表面分割算法

[小组成员]

严枭	2013310584	计研 133 班
杨涛	2013310587	计研 133 班
俞承驰	2013310xxx	计研 133 班

[问题背景]

在科研工作中，特别是在几何处理等领域，三维模型会经常接触到，事实上，对三维模型的处理也是计算几何中的一个重要方向。对三维模型的处理，按照目的可以大致分为两类，其一是得到更加“符合要求”的模型，这包括对模型进行切割、对面片进行正规化操作等等；其二则是得到新的模型，其中的 Constructive Solid Geometry(CSG)就是利用多个三维模型进行布尔操作从而得到所需要的复杂模型。在实际的科研中，我们遇到了对模型进行切边操作的问题，为了解决这一问题，我们提出了自己的表面分割算法，并在实际中得到很好的应用，求得了满意的效果。

[问题提出]

三维模型的复杂是难以想象的，由于精力和时间的限制，我们缩小了研究的方向，将问题定位于通过一个封闭的模型（water tight）来对另一个模型进行分割。这里的分割本质上属于模型的布尔操作，但我们的目的并不在于获得更加复杂的组合模型，而是通过分割模型将被分割模型切割为独立的几个部分。

传统的面片分割可以通过 Marching Cubes 算法实现，但是如图 1 所示的那样，如此得到的表面存在比较严重的锯齿状边缘，我们希望能够实现较为平滑的切割。

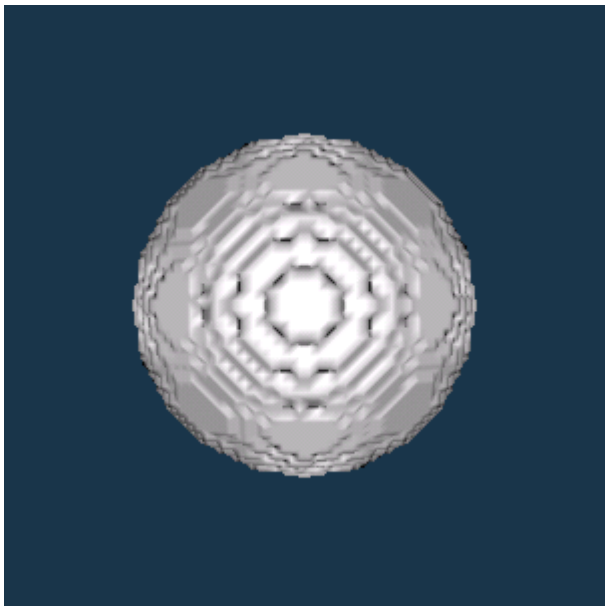

图 1

另外，我们强调了只针对封闭模型（water tight）进行处理，其中的原因在于，如果模型不封闭，可能会存在被分割模型无法被分割开的缘故，如图 2 所示，可以显见，红色的模型没有被完全的切割。

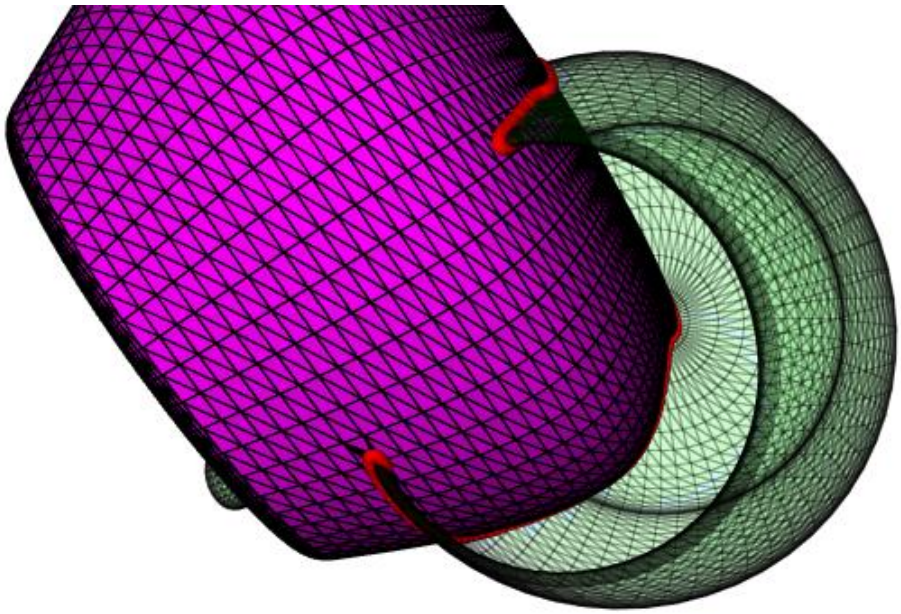

图 2

[算法概括]

我们的主要算法流程可以分为以下几步：

1、碰撞检测；
2、确定交点；
3、连接交线；
4、三角划分；
5、Flood Fill 分解模型。

其中碰撞检测顾名思义，主要为了检测两个输入模型（分割模型和被分割型）彼此之间相交的面片对，确定交点则是为了确定得到的面片对相交的位置，得到相交点的位置之后，可以把这些点连接成线，实际上这些连线也就实现了对被分割模型表面的分割，但是这样简单连线之后，会出现连线所在面片不再规整，为此，我们需要进行下一步的三角划分操作，以重新三角化面片，最后则采用简单的 Flood Fill 方向，确定各个分割开的部分。以上简单陈述了一下基本过程，下面将分别进行详细介绍。

[碰撞检测]

碰撞检测是计算机学科特别是图形学领域中的一个常用概念，针对三维模型

的面片，存在不少碰撞检测的方法。主要的算法包括蛮力的对所有面片两两检测、AABB、OBB 以及相对应的 AABB 树和 OBB 树，其中的蛮力算法虽然实现简单，但是显然在效率上不能令人满意，所以果断放弃。下面首先简单介绍一下所谓的 AABB、OBB 以及相应的树。

所谓的 AABB 实际上就是对模型中的每一个面片设置一个与 x,y,z 坐标都相互平行的包围盒，通过检测包围盒是否相交来确定相对应的面片是否相交。其包围盒的基本形式如图 3 所示。

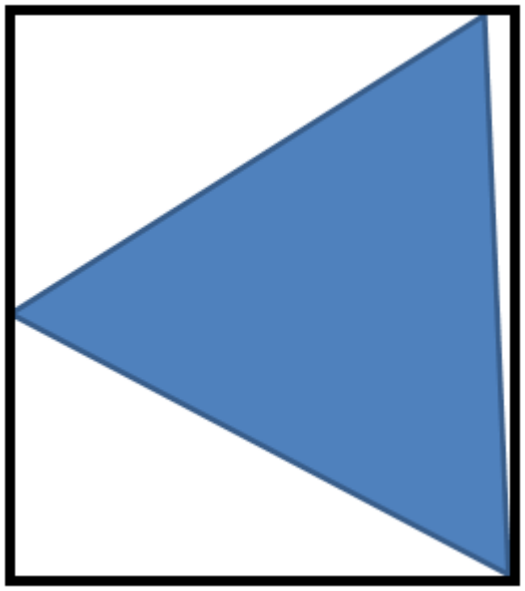

图 3

可以看出，如果单纯的采用 AABB，与蛮力算法相比并没有什么本质上的提升，但是其优势在于可以优化，为此，出现了 AABB 树的概念。所谓的 AABB 树就是将每个面片得到的 AABB 包围盒结合起来，从上到下构建出一棵树。其一般的结构形式如图 4 所示。

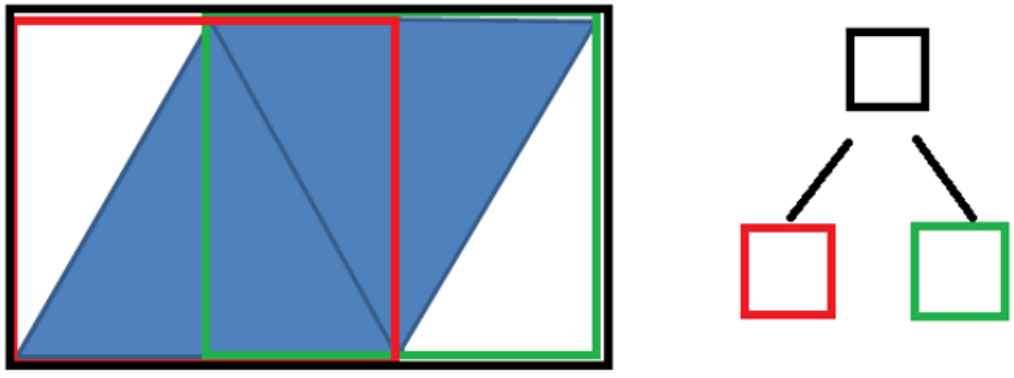

图 4

其中的红色包围盒和绿色包围盒分别是对应的三角面片的 AABB，而黑色的框则是包含红色框和绿色框在内的一个更大的包围盒，可以认为是一个父包围盒，

其树状结构如图 4 右边所示。如此这般，便可将整个模型的 AABB 包围盒构成一棵完整的树。两个模型分别对应于一棵独立的树，通过两棵 AABB 树相交的情况进行判断，非相交的 AABB 树节点就不用再继续递归的判断下去了，从而实现了效率的大提升。

与 AABB 模型中的“正规”包围盒不同，OBB 模型提出了一种有方向的包围盒，其中的“O”就是“oriented”的缩写。OBB 的核心思路在于寻找沿着面片方向的的包围盒，从而使得包围盒最小，同时也有效的减小了包围盒之间碰撞检测的次数。其一般的模型形式如图 5 所示。

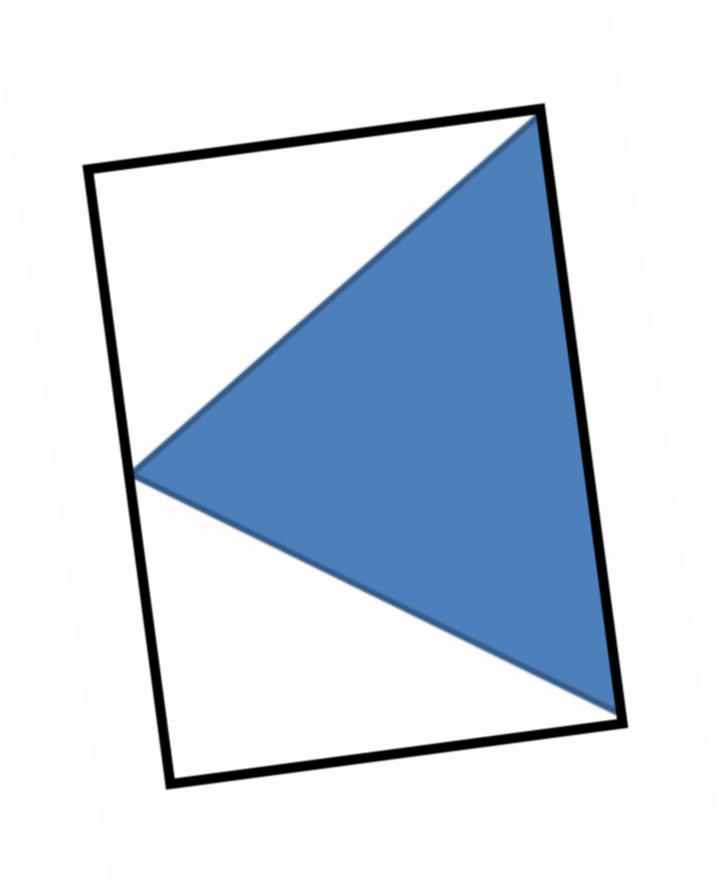

图 5

得到 OBB 包围盒的一般步骤可以概括为：

1、求凸包；
2、对凸包进行均匀采样；
3、利用采样点求出相应的协方差矩阵和中心，从而得到 OBB 包围盒。

由于 OBB 的实现较为复杂以及需要较为繁琐的证明，在这里不详细介绍，更多的详细内容可以参见“OBB-Tree: A Hierarchical Structure for Rapid Interference Detection”这篇文章。可以看出采用 OBB 包围盒可以极大的减少包围盒碰撞检测的次数，避免很多利用 AABB 包围盒需要检测但实际上却并不相交的情况。

与 AABB 模型相同，OBB 模型也可生成相应的树，从而得到 OBB 树，利用 OBB 树同样实现了碰撞检测效率的大提升。

为了评估以上提到的五种碰撞检测的效率，我们分别对五种方法进行了实现并测速，最终的比较效果如表 1 所示。

检测方法	8800 个面片的圆柱 + 8000 个面片的管型
两两判断	0.736s
AABB	0.148s
OBB	0.209s
AABB-Tree	53ms
OBB-Tree(rapid library)	2ms

表 1

通过对比可以看出，采用 OBB 树的方法可以极大的提升碰撞检测的效率，因此在最终的程序实现中，我们采用了 OBB 树的方法进行面片的碰撞检测。

[确定交点]

通过前面的碰撞检测，我们已经得到了两个模型中彼此相交的面片对，为了精准、平滑的切割面片，我们需要求得所有面片对相交的位置信息，这一步可以直接通过几何计算得出，具体的计算以及实际操作中的情况可参见图 6。

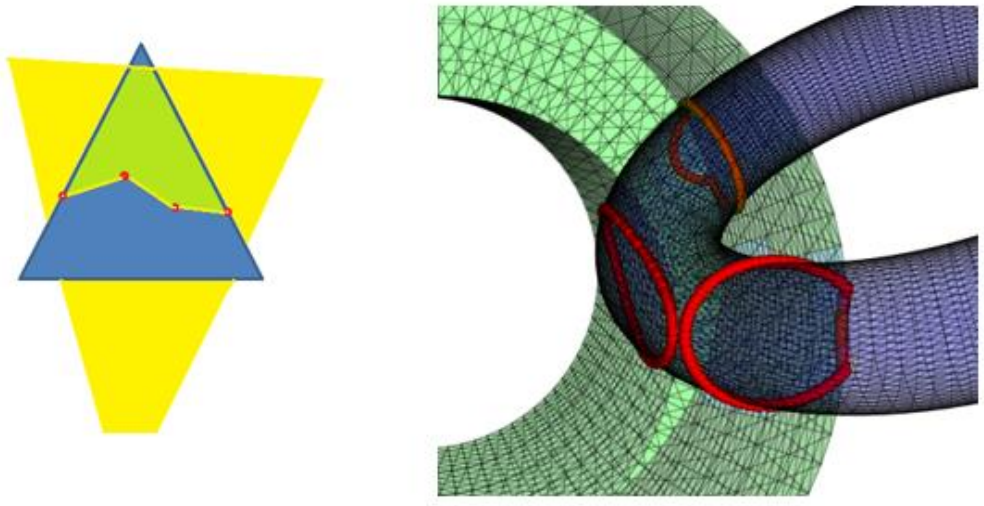

图 6

[连接交线]

确定了相交面片对相交的点之后，下一步就是要通过将这些点连接成线，从而实现将模型的各个部分进行切割。连接交线的主要算法过程如下：

1、得到每个交点对应的两个面片；
2、建立交点与面片的正索引和反索引；
3、将交点对应的面片排序；
4、得到被分割模型面片上的一系列交点，以及这些交点对应的分割模型的一系列面片；
5、在每个被分割模型的面片上建立一系列连线。

具体的实现情况可参见图 7。需要特别注意的是在第一步中，位于面片边缘的交点可能对应于多个面片，这时候需要考虑对交点进行复制操作。

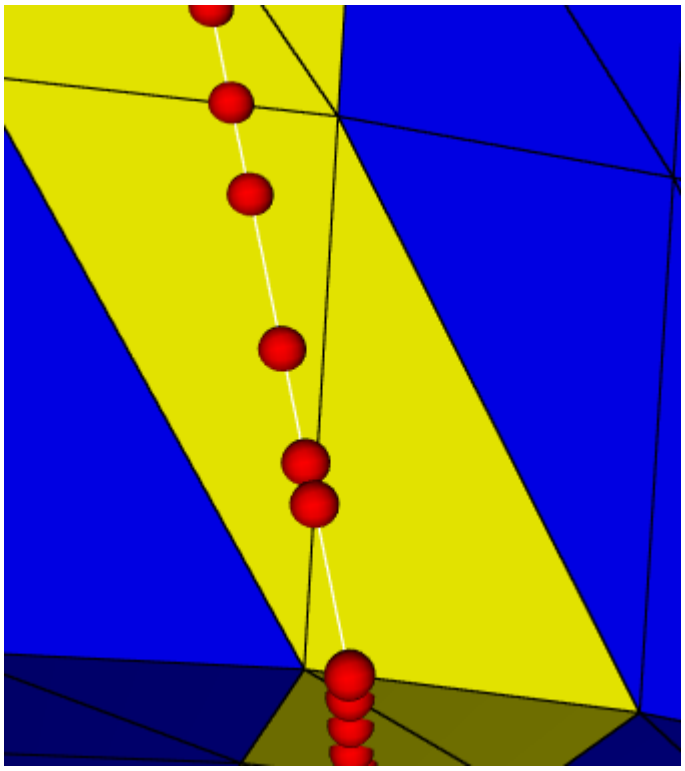

图 7

[三角划分]

由于分割模型是简单的封闭多面体，所以三角形上的边不存在交错，可以先做一次三角划分，然后再对之前得到的连边对所有分解出来的三角形进行再细分。以上是我们最初的设想和实现，但是在后来的查阅中，我们发现了带约束的 Delaunay 三角划分方法，采用带约束的 Delaunay 三角划分可以得到更加优质和鲁棒的三角面片，所以在最终的实现中，我们改成了带约束的三角划分方法。

带约束的 Delaunay 三角划分一般过程如下：

1、先忽略约束，采用 Delaunay 三角划分方法进行三角划分；
2、加入约束边，去除与约束边相交的边；

3、对约束边两边新形成的多边形进行三角划分。
具体的流程如图 8 所示。

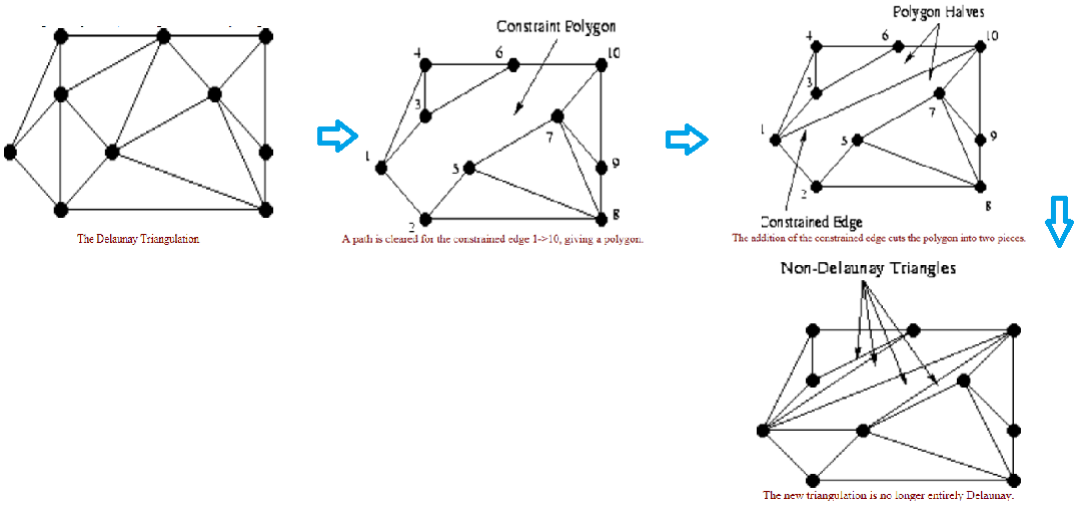

图 8

[Flood Fill 分解模型]

因为 Flood Fill 的通用性和广泛性，这里只简单介绍一下我们的实现流程。首先建立所有面片之间的连接关系，利用 Flood Fill 方法，任选一个没有分类的面片，考虑与其连接的所有面片，如果和这个面片并不是通过之前求得的连线而连接，则继续 Flood 过程，否则停止。如此一直重复下去，直至所有的面片都得到分类。

[时间复杂度分析]

对于碰撞检测，通过前面的分析可知，最槽糕的情况应该是 O(n^2)，但是如果采用 OBB 树优化，最终的复杂度只有 O(s)，这里的 s 表示相交的面片数。由此，可继续分析得到，确定交点的时间为 O(s)，连接交线的时间为 O(nlogn)+O(slogs)，三角划分的时间为 O(slogs)，考虑到约束边个数的不确定，复杂度可能更大，最后的 Flood Fill 分解模型算法则需要 O((n+s)log(n+s))。

实际运行的结果也基本验证了上面的复杂度分析，具体的如表 2 所示。

步骤	时间(8800 个面片的圆柱 + 8000 个面片的管型)
OBB 检测	2ms
确定交点	1ms
连接交线	96ms

三角划分	663ms(瓶颈)
Flood Fill	375ms

表 2

可以看出本算法的瓶颈是在三角划分部分。但是在实际的测试中，我们发现我们的算法同样可以处理规模较大的三维模型的分割，这说明了我们的算法具有良好的壳扩张性。

[讨论与总结]

虽然在最终的实现中，我们的算法获得令人满意的结果，但是还是存在很多的限制，比如分割模型必须是 water tight 的，以及没有考虑流形间的表面分割等等，所有这些都需要在后续的工作中进一步的改进和完善。

最后在一学期的学习，我们都收获了很多，感谢老师的辛苦付出。

[参考文献]

[1] OBB-Tree: A Hierarchical Structure for Rapid Interference Detection. S.Gottschalk, M.C. Lin, D.Manocha.
[2] http://www.geom.uiuc.edu/~samuelp/del_project.html
[3] TRICUT: a program to clip triangle meshes using the rapid and triangle libraries and the visualization toolkit (Projection Method).
[4] Multiple-Fluid SPH Simulation Using a Mixture Model (our paper).
[5] Constrained Delaunay Triangulations.
[6] visualization toolkit. http://www.vtk.org/.org/