2 维及 3 维 Voronoi 图的 3 维解释

汤慧赟 荆明轩 郭碧川

1. 问题背景

Voronoi 图，又叫泰森多边形或 Dirichlet 图，它是由一组由连接两邻点直线的垂直平分线组成的连续多边形组成。N 个在平面上有区别的点，按照最邻近原则划分平面，每个点与它的最近邻区域相关联，如图 1 所示，每个区域被叫做一个 cell，每个 cell 的核心是 site。当确定了 site 的位置的时候，cell 的划分就能给出来了。

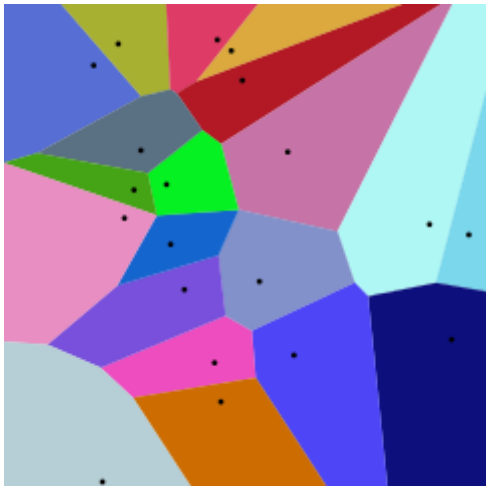


图 1. 2 维 Voronoi 图[1]

Voronoi 图在现实中有很多观察和应用，笛卡尔将宇宙爆炸的结构描述为这种结构，生物学家描述每棵树能够获得的面积也可以看作是 Voronoi 图的分割，设计师也利用 Voronoi 图设计了很多丰富多彩的作品……在 Okabe[2]等人的著作中，对 Voronoi 图及其应用做了详实的介绍。对于 Voronoi 图的研究，以及不同形式的展示总是一个很新鲜的话题。

Voronoi 存在一种对偶的几何结构，也就是我们熟知的 Delaunay3 角剖分，链接任意相邻的两个 cell 中的 site，能够得到这些点集构成的凸包的 3 角剖分。实际上 Delaunay3 角剖分满足空圆特性和最大化最小角特性，空圆特性指在 Delaunay3 角形网中任一 3 角形的外接圆范围内不会有其它点存在，最大化最小角特性指 Delaunay3 角剖分所形成的 3 角形的最小角最大。因为以上两条特性，Delaunay3 角剖分是在实际中应用最多的 3 角剖分了。由于 Voronoi 图和 Delaunay3 角剖分的对偶性质，知道其中一个的结构，也就能知道另一个的结构了。

计算几何中的一个基本的问题是凸包的研究问题，凸包指的是对于给定的点集 Q，能够将 Q 中所有的点都包含在其内的凸多边形就是凸包。凸包的研究有比较成熟的理论和算法，比如 Jarvis March 算法和 Graham Scan 算法。比较成熟的工具包 Qhull 能直接生成 2,3,4 维的凸包，他们采用的是 Quickhull 的算法。

本小组基于以上的理论以及几何变换的知识，对 2 维和 3 维的 Voronoi 图的生成给出了它们在 3 维的解释。对于 2 维的 Voronoi 图，我们给出了它们在 3 维中由锥体增长相交后到 2

维的投影情况。对于 3 维的 Voronoi 图，我们给出了它们在 3 维中球的膨胀相交后得到的结果。

2. 凸包的构成

我们利用工具包 Qhull 来获得凸包和 Voronoi 图的几何。Qhull 采用 QuickHull 计算凸包的几何结构。QuickHull 使用一个类似 QuickSort 的分而治之的做法，平均意义下的复杂度为 O(nlogn)，在最差的情况下是 $O(n^2)$。

算法可以被描述为下面几个步骤：

I. 找到点集中最左边和最右边的两个点，这两个点一定是凸包上的点；

II. 连接这两个点的线将点集分成两个子集，这两个子集一个处于线的上方，一个处于线的下方；

III. 对于任意一个子集而言，找到该子集中距离线最远的那个点，该点一定也是凸包上的点，将该点个线的两端连接起来，处于 3 角形内部的点不可能是凸包上的点了，只需要考虑在 3 角形外部的点；

IV. 以新连接的线作为基准线重复步骤 III；

V. 循环这样的过程，直到没有剩余的点为止，这样凸包就构造成功了。

这个算法可以扩展到 d 维，伪代码如下：

QuickHull 算法[3]

```
输入：d 维的数据点集 P
输出：P 构成的凸包 CH(P)
初始化：选取 d+1 个极点构成一个单纯型

for 超平面 F
    for 每个没有被封杀的点 p
        if p 在 F 的上方
            把 p 添加到 F 的外部集合 S 中
while 有非空的外部集合 S 的超平面 F
    选择 S 中距离 F 最远的那个点 p
    for 每条边 L in F
        L 和 p 能构成一个新的超平面
        将这个超平面加入到当前的凸包中
```

由于 Qhull[4]基于 QuickHull 的算法，是比较强大的计算 2D，3D 以及 4D 的凸包，Delaunay3 角剖分和 Voronoi 图的工具包，Matlab 和 Octave 都是基于它来提供计算几何的功能的，Mathematica 使用它来实现 Delaunay 网格构造。它能调用不同命令得到不同的计算结果，通过不同的功能选项来控制程序的行为，还能选择特定的输出结果。

3. 几何变换

3.1 2 维 Voronoi 图在 3 维中的形成

我们采用圆锥体相交投影的方法更形象地描述 2 维的 Voronoi 图[5]，过程如下所示：

3 维中圆锥相交和 2 维 Voronoi 图的关系

对 2 维图中的任意一个点p_i，它的坐标为(x_i, y_i)
以该点为圆锥的顶，以 45° 为母线和平面的夹角生成圆锥
如果有两个圆锥他们能够相交，那么在相交处能形成 3 维空间中的一条抛物线
将该抛物线投影到 2 维平面中就是生成的 2 维 Voronoi 图

我们给出能这样做的数学推导：以任一点p_i形成的圆锥面的表达式为：

$$\mathrm{z} = \sqrt{(\mathrm{x} - x_i)^2 + (\mathrm{y} - y_i)^2}$$

则任意两个圆锥面相交得到的交线是由下面的方程来描述的：

$$\begin{cases} \mathrm{z} = \sqrt{(\mathrm{x} - x_j)^2 + (\mathrm{y} - y_j)^2} \\ \mathrm{z} = \sqrt{(\mathrm{x} - x_i)^2 + (\mathrm{y} - y_i)^2} \end{cases}$$

这是一条曲线的表达式，如果投影到 x-y 平面，就能得到

$$2(x_i - x_j)\mathrm{x} + 2(y_i - y_j)\mathrm{y} + (x_j^2 - x_i^2) + (y_j^2 - y_i^2) = 0$$

这其实是p_i和p_j的垂直平分线的表达式，也就是 Voronoi 图中每个 cell 的交界线。

我们通过圆锥相交的方式，给出了 2 维 Voronoi 图生成的另一种形象化的描述，并将生成的过程展示了出来。

3.2 3 维 Voronoi 图在 3 维中的形成

3 维 Voronoi 图依照理论依据依然可以由一个四维锥体相交投影到 3 维中形成，但是四维空间我们已经无法观察，所以我们利用 Voronoi 图的空圆性质，利用球的增长来描述 3 维 Voromoi 图的生成过程。

2 维中获得 Voronoi 图有比较好的算法比如分而治之的算法，扫描线算法，但是 3 维中再这样做就相当繁琐，所以需要用到几何变换的方法。

3 维 voronoi 图几何变换生成法[6]

将 3 维中的点投影到 4 维的抛物面$\mathrm{w} = x^2 + y^2 + z^2$上，那么 4 维中的任意一点的坐标就是$(x, y, z, x^2 + y^2 + z^2)$
对 4 维中的点利用 QuickHull 的方法求它的凸包
将下凸包投影至 3 维，则得到了 3 维中的 Delaunay 三角剖分
依据 Delaunay 三角剖分和 Voronoi 图的对偶关系，得到 Voronoi 图的结构

能够做出这样的映射是因为 Delaunay 三角剖分和凸包满足这样的性质：

a) 3 维中的 4 个点能构成 Delaunay 三角剖分的一个外接球 S，并且这个外接球中没有其他的点。
b) 4 维中的 4 个点构成的凸包的一个超平面 P 能使得凸包中的其他点都在它的一侧。

由于超平面 P 投影到 3 维是球 S，依据(b)不会有任何其他的点出现在 P 内，也就意味着不会有其他点出现在 S 内，这就意味着 P 与 S 存在等价的关系。

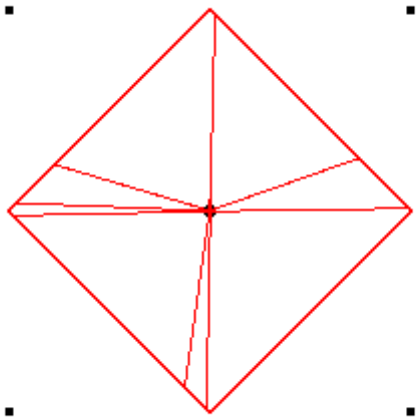
通过以上的映射的方法我们能够得到 3 维 Voronoi 图的结构，这个与我们通过球的增长获得的 Voronoi 图进行比较，可以验证算法的正确性。

4. 系统设计

演示程序提供以下演示例子：绘制二维点或从文件读入二维/三维点；绘制 Voronoi 计算结果（二维分割边，三维分割面）；动画展示二维 Voronoi 图的扫描线算法三维过程；动画展示二维 Voronoi 图的三维意义；动画展示三维 Voronoi 图的原理。

我们的演示系统分成以下几个部分：

4.1 Voronoi 计算内核：这部分负责管理 sites、Voronoi vertex 和 Voronoi cell 的数据，在数据生成方面包括随机生成 sites、手动指定 sites、按照文件生成 sites。在数据计算方面，我们调用了 Qhull 计算 Voronoi 图的接口，能够获得 Voronoi vertex 和 Voronoi cell（按照逆时针方向的 vertex 排列表示）的信息。为了通过这些信息绘制完整的 Voronoi 图，我们在无穷远处（包括正负）添加了 2^{dim} 个辅助点，这样我们感兴趣的 Voronoi cell 都是有界的，从而能够完整绘制（如下图所示，四角上的点是无穷远点，我们感兴趣的点都集中在中心，充分放大即可展示我们的结果）



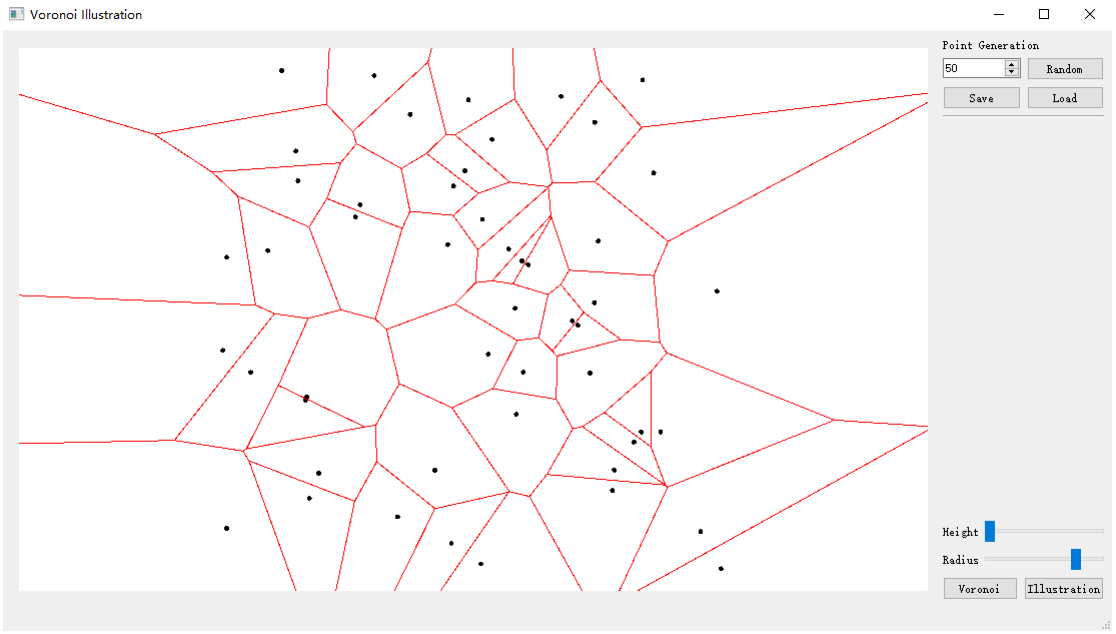
4.2 展示模块：得到二维/三维空间中的 Voronoi 图后，对于二维情况，我们先将点的坐标 Z 值置为零从而将二维点放入三维的 XY 平面上，以每个点为顶点，绘制高度和底面半径可调的圆锥（使用由顶点和圆锥底面圆周上相邻亮点构成的三角形面近似圆锥曲面。其中，随着圆锥的高度和底面半径增大，两个不同圆锥面会相遇。两个面相遇后，形成的交线即为 Voronoi 图的边。对于三维情况，则是通过在每个点处形成球，球的半径随着动画的播放而增大，直到两个球面相遇，形成交界面，即三维 Voronoi 图的分割面。在程序中，可以通过拖动鼠标移动视角，也可以通过按钮选择俯视/透视视角。

4.3 IO 模块。IO 模块包括按照固定的格式将 sites 读入计算内核，将 sites 从计算内核保存至文件，以便重现结果。

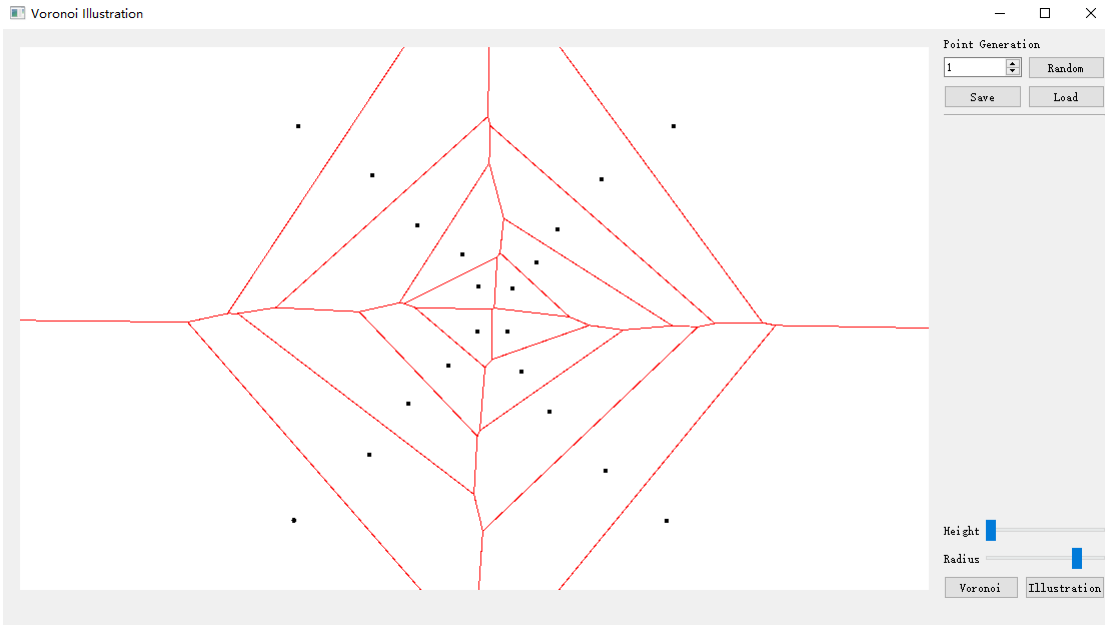
5. 实验结果

5.1 Voronoi 图计算结果

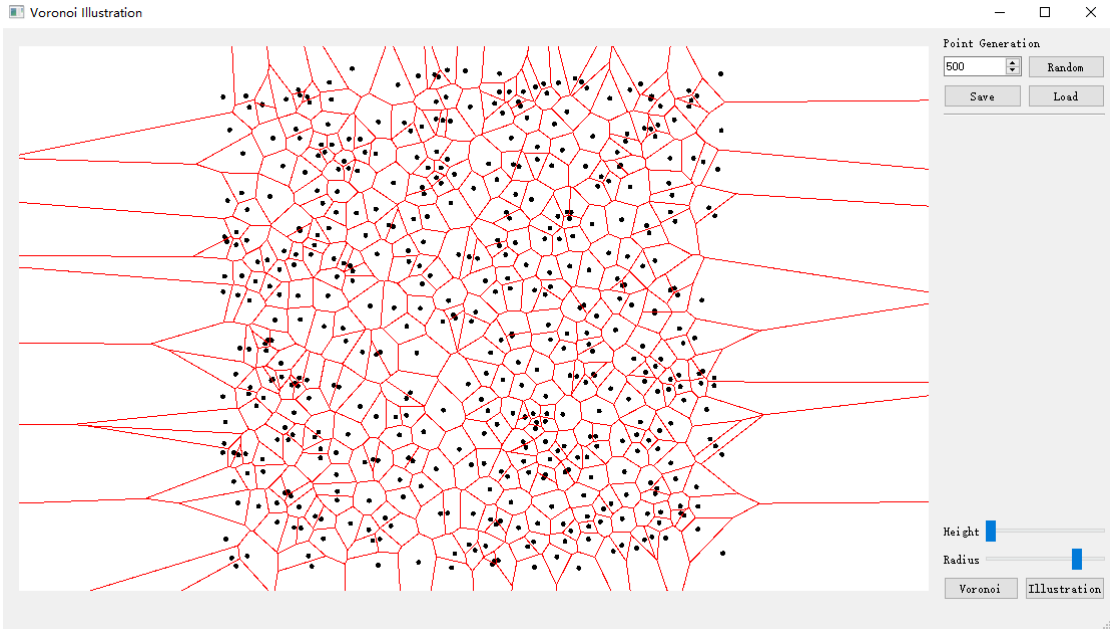
以下是对随机生成的 50 个点计算 Voronoi 图的结果：



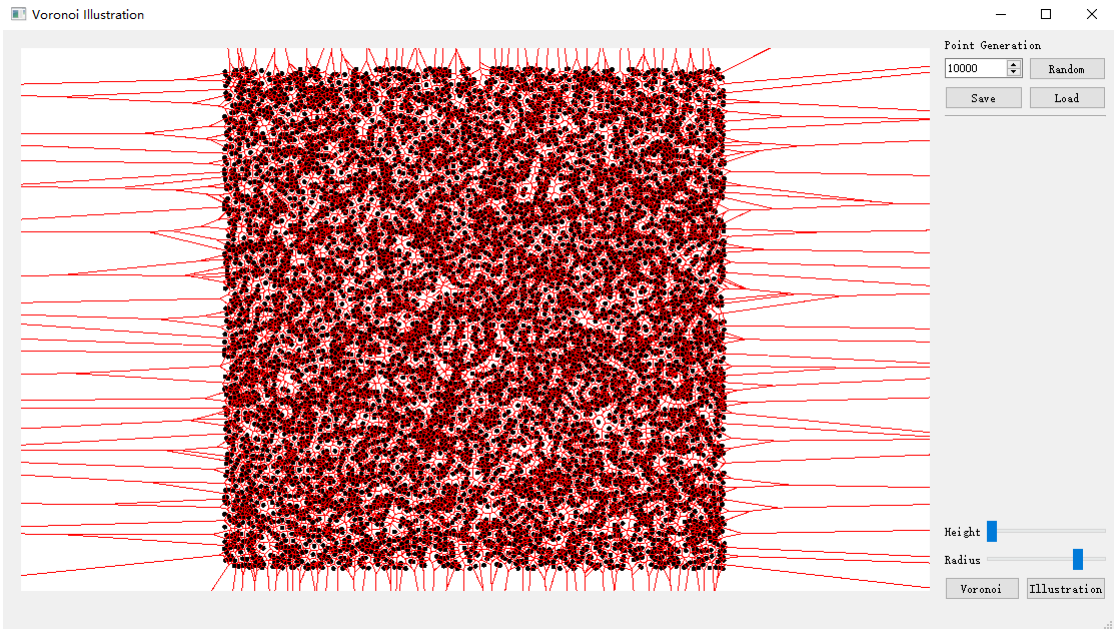
对手动生成的 20 个点计算 Voronoi 图的结果：



对随机生成的 500 个点计算 Voronoi 图的结果：



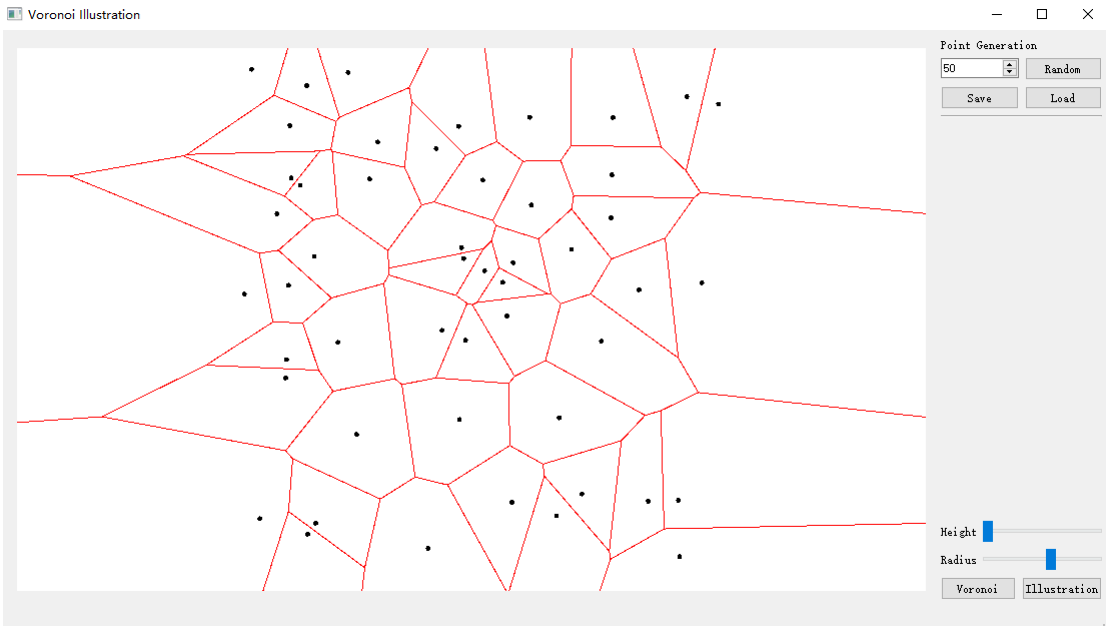
对随机生成的 10000 个点计算 Voronoi 图的结果：



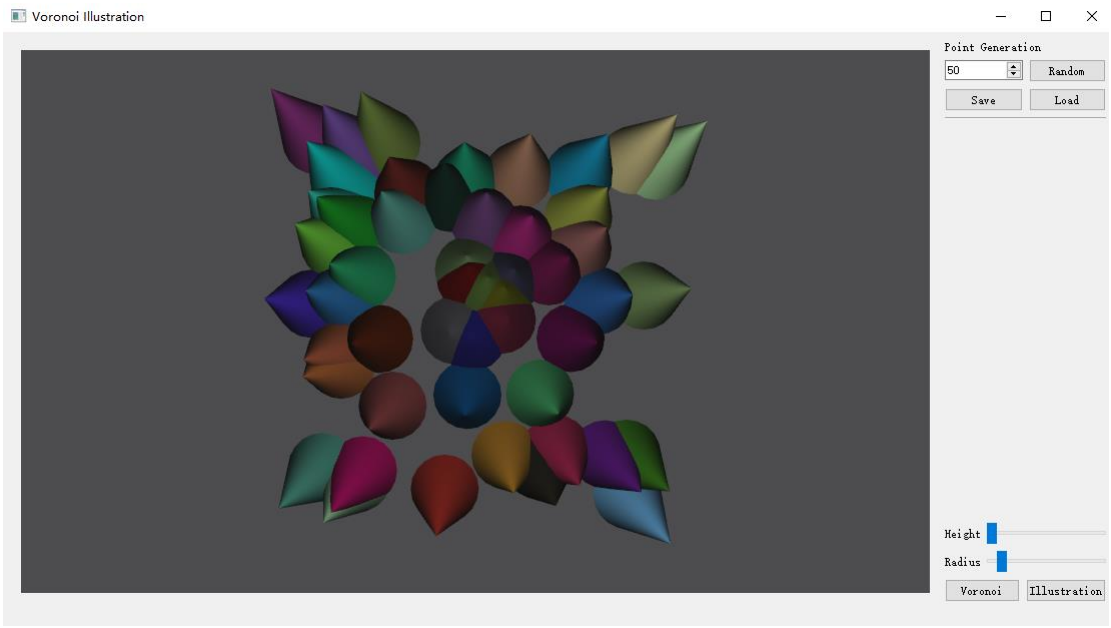
5.2 2 维 Voronoi 图的三维解释（圆锥）

以下是对随机生成的 50 个点利用圆锥展示 Voronoi 图生成原理：

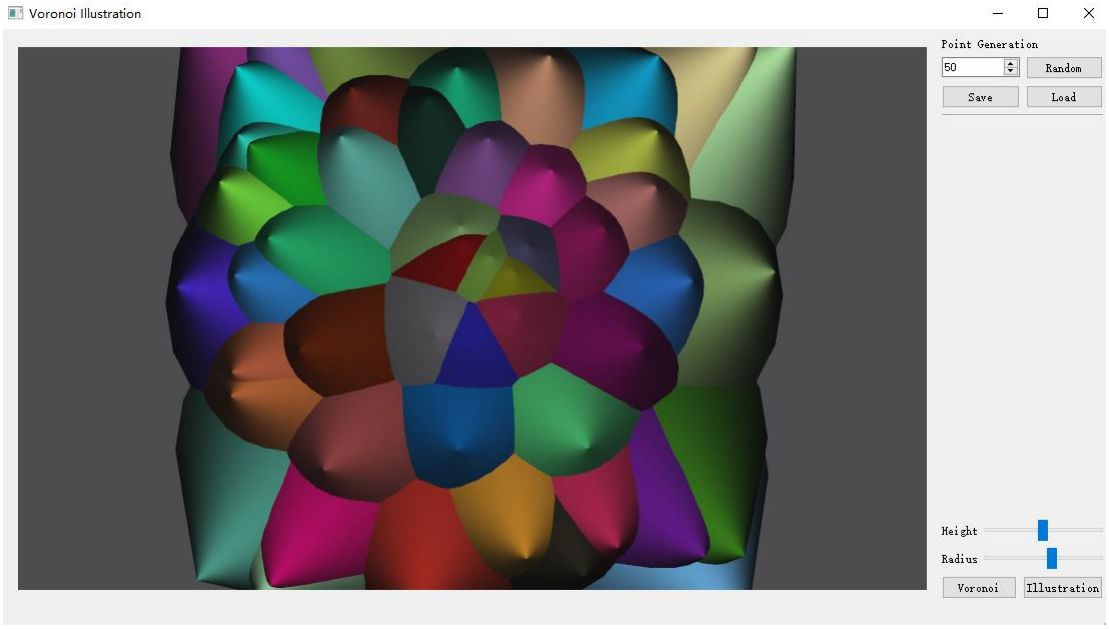
可以清晰地看到它和 Voronoi 图的关联：



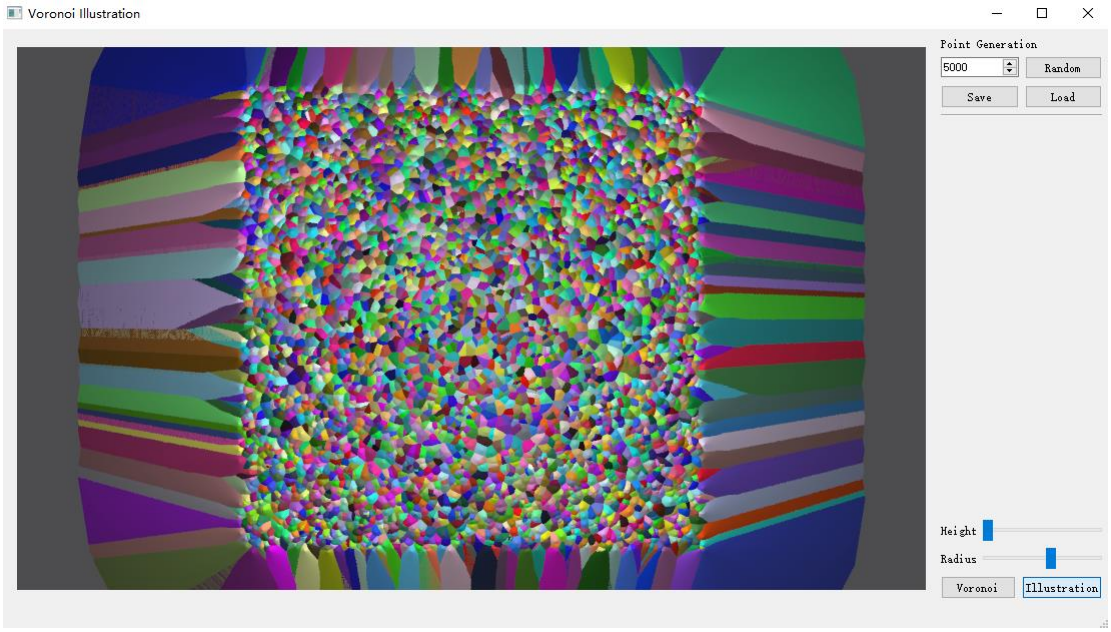
调节圆锥底面半径，可以看到 cell 生长过程：



调节圆锥高度，可以形象地看到圆锥的相交情况：



以下是对随机生成的 5000 个点利用圆锥展示 Voronoi 图生成原理：



由于 3D 绘制消耗大量内存，不再做更高规模的测试。

5.3 规模测试

对不同规模的 2 维随机生成点，我们用演示程序计算 Voronoi 图，耗时如下：

点规模	计算耗时（ms）
50	7
500	29
5000	179
50000	1943
99999	3938

6. 总结

Voronoi 图具有广泛的应用，为了更形象地理解它，我们对二维和三维的情况进行了可视化的描述。在 2 维中，我们利用圆锥的增长、相交、投影在 3 维来展示 2 维 Voronoi 图的生成过程。在 3 维中，我们利用球的增长、相交来展示 3 维的 Voronoi 图的结构。

为了完成这项工作，我们利用了 Qhull 计算几何算法中的 qvoronoi 模块、OpenGL 的绘制模块、Qt3D 中的绘图模块等工具，不仅对计算几何中的算法原理有了更深的认识，同时也学习了 OpenGL、Qt GUI 编程等实用工具，得到了很多收获。

7. 参考文献

[1] https://en.wikipedia.org/wiki/Voronoi_diagram

[2] Okabe, A., Boots, B., Sugihara, K., & Chiu, S. N. (2009).Spatial tessellations: concepts and applications of Voronoi diagrams(Vol. 501). John Wiley & Sons.

[3] Barber C B, Dobkin D P, Huhdanpaa H, et al. The quickhull algorithm for convex hulls[J]. ACM Transactions on Mathematical Software, 1996, 22(4): 469-483.

[4] http://www.qhull.org

[5] http://groups.csail.mit.edu/graphics/classes/6.838/S98/meetings/m25/mm.html

[6] Hugo Ledoux. Computing the 3D Voronoi Diagram Robustly: An Easy Explanation.2007, 10.1109/ISVD.2007.10