Voxelization

1. **体素是什么**

Voxel，即体积元素volume Pixel，是数字数据在三维空间分割上的最小单位，类似2D空间像素，包含体素的立体可以通过立体渲染或者提取给定轮廓的多边形等值面表现出来。

本身不包含空间中位置的数据（即坐标，标量），但可以通过他们相对于其他体素的位置来推敲，也就是说他们在构成单一张体积影像的数据结构中的位置。

体素的边界在相邻晶格的中间位置，“体素”这个术语仅用来表示最邻近的插值，而不表示三次线性、立方等高次插值。

插值：对原图像的像素重新分布，从而改变像素数量（的一种方法）。在图像放大过程中，像素也相应地增加，增加的过程就是“插值”发生作用的过程，“插值”程序自动选择信息较好的像素作为增加、弥补空白像素的空间，而并非只使用临近的像素，所以在放大图像时，图像看上去会比较平滑、干净。不过需要说明的是插值并不能增加图像信息，尽管图像尺寸变大，但效果也相对要模糊些，过程可以理解为白酒掺水。（百度百科）

几乎应用于所有需要进行图像缩放功能的领域

方法：最近邻插值、双线性插值、高阶插值

图像插值：利用已知邻近像素点的灰度值（或者RGB图像中的三色值）来产生未知像素点的灰度值，以便让原始图像生成更高分辨率的图像

~~用于三维成像、科学数据、医学影响等领域~~

1. **体素成像**

通过体素成像，三维空间采样点的集合可以转换成计算机二维屏幕上有意义的图像。体素成像与传统图形显示方法的思想截然不同，它的最大特点就在于放弃了传统图形学中体由面构造这一约束，在不构造物体表面几何描述的情况下直接对体数据进行显示，采用体绘制光照模型直接从三维数据场中绘制出各种物理量的分带晴况，也就是说，直接研究光线穿过三维体数据场时的变化，得到最终的绘制结果。

1. **体素化是什么**

将物体的几何表示转换成最接近该物体的体素表示形式，产生体数据集，包含模型的表面信息，而且能描述模型的内部属性

体素化能对模型进行简化得到均匀的网络，在求模型的测地线，求交等过程有较好的应用

表示模型的空间体素跟表示图像的2D像素相似，只不过是从2D点扩展到3D的立方体单元

。。

1. **体素化操作**

假设模型体素化的分辨率为N\*N\*N，之后①对模型表面体素化②对模型内部体素化。

①首先计算出模型的AABB包围盒，然后根据空间分辨率对包围盒进行划分，得到每个大小为（X/N）\*（Y/N）\*（Z/N）空间像素列表，然后对构成3D模型的多边形或三角形列表进行遍历，得到这些基本体元所对应的包围盒，然后有AABB求交运算得到这些基本体元所能影响到的体素单元，将这些体素单元作为待判断的基本对象，为进一步精确判定，使用三角形与AABB的求交算法确定这些基本体元所能影响的最终体素，并将这些体素标记为非空

②模型表面体素化完后得到对模型体素表示的一个“外壳”，接下来要做的操作就是进行模型的内部体素化操作。

首先将对应的3D模型建立空间八叉树，这棵八叉树主要用于进行基本体元面片的求交操作。然后对模型AABB中的所有空体素，从其中心位置以轴对齐方向来发射两条射线，这两条射线的方向相反，但基本方向都是轴对齐的。对于这两条的射线利用空间模型的八叉树来得到其与3D模型的相交位置，并得到相交点的法向量及到相交点的距离，然后根据这两点法向量之间的关系来判断得到当前体素是在3D模型的内部或是在3D模型的外部。将这样的操作施加于每一个空的体素之后就可以完成对3D模型内部的体素化操作。

但是将这样的操作施加于每一个空的体素速度比较慢，故而此处可以采用扫描的方法来进行加速处理。如果判断得到某个体素的位置为模型内部后，就可以根据射线的方向及这两条射线与模型的交点处的距离来对当前体素相邻的体素进行扫描，这样不需要再做判断就可以标记出相邻体素的状态，这样就加速了整个模型内部的体素化操作。

包围盒算法，一种求解离散点集最优包围空间的方法，用体积稍大且特性简单的几何体（包围盒）来近似代替复杂的几何对象

3D空间中的AABB盒**Axis-aligned bounding box** ||3D空间包围球Bounding Sphere

1. max-min表示法：使用右上角和左下角的点唯一的定义一个包围体
2. center-radious表示法：center点表示中点，radious数组保存包围盒在x、y、z方向上的半径
3. mid-width表示法：min表示左下角的点，width保存x、y、z方向上的长度。

！不同方法的碰撞检测算法不同，适用于不同的情形

四叉树和八叉树:

即2D和3D的“二分法”，搜索过程与二叉树搜索类似，二叉树中是将数组sort后存入二叉树中，从而在查找中实现时间复杂度为log2N；

四叉树/八叉树是按平面/空间范围划分有序node，将所有points（坐标已知，但是每个点的point在vector中的index可以认为是随机的，没有规律的，所以不能直接根据index取出point(x,y)）放入所属node中，实现所有points的sort，进而在搜索时，实现时间复杂度为log4N/log8N---WellP.C

！树中任一节点的子节点恰好只会有八个，或零个，也就是子节点不会有0与8以外的数目。

八叉树可以用来碰撞检测、邻域检索、空间变化检测、压缩等功能。

优点，使用八叉树可以快速进行三维目标的集合运算，如交、并、补、差等，亦可快速进行最邻近区域或点的搜索。

缺点，存储空间消耗（空间换时间）