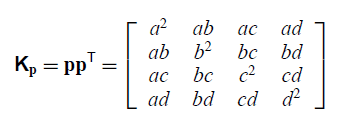
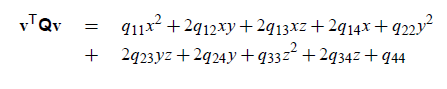
1. 算法流程
2. 给每个初始点计算Q矩阵。对于某个点所在的一个三角面片，设三角面片所在的平面方程为ax+by+cz+d=0(a2+b2+c2=1),令p=(a,b,c,d)T,那么空间中一个点v(vx,vy,vz,1)T到这个平面的距离的平方为(pTv)2 ，将它转化为二次型形式(pTv)2 =（vTp）(pTv) = vT(ppT)v，（p,v都是列向量，pTv= vTp），令Kp=(ppT):



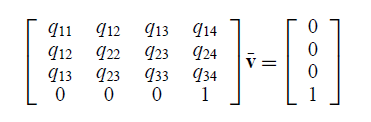
某个顶点所在的所有面片的K累加，则得到Q。K是对称矩阵，Q也是对称矩阵。

为了保留边界，对于在mesh边界上的边，这条边所在的三角面片有且只有一个，假设这个三角面片为t，构造一个与t垂直且穿过这条边的平面，将这个平面的K矩阵乘以一个很大的权重，加到这条边两个端点的Q矩阵中。这样做的意义在于，使得合并之后的点与这个面的距离很小，从而达到保留边界的目的。

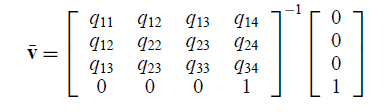
1. 选择有效点对（pairs）。如果v1,v2之间有一条边，或者v1,v2的距离小于某个阈值，那么（v1,v2）就是一个有效点对。
2. 为每一个有效点对计算最优的目标位置。V1对应的Q为Q1，V2对应的Q为Q2，合并之后的点v的Q=Q1+Q2,那么把二次型vT(Q1+Q2)v作为这个有效点对合并的代价（cost），



为了求得cost的最小值，对x,y,z分别求偏导数，令偏导数等于0，得到：



假设左侧矩阵可逆，则：



如果不可逆，从这条边的中点和两个端点中选择一个最优位置（论文里面提到先在线段v1v2上找最优位置，没看懂）。

为了防止合并前后某些三角面片方向变化过大，对v1v2所在的所有三角面片，计算合并前后的法向量，如果法向量夹角大于90度，则禁止该合并。

1. 将所有有效点对按照cost大小放入优先队列，cost小的在前。
2. 循环取出优先队列中的点对，进行合并，将v1的坐标改为最优位置，删掉v1v2共有的三角面片，将v2所在的三角面片赋给v1（如果重复的话直接删掉），将v1v2这条边删掉，将v2的边赋给v1（重复的话删掉）。然后更新优先队列，v1所在的边要更新，v1邻接顶点所在的边也要更新（主要针对更新前后某个面的法向量夹角过大的情况，可能之前法向量夹角过大，但是现在不大，所以cost会改变）。
3. 注意事项
4. 基本思想：合并顶点对（contraction of vertex pairs），两个顶点之间不一定有边相连，好处是可以连接不连通区域，支持非流形模型。合并顶点对是边收缩（edge contraction）的扩展形式，边收缩要求两个顶点之间有边相连。
5. 算法针对三角网格，如果是多边形，可以三角化为三角形。
6. 在点合并时，合并之后的点的Q直接赋为两个点的Q之和，那么Q矩阵里面包含的三角面片会有重复，在简化过程中，每个三角面片最多重复三次。如果是非流形模型，可以记录每条边所在的两个三角面片，两个点合并时，两个点所在的所有三角面片的交集就是这条边所在的两个三角面片，减去这个交集就可以解决重复问题（容斥原理）。
7. 优点
8. 速度快。两个顶点合并为一个顶点之后，两个顶点的Q矩阵直接相加赋给新顶点的Q矩阵（可以直接相加的一个潜在原因是cost是新顶点到相应三角面片距离的平方之和，如果是距离，则这个距离有正有负，不能直接相加）。
9. 精度较高，可以保留原始特征。
10. 如果模型有多个不连通区域，算法可以连接不连通区域，论文里面把它称为聚合（aggregation）。支持非流型模型。
11. 一个潜在的优点：如果存在重复顶点，这两个顶点可以很容易被合并；而对于边收缩的算法，两个顶点可能不会很容易被合并，因为两个顶点之间可能没有边。
12. 缺点
13. 不能保持模型的拓扑结构。
14. 产生非流形区域（两个点之间没有边的时候很容易产生非流形顶点）。