CSCE 625
Programing Project #1: Implementation (writeup and video requirements is separate)
Due time: Tuesday 2/20

This project is to be done individually. The software will be checked for plagiarism using Moss so be very careful. It is ok to discuss the project with each other on a general level, e.g., "I'm using a stack instead of a queue," but not at the "here is my function call" level. A "let's sketch out the main program together" can be fine if it is clear that the collaborating students did their own coding. Unfortunately most programming assignments (such as this one) are small enough that two students working off the same detailed design will produce nearly identical code and thus the project would not be considered to be done individually.

Topic:

You are to implement one of the following two algorithms in Section IV from "UAV Intelligent Path Planning for Wilderness Search and Rescue" which is attached.

* Complete-coverage Algorithm (CC)
* Local Hill Climbing Algorithm (LHC)

If you really don't like any of these recommended algorithms, you can discuss with the instructor about choosing one of the other algorithms in that sections, but the methods for those algorithms were not emphasized in class or in the book and you would get much less support from the TA.

The intent was to give you a choice but at the same time minimize the effort you have to spend in getting "hands on" experience with a search algorithm. It also provides experience in taking a paper and implementing what it described.

Implementation requirement:

• Programming Language: you are free to choose from (C++, Python, Java, Matlab) for algorithm implementation.
• Platform: you can use any platform, such as your own laptop or department server, but you must provide a README file and make files so that the TA can compile and run the program on his machine.

• Input/Output Format:

```
        Y------------------------>
         0     1     2   ..  99
X 0   (0,0) (0,1) (0,2)    (0,99)
↓ 1   (1,0) (1,1)
↓ 2   (2,0)
↓ :
↓ 99  (99,0)
```

Input (Heatmap) file format: The input file will be a .txt file (e.g., heatmap_1.txt) in which each line contains three numbers representing (X_coordinate, Y_coordinate, probability) of gridel. You can assume the size of coordinate is (100 x 100) and the value of the gridel probability is a real number from [0.0, 1.0]. You will take as input a .txt map with a list of normalized probabilities. And the summation of all probabilities is equal to 1.

For example:
0 0 0.0
0 1 0.1
0 2 0.1
…
99 99 0.0

Output (search path) file format: The output file should be a .txt file (e.g., path_1.txt) in which each line contains two numbers representing (X_coordinate, Y_coordinate) of gridel, so that the UAV will visit the gridels in the first to last order. You can assume that the UAV will always start from (0,0).

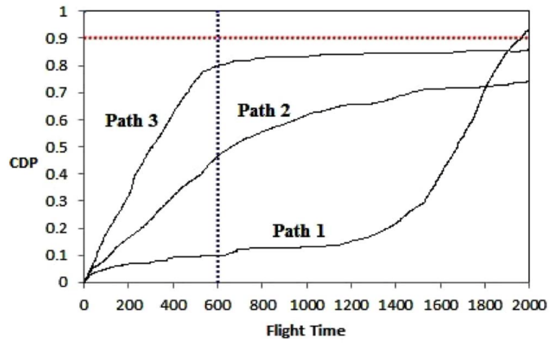
For example, path_1.txt might be:
0 0
50 50
50 51
…
99 99

Where the UAV starts at (0,0), flies in a straight line to (50,50), then to (50,51), and so on. The number of "waypoints" should be less than the 10,000 elements in the grid but the path should cover all 1,0000 elements (which was the point of the Lin and Goodrich paper).

Evaluation:

Each search path should allow the UAV to visit ALL the elements in the grid, though the ones with highest probability would be visited first. For each search path, we will use following two metrics to evaluate its quality.

Please report following things in a word (or pdf) file:
• Figure (CDP v.s. time): assume the speed of UAV is 1 dist/second, and the distance between two point is euclidean distance. Please draw with matlab or other graphing software a figure showing the change of cumulated detection probability (CDP) over time. The CDP is the summation of gridel probability the UAV has visited. One sample figure is:



• Table (EfficiencyLB v.s. time): please read the section V of paper "UAV Intelligent Path Planning for Wilderness Search and Rescue" for EfficencyLB calculation. For each search path you generated, please report the EfficencyLB at time [100, 200, 300, 600, 900].

What to Turn in:

• You need to turn in your source code and output files of given heatmap examples in a .zip file using the eCAMPUS. Please include any files that are needed to compile and run the program.
• Include a README file in your submission which provides any instructions necessary for compiling and running your program. It is your responsibility to make sure the TA can compile and test your program.
• Include a word (or pdf) file which report the Figure (CDP v.s. time) and Tabel (EfficencyLB v.s. time) of each search path you generated.

Grading Rubric:

• [10 points] Can the program generate the search path (in required format and the supporting graph and table) successfully with different heatmaps as input? We will test your program with a new file that is similar to, but different, than the two examples. We will also try your program with a much harder bonus file and give you another 10% if the program runs with that.

• [5 points] Does the search path provide complete coverage of the polygon (in this case a 100x100 square)? The TA is writing a script to check this.

• [15 points] Can the search path reach the highest probability gridels with priority? (Note that the original goal of minimizing turns has been eliminated because we didn't find an algorithm that could do that.)