一种简单快速的 Delaunay三角网逐块生成算法

刘永和[①]，王燕平[②]，齐永安[①]
(①河南理工大学资环学院，河南焦作 454000；②河南理工大学图书馆，河南焦作 454000)

【摘 要】分块式生成 Delaunay三角网是加快构网速度的一个基本思路。已有的分治算法和其他分块合并算法能使平均时间复杂度接近线性，但算法复杂，编程难度大，且容易产生计算误差导致的错误。本文作者曾提出过一种基于三角网扩张法的逐块归并算法，它也是一种快速算法，但在算法中需要增加避免错误的判断规则，使程序变得较复杂。本文中的逐块生成法是对逐块归并法的改进，它继承了逐块归并法高效的优势，而且减少了判断规则，步骤更加简单。

【关键词】Delaunay三角网；分块合并算法；LOP优化；不规则三角网；时间复杂度
【中图分类号】P224 【文献标识码】A 【文章编号】1009-2307(2008)06-0133-03
DOI: 10.3771/j.issn.1009-2307.2008.06.046

1 引言

平面 Delaunay三角网是计算几何学中的重要研究内容，它是构造不规则三角网(TIN)数字高程模型(DEM)和生成平面 Voronoi图的基础。不规则三角网 DEM精确表达的优点使它在三维地形可视化、等值线生成等方面的应用日趋广泛。而基于点生成元的平面 Voronoi图在空间决策中也起着重要作用。目前已有很多算法可用于由平面离散点生成 Delaunay三角网，具有代表性的基础算法是三角网扩张法[1-3]、逐点插入法[4]，它们的平均时间复杂度为 $O(n^2)$，其中后者较前者快一些。当数据量增多时，这二种算法的耗时呈指数次增加；Shamos和 Hoey提出了分治算法[5-8]，他们采用递归分块的算法使复杂度接近 $O(n)$，但其缺点是递归过程需要大量内存来保存中间数据，而且在子块合并时还要将相邻子块的两侧凸包边界采用自下而上的连接算法，需要大量复杂的判断，使浮点数误差错误的发生机率增大。国内不少学者提出了各种分块合并算法，解决了分治算法在空间复杂度方面的问题，但点集分块方法及合并算法都比较复杂[9-13]。笔者曾提出一种基于三角网扩张法的逐块归并算法，该算法实际上是对三角网扩张法的一种简单修改，比较易于理解且复杂度也接近线性，但该算法必须添加一个用于判断的补充算法来避免错误，这个补充算法需要考虑的图形拓扑关系较多，较为复杂。于是，笔者修改了逐块归并算法，仍采用逐块生成的思想，但不再使用合并算法，时间复杂度未变，但算法变得更简单了。

2 Delaunay三角网的数据结构

Delaunay三角网一般需要顶点、边、三角形三种数据类型，使用将 C 语言表达如下：

```
//顶点类
classDPoint
{
    public float x;     //顶点的横坐标
    public float y;     //顶点的纵坐标
}
//有向边类
classDEdge
{
    public int beginpoint;  //边的起点索引
    public int endpoint;    //边的终点索引
    public int Ltri;        //边的左邻三角形索引
    public int Rtri;        //边的右邻三角形索引
}
classDTriangle
{
    public int[] points= new int[3]; //指向三个顶点的索引数组
    public int[] edges= new int[3];  //指向三条边的索引数组
}
```

为了保存所有离散点、有向边和三角形信息，还要使用下面的 C 数组类容器数据结构来存放由上面的类生成的对象：

① List<DPoint> points;
② List<DEdge> edgelist;
③ List<DTriangle> trianglelist;

所有其他数据类型和数据结构中用到的有关顶点、边、三角形的信息均为上面这 3个数组的索引。在三角网生成完毕后可以直接将这 3个数组串行化为磁盘文件。这三个数组也可换成字典式结构。

在三角网扩张法中，还需要下面的辅助容器数据结构，来记录临时边信息的数组，它们同时起到栈和集合的功能。

List<int> edgestack;

3 三角网扩张法

三角网扩张法是一种构造 Delaunay三角网的基础算法，即可以用来在其他分块算法中构造三角网子集。为便于三角形的扩张和拓扑关系的自动构建，需要在生成三角形时遵循下面的约定：三角形的 points和 edges的索引顺序应满足在二维空间按逆时针方向排列的要求，并要有对应关系(如图 1所示)：edges[0]的起始点与 points[0]为同一顶点，同理，edges[1]、edges[2]的起始点分别与 points

作者简介：刘永和(1976-)，男(满族)，在读博士，讲师，河南理工大学资源环境学院地信系，主要从事地理信息系统教学和地层三维可视化研究。
Email: sucksis@163.com

收稿日期：2007-06-13
基金项目：国家自然科学基金项目(40572012)

[1]、points[2]为相同顶点；三角形的边有向边，各边以逆时针方向围绕三角形。

从以上约定可知，在生成一个三角形 T时，即可按照其顶点的顺序生成 T的 3条有向边，这些边的左邻三角形即为三角形 T，右邻三角形为与 T邻接的三角形。可见，三角网向外扩张的过程就是为三角形各边寻找右邻最优点、构建右邻三角形的过程。

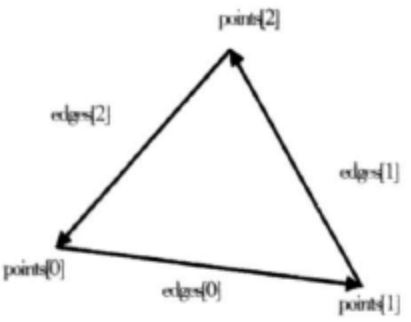


图 1 DTriangle类中 points和 edges数组的排列

三角网扩张法的具体步骤为：

1）构造初始边，在离散点中任选一个顶点作为初始边的第一个顶点 P_0，找出距离 P_0最近的点 P_1，连接 P_0、P_1为初始边 e_0，将 e_0加入 edgestack等待被扩展。

2）扩展三角形，若 edgestack不空时，重复做以下各步：①从 edgestack中弹出一条边 e；②从 points数组中选择 e右侧的 1个最优点，构造新三角形 tri；同时对 tri的各边要作以下处理：判断 edgestack中是否有与新边方向相反的相同边，若有则将找到的边从 edgestack中删去记录，否无则将新边加入 edgestack以等待扩展。

在以某边扩展三角形时，要选择该边右侧的一个最优点，此点与该边的原有 2顶点构成的三角形 tri应满足三角网的空圆特性，归结为求与待扩展边的两端点连线张角最大的顶点，可以使用余弦定理 $\cos C=\frac{a^2+b^2-c^2}{2ab}$，满足 cosC值最小的点即为最优点。

4 逐块归并算法介绍

逐块归并算法是基于分块的思想，分块的方法可以采用按横向切割分块或按纵向切割分块。算法的主要步骤为：①用纵向切割或横向切割将点集分成若干个子块，要保证各子块中的点数大于 3；②将所有子块分别用三角网扩张法构建 Delaunay三角网；找出各子三角网的边界；③从边界边出发，采用三角网扩张法依次将相互邻接的三角网子集合并；④用 Lawson提出的 LOP[4]优化方法优化与各子集凸包边相邻的两侧三角形。

该算法中三角网合并的基本思想是以相邻两个子网的边界边作为待扩展的边，向这些边界顶点（边界边的端点）扩展。如图 3 由 $\overrightarrow{P_1P_2}$扩展到的最优点为 P_4，由 $\overrightarrow{P_4P_3}$扩展到的最优点为 P_1。

搜索边界边的方法归结为寻找右邻三角形为空的边。对相邻的两个三角网合并的方法是：首先要将两个子网的所有边界边存入空 edgestack；然后使用与构建子网相同的三角网扩张算法来完成边界上的三角形构网。

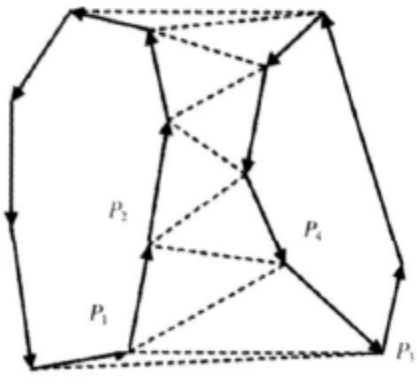


图 2 相邻两个子三角网的合并

逐块归并算法在寻找最优的顶点时只从所有边界顶点集中查找，减少了寻找最优点时的遍历时间，使得在数据量较大分块较多的情况下时间复杂度接近 O(n)。在点数较少的情况下，这种完全基于三角形扩张法的合并算法能正确完成子网的合并，但当子网中点很多时会出现类似于图 4中的特殊情形，如对边 e扩展时找到的最优点 P使新生三角形跨进了一个子网的内部，P为非法点。因此，在合并三角网的算法中，遍历边界点数组以寻找最优点时还需要对遇到的点 P_w判断是否合法，合法的标准是 $\frac{1}{2}$、$\frac{1}{2}$（$\frac{1}{2}$、$\frac{1}{2}$分别是 P_w和待扩展边两端的连线）与所有边界边 e_b没有交点。这个附加判断虽然简单，但计算量较大，所依赖的条件易受浮点数误差影响而导致错误。

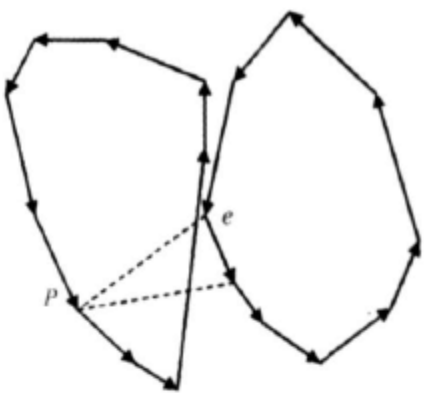
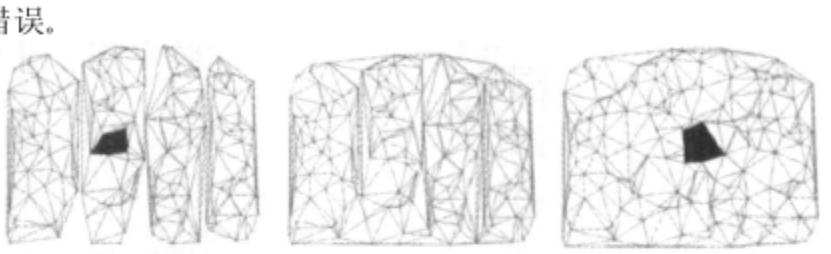


图 3 算法的特殊情形



a分块生成的子三角网 b合并后的子三角网 c LOP优化后的合并三角网

图 4 子三角网的合并与优化

5 思路与步骤

逐块归并算法需要先将所有子三角网建成，然后逐块归并，归并的方法是从相邻三角网一侧的凸包边出发向另一侧凸包边顶点扩张，即合并边界。而本文中的逐块生成法是对逐块归并法的改进，即仍采用水平或垂直分块，但与后者不同的是，本算法不采用一次生成所有子三角网的做法，而是逐次从当前子三角网的凸包边界向下一个点子集扩张，扩张时不再用到右侧的凸包边界。具体步骤如下：

①将点集以 x坐标排序；②将点集分成数块，每块中含有的点数相同或相近；③取第一个子块中最前面的三个点，将其连成一个逆时针顺序的三角形；将三角形的三边的记录加入集合 edgestack；④从 edgestack中取出一条边 e（同时要删去它在 edgestack中的记录），从 e出发向当前点子集中剩下的点中扩张最佳三角形；如此重复执行，直至 edgestack为空；⑤在第④步生成的边中查找所有凸包边界边（即右邻三角形为空的边），将它们加入 edgestack；⑥重复执行第④～⑤步，直至所有分块全部连成三角剖分；⑦应用 LOP算法优化三角网中的窄细三角形。

6 复杂度分析与耗时测试

三角网扩张法在最坏情况下的时间复杂度为 $O(n^2)$，随着离散点个数的增加，算法耗时呈指数次增加。基于三角网扩张法的逐块生成法与逐块归并法类似，都是将原有离散点按横坐标或按纵坐标分块，使在寻找最优点时的搜索范围减小，搜索时间随点数的指数次增长，当数据量很大时平均时间复杂度接近 O(n)；最后的 LOP优化只是对所有三角形判断或优化数遍，复杂度为 O(m)，这里的 m为三角形数。因为 m与 n接近或成正比，所以总的平均时间复杂度为 O(n)。

由该算法中的分块方案可知：① 若每块的离散点个数一定，则构网耗时应该与总块数成正比关系；② 对于离散点总数固定的情况，一般地，分块数越多，构网耗时越少，但分块数超过某一最优限值时，构网耗时趋于稳定，因为这时需要优化的长条三角形数增多，优化耗时增加。分块数的最优限值随离散点分布的情况不同而不同，设分块的宽度为单位距离，则分块内一个单位距离的高度内的离散点数要大于 10。

笔者在 Microsoft net Framework平台上用 C#编写了该算

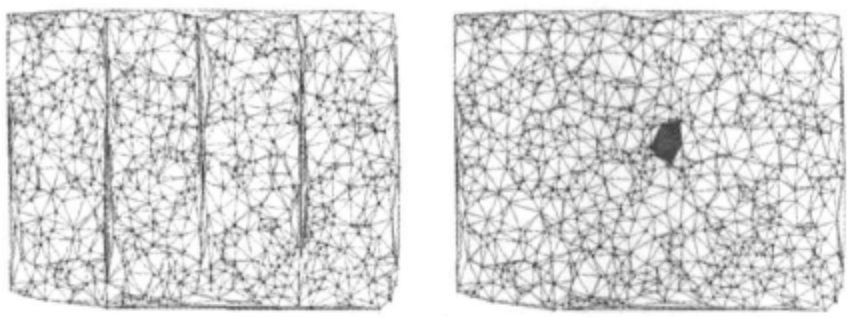


图 5　逐块生成的三角剖分和优化后的 Delaunay三角网

法程序，运行结果正确，且能够保证三角形之间具有正确的邻接关系（如图 5　左边为未优化时的三角剖分，右边的三角网是优化后的效果，其中三角网中的黑色区域是被选中的一个三角形和其三个邻接三角形，这表明它们的邻接关系是正确的）。笔者作了如下测试：①规定每块有 100个离散点，分别统计用 1～10块（即 100～1000个）离散点生成 Delaunay三角网的多次测试的平均耗时（见图 6），由图 6可见，耗时曲线基本接近线性，与上面分析得到的时间复杂度 O(n)相吻合；②用 5000个点分 1～15块分别生成三角网，统计各种分块方案的多次测试平均耗时，如图 7 可见随着分块数的增加，耗时显著减少；③对随机分布在 1000×700的电脑屏幕范围内的 1万个点，分 10块时平均构网耗时 7113ms，分 20块时耗时 3900ms，分 40块时耗时 2430ms，分 100块时耗时 1600ms，分 200块时耗时 2400ms 左右；④分别由逐块归并法和逐块生成法对 1万个点分 10块的情况生成三角网，前者耗时 7232ms，后者耗时 7113ms，二者较为接近。

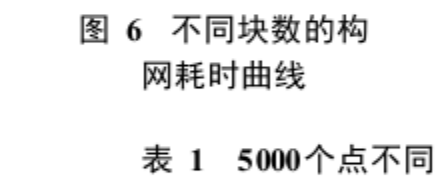


图 6　不同块数的构网耗时曲线

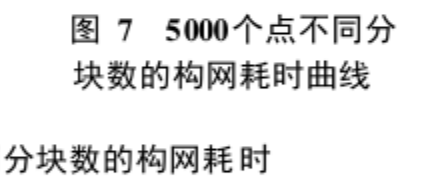


图 7　5000个点不同分块数的构网耗时曲线

表 1　5000个点不同分块数的构网耗时

分块数	1	2	3	4	5	6	7	8
耗时（ms）	20320	9167	5986	4372	3580	2977	2614	2336
分块数	9	10	11	12	13	14	15	
耗时（ms）	2121	1922	1805	1697	1592	1503	1433	

7　结束语

逐块生成算法与逐块归并法都具有 O(n)的时间复杂度，本质上都是对三角网扩张法简单修改，不过前者不再用到后者的补充判断条件，节省了运行时间，但同时使用了 LOP优化法又要对所有三角形遍历 1趟，又增加了处理时间，二者相互抵消，总体效率上没有显著变化。为了节省优化时间，总的来看，逐块生成算法继承了逐块归并算法优点，但算法步骤更为简单。一般情况下，对于由一定个数的离散点生成 Delaunay三角网，分块数越多耗时越少，但分块数过多时耗时曲线趋于稳定。

参考文献

[1] Green P J, Sibson R. Computing Dirichlet Tessellations in the Plane [J]. The Computer Journal, 1978, 21(2): 168-173.

[2] 王家耀. 空间信息系统原理 [M]. 北京：科学出版社，2001.

[3] 吴立新，史文中. 地理信息系统原理与算法 [M]. 科学出版社，2001.

[4] Lawson. Software for C¹ Surface Interpolation [C] // In Mathematical Software III (J R Rice, Ed.), Academic Press, New York, 1977: 161-194.

[5] M I Shamos, D Hoey. Closest-point Problem [C] // Proceedings of 16th IEEE Symposium on Foundations of Computer Science, Berkeley, California, 1975: 151-162.

[6] Lee D T, Schachter B J. Two Algorithms for Constructing a Delaunay Triangulation [J]. International Journal of Computer and Information Science, 1980, 9(3): 219-242.

[7] Rex A Dwyer. A fast Divide-and-Conquer Algorithm for Constructing Delaunay Triangulations [J]. Algorithmica, 1987, (2): 137-151.

[8] 蒋红斐. 基于分治算法构建 Delaunay三角网的研究 [J]. 计算机工程与应用，2003, 16: 81-83.

[9] 胡金星，潘懋，马照亭. 高效构建 Delaunay三角网数字地形模型算法研究 [J]. 北京大学学报（自然科学版），2003, 39(5): 736-741.

[10] 蒲浩，宋占峰，詹振炎. 快速构建 Delaunay三角网算法研究 [J]. 铁道学报，2001, 23(5): 85-91.

[11] 徐青，常歌，杨力. 基于自适应分块的 TIN的三角网建立算法 [J]. 中国图像图形学报，2000, 5(6).

[12] 武晓波，王世新，肖春生. 一种生成 Delaunay三角网的合成算法 [J]. 遥感学报，2000, 4(1): 32-35.

[13] 刘少华，等. Delaunay三角网中点目标快速定位算法研究 [J]. 测绘科学，2007, 32(2).

[14] 郭兆胜，张登荣. 一种改进的高效 Delaunay三角网的生成算法 [J]. 遥感信息，2005, 1: 15-17.

A simple and quick block-by-block generating method for generating delaunay triangulation from points in the plane

Abstract: Dividing the points into blocks and generating Delaunay triangulation from each block is the cardinal idea for fast creating large Delaunay triangulation. Divide-and-Conquer algorithm and other divide-and-merge method at present have the time complexity of linearity but their steps are more complex and difficult to program, and it also raises the probability of occurring bugs from float point errors. The author of this article proposed a sequential merging algorithm based on triangle-expanding method and it's average time complexity is close to O(n), but in order to avoid float point bugs, a judging rule must be added and thus the steps is some more complicated. The block-by-block generating method in this article is an improved modification of sequential merging method and inherits the high efficiency but more simple.

Key words: delaunay triangulation; divide-and-merging method; LOP optimizing; triangulated irregular networks; time complexity

LIU Yong-hua①, WANG Yan-ping②, QI Yong-an① (① School of resources & environment science, Henan Polytechnic University, Jiaozuo 454000, China; ② Library, Henan Polytechnic University, Jianzuo 454000, China)