ECE 4984 & 5984: (Advanced) Robot Motion Planning (Spring 2017)*

Homework 2

Due: Thursday February 16th, 9PM

February 9, 2017

Instructions. This homework constitutes 5% of the course grade. You must work on it individually. It is okay to discuss the problems with other students in class. However, your submission must be your original work. This implies:

- If you discuss the problems with other students, you should write down their names in the pdf report.
- If you refer to any other source (textbook, websites, etc.), please cite them in the report at the relevant places.
- The answers in the pdf report must be written entirely by you from scratch. No verbatim copy-paste allowed without citations.
- Any software you submit must be written entirely by you with no copy-pasting of significant portions of code from other sources.

Please follow the submission instructions posted on canvas exactly. You must submit your assignment online on canvas by the due date. Your submission must include one pdf file with the answers to all the problems and one or more files containing your code. It is okay to scan your answers and create the pdf submission.

Problem 1 — 10+10 points

Suppose we have a graph where all the vertices represents points on a 2D integer grid. Let G_4 be a graph where each vertex has edges only to one or more of its four neighbors (N,W,S,E) with N representing the North neighbor and so on. Let G_8 be a graph where each vertex has edges only to one or more of its eight neighbors (N,NW,W,SW,S,SE,E,NE). The cost of an edge is the Euclidean distance between the points. The cost of a path is the sum of the costs of the edges along the path.

Recall the definition of an admissible (i.e., under-estimate) heuristic. *A heuristic function, $h()$, is said to be* admissible *if for any vertex i, $h(i)$ is less than or equal to the cost of the optimal path starting from i and terminating at the goal vertex G.*

In class, we saw that Euclidean distance function is an admissible heuristic for these graphs. Now, consider the Manhattan distance function:

$$h_M(i) = |x_i - x_G| + |y_i - y_G| \tag{1}$$

where (x_i, y_i) are the grid coordinates of vertex i and (x_G, y_G) are grid coordinates of the goal vertex G.

*Correction posted on Feb 9, 2017 highlighted in red

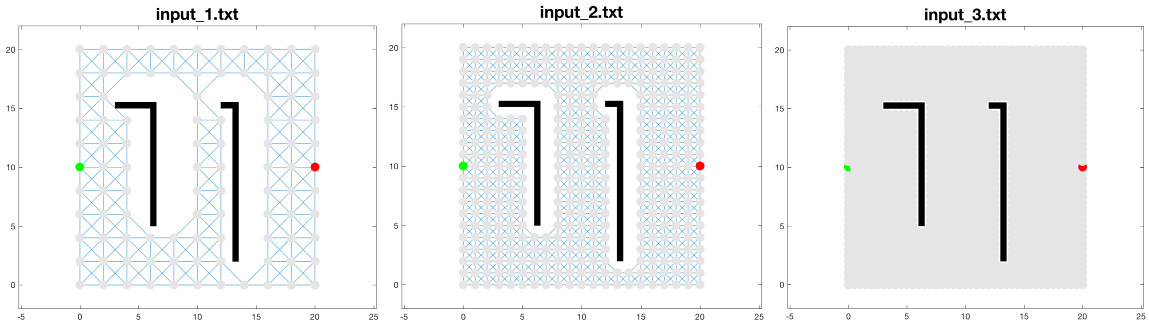

Figure 1: The three input files represent the same grid environment with increasing grid resolutions. The start (green) and the goal (red) vertices are at the same grid position in all three graphs but have different vertex numbers since the graph sizes are different.

(a) Is $h_M()$ an admissible heuristic for G_4? If yes, justify. If not, give a counter-example of $h_M()$ leading to over-estimate.

(b) Is $h_M()$ an admissible heuristic for G_8? If yes, justify. If not, give a counter-example of $h_M()$ leading to over-estimate.

Problem 2 — 80 points

Implement and compare the performance of Dijkstra, A*, and Weighted A* algorithms to solve for minimum cost paths on given graphs. We will consider only 2D grid graphs in this assignment. The edge costs are the distances between the two vertices. Each vertex can be connected to at most eight of its neighbors (N,NW,W,SW,S,SE,E,NE).

Input Description. You are given three graphs that represent the same environment but with increasing grid resolutions (Figure 1). There are two input files corresponding to each graph: (1) the `input_*.txt` file specifies the input graph along with the start vertex and goal vertex in the same format as Homework 1; (2) the `coords_*.txt` file lists the x and y coordinates of the vertices starting with vertex numbered 1 on the first line.

For example, the `input_1.txt` file contains 99 vertices. The `coords_1.txt` file specifies the actual coordinates of the 99 vertices starting with vertex numbered 1 to vertex numbered 99.

The second input file will allow you to compute the heuristic (Euclidean distance to the goal vertex). You can also use this file to make it easier to visualize the paths produced by the algorithm. For example, in MATLAB R2016b you can use the examples shown on this page (`https://www.mathworks.com/help/matlab/ref/graph.plot.html#buzeikk`) to display the graph. Similar plotting functions are available for C/C++ and python. See, for example, Graphviz `http://www.graphviz.org/Home.php`.

You do not need to visualize the paths or submit your visualizations. This is meant only to help you debug and understand what is happening in your code.

Output Files. You may implement your algorithm using one of the following programming languages: MATLAB, C/C++, or Python. Your implementation must produce a single executable (for C/C++) or script (for MATLAB, Python) named `problem2` that reads in all six input files (saved in the same directory as the executable or script) and produces two output files named `output_costs.txt` and `output_numiters.txt`. Each output file must contain a 3×6 table with the first row (i.e., first line of the file) corresponding to `input_1.txt`, the second row for `input_2.txt`, and the third row for `input_3.txt`. Each line must list six numbers (i.e., six columns): for `output_costs.txt` these should give the cost of the paths returned by Dijkstra, A* with Euclidean distance as heuristic, weighted A* with $\epsilon = 2$, weighted A* with $\epsilon = 3$, weighted A* with $\epsilon = 4$, weighted A* with $\epsilon = 5$ for the corresponding files. For `output_numiters.txt`, the six numbers should correspond to the number of iterations taken by the algorithm to find the output path (listed in the same order as the other output file). The number of iterations refer to the number of times we remove an entry from the closed list which is also the same as the size of the closed list at the end.

Your main executable or script must read in all input files, make specific calls to the Dijkstra and weighted A* functions, and generate the two output files. Note that instead of writing six separate functions, you can just write one function which can be configured to all six algorithms.

Report. In addition to submitting code, you must write a short paragraph describing your implementation in the pdf report. This description should include instructions on how to compile and run your code. You must also include the two tables in the pdf report.

We will test your code on instances other than the three input files supplied. Make sure you follow the input/output conventions exactly. Submit all files necessary to compile and run your code.