，以(K,V)来描述它，其中K指的是对连接性的描述，而V指的是所有三维空间中描述该网格的点，即$V=\{v_1,v_2,\ldots,v_n\}$我们定义一个邻居环$\mathcal{N}_i=\{j|(i,j)\in K\}$，定义度数$d_i$为每个点$v_i$对应的邻居环中的元素个数。

我们用一组差分$\Delta=\{\delta_i\}$来描述网格的几何形状，其中坐标i将被用v_i与邻居差的平均值来表示，即$\delta_i=\mathcal{L}(v_i)$

当使用**均匀权重**时，$\mathcal{L}(v_i)=v_i-\dfrac{1}{d_i}\sum\limits_{j\in\mathcal{N}_i}v_j$，根据这个权重怎么确定从$V$到$\Delta$的变换，可以用矩阵代数来看：

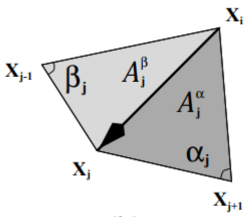
取A为网格的邻接矩阵，$D=diag(d_1,d_2,\ldots,d_n)$为度数矩阵，则有$\Delta=LV$其中$L=I-D^{-1}$

注意我们一般认为这个L是对具有邻接矩阵A的网格的拉普拉斯算符。

当使用**cot权重**时，我们的方法是这样的：

对边界点，不做特殊处理，直接在L矩阵对角线填1。

对非边界点，找该点的所有邻接点，先将该点到临界点的向量归一化，再对每个临接点计算`cot`权重，如下图所示



(b)

取点x_i在L中对应的行里，x_j对应的列的权重值为$cot(\beta_j) + cot(\alpha_j)$

并最后在对角线填这些权重值的求和的负数。

由拉普拉斯坐标还原绝对坐标

因为L矩阵的秩为$n-1$，也即当我们固定了一个顶点的位置，就可以通过解线性方程求出其他顶点的坐标。

一般来说，我们可以固定一组顶点的绝对位置，$v'_i = u_i, i \in \{m, \dots, n\}, m < n$，然后将形状$V'$与所得到的$\Delta$拟合，来求解其他顶点的坐标。

当约束$\{u_i\}$在最小平方的意义上被满足，整个网格编辑结果表现会更好，也即我们要最小化以下误差函数：

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

论文希望去针对每个顶点i，计算T_i，使得$T_i(V')$为V'的函数，并且取误差函数为：

$$E(V') = \sum_{i=1}^{n} ||T_i(V')\delta_i||^2 + \sum_{i=m}^{n} ||v'_i - u_i||^2$$

为了解决问题，我们要找到T_i，这可以通过寻找将v_i和它的邻居转换到V'的过程求出：

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

作者通过分析写出T_i对应的矩阵

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

这样就可以写出T_i与V'的线性关系，取$(s_i, h_i, t_i)^T$为T_i的未知向量。我们希望最小化：

$$||A_i(s_i, h_i, t_i)^T - b_i||^2$$

而其中A_i包含v_i和其邻居的位置信息，b_i包含v'_i和其邻居的位置信息

这样就有：

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

最终这个问题通过

$(s_i, h_i, t_i)^T = (A_i^T A_i)^{-1} A_i^T b_i$ 解决，注意A_i通过初始网格可知，而b_i则来自V'

L与Δ的确定

关于矩阵L，根据不同的权重选取，可以通过计算来获得每一个点对应行所填的值，详细的讨论已经在第一部分给出。

$\Delta = LV$，通过这个可以直接求出网格在拉普拉斯坐标系下的表示。

添加对handle的rotation

$$\mathbf{R_q} = \begin{bmatrix} 1-2y^2-2z^2 & 2xy-2wz & 2xz+2wy \\ 2xy+2wz & 1-2x^2-2z^2 & 2yz-2wx \\ 2xz-2wy & 2yz+2wx & 1-2x^2-2y^2 \end{bmatrix}$$

通过确定旋转四元数的值，可以调用相关的结论，并直接对矩阵$L * R'$进行变换。

实现细节

L的计算

主要是注意在matlab中判断需求的是统一权重或是cot权重，分别针对不同的权重填入L矩阵的值。

矩阵Ai的构造

按照论文，点i与其邻接点，每个占三行的位置，将所知晓的x, y, z坐标按要求填入相应的位置。

```
    A = zeros(3 * n_i, 7);
    for r=1:length(Neighbors)
        A(r,:)        =  [V_i(1,r) 0 V_i(3,r) (-1)*V_i(2,r) 1 0 0];
        A(r+n_i,:)    =  [V_i(2,r) (-1)*V_i(3,r) 0 V_i(1,r) 0 1 0];
        A(r+2*n_i,:) =  [V_i(3,r) V_i(2,r) (-1)*V_i(1,r) 0 0 0 1];
    end
```

矩阵Ti的确定

根据矩阵Ai确定未知量，s和h1,h2,h3，

```
    Ainv = pinv(A);
    s = Ainv(1,:);
    h1 = Ainv(2,:);
    h2 = Ainv(3,:);
    h3 = Ainv(4,:);
```

此外这里不必考虑tx，ty，tz

```
T_delta = [s * delta_ix         - h3 * delta_iy + h2 * delta_iz
           h3 * delta_ix        + s * delta_iy  - h1 * delta_iz
           (-1) * h2 * delta_ix + h1 * delta_iy + s * delta_iz];
```

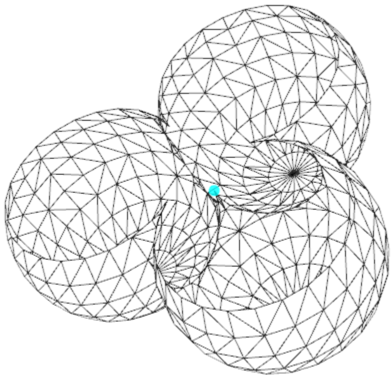
最终将所得的变化附加于矩阵L中

再根据L以及D和新的Δ计算变形后网格在绝对坐标系的坐标。

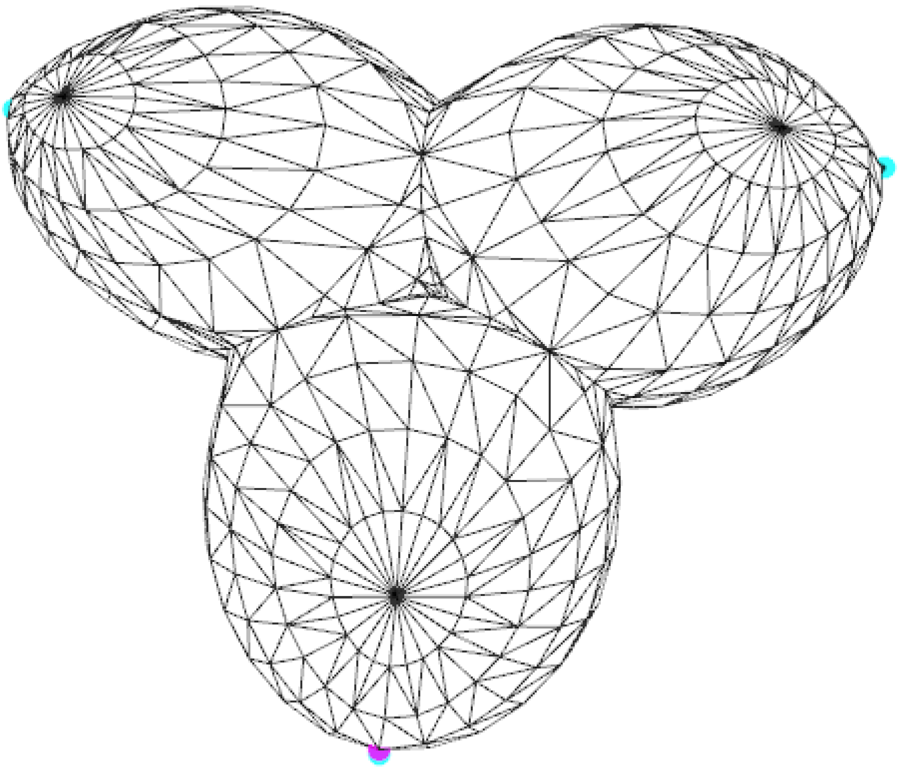
```
B = D' * D;
rhs = B * V;
A = L + B;
V_new_pos = A\rhs;
```

实验结果

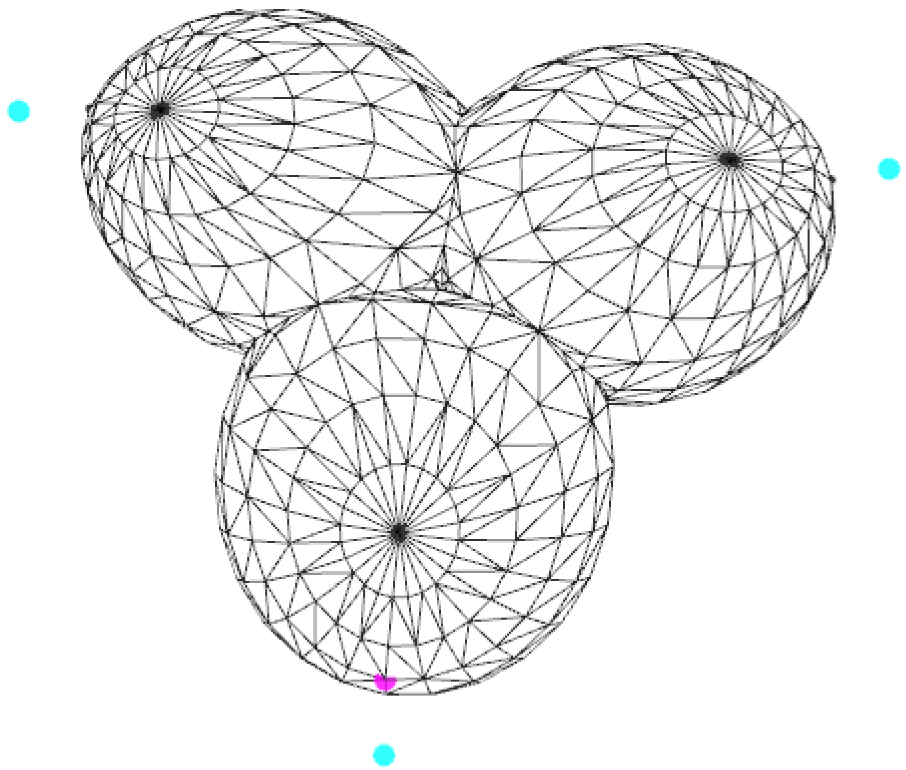
注意由于该算法的问题，应当先固定这个三圆网格的的中心尖锐点，再对其他进行编辑，否则这个点会有点影响效果：



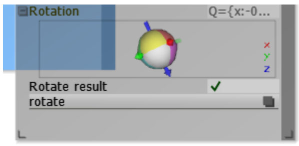
uniform weight

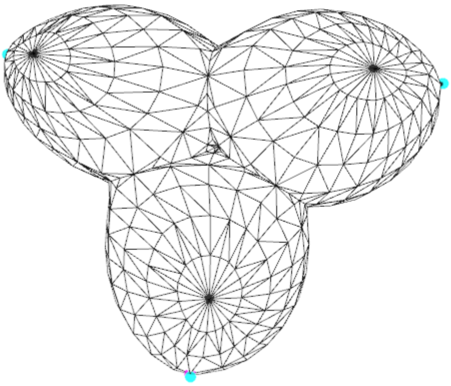


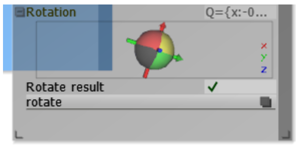
cot weight

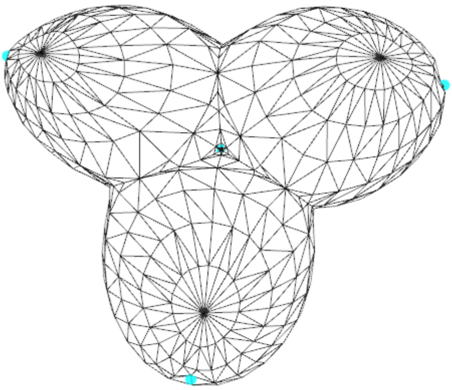


with rotation


Rotation
Q={x:-0...
x
y
z
Rotate result
rotate




Rotation
Q={x:-0...
x
y
z
Rotate result
rotate



camelhead