1. **查找一组点的边界边**

\*辅助函数seleced\_flags，将整个mesh被选中顶点以bool值标记

1. 确定开始边：获取其中一个顶点指向的半边，设置为start\_he，将当前半边cur\_he赋值为start\_he；利用seleced\_flags判断start\_he的终点和start\_he的下一条半边的终点是否是被选择的顶点，如果前者为真后者为假，则将当前边设置为cur\_he，并加入到选中边的集合selecedEdge
2. 绕点循环：将当前边设置为当前边的下一条半边的反向半边；

如果当前边的下一条边的起点或终点有一个不是被选择的顶点则继续循环 2

1. 如果当前边的下一条边的起点和终点都是被选中的顶点，则将当前边加入边集，并将当前边设置为当前边的下一条边
2. 如果当前边与起始边不是同一条则循环2-3
3. **查找一条边的邻域边**

第一种方法：

1. 起始边设为选定边的第1条半边，当前边设置为起始边
2. 如果当前边的下一条边反向边不是起始边，则循环3-4
3. 将当前半边设为当前半边的下一条的反向半边
4. 如果当前半边的下一条的反向半边不是起始半边，则将当前半边的前一条半边加入边集
5. 起始边设为选定边的第2条半边，当前边设置为起始边（或将起始边设为当前边的下一条）
6. 从2开始

第二种：1.找到选中边的两端点的所有领域面

1. 每一个面的边加入边集
2. 按id排序边集
3. 迭代边集，剔除所有出现次数超过1的边，留下的边即为邻域边
4. **查找一组顶点的邻域边**
5. 循环遍历每一个选中的顶点，获得每个顶点的一层邻域边和放射边并加入一整个集合E中
6. 遍历放射边集合和E集合，遇到相等的边则在E集合中删除整个边，E集合剩下的边就是结果边
7. **查找一组面的边界边**

算法思想：获取所有面的边界边加入一个集合中，重复出现的边则删除，余下的边就是边界边

1.循环遍历每一个选中的面，获得每个的边界边的索引，加入边集E中

2.对边集E进行排序

3.用c记录E中第一个元素

4.从第二个元素开始遍历E，如果当前的元素于c相等，则将前一个元素和当前元素的值置为-1，否则将当前元素赋值给c

5.再次循环遍历E，如果元素值为-1则删除

**五．查找边界**

辅助变量，设置半边flag记录半边是否被访问过；

Hole数组，存放一个组的边界半边

all\_boundary数组，存放所有hole数组

1.遍历所有半边

2.将i设为起始边，起始边设为当前边

3.如果当前半边是边界并没有被访问过，则设置为访问过，加入hole中

4.如果当前半边的下一条不是边界，则循环5

5.将当前半边设置为当前半边的下一条的反向半边

6.如果当前半边的下一条未被访问过，则当前半边设为下一条，加入hole,设为访问过，否则结束循环

7.如果当前半边不是起始半边则循环4-6

8.如果当前半边是起始半边（不是因为break中途停止循环），则将将hole加给all\_boundary