算法的简单说明

1输入网格

最好是无边界、流形、封闭的网格，且不存在特别小的以至于无法在容忍度下计算法向的面。

2网格求交线

预处理：

首先判断两个网格是否有重叠的面，如果存在重叠的面，我通过将mesh0的重叠面记录下来，然后沿着面法向平移偏移0.001长度的方法来避免掉重叠面的麻烦。注意，此步操作是会改变原始网格数据的。

三角形求交：

三角形求交有许多种情况，主要分为不相交、交于一点和一个线段的情况。

如果两个三角形只交于一点的话，是不会将此点记录下来的，即认为不相交就好了。

两个三角形交于一条线段的话，又主要分为两种情形。一种是一个三角形的边在另一三角形平面上且与其相交，另一种就是最普通的相交。第一种相交的情况下相当于是交于某个三角形边界的情况，需要判断此边界是否是切线，如果是的话，不能把该交线段算作相交环的一部分，即认为相切就相当于不相交。

求交这一步里还有一个重要问题的处理就是判断得到的Vertex是否已经存在。为准确的解决这一问题，在Vertex的数据结构中，添加了关于两个网格的是否处于顶点，是否处于半边，是否处于三角形内部，以及处于哪个顶点、半边和三角形内部的信息。有了这些信息，判断两个vertex相同的话，除了要求空间距离小于容忍度，还要求这些信息的匹配。这样的判断就会比较准确。

3网格重新三角化

在得到了正确的求交结果之后，只需要通过segment来获得相关的面，给每个面建立一个RemeshTriangle，然后将所有的交点和新产生的线段的信息存入进去，然后调用triangle来重新三角化就好了。

4判定内外和拓扑重构

判定内外和拓扑重构需要在remesh后的网格上进行。

内外判断：

通过交线段处两个网格的四个面的法向来判断内外。线段有一个方向，相当于一个半边，此半边在两个网格中对应两个面，线段的方向和这两个面的方向可以判断出这两个面的内外关系。得到了一个内外关系之后，再通过扩散的方法确定一个区域的内外。因为被交线围起来的区域的内外性质是一致的。

拓扑重构：

有了内外信息之后，拓扑重构就简单了。

以bool\_union为例：只需将两个网格都标记为外部的面拼起来就完成了。

bool\_intersect：只需将两个网格都标记为内部的面拼起来。

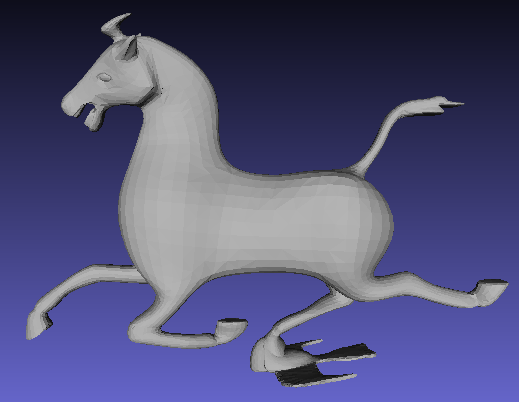
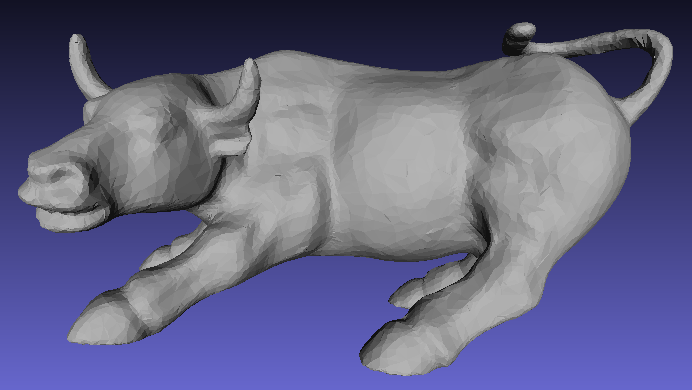
bool\_substract：将一个网格的外面和另一个网格的里面拼接起来，注意拼接标记为内部的面时，需要倒转一下面的法向量。

5运行环境

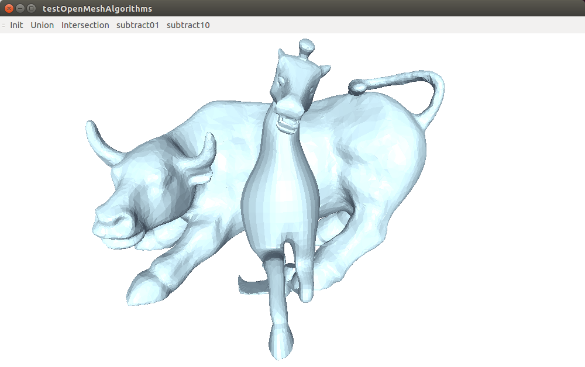
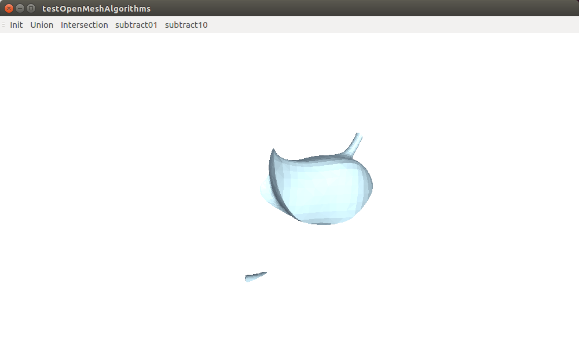
Lib里的依赖库是Ubuntu14.04和gcc4.9.4编译的，其它eigen和openmesh需要自己配

6一个结果

Mesh0 mesh1



Intersection union



Subtract10 subtract01

