

Vector-based Realization of Multiplicatively Weighted Voronoi Diagrams with ArcGIS Engine

Tian Song, Cui Ximin, Sun Yunhua, Liu Qiang
College of Geoscience and Surveying Engineering,
China University of Mining and Technology,
Beijing, China

Gong Yu
South China Sea Branch,
State Oceanic Administration,
Guangzhou, China

Abstract—As a basic data structure in geometry, theoretical and computational aspects of Voronoi diagrams (VDs) and multiplicatively weighted Voronoi diagrams (MWVDs) have been widely researched and applied. But few algorithms are implemented in combination with geographic information system (GIS) softwares for practical use. A vector-based algorithm based on ArcGIS Engine (AE) is proposed to generate VDs and MWVDs for points through the methods of regions division and regions merging. New algorithm is simple and easy to understand with high precision and practicability. Generated VDs and MWVDs are convenient to store, display, output and overlay with other layers to support both vector-based and raster-based spatial analyses. It expands the application range of MWVDs in geographical field.

Keywords- VDs; MWVDs; regions division; regions merging; ArcGIS Engine

I. INTRODUCTION

A Voronoi diagram (VD), first mentioned by Georges Voronoi [1], is a way of dividing space into several regions from a set of points (called seeds, sites, or generators) as is shown in Fig.1. For each generator there will be a corresponding region (called a Voronoi region or a Voronoi cell) consisting of all points closer to that seed than to any other. A weighted Voronoi diagram (WVD) is a Voronoi diagram for which the Voronoi cells are defined in terms of a distance defined by some common metrics modified by weights assigned to generator points [2]. The WVD contains several types: multiplicatively weighted Voronoi diagram (MWVD), additively weighted Voronoi diagram (AWVD), additively weighted power Voronoi diagram, etc. The MWVD is defined when the distance between points is multiplied by positive weights [3]. In the plane under the ordinary Euclidean distance, the MWVD is also called a circular Dirichlet tessellation [4] and its edges are circular arcs or straight line segments (Fig.2).

As an important branch of computational geometry, VDs and MWVDs play enormous roles in the field of geography, which are extensively applied in solving transportation problems [5], spatial regions division [6], optimal path choice [7], optimum allocation of resources [8], regional analysis [9] and sites selection analysis [10], etc. Although theoretical and computational aspects of VDs and MWVDs have been widely researched and applied, limited efforts have been made to bring

them into GIS for practical use, especially MWVDs, because the construction of MWVDs is computationally more difficult than the VDs and weighted Voronoi regions can not be easily constructed within any standard GIS package. The most famous GIS software ArcGIS, developed by the Environmental Systems Research Institute (ESRI), provides a function to generate Voronoi polygons, but weights are not taken into account. In ArcGIS 10.2 released by ESRI in 2013, ordinary Voronoi diagrams (OVDs) can be generated in vector format by using the “Create Thiessen Polygons” in Analyst Tools, or can be created in raster format with the “Euclidean Allocation” function in Spatial Analyst Tools. However, none of the Tools can generate MWVDs.

Several references and programs are noteworthy in bringing MWVDs into GIS. The specialist package GAMBINI developed by Tiefelsdorf and Boots [11] is capable of calculating, drawing and exporting MWVDs, but the polygons can not be properly built. Gahegan and Lee [12] proposed a generalized Voronoi data structure to construct six Voronoi diagram variants including the MWVD dynamically to support interactive spatial analysis. Mu [13] demonstrated the composition and decomposition of multiplicatively weighted Voronoi polygons and provided precise computations of MWVDs. The result can be saved in shapefile formats and used in many GIS packages such as ArcGIS. Yongxi Gong, Guicai Li and Yuan Tian [14] developed a vector-based algorithm to generate and update MWVDs for points, polylines, and polygons in C#. But it is loosely linked to GIS software. Pinliang Dong [15] introduced a raster-based approach to generate VD and MWVDs for point, line and polygon using ArcObjects, and implemented it seamlessly as an ArcGIS extension. But the algorithm efficiency is low, creating a MWVD for 200 points requires about 15 minutes. FAN Xi-wei [16] introduced a method to generate VDs and MWVDs for points and line segments based on AE, which was restricted to process with a small amount of points.

For better integration of VDs and MWVDs with GIS to support various applications, we propose a vector-based algorithm to generate OVD and MWVD based on the method of regions segmentation and regions merging with AE. Following this introduction, section II introduces mathematic definitions and several properties of OVD and MWVD, and then presents some knowledge about AE interfaces relevant to algorithm implementation. In section III the idea of generating

OVD and MWVD based on AE is put forward and implemented by using several AE interfaces. Section IV represents several examples and discusses the advantages and limitations about the algorithm. At last, some conclusions are drawn in section V.

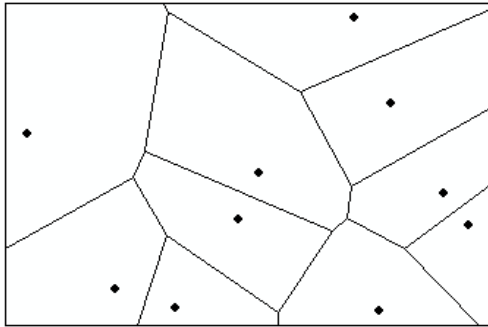


Figure 1. VD

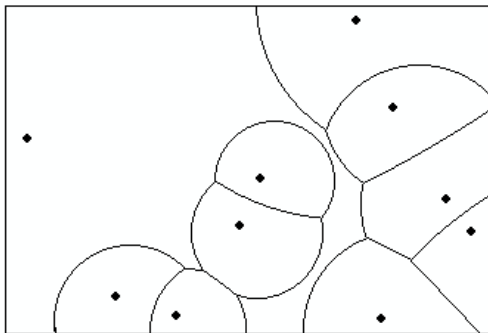


Figure 2. MWVD sharing the same generators with Fig.1

II. PRELIMINARY KNOWLEDGE

The OVD or MWVD enables a two-dimensional subdivision with Voronoi or weighted Voronoi regions without producing holes or overlaps. For an easier understanding of their underlying theories, their definitions are given. All definitions and denotations are according to [17-19].

A. Definitions

Definition 1: Let $P = \{p_1, p_2, \dots, p_n\}$ be a set of n distinct points in \mathbb{R}^2 with coordinates $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$. The subdivision of \mathbb{R}^2 into n Voronoi regions $V(p_i)$, with the property that a point $q(x, y)$ lies in the Voronoi region $V(p_i)$ if and only if $d(p_i, q) < d(p_j, q)$ for each $p_i, p_j \in P$ with $i \neq j$, is defined as the Voronoi diagram $V(P) = \{V(p_1), V(p_2), \dots, V(p_n)\}$, briefly OVD. The denotation $d(p_i, q)$ denotes a specified distance function between the site p_i and the point q . An OVD $V_o(P)$ is a Voronoi diagram using the Euclidean metric, defined by

$$d_o(p_i, q) = \|p_i - q\| = \sqrt{(x_i - x)^2 + (y_i - y)^2} \quad (1)$$

Definition 2: Let $P = \{p_1, p_2, \dots, p_n\}$ be a set of n distinct points in \mathbb{R}^2 , $W = \{w_1, w_2, \dots, w_n\}$ be a set of n positive real numbers. Each site $p_i \in P$ corresponds to a parameter $w_i \in W$.

The subdivision of \mathbb{R}^2 into n multiplicatively Voronoi regions $V_{mw}(p_i)$, with the property that a point $q(x, y)$ lies in the region $V_{mw}(p_i)$ if and only if $d(p_i, q)/w_i < d(p_j, q)/w_j$ for each $p_i, p_j \in P$ with $i \neq j$, is defined as the multiplicatively Voronoi diagram $V_{mw}(P, W) = \{V_{mw}(p_1, w_1), V_{mw}(p_2, w_2), \dots, V_{mw}(p_n, w_n)\}$, briefly the MWVD. A MWVD uses the following distance function between a generator p_i with its assigned weight $w_i \in W$ and a point $q(x, y)$:

$$d_{mw}(p_i, w_i, q) = \frac{\|p_i - q\|}{w_i} = \frac{\sqrt{(x_i - x)^2 + (y_i - y)^2}}{w_i} \quad (2)$$

For simplicity, we suppose the term “ $V(i)$ ” as the Voronoi or weighted Voronoi region of p_i except an explicit specification is made.

B. Properties

Let $p_i(x_i, y_i), p_j(x_j, y_j)$ be two points with weights w_i and w_j, L_{ij} be the bisector of p_i and p_j .

L_{ij} of an OVD is expressed as follows:

$$\sqrt{(x - x_i)^2 + (y - y_i)^2} = \sqrt{(x - x_j)^2 + (y - y_j)^2} \quad (3)$$

After the reduction, we can get:

$$2(x_i - x_j)x + 2(y_i - y_j)y - (x_i^2 + y_i^2 - x_j^2 - y_j^2) = 0 \quad (4)$$

Clearly, L_{ij} is a perpendicular bisector that separates all points in \mathbb{R}^2 closer to p_i from those closer to p_j .

For a MWVD, L_{ij} of p_i and p_j can be represented as:

$$\frac{\sqrt{(x - x_i)^2 + (y - y_i)^2}}{w_i} = \frac{\sqrt{(x - x_j)^2 + (y - y_j)^2}}{w_j} \quad (5)$$

After expanding and simplifying, we can obtain:

$$\left(x - \frac{w_i^2 x_j - w_j^2 x_i}{w_i^2 - w_j^2}\right)^2 + \left(y - \frac{w_i^2 y_j - w_j^2 y_i}{w_i^2 - w_j^2}\right)^2 = \left(\frac{w_i w_j}{|w_i^2 - w_j^2|} d(p_i, p_j)\right)^2 \quad (6)$$

$d(p_i, p_j)$ denotes an Euclidean distance function between p_i and p_j .

It is known that the bisector of p_i and p_j forms a circular arc and the coordinate of circle center is:

$$\left(\frac{w_i^2 x_j - w_j^2 x_i}{w_i^2 - w_j^2}, \frac{w_i^2 y_j - w_j^2 y_i}{w_i^2 - w_j^2}\right) \quad (7)$$

Radius is:

$$\frac{w_i w_j}{|w_i^2 - w_j^2|} d(p_i, p_j) \quad (8)$$

C. Usages of ArcGIS

ArcGIS is a geographic information software developed by ESRI to work with maps and geographic information. AE, a component of ArcGIS, is a complete GIS library for developers to build a variety of specialized applications with .NET. As a development tool, AE provides several relevant interfaces to generate OVD and MWVD. Boundaries generation, regions division and regions merging are three key points to implement the algorithm. Some preliminary knowledge about these three aspects will be introduced.

1) Boundaries Generation

Let $P = \{p_1, p_2, \dots, p_n\}$ be a set of n distinct points in \mathbb{R}^2 , MBR be the minimum bounding rectangle of P with four vertices $(x_{min}, y_{min}), (x_{max}, y_{min}), (x_{max}, y_{max})$ and (x_{min}, y_{max}) , $p_i(x_i, y_i)$ and $p_j(x_j, y_j)$ be two non-intersecting points of P , L_{ij} be the bisector of p_i and p_j . According to the above definitions and properties, the boundaries of two points of OVD and MWVD are respectively a straight line segment and a circular arc.

A line segment can be generated by ILine interface with two properties of "FromPoint" and "ToPoint". Two parameters can be gained by the following steps:

Suppose $p_{from}(x_{from}, y_{from})$ be the start point of L_{ij} intersected with MBR and $p_{to}(x_{to}, y_{to})$ be the end point, define $x_{from} = x_{min}$ and $x_{to} = x_{max}$, according to (4), coordinates of these two points can be obtained:

$$\begin{cases} x_{from} = x_{min} \\ y_{from} = -\frac{x_i - x_j}{y_i - y_j} x_{min} + \frac{x_i^2 + y_i^2 - x_j^2 - y_j^2}{2(y_i - y_j)} \end{cases} \quad (9)$$

$$\begin{cases} x_{to} = x_{max} \\ y_{to} = -\frac{x_i - x_j}{y_i - y_j} x_{max} + \frac{x_i^2 + y_i^2 - x_j^2 - y_j^2}{2(y_i - y_j)} \end{cases}$$

AE offers an interface IConstructCircularArc. ConstructCircle(IPoint CenterPoint, double Radius, bool isCCW) to create a circle, where, CenterPoint signifies the coordinate of circle center and can be obtained by (7), Radius represents the circle radius and can be got by (8), isCCW stands for "whether the circle is counter clockwise". If isCCW = true, the circle is counter clockwise, otherwise, the circle is clockwise.

2) Regions division

As seen in Fig.3, supposing $V(i)$ be the Voronoi region of p_i and p_j be the insertion point, ITopologicalOperator interface offers a method Cut(IPolyline cutter, ref IGeometry leftGeom, ref IGeometry rightGeom) to divide $V(i)$ into two parts: leftGeom and rightGeom, where cutter is the bisector of p_i and p_j ; leftGeom corresponds to $V(i)$ (Voronoi region of p_i) and

rightGeom corresponds to $V(ij)$ (Voronoi region of p_j separated from $V(i)$). With the ITopologicalOperator.Cut() method, we can also partition a multiplicatively weighted Voronoi region into two parts.

3) Regions merging

Let S_1 be a polygon and S_2 be the other polygon, two polygons can be merged by function Union(IGeometry other) of ITopologicalOperator, where interface itself represents S_1 , and parameter other indicates S_2 , the output is a merged polygon S_3 (Fig.4).

Let $S = \{S_1, S_2, \dots, S_n\}$ be a set of polygons. These polygons can also be merged to generate one polygon S_n with Union() function. The specific steps are in the following: Combine S_1 and S_2 then generate a new polygon S_2 , integrate it with S_3 and generate S_3 , in turn, S_n can be obtained.

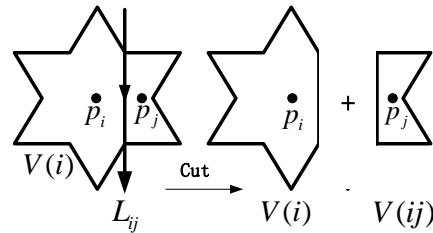


Figure 3. Region division by a line segment

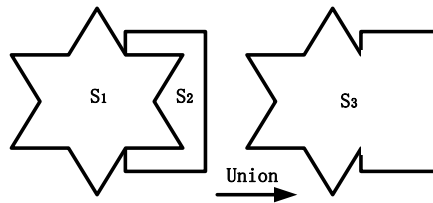


Figure 4. Regions merging

III. ALGORITHM IMPLEMENTATION

Let $P = \{p_1, p_2, \dots, p_n\}$ be a set of points in \mathbb{R}^2 , L_{ij} be the bisector of the site p_i and an insertion point p_j , $V(i)$ be the Voronoi or weighted Voronoi region of p_i . The algorithm uses the idea of incremental method: Starting from $V(2)$, everytime a new point p_j is inserted, regions $V(i)$ ($i = 1, 2, \dots, j-1$) will be redivided to get $V(j)$. Specific steps are as follows:

- Define MBR of P as $V(1)$ to be the Voronoi or weighted Voronoi region of p_1 .
- Insert p_2 , generate bisector L_{12} of p_1 and p_2 according to contents about "Boundaries Generation" in section II part C. Divide $V(1)$ into two parts with L_{12} according to contents about "Regions division" in section II part C. The one intersected with p_1 is still defined as $V(1)$ and the left one is defined as $V(2)$ belonging to p_2 .
- Insert p_3 , generate edges L_{13} of p_1 and p_3 and L_{23} of p_2 and p_3 . Divide $V(1)$ into two parts with L_{13} : $V(13)$, a polygon separated from $V(1)$ belonging to p_3 and $V(1)$,

the Voronoi or weighted Voronoi region of p_1 after division. Identically, $V(2)$ can be divided into $V(23)$ and $V(2)$ with L_{23} . Then union $V(13)$ and $V(23)$ to generate $V(3)$ according to contents about “Regions merging” in section II part C.

- Insert p_j , process $\{V(1), V(2), \dots, V(j-1)\}$ to obtain $V(j)$. Algorithm is over until all points are processed.

As OVD is an exception of MWVD when all weights of P are equal, pseudocode of generating MWVD (algorithm 1) is given first.

Algorithm 1 *CreatingMWVD*

Input: a set of n points $P = \{p_1, p_2, \dots, p_n\}$; a set of n weights $W = \{w_1, w_2, \dots, w_n\}$; MBR $V(1)$
Output: a MWVD V , composed of a set of n multiplicatively weighted Voronoi regions $\{V(1), V(2), \dots, V(n)\}$.

```

begin
  Define  $V(1)$  as the weighted Voronoi region of the first point  $p_1$ 
  Initialize a dynamic array  $V$  storing weighted Voronoi regions
  % Divide  $V(1)$  with the rest points in  $P$ 
  for  $j:=2$ ; step 1 until  $n$  do
    Get point  $p_j$  from  $P$ 
    Get weight  $w_j$  from  $W$ 
    for  $i:=1$ ; step 1 until  $j-1$  do
      Get point  $p_i$  from  $P$ 
      Get weight  $w_i$  from  $W$ 
      if ( $w_i = w_j$ ) then
        % Divide  $V(i)$  with a perpendicular bisector
         $V(ij) = DivideRegionWithPB(p_i, p_j, V(i))$ 
      else
        % Divide  $V(i)$  with a circular arc
         $V(ij) = DivideRegionWithArc(p_i, p_j, V(i), w_i, w_j)$ 
      endif
    endfor
    % Merge  $V(ij) = \{V(1j), V(2j), \dots, V(cj)\}, c=j-1$  and generate a weighted Voronoi region  $V(j)$  belonging to  $p_j$ 
     $V(j) = UnionRegion(V(ij))$ 
    % put  $V(j)$  in  $V$ 
     $V \leftarrow V(j)$ 
  endfor
  ExportVorRegions( $V$ )
end

```

The algorithm is composed of three main functions: *DivideRegionWithPB()*, *DivideRegionWithArc()* and *UnionRegion()*.

DivideRegionWithPB() is a method of region division with a straight line segment. It needs three parameters, p_i , p_j and $V(i)$. First, create a perpendicular bisector of p_i and p_j (see contents about “Boundaries Generation” in section II part C) and divide $V(i)$ into two regions with it: $V(i)$ and $V(ij)$ (see contents about “Regions division” in section II part C), then output $V(ij)$. Similarly, *DivideRegionWithArc()* is a function of region division with a circular arc which receives five

parameters: p_i , p_j , $V(i)$, w_i and w_j . The only difference between them is that the first method is to generate a straight line segment to divide region with two parameters p_i and p_j while the latter one is to create a circular arc to divide region with four parameters p_i , p_j , w_i and w_j .

UnionRegion() method achieves the function of merging a set of polygons into one. It receives one parameter $V(ij)$, a set of polygons separated from $V(i) \{i=1, 2, \dots, j-1\}$. Using the method “Regions merging” in section II part C, $V(ij)$ can be integrated together to generate $V(j)$.

The algorithm *CreatingOVD* is similar to algorithm 1, as weight factor need not to be taken into account, so the function *DivideRegionWithPB()* is retained and the method *DivideRegionWithArc()* is excluded.

Algorithm 2 *CreatingOVD*

Input: a set of n points $P = \{p_1, p_2, \dots, p_n\}$; MBR $V(1)$
Output: an OVD V , composed of a set of n Voronoi regions $\{V(1), V(2), \dots, V(n)\}$.

```

begin
  Define  $V(1)$  as the Voronoi region of the first point  $p_1$ 
  Initialize a dynamic array  $V$  storing Voronoi regions
  % Divide  $V(1)$  with the rest points in  $P$ 
  for  $j:=2$ ; step 1 until  $n$  do
    Get point  $p_j$  from  $P$ 
    for  $i:=1$ ; step 1 until  $j-1$  do
      Get point  $p_i$  from  $P$ 
      % Divide  $V(i)$  with a perpendicular bisector
       $V(ij) = DivideRegionWithPB(p_i, p_j, V(i))$ 
    endfor
    % Merge  $V(ij) = \{V(1j), V(2j), \dots, V(cj)\}, c=j-1$  and generate a Voronoi region  $V(j)$  belonging to  $p_j$ 
     $V(j) = UnionRegion(V(ij))$ 
    % put  $V(j)$  in  $V$ 
     $V \leftarrow V(j)$ 
  endfor
  ExportVorRegions( $V$ )
end

```

IV. EXAMPLES AND DISCUSSIONS

Fig.5 shows an OVD and a MWVD for 200 points with algorithms *CreatingOVD* and *CreatingMWVD*.

This paper introduces a new approach for the generation of OVD and MWVD based on AE. Some advantages and limitations of the algorithm should be discussed. Contrary to existent algorithms, new algorithm works seamlessly with GIS applications, and it can generate OVD and MWVD with high precision and practicability from point features by using methods of regions division and regions union. The input non-spatial attributes can be automatically assigned to output Voronoi or weighted Voronoi polygons through attributes joining. Initial point data and generated VDs and MWVDs can be stored together in geodatabase, which are easy to read, edit, output and display. It brings MWVD into GIS and provides a spatial data model to enrich the applications of Voronoi diagrams in the field of geography. Meanwhile, combining the VD or MWVD layouts with other vector-based or raster-based

data for overlaying analyses, more useful information will be acquired.

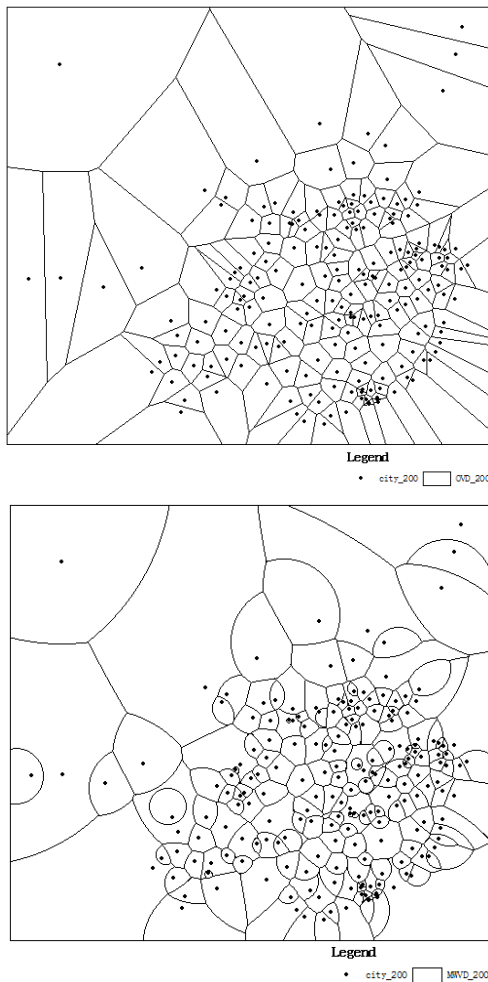


Figure 5. OVD and MWVD for 200 points

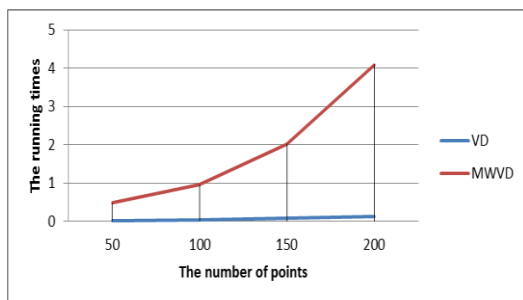


Figure 6. The running times of the algorithm for 50, 100, 150, 200 points

Some existing or possible issues with the algorithm are also addressed: the basic idea of algorithm is simple and easy to understand, but the process for generating MWVD can be relatively slow while facing massive input point features because the process loops through all features one by one and topological operations including regions segmentation and regions merging consume a great deal of time. Using four

Core i5-3337U CPUs each with 1.8 GHz, the computation of VDs and MWVDs for 50,100,150,200 are shown in Fig.6. Creating a MWVD for 200 points requires about 4 minutes, much faster than the raster-based MWVD generation algorithm based on ArcGIS in [15], which requires about 15 minutes for 200 points. For many GIS applications, this speed is acceptable. For generating OVDs, topological operations are much simpler than that of MWVDs, so the processing is much faster.

V. CONCLUSIONS

For better integration of VDs and GIS applications, we propose a vector-based algorithm to generate OVD and MWVD based on AE, which provides a spatial data model for several practical geographical applications. New algorithm is simple and easy to understand and results are convenient to store, display, output and overlay with other layers, etc. It works well for point datasets, but is not efficient for line segments and area features. In future work, problems about efficiency and processing with line segments and area features will be studied extensively.

REFERENCES

- [1] G.Voronoi, "Nouvelles applications des paramètres continus à la théorie des formes quadratiques," Premier Mémoire: Sur quelques Propriétés des formes quadratiques positives parfaites, *J.Reine Angew.,Math.*, 133, 1907,pp. 97-178.
- [2] http://en.wikipedia.org/wiki/Voronoi_diagram
- [3] Elena Deza and Michel Deza,"Dictionary of distances," Elsevier Science, 2006 ,pp.225-226.
- [4] Peter F. Ash1, Ethan D. Bolker, "Generalized Dirichlet tessellations," *Geometriae Dedicata*, Vol.20, no.2,1986, pp.209-243.
- [5] Darius Geiß,Rolf Klein,Rainer Penninger, "Optimally solving a transportation problem using Voronoi diagrams,"*Computational Geometry*, vol.46,no.8,2013,pp.1009-1016.
- [6] John Gunnar Carlsson, Raghuvveer Devulapalli, "Dividing a Territory Among Several Facilities," *INFORMS Journal on Computing*, vol.25,no.4, 2013, pp.730-742.
- [7] Mehdi Sharifzadeh,Cyrus Shahabi, "Processing Optimal Sequenced Route Queries Using Voronoi Diagrams," *GeoInformatica*, vol.12,no.4, 2008,pp. 411-433.
- [8] XUE Jiao, "Optimal Allocation Model Based on Weighted Voronoi Diagram of Primary School," Master thesis:Hebei Normal University,2012.
- [9] FENG She-miao, "Service Coverage of Civil Aviation Airports Based on Weighted Voronoi Diagram," *Journal of Transportation Systems Engineering and Information Technology*, vol.13,no.5, 2013,pp.140-144.
- [10] Zhu Xiangqian, "Research of Electric Power GIS and Transformer Substation Site Selection Based on Voronoi Diagrams," Master thesis:Zhengzhou University,2006.
- [11] Tiefelsdorf M. and Boots B.N., "GAMBINI, A GIS utility program to calculate multiplicatively weighted Voronoi diagrams (v1.01),"1997, Available at:<http://www.wlu.ca/~wwwgeog/special/download/gambini.htm> (2006-02-22).
- [12] Gahegan, M., Lee, I., "Data structures and algorithms to support interactive spatial analysis using dynamic Voronoi diagrams," *Computers, Environment and Urban Systems*,vol24,no.6, 2000,pp.509–537.
- [13] Mu L., "Polygon characterization with the multiplicatively weighted Voronoi diagram," *The Professional Geographer*,vol.56,no.2, 2004,pp.223-239.

- [14] Yongxi Gong, Guicai Li, Yuan Tian, etc, "A vector-based algorithm to generate and update multiplicatively weighted Voronoi diagrams for points, polylines, and polygons," *Computer & Geoscience*, vol.42, 2012, pp.118-125.
- [15] Pinliang Dong, "Generating and updating multiplicatively weighted Voronoi diagrams for point, line and polygon features in GIS," *Computer & Geoscience*, vol.34, no.4, 2008, pp.411-421.
- [16] FAN Xi-wei, "The Vector Construction Algorithm and Implementation of Weighted Voronoi Diagram," Master thesis: XiBei University, 2011.
- [17] Aurenhammer F, "Voronoi diagrams-A survey of a fundamental geometric data structure," *ACM Computing Surveys*, vol.23, no.3, 1991, pp.345-405.
- [18] Zhang You-hui, "A method to draw Weighted Voronoi Diagram," *Computer Science*, vol.28, no.6, 2001, pp.126-130
- [19] M. Balzer and O. Deussen, "Voronoi treemaps," in *Proceedings of the IEEE Symposium on Information Visualization (InfoVis'05, poster)*, 2005, p. 7