Spatial Analysis in PostGIS based on Voronoi diagram/ Delaunay triangulation

Quach Dong Thang

HoChiMinh City GIS Center

ABSTRACT: Voronoi diagram/ Delaunay triangulation is a basic problem in computational geometry and has been applied in many fields, especially in spatial analysis. This paper presents the ability of constructing Voronoi diagram/ Delaunay triangulation in PostgreSQL/ PostGIS and exploit it to enhance the spatial analysis functions at DBMS level for GIS developers.

Keywords*: Spatial Analysis, PostGIS, Voronoi diagram, Delaunay triangulation, PL/Java, PL/R, R, deldir package.*

1 INTRODUCTION

1.1 General introduction

The Voronoi diagram/ Delaunay triangulation (VD/ DT) and related problems have many practical applications as well as supporting other complex/ advanced spatial analysis problems in GIS applications. Most current GIS softwares support end-user tools to create VD/ DT. However, for GIS programmers, manipulating those functions at DBMS level is essential to easily control and reuse for different kinds of application.

PostGIS is an extension of the open source DBMS PostgreSQL allows spatial data management. Currently PostGIS ver 2.0 does not support built-in functions to create VD/ DT. This paper describes how to use third-party functions via Procedural language such as PL/ Java and PL/ R to construct Voronoi diagram/ Delaunay triangulation for point features in PostGIS. Based on this, the paper proposes how to exploit VD/ DT to solve Voronoi-related spatial analysis problems, as well as evaluating the ability to develop, expand functions in PostGIS for GIS developers.

1.2 Voronoi diagram

Basically, Voronoi diagram is defined as follow:

- Let $P = \{p_1,p_2,p_3,\ldots,p_n\}$ be a set of n points in Euclidean space.
- $d(p_i,p_j)$: Euclidean distance between p_i and p_j.
- Voronoi cell of p_i – denoted $V(p_i)$ is defined:

$$V(p_i) = \{ q : d(p_i, q) < d(p_j, q)\ \forall\ j \neq i \}$$

- Voronoi diagram of a set of points P, denoted V(P) is the union of all Voronoi cells of P.

In other words, Voronoi diagram is a partition of P into n regions. Each region contains exactly one point of P so that if a point q is within the region of p_i, then the distance between q and p_i is the smallest compared to other points of P.

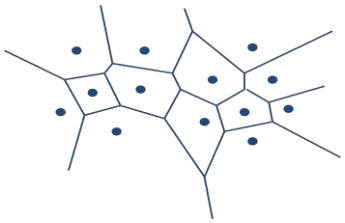


Figure 1: Voronoi diagram

Algorithms for constructing Voronoi diagram:

- Naive algorithm: for each point p_i, calculates the intersection of all half-plane of the points $p_j \neq p_i$ to build Voronoi cell $V(p_i)$. The complexity is O (n^2logn).
- Divide and conquer: O(nlogn) complexity.
- Fortune's line Sweep: O(nlogn) complexity.

1.3 Delaunay triangulation

Delaunay triangulation for a set of points P is a triangulation with vertices are the points of P so that the circumcircle of each triangle does not contain any point of P.

Delaunay triangulation is the dual graph of Voronoi diagram. The complexity of construct Delaunay triangulation is equivalent to Voronoi diagram (O (nlogn)).

Delaunay triangulation is used for creating TIN model in 3D GIS and 3D graphics.

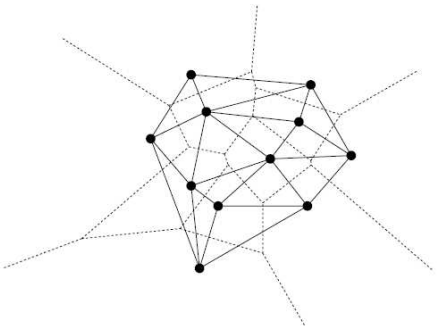


Figure 2: Voronoi Diagram and Delaunay triangulation

1.4 PostgreSQL/PostGIS

PostGIS is an extension supporting spatial data management in PostgreSQL DBMS. The PostGIS ver 2.0 has 890 functions for manage spatial data (compliant *OGC Simple Feature Specifications for SQL*). However, for complex spatial analysis problems like creating Voronoi diagram, Delaunay triangulation, find the closest pair of points, find the largest empty circle,...PostGIS does not support yet. Therefore, study and manipulate spatial analysis at DBMS level is essential for GIS developers for the following reasons:

- Developers can implement spatial analysis in database and reuse for different kinds of application (desktop GIS, web GIS).
- Simple to use through SQL queries.
- Results of spatial analysis can easily export to different formats: Well known Text - WKT, Well Known Binary - WKB, shape file,….

2 EXPERIMENT

PostgreSQL supports adding user-defined functions through 04 ways:

- Query language functions (written by SQL).

- Procedural language functions (written by PL/ Pgsql, PL/ Java, PL/ R, PL/ Tcl, PL/ Perl and PL/ Python)
- Internal functions.
- C-language functions.

The following section shows how to create Voronoi diagram and Delaunay triangulation using Procedural language functions. In details:

- Constructing Voronoi diagram: use the deldir package of R via PL/ R language.(deldir stands for: Delaunay Triangulation and Dirichlet/ Voronoi Tessellation)
- Constructing Delaunay Triangulation: use ST_DelaunayTriangles() function in Jaspa (based on PL/ Java).

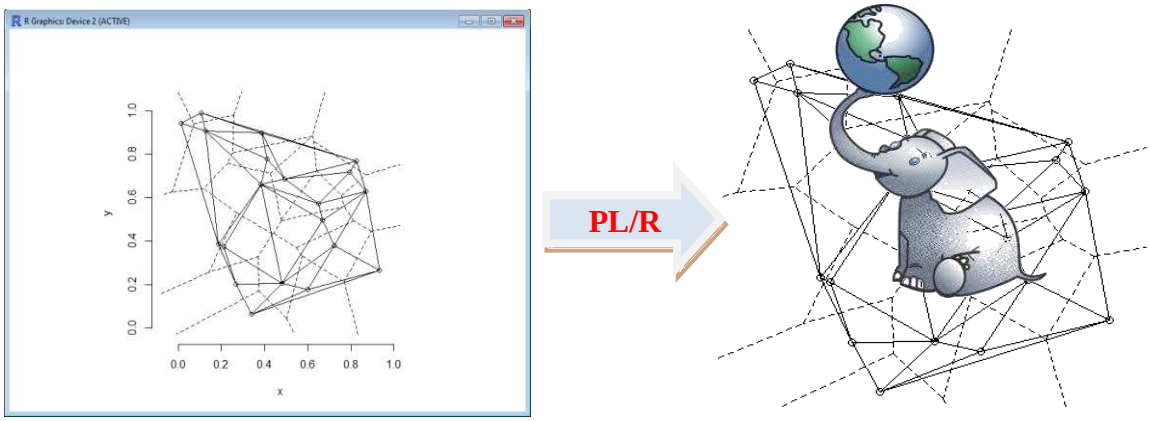
2.1 Constructing Voronoi diagram using deldir Package of R via PL/ R

R is a free software environment for statistical computing and graphics.

PL/R is a PostgreSQL language extension that allows user to write PostgreSQL functions and aggregate functions in the R statistical computing language.

Deldir package is created by Rolf Turner (http://www.math.unb.ca/~rolf/) to compute the VD/ DT for a set of points.

The proposed solution is to use PL/ R function contacts with the deldir package of R to construct Voronoi diagram for point features in PostgreSQL/ PostGIS.



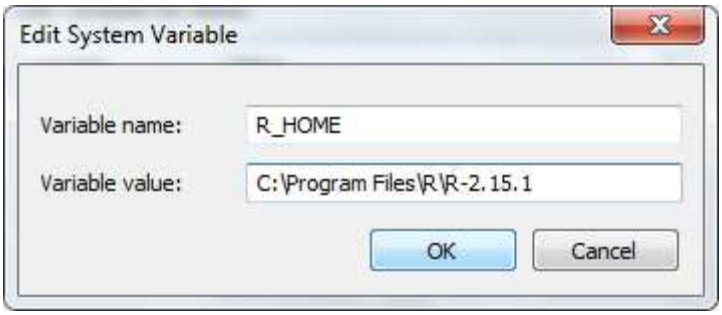
Construct Voronoi diagram/ Delaunay triangulation in R using deldir package

Construct Voronoi Diagram/ Delaunay triangulation in PostGIS using PL/R function

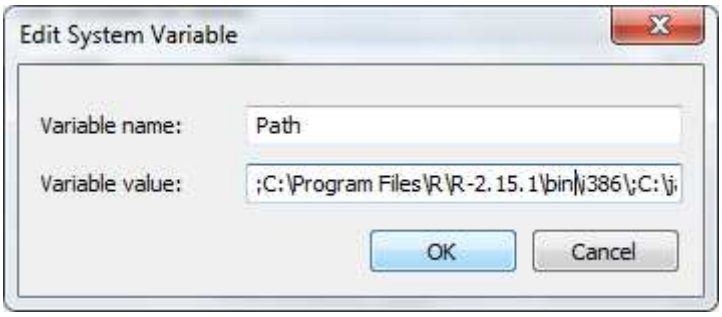
Figure 3: Construct Voronoi diagram using deldir package via PL/R

Installing Prerequisites (using PostgreSQL ver 9.1, PostGIS ver 2.0, R ver 2.15.1):

- Install PostgreSQL/ PostGIS, create a database named "voronoi" using template PostGIS and table "nodes" contains point features.
- Install R and deldir package.
- Install PL/R (For Windows user):
  - Download PL/R, copy and paste plr.dll to the lib folder of PostgreSQL.
  - Create System variable R_HOME (value: C:\Program Files\R\R-2.15.1).



- Add the file path of R into the Path variable (C:\Program Files\R\R-2.15.1\bin\i386\).



- Restart PostgreSQL service.

Create Voronoi diagram using PL/R:

- Execute the script file plr.sql (in PL/R folder) to install PL/R language in PosgreSQL.
- Execute file voronoi.sql (written by Mike Leahy) to create voronoi function in PostgreSQL/ PostGIS based on voronoi function from deldir package.

```
create type voronoi as (id integer, polygon geometry);
create or replace function voronoi(text,text,text) returns setof voronoi as '
  library(deldir)
   # select the point x/y coordinates into a data frame...
  points <- pg.spi.exec(sprintf("select x(%2$s) as x, y(%2$s) as y from %1$s;",arg1,arg2))
    # calculate an approprate buffer distance (~10%):
  buffer = ((abs(max(points$x)-min(points$x))+abs(max(points$y)-min(points$y)))/2)*(0.10)
    # get EWKB for the overall buffer of the convex hull for all points:
  buffer <- pg.spi.exec(sprintf("select buffer(convexhull(st_union(%2$s)),%3$.6f) as ewkb from %1$s;",arg1,arg2,buffer))
  # the following use of deldir uses high precision and digits to prevent slivers between the output polygons, and uses
  # a relatively large bounding box with four dummy points included to ensure that points in the peripheral areas of the
  # dataset are appropriately enveloped by their corresponding polygons:
  voro = deldir(points$x, points$y, digits=22, frac=0.000000000000000000000001,list(ndx=2,ndy=2), rw=c(min(points$x)-
abs(min(points$x)-max(points$x)), max(points$x)+abs(min(points$x)-max(points$x)), min(points$y)-abs(min(points$y)-
max(points$y)), max(points$y)+abs(min(points$y)-max(points$y))))
  tiles = tile.list(voro)
  poly = array()
  id = array()
  p = 1
  for (i in 1:length(tiles)) {
    tile = tiles[[i]]
     curpoly = "POLYGON(("
      for (j in 1:length(tile$x)) {
        curpoly = sprintf("%s %.6f %.6f,",curpoly,tile$x[[j]],tile$y[[j]])
    }
    curpoly = sprintf("%s %.6f %.6f))",curpoly,tile$x[[1]],tile$y[[1]])
    # this bit will find the original point that corresponds to the current polygon, along with its id and the SRID used for the
    # point geometry (presumably this is the same for all points)...this will also filter out the extra polygons created for the
    # four dummy points, as they will not return a result from this query:
    ipoint <- pg.spi.exec(sprintf("select %3$s as id, intersection(''SRID=''||srid(%2$s)||'';%4$s'',''%5$s'') as polygon from %1$s where
intersects(%2$s,''SRID=''||srid(%2$s)||'';%4$s'');",arg1,arg2,arg3,curpoly,buffer$ewkb[1]))
    if (length(ipoint) > 0)
    {
```

```
        poly[[p]] <- ipoint$polygon[1]
        id[[p]] <- ipoint$id[1]
        p = (p + 1)
      }
    }
    return(data.frame(id,poly))
' language 'plr';
```

- Using voronoi function to create voronoi diagram for table “nodes”.

```
create table voronoidiagram (gid serial primary key);
SELECT AddGeometryColumn ('public','voronoidiagram','the_geom',4326,'POLYGON',2);

insert into voronoidiagram (gid, the_geom)
   SELECT v.id, v.polygon
     FROM voronoi('(SELECT gid, the_geom FROM nodes ) as vor', 'vor.the_geom', 'vor.gid') As v
```

2.2 Constructing Delaunay triangulation using PL/Java and Jaspa

Jaspa (Java Spatial) is an extension for spatial data management in PostgreSQL and H2. Just like PostGIS, Jaspa creates a template contains functions to support spatial data management (compliant *OGC Simple Feature Specification for SQL Specifications*). Jaspa manages spatial data in schema named “Jaspa”, which contents ST_DelaunayTriangles() function to construct Delaunay triangulation for point features.

Suppose we have a PostGIS database called “voronoi” contains table “nodes” needed to build Delaunay triangulation. We will install Jaspa into existing “voronoi” database and use ST_DelaunayTriangles() function of Jaspa to construct Delaunay triangulation for table “nodes”.

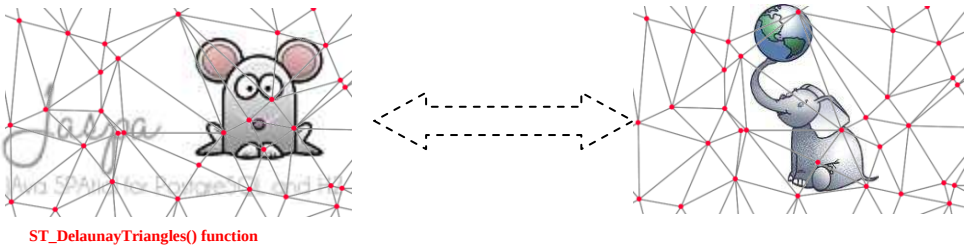


Figure 4: Construct triangulation using Jaspa’s ST_DelaunayTriangles() function

Install Jaspa:

- Download Jaspa.
- Unpack it into C:\jaspa4pg folder. The directory structure is as follows:
  - C:\jaspa4pg\lib
  - C:\jaspa4pg\bin
  - C:\jaspa4pg\sql
  - C:\jaspa4pg\doc
  - C:\jaspa4pg\jre
- Append the followings paths to the end of Path variable (in System Variable):

```
C:\jaspa4pg\jre\bin;C:\jaspa4pg\jre\bin\client;c:\jaspa4pg\bin;C:\jaspa4pg\jre\bin\server; C:\Program Files\PostgreSQL\9.1\bin
```

- Configure PL/Java: Copy the content of file C:\jaspa4pg\doc\html\ pgconf-wint.txt and append it to the content of C:\Program Files\PostgreSQL\9.1\data\ postgresql.conf

- Restart PostgreSQL service.
- Execute the script file C:\jaspa4pg\sql\pljavainstall.sql to install PL/Java language.
- Execute the script file C:\jaspa4pg\sql\jaspa.sql to create the spatial data functions.
- As a result, a schema named "jaspa" is created with 569 functions for spatial data management, including ST_DelaunayTriangles() function:

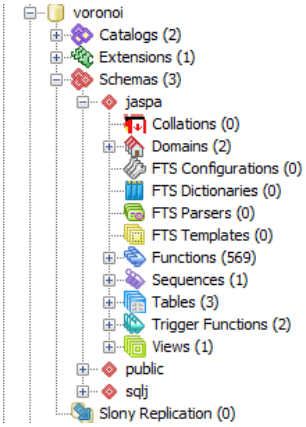


Figure 5: Schema Jaspa in "voronoi" PostGIS database

- Use jaspa.ST_DelaunayTriangles() function to construct Delaunay triangulation for table "nodes" in "voronoi" database:

```
-- Create table delaunay to hold the Delaunay Trianagulation
create table delaunay (gid serial primary key);
SELECT AddGeometryColumn ('public','delaunay','the_geom',4326,'POLYGON',2);

--  Create delaunay triangulation using jaspa.ST_DelaunayTriangles function.
insert into delaunay (the_geom)
SELECT
ST_GeomFromEWKB(ST_AsEWKB(jaspa.ST_Dump(jaspa.ST_DelaunayTriangles(jaspa.ST_GeomFromEWKB(ST_AsEWKB(ST_Collect(th
e_geom)))))))) from nodes
```

The figure below displays the Voronoi diagram and Delaunay triangulation of table "nodes" in gvSIG (open source software).

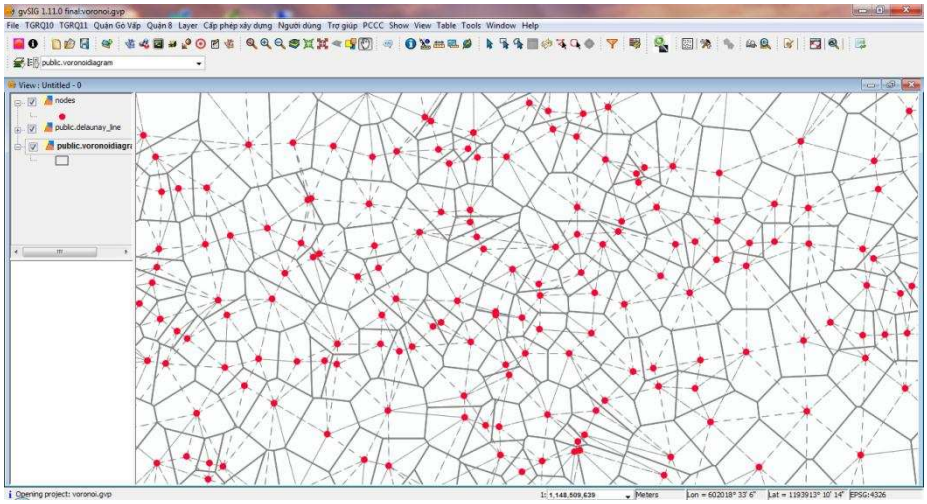


Figure 6: Display VD/ DT table in PostGIS using gvSIG open source software

3 SPATIAL ANALYSIS IN POSTGIS BASED ON VORONOI DIAGRAM/ DELAUNAY TRIANGULATION

We can start with an example of finding nearest pair of points from a point set. Ordinary, a GIS programmer could solve it by a simple SQL (and of course with a "naive" thinking and it is something like exhausted algorithm):

```
SELECT n1.gid, (select gid from nodes n2 where n1.gid <> n2.gid order by ST_Distance(n1.the_geom, n2.the_geom) limit 1 ),
(select ST_Distance(n1.the_geom, n2.the_geom)  from nodes n2 where n1.gid <> n2.gid order by ST_Distance(n1.the_geom,
n2.the_geom) limit 1 ) as dist.
FROM nodes n1
order by dist
```

With n = 4469 points, execution time in PostgreSQL is about 60s. Conspicuously, with the exhausted method, when n increases, execution time will increase highly (the complexity is O (n^2)). Meanwhile, this problem can be solved with smarter algorithms, or use the available Voronoi diagram to reduce the computed time and resources: for each point p_i, just measure the distance to the point within the adjacent Voronoi cells of p_i – or simply finding the smallest Delaunay edge touches p_i.

For example, the SQL statement to find the closest pair of points by using Delaunay edge (dual graph of Voronoi diagram) is:

```
SELECT n.gid from nodes n
where ST_touches(n.the_geom, (select the_geom from delaunay_line d order by ST_length(the_geom) limit 1))
```

The properties of the Voronoi diagram are used to solve many different problems:

- Find the nearest point in a given set of points P to a query point q: Find the Voronoi cell V(p_i) contains point q, and p_i is exactly the closest point to q.

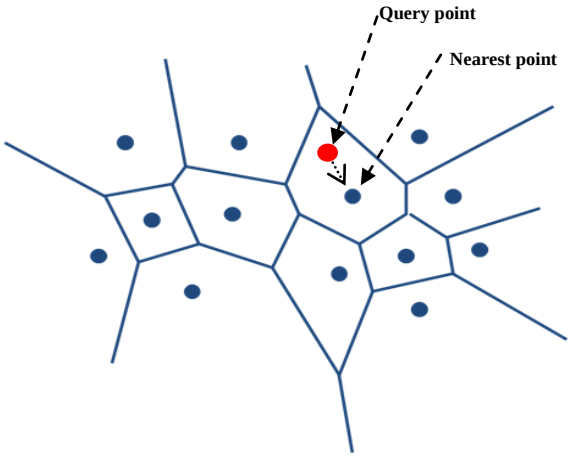


Figure 7: Find nearest point based on Voronoi cell

- Find the nearest pair of points in a given point set P: for each point p_i, just compare the distance to the points within the adjacent Voronoi cells of p_i - or the edge of the Delaunay triangles touches the point p_i to find the closest pair of points (instead of comparing the distance from each point p_i to the remaining n-1 points of P).

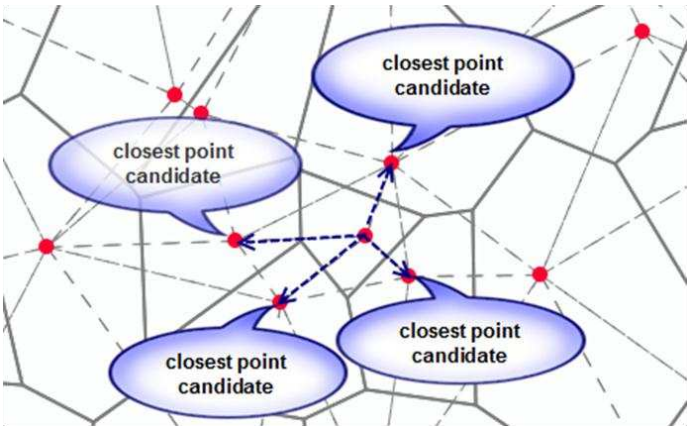


Figure 8: Find nearest pair of points based on Voronoi diagram

- Find k nearest neighbors for each point in a given set of points.
- Find medial axis/ skeleton for polygon: applied in finding the center line of roads, rivers (in spatial data digitization).
- Find the minimum spanning tree: Using the Delaunay triangulation and Kruskal's algorithm.
- Find largest empty circle in a set of points: just compare the circles whose center is the Voronoi vertex.
- Clustering.
- Classification.

Some practical applications of the Voronoi diagram / Delaunay triangulation:

- Business: A supermarket chain with many branches located in different locations. The branch managers want to send the discount leaflets to their customers. The question is how to partition the customer area so that the distance from each branch to its customer area is minimal compared to the other branches. The "customer area" is exactly the Voronoi cell of each branch.
- Hydrology: Voronoi diagram is used to calculate the average rainfall of a river basin (also known as Thiessen polygons).
- Epidemiology: determine the impact area of an infectious disease case.
- Environment: Suppose that the environmental manager need to build a garbage treatment within a residential area. The requirement is maximize the distance from the garbage treatment as possible to the residential houses. It can be solved by finding the largest empty circle based on Voronoi diagram.
- …

Some further problems of the Voronoi diagram:

- Voronoi diagram for farthest pair of points: is applied in finding the smallest enclosing circle for a set of points (the center of the smallest enclosing circle must be a vertex of the farthest-point Voronoi diagram).
- Second-order Voronoi diagram: Voronoi cell V(pi,pj) contains points which are nearest to 2 points pi and p_j.
- K-order Voronoi diagram: Voronoi cell V(pi,pj,…pk) contains points which are nearest to k points p_i, p_j,… p_k.
- Weighted Voronoi diagram.
- Line segment Voronoi diagram.
- 3D Voronoi diagram.
- …

4 CONCLUSION

In conclusion, this paper just "gather" some "tips and tricks" to construct VD/ DT in PostGIS database and exploit it to enhance the spatial analysis functions in PostGIS.

For that reason, the algorithm or detailed further problems of VD/ DT,… is out of scope of this paper.

The experiment in this research demonstrates the ability of using third-party functions to extend the spatial analysis capabilities in PostGIS (or PostgreSQL in general). In this case is the interoperation of:

- PostGIS and R through PL/R language: thanks to the author of deldir package (Rolf Turner) and PL/R voronoi function (Mike Leahy).
- PostGIS and jaspa through native SQL statement: thanks to the author of the jaspa extension and the built-in Delaunay function in jaspa template.

This paper may be not useful or bring interest to the end-users, but I think it can meet some GIS developers' needs to manipulate and extend spatial analysis functions directly in DBMS (transparent to the end-user).

Bibliography

1) Boissonnat, J.-D. (January 2010). *Convex Hulls, Voronoi Diagrams and Delaunay Triangulations.* ENS-Lyon.
2) Edwards, R. (December 2, 2010). *Determining the Skeleton of a Simple Polygon in (Almost) Linear Time.* Oak Ridge, Tennessee.
3) Francis Chin, Jack Snoeyinky, Cao An Wangz. *Finding the Medial Axis of a Simple Polygon.*
4) Franz Aurenhamme, Rolf Klein. *Voronoi Diagrams.*
5) Goswami, P. P. *Introduction to Computational Geometry.*
6) Inkulu, R. *Voronoi diagrams: Higher order.*
7) Kreveld, M. v. *Computational Geometry: its objectives and relation to GIS.*
8) Leo Hsu and Regina Obe. (n.d.). *bostongis*. Retrieved from PL/R and PostGIS: http://www.bostongis.com/?content_name=postgresql_plr_tut02#98
9) Maksym Zavershynskyi, Prof. Evanthia Papadopoulou. *Higher-Order Voronoi Diagram of Line Segments.*
10) Mark de Berge, Otfried Cheong, Marc vn Kreveld, Mark Overmars. (2008). *Computational Geometry - Algorithms and applications.* Springer.
11) Martinez, P. J. (n.d.). Retrieved from http://jaspa.upv.es/blog/
12) Nandy, S. C. *Voronoi Diagram.*
13) Nehab, D. *Medial Axis.*
14) Obe, L. H. *Cross Comparison of Spatially enabled databases: PostGIS, SQL server and Jaspa.*
15) Schuster, M. *The Largest Empty Circle Problem.*
16) Sharifzadeh, M. (2007). *Spatial Query Processing using Voronoi diagram.*
17) Smid, M. (1997). *Closest-Point Problems in Computational Geometry.* Magdeburg, Germany.
18) Turner, R. (n.d.). Retrieved from http://cran.r-project.org/web/packages/deldir/index.html

19) Vennemann, K. *Beyond PostGIS: new developments in opens source spatial databases.*

Paper Reference No.: PN-19

Title of the paper: Spatial analysis in PostGIS based on Voronoi diagram/ Delaunay triangulation

Name of the Presenter: Quach Dong Thang

Author (s) Affiliation: Quach Dong Thang

Mailing Address : 244 Dien Bien Phu Street, Ward 7, District 3, HoChiMinh City

Email Address : quachdongthang@yahoo.com

Telephone number (s): (+84) 933 908 919

Fax number (s) : (+84) 839320963

Author Photograph:

Brief Biography:

Full name: Quach Dong Thang

Day of birth: 12/04/1982

Education: Master of Science in Maps - Remote Sensing and GIS, HoChiMinh City University of Technology, 2008.

Position: Head of Technical Division, HoChiMinh City GIS Center.

Fields of interest: Open source GIS software, Computational Geometry, Agent based modeling, 3D GIS.

Mobile: (+84) 933 908 919

Homepage: hcmgisportal.vn

Email: quachdongthang@yahoo.com