**COMP 6721 Project 1 Report**

Authored by Junwei Zhang

ID - 40050122 zhangjw248@gmail.com

1. Introduction
   1. Project manage to do

In this project, we are supposed to design one or two heuristic algorithms to generate optimal path(path length and its cost) between two points in the map (presented by longitude and latitude coordinate of each point).

The map information which is given by *.shp* file. And we are given by a certain area within city Montreal.

Each point shows on map are include crime rate(one point represent one crime), geometry Point, and several information. From the data been given, we are about to use crime rate as obstacle for our path to certain point.

* 1. Subdivide the problem

For project, we need to decide which area’s crime rate is considered risk as blocked area and which area is accessible, in order to do that, we must first create certain area by our self (grid in map) and calculate total crime rate on each grid we assume.

Second, to create grid on map we should first read original map which is given and plot the given area on Screen within x, y 2D coordination.

Third, after we divided each part of the map we need able to count each part total crime rate and store it for later usage.

Fourth, using certain function decide if a grid is considered as risk area and marked as **“blocked”**.

Try to use different color to separate it which is easy to show to customers.

Last, design our heuristic algorithm, find the path and print it on map. And that is all we want.

* 1. Detail for extension

First, when we create grid on map, it is important that we could change each grid size as we want.

Because small size of a grid is always means more accuracy.

Second, how do we define a risky area is depends on some kind of threshold for total crime rate on map, and we should be able to change that to see some different result.

Third, all edges of map are inaccessible simply because we can not define if it risky or not.

Fourth, when there is possible we can’t find optimal path to termination, we need to show some message to notice customers and should terminate the program.

Last, analysing geospatial data means we need to involve some geo-data analysis tool in python.

1. Data analysis
   1. Organize data

In this project, the map contains a lot of information and I need to organise these useful data for later usage. (designing algorithm)

First, I opened the file by using *‘geopandas’* and plot on the prompt to see specific area I was working on, and record 4 points of edges.

Second, since each point represent a crime and include geometry coordination, I use pandas and then numpy to form a matrix include each point coordinate (That is the only important information we need to get).

Third, we declare a value represent grid\_size and use ‘meshgrid’ to create\_grid on map, also we generate two more matrix each represent x\_coordinate information and y\_coordinate information.

Fourth, for each grid, count how many points inside and stored as another matrix for later usage.

Fifth, use fourth matrix calculate total mean, standard deviation, each grid crime rate and print on screen to show, and use these information decide threshold (mean value).

Last, for each grid, if its crime rate over threshold marked as ‘risky’ (1) , others marked as ‘safe’ (0).

Using above information painting color for each grid (yellow means ‘risky’, blue means ‘safe’)

The final matrix (0, 1) we create is used by counting path.

* 1. Algorithm description

In my program, I was designed an A\* algorithm for my own to find the optimal path. The reason why was I using and how to design is as follow:

For data information as large as map, we need both consider complete and time complexity. Since A\* algorithm has h(n) evaluate function to reduce the cost and it is also a ‘traceback’ algorithm, I decided to use that. And for h(n) I use *manhattan* function.

First, create an *openlist* and a *closelist* for store nodes (use points to represent), confirm *startNode* and *endNode*, make *startNode* as *currentNode*.

Second, the cost function for each node is f(n) = g(n) + h(n), start with startNode and put it into openlist.

Third, get the startNode and set to currentNode and check the 8 points around this node. Checking if each of its node can be accessed and if it is calculate each f(n). Then, put all accessible node into openlist and set each of these node’s father node currentNode.

Fourth, put currentNode into closeList and we don’t check inside closelist.

Fifth, get the min\_value f(n) in openlist and set to currentNode, check 8 points around it. Ignore the node that inside closelist or inaccessible, if node is not inside openlist put it into openlist and set their father node as currentNode.

Sixth, if node is inside openlist, check if this node has smaller g(n), if it does set this node father as currentNode and update its f(n). and continue to fourth step.

Util endNode is inside openlist or openlist is empty, for previous, we print endNode recursively its father node all the way throught the beginning startNode to generate a path. For later, we simply can not find path because of obstacle and print some notification message.

* 1. Algorithm detail

First, for each node g(n) is currentNode to its actual path cost, h(n) I use manhattan function (please see in the program).

Second, the edges of map are considered as inaccessible.

Third, how do I find it is obstacle, I use matrix (0,1) as comparison, if for each grid, if I met 1, which means this grid is yellow and it is ‘risky’.

Fourth, for present algorithm, I create another map matrix according each 4 points of a grid coordinate. Using ‘8’ as notification of pathTag and ‘0’ for all the others.

Last, the reason I created lots of matrix by using numpy is simply because numpy way much faster calculating than python, it can reduce actual machine time.

1. Comparison for result

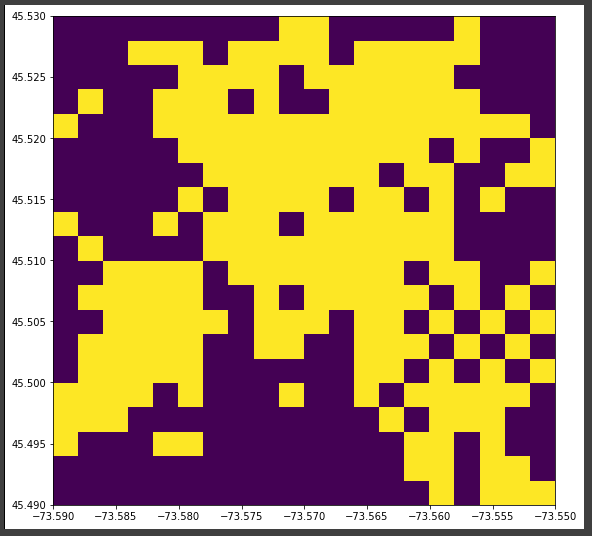
According to different threshold and grid size, it affects my algorithm calculating time.

If threshold is high that means less grid can reach that threshold level to become ‘risky’ area. In that case, there are more grid are accessible and cost may less.

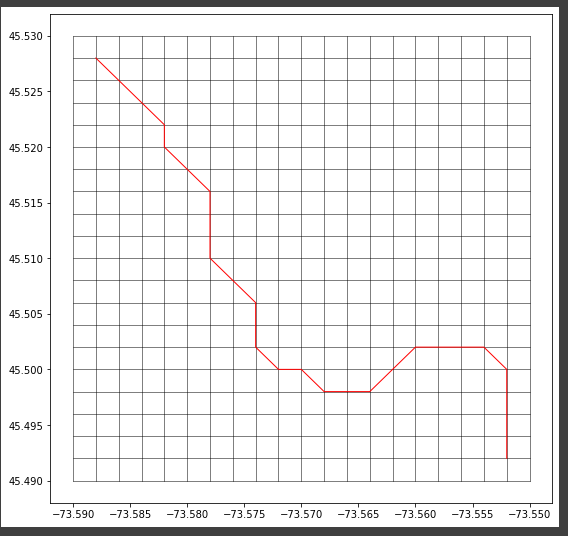
Otherwise, if threshold is low enough, more grid can reach threshold level and become ‘risky’ area. In that case, less grid is accessible and may cost more to reach goal or couldn’t find way to pass.

For grid size, the smaller the grid size the more time to take to search path for sure, because that means there are more points need to be searched.

Showing the result as follow:

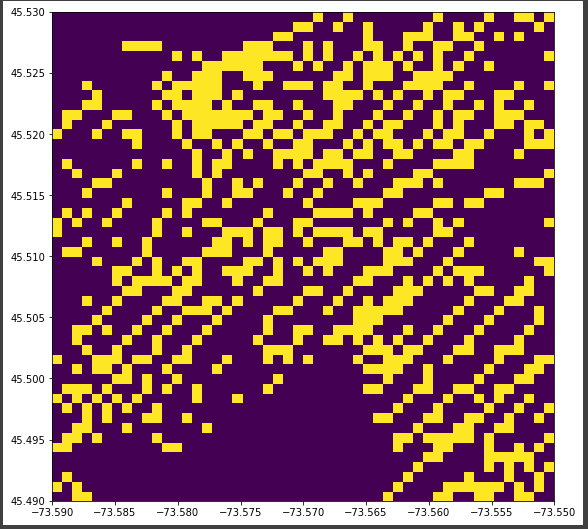


**Fig. 1.** Painted area (yellow block is ‘risky’) grid size = 0.002\*0.002, and threshold = 0.25.

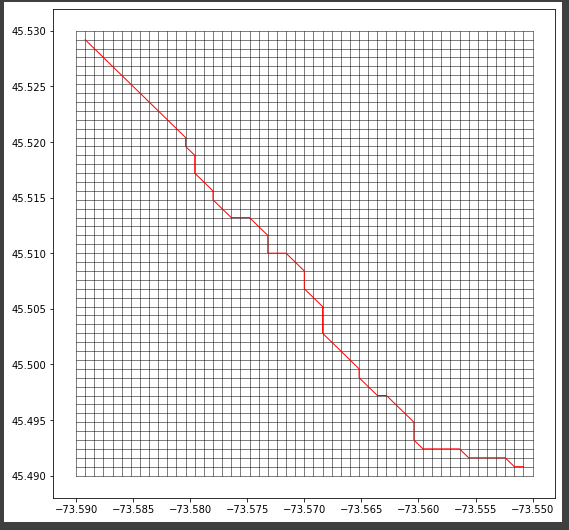


Time used to generate path is: 0.021062700000129553

**Fig. 2.** The path generated from **Fig.1**.



**Fig. 3.** Painted area (yellow block is ‘risky’) grid size = 0.0008\*0.0008, and threshold = 0.25.



Time used to generate path is: 0.03521790000013425

**Fig. 4.** The path generated from **Fig.3**.

1. Difficulty encountered

Since I’m new to python, for beginning it is really hard to find a way to learn numpy, pandas geopandas, matplotlib and some other module, it takes me lot of time through watching video on youtube and searching tutorial document. But after that, for building a map part is rather easier to implement.

For algorithm, the most challenging part is path is from point to point, not grid to grid. So, I need to use two matrix and compare each other and make lots of ‘if statement’ to determine if a point is inaccessible or if its g(n) cost is different.