

1 几何表示

1.1 几何形状表示方法

几何形状 (geometric shape) 是指空间中的一组特定点的集合. 对于二维空间, 常见的几何图形包括线段, 多边形, 圆等等; 对于三维空间, 常见的几何图形包括线, 面, 体等.

1.1.1 连续函数表示法

和平面图形一样, 空间图形也可以用连续函数表示. 例如, 空间中的直线可以表示为

$$\{(x, y, z) : Ax + By + Cz + D = 0, x, y, z \in \mathbb{R}\}$$

空间中的球面可以表示为

$$\{(x, y, z) : x^2 + y^2 + z^2 = R^2, x, y, z \in \mathbb{R}\}$$

定义 1.1 连续函数表示法 一般地, 对于空间中的几何形状 M , 可以用一个特定的连续函数 $S_M(x, y, z)$ 刻画其表面 ∂M :

$$\{(x, y, z) : S_M(x, y, z) = 0, x, y, z \in \mathbb{R}\}$$

连续函数可以精确地刻画几何形状, 也方便研究其性质. 然而, 对于复杂的图形 (尤其是复杂的曲线或曲面), 难以找到合适的函数来表示; 并且这一表达形式是隐式的, 不能直接将几何形状呈现出来. 这就需要离散化的办法.

1.1.2 点云

通过类似雷达的工作方式对几何形状 M 进行扫描, 可以获知 ∂M 上一系列离散的点.

定义 1.2 点云 点云 (Point Cloud) 是一组三维空间中有限个点构成的集合:

$$\{(x_i, y_i, z_i) : i = 1, \dots, N\}$$

这集合描述的几何形状 M 满足: 对任意点云中的点 (x_i, y_i, z_i) , 都有 $(x_i, y_i, z_i) \in \partial M$, 即

$$S_M(x_i, y_i, z_i) = 0$$

点云作为原本几何形状的采样结果, 保留了原形状的一部分几何信息, , 使得我们可以在点云上进行一些表面性质的计算, 例如计算法向和曲率.

然而采样总是伴随着信息的损失, 点云也不例外. 点云的采样密度决定了它对几何细节的表示能力. 更重要的是, 由于点云本身的非结构化和无序性, 几何形状的拓扑关系往往是最容易在点云表示中变得模糊不清的. 这为基于点云的几何形状计算和处理带来了困难.

1.1.3 网格模型

为了解决点云对于拓扑形状表示的不足, 我们考虑对曲面表面进行线性近似, 即用一系列小的多边形片段拼接成曲面. 为此, 把点云中的点按照 M 的形状进行连接, 可以得到一个由多边形构成的网格.

定义 1.3 网格模型 网格模型 (Mesh Model) 是由一组顶点, 边和面构成的三元组:

$$\mathcal{M} = (\mathcal{V}, \mathcal{E}, \mathcal{F})$$

其中 $\mathcal{V} = \{\mathbf{v}_i = (x_i, y_i, z_i) : i = 1, \dots, N\}$ 是顶点集合, 每个顶点对应点云中的一个点; $\mathcal{E} = \{\mathbf{e}_{ij} = (v_i, v_j) : i, j = 1, \dots, N\}$ 是边集合, 每条边连接两个顶点; $\mathcal{F} = \{\mathbf{f}_k\}$ 是面集合, 每个面 \mathbf{f}_k 由数个顶点组成.

\mathcal{M} 描述的几何形状满足: 对于任意面 \mathbf{f}_k , 其上所有点都在 ∂M 上, 即

$$\forall (x_k, y_k, z_k) \in \mathbf{f}_k, \quad S_M(x_k, y_k, z_k) = 0$$

一般而言, 我们要求网格是流形的, 即每条边总是且仅被两个面共享.

1.2 网格表示

网格模型是计算机图形学中最常用的几何表示方法, 我们已经在上一节介绍过其定义. 在计算机中存储时, 通常使用顶点列表, 边列表和面列表作为储存多边形网格的数据结构; 有时也会设计额外的数据结构方便邻边查找等操作.

1.2.1 三角网格表示法

我们从最简单的三角网格开始, 即所有面都是三角形的网格. 最简单的表示方法就是记录每个三角形的三个顶点, 即

$$\triangle_i = (\mathbf{v}_{i0}, \mathbf{v}_{i1}, \mathbf{v}_{i2}), \quad \text{where } \mathbf{v}_{ij} = (x_{ij}, y_{ij}, z_{ij})$$

定义 1.4 三角形乱序集合 三角形乱序集合 (Triangle Soup) 是由一组三角形构成的集合, 每个三角形包括其顶点信息.

显然, 每个顶点几乎都会被多个三角形共用, 因此上面的表示方法在空间上由很大冗余. 并且由于存储的乱序性, 我们也不易对模型进行拓扑关系的考察.

为了减少空间开销, 我们可以先单独存储顶点, 然后只存储每个三角形顶点在顶点列表中的索引.

定义 1.5 索引三角形网格 索引三角形网格 (Indexed Triangle Set) 由顶点列表 \mathcal{V} 和三角形列表 \mathcal{F} 构成. 其中 $\mathcal{V} = \{\mathbf{v}_i = (x_i, y_i, z_i) : i = 1, \dots, N\}$ 是顶点列表; $\mathcal{F} = \{\triangle_k\}$ 是三角形列表, 每个三角形 \triangle_k 由三个顶点索引组成.

特别地, 为了方便处理, 我们在存储顶点索引时可以按照逆时针方向存储. 这可以保证每个三角形的法向方向一致, 从而方便后续的渲染等操作.

1.2.2 半边数据结构

在网格中, 我们经常会面对顶点邻接关系的查询, 也需要有序地遍历顶点和面. 在一般的索引三角形网格中, 我们只能通过遍历所有三角形来找到某个顶点的邻接顶点, 这显然效率很低. **半边数据结构 (Half-Edge Data Structure)** 就是一种更高效的网格表示方法.

我们考虑的几何形状几乎都是流形, 因此每条边总是被两个面共用. 我们可以把每条边拆成两个方向相反的**半边 (half-edge)**, 每个半边只属于一个面. 对于每个半边, 我们记录其起点, 终点, 所属面, 上一个和下一个半边, 以及与其配对的另一个半边. 这样, 我们就可以通过半边快速找到顶点的邻接顶点和邻接面.

定义 1.6 半边数据结构 半边数据结构 (**Half-Edge Data Structure**) 相比一般的网格数据结构增加了对每个边的半边表示. 每个半边 e 包含以下信息:

- $e.from$: 半边的起点;
- $e.twin$: 与该半边配对的另一个半边;
- $e.face$: 该半边所属的面;
- $e.next$: 该半边在所属面中的下一个半边;
- $e.prev$: 该半边在所属面中的上一个半边.
- $e.to$: 半边的终点, 等于 $e.twin.from$.

使用半边数据结构能完成对网格的许多拓扑操作

1.3 网格细分

不难看出, 通过增加组成几何表面的网格面片数量, 减小每个面片的面积, 可以使几何表面看起来更加光滑.

定义 1.7 网格细分 网格细分 (**Mesh Subdivision**), 又称作网格的上采样, 是指通过反复细分初始的多边形网格, 不断得到更精细的网格的过程.

1.3.1 Catmull-Clark 细分

Catmull-Clark 细分是最常用的几何表面细分方法之一, 主要应用于四边形网格的细分上. Catmull-Clark 细分的具体步骤如下:

1. **增设面点**: 对多面体的每个面片计算一个面点, 该面点是该面片所有顶点的平均值:

$$\mathbf{v}_{\text{face}} = \frac{1}{N} \sum_{n=1}^N \mathbf{v}_n$$

2. **增设边点:** 对多面体的每条边计算一个边点, 该边点是该边两个端点 v_1 和 v_2 , 以及相邻两个面片的面点 $v_{\text{face},1}$ 和 $v_{\text{face},2}$ 的平均值:

$$v_{\text{edge}} = \frac{1}{4}(v_1 + v_2 + v_{\text{face},1} + v_{\text{face},2})$$

3. **更新顶点:** 对多面体原有的每个顶点 v , 使用下面的加权平均算法更新其位置:

$$v_{\text{vertex}} = \frac{F + 2R + (N - 3)v}{N}$$

其中 F 是所有以 v 为顶点的面片的面点的均值, R 是所有以 v 为端点的边的中点 (注意不是 2. 中的边点) 的均值, N 是与该顶点相邻的顶点数.

4. **重新连接:** 将每个面点 v_{face} 与该面片所有边对应的的边点 v_{edge} 相连; 将每个新顶点 v_{vertex} 与原有顶点所有相邻边的边点 v_{edge} 相连. 于是就形成新的细分过后的面片.

1.3.2 Loop 细分

Loop 细分是另一种常用的几何表面细分方法, 主要应用于三角形网格的细分上. Loop 细分的具体步骤如下:

1. **增设新顶点:** 对于每一条边, 如果这条边被两个三角形面包含, 则根据这条边的两个端点 v_0, v_2 和这两个三角形除这条边外各自的顶点 v_1, v_3 加权平均得到新顶点 v^* :

$$v^* = \frac{3}{8}(v_0 + v_2) + \frac{1}{8}(v_1 + v_3)$$

如果这条边只被一个三角形面包含, 则取边的中点得到新顶点 v^* .

2. **更新原有顶点:**