CSE555 Spring 2019 - Computational Geometry - Project Description

Chi Kien Huynh

May 24, 2019

Contents

1 The problem

Given a point set S in the unit square and let $(0, 0) \in S$, find a set of non-overlapping valid rectangles so that they cover the maximum possible area. A rectangle R_i is considered valid if (a) a point $p_i \in S$ is its bottom left corner and (b) no other point $p_j \in S$ lies inside it.

2 Analysis

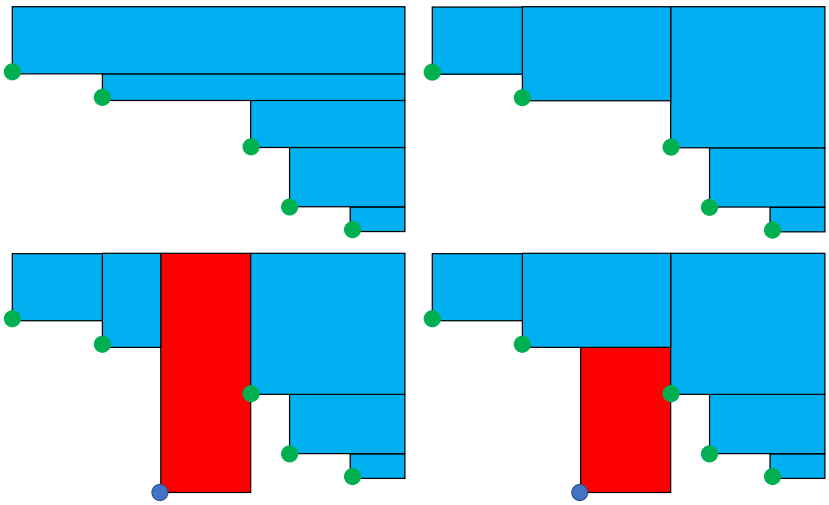

Figure 1: Choosing rectangles for the topmost frontier. The blue point does not belong to the frontier.

The topmost frontier of the points can be computed independently from other points. As shown in Figure 1, for the topmost frontier, regardless of which combination we choose, the covered area will stay the same. Moreover, even if we choose a rectangle from a point below them first, the overall area will also stay the same. Therefore, the upper most frontier of the point set can always be computed first.

This is not true for lower frontiers. Regardless, the algorithm used here is built around frontiers.

3 Algorithm

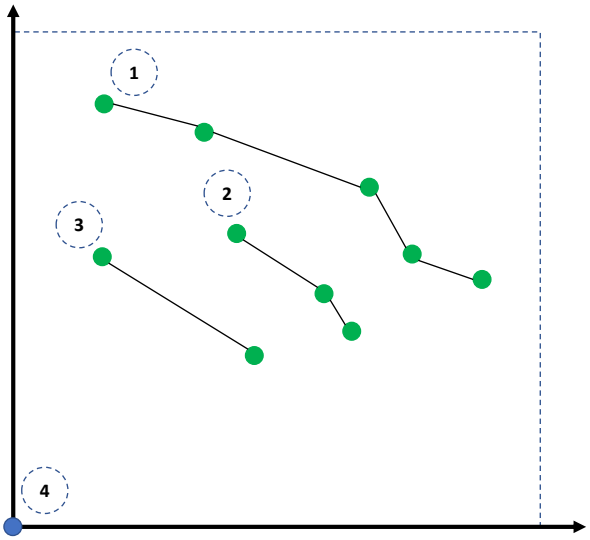

Figure 2: A set of points divided into 4 frontiers. In this particular set, the first frontier will be processed first, then the second, and so on.

The proposed algorithm has the following steps:

- (1) Divide the point set into different level of Pareto frontiers.
- (2) For each set of frontier, from the highest frontier to the lowest:
- (2.1) Find the biggest rectangle associated with a point p.
- (2.2) Remove p fromt the frontier.
- (2.3) Do step (2.1) and (2.2) until there is no point left in the frontier.

4 How to run

The program is coded in C++, it is provided along with FLTK to render the GUI. Download the FLTK library here `https://www.fltk.org/software.php` if:

- Somehow FLTK is not provided with the package.
- Or you are running this on MacOS or Linux. The provided package (.lib files) is for Windows.

- Or you cannot run the .lib files due to different compiler problem in Windows.

Once the FLTK source is downloaded, follow FLTK instruction (README.XX.txt, XX is your operating system) to compile the related static libraries files (.a on linux or .lib on windows) before compiling the main program.

The default structure of the project is as follow:

```
parent_dir
├─ maximum_area_coverage
│   ├─ gui.*
│   ├─ pointcloud.*
│   ├─ main.cpp
│   ├─ config.h
│   └─ input.txt
├─ include
│   └─ FL
└─ lib
```

To compile and run it, do the following steps:

- (1) Set ./parent_dir/maximum_area_coverage/include/ to be an additional include directory.
- (2) Set ./parent_dir/maximum_area_coverage/lib/ to be an additional lib directory.
- (3) Link the following files when compile: fltkgld.lib; fltkd.lib; fltkjpegd.lib; fltkpngd.lib; fltkimagesd.lib; (In Visual Studio: Open Project → Properties → Linker → Input).
- (4) Compile and run.

5 The GUI

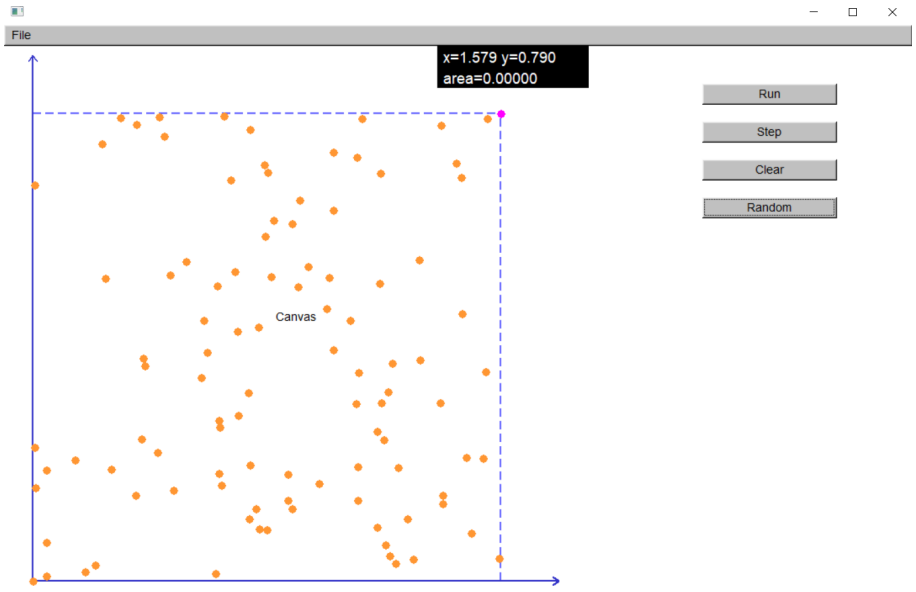

Figure 3: The graphics interface.

- You can draw points on the canvas by left-clicking inside the unit square
- Random points can be generated by hitting the Random button.
- To run the entire algorithm from start to finish, press Run.
- To run the algorithm, frontier-by-frontier, hit Step.
- To clear all points and rectangles from the canvas, press Clear.
- To load points from a file, go to File→Open. Choose the file you want (a sample file that goes with the package is input.txt). The format of the input is as follow: first line depicts the number of points in the file; after that each line consists of the coordinates of each point x and y separated by a space.