Recomendaciones de formato y contenido de la memoria https://www.fi.upm.es/?pagina=2031

FITTING PLANE PARTITIONS THROUGH VORONOI DIAGRAMS

Supervisor Manuel Abellanas mabellanas@fi.upm.es
Student Guillermo Alonso Núñez guillermo.alonso.nunez@alumnos.upm.es
Universidad Politécnica de Madrid (UPM)

<<<IMAGE OF THE SOFTWARE>>>

INDEX

SUMMARY

Voronoi diagrams have practical and theoretical applications to a large number of fields, mainly in science, technology and visual art. The aim of my project is to study the inverse Voronoi diagram problem and design, compare and analyze different strategies for its solving. The inverse Voronoi diagram problem consists on detecting whether a given plane partition is a Voronoi diagram and finding the seed points that would generate such a partition. For a partition which does not come from a Voronoi diagram, it would be interesting to find the best fitting Voronoi diagram. At this point, we need a way to measure how good a candidate solution is. We will be using the total symmetric difference between the two partitions.

Note that despite the search space is bounded, it being continuous grants an infinite number of solution candidates. Therefore, we will try to solve the problem using different metaheuristic, which makes it impossible to tell whether the obtained solution is optimum.

The basic steps of all the strategies are:

- For each input, calculate a set of seed points from which we will start.
- Move each point slightly following some criteria and check if we improved the current best solution.
- Repeat last step until we cannot keep improving or we are satisfied with the result.

RESUMEN

Los diagramas de Voronoi tienen aplicaciones tanto prácticas como teóricas en muchos ámbitos, la mayoría relacionados con la ciencia y tecnología, aunque también se aplica en otros como el arte visual. El objetivo de mi proyecto es estudiar el problema inverso del diagrama de Voronoi y diseñar, comparar y analizar diferentes estrategias para su resolución. Dicho problema consiste en detectar si una partición dada es o no un diagrama de Voronoi, y calcular las semillas que lo generan en caso afirmativo. Para particiones que no lo son, sería interesante encontrar el diagrama de Voronoi que mejor se aproxima. Para ello, necesitaremos alguna forma de evaluar como de buena es una solución. Usaremos la diferencia simétrica entre las dos particiones para tal fin.

Nótese que pese a que el espacio de búsqueda es acotado, al ser continuo hay infinitos candidatos que habría que probar, lo cual es inviable. Por tanto, trataremos de resolver el problema mediante diferentes metaheurísticas, aunque en ningún caso podremos asegurar que la solución obtenida es óptima.

El esquema básico de todas las estrategias es el siguiente:

- Para cada datos de entrada, calcular un conjunto de puntos semilla a partir de los cuales comenzaremos la búsqueda.
- Desplazamos los puntos ligeramente según algún criterio y comprobamos si hemos conseguido mejorar nuestra solución, actualizando ésta en ese caso.
- Repetimos el paso anterior hasta que no podamos seguir mejorando o hasta que la solución obtenida se considere lo suficientemente buena.

INTRODUCTION

There are some concepts the reader should be familiar with in order to follow this document without problems. Those are:

- Partition of a set. Partition of the square unit.
- Voronoi diagrams.
- Symmetric difference.

Partition of a set. Partition of the square unit.
A partition of a set is a grouping of the set's elements into non-empty subsets, in such a way that every element is included in one and only one of the subsets.

In our case, we will be working with partitions of the square unit. That is, a set of polygons such that the union is the square unit and the intersection of two polygons is, at most, a segment. Note that there might be points contained in more than one polygon.

Voronoi diagrams.
A Voronoi diagram is a partitioning of a plane into regions (called Voronoi cells) based on distance to points (called seeds or generators) in a specific subset of the plane. In this project we will only be using the simplest case of Voronoi diagrams: the seeds are given as a finite set of points in the Euclidian plane.

The lines that appear in a Voronoi diagram are the points of the plane that are equidistant to two or more of the nearest seeds.

Symmetric difference.
For measuring how good a given solution is, we will rely on the symmetric difference as an indicator. For two given polygons A and B, the symmetric difference can be calculated as follows:

$$SD(A, B) = Area(A \bigcup B) - Area(A \bigcap B)$$

Applying this concept to two partitions, we can define the symmetric difference of two partitions as the sum of all the symmetric differences for each pair of related polygons.

Another way of calculating it would be to substract from the area of the unit square, the area of the intersections of every pair of related polygons.

We will be treating this value as a ratio with the total area of the unit square, so a 0% value will mean the two partitions are exactly the same.

OBJECTIVES

The objectives of this project are:

- Fit a given partition of the square unit as much as we can through Voronoi diagrams. Therefore, the variable we want to minimize is the resulting symmetric difference between the input and the solution given.
- If the given partition is in fact a Voronoi diagram, we would like to arrive to that conclusion. Even though this looks trivial, it will be very difficult to get there for non trivial input, since the number of local minimums that have to be avoided to find the seeds increases in a different magnitude compared to the number of polygons.

DEVELOPMENT

Why Processing. Pros and cons of Java.
ga

Line and polygon clipping. Divide and conquer. Robustness. a

Input data.
The input data for the problem is a partition of the square unit. That is, a set of polygons whose union is the square unit and, for every two polygons, its intersection is at most a straight line. More precisely, each polygon is given as a set of ordered 2D points. At first, we know we will be working only with convex polygons (that is, all internal angles are less or equal than 180 degrees).

Output data.
The program generates various files after being executed. The files will be located in the "voronoi" folder, and will be the following:

- output.csv: CSV file in which each line represents each iteration of the algorithm. Each line contains information of where each seed point was at that stage and the symmetric difference in that precise step.
- before.png: Image that displays the first approximation of the Voronoi diagram.
- after.png: Image that displays the final approximation of the Voronoi diagram.

Gradient method.
a

Simmulated annealing.
a

Getting rid of the "repetition" problem: randomizing point order in each state.

Bibliografía

https://processing.org – Open source programming language and IDE in which the whole project is coded

http://leebyron.com/mesh/ - External processing library used for calculating the Voronoi diagrams

http://www.lyx.org/ - Document processor used for writing this thesis

https://github.com/Flood1993/TFG_voronoi - Git repository containing everything about this project