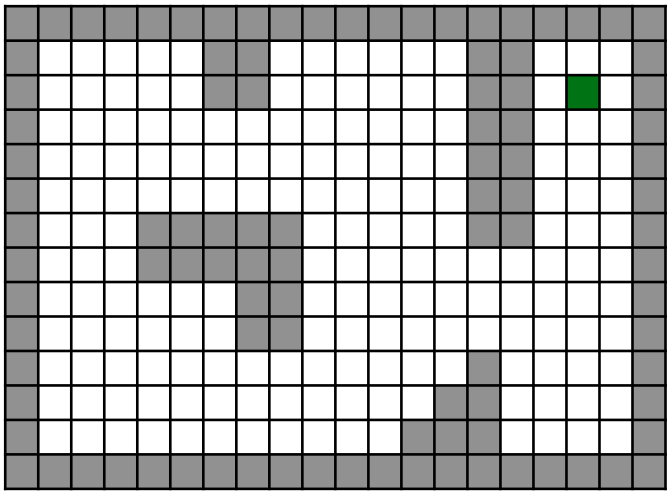
Autonomous Robots

Potential Functions – Brushfire algorithm and Wavefront planner

This document will guide you through the practical work related to path planning algorithms based on potential functions. In particular, this practical exercise consists on programming the brushfire and wavefront planner algorithms to solve a simple path planning problem. All the code has to be programmed in Matlab.

1. Environment

The problem consists in generating the potential of repulsive obstacles (brushfire algorithm) and finding the optimal trajectory (wavefront planner) towards the goal in a finite 2D environment that is closed and contains some obstacles (in grey) as shown in the figure:



The size of the environment is 14 rows by 20 columns. The goal position is marked in green, and it is located in row 3 column 18. The environment will be represented as a matrix in the following way:

```
map=[
     1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1;
     1 0 0 0 0 0 1 1 0 0 0 0 0 0 1 1 0 0 0 1;
     1 0 0 0 0 0 1 1 0 0 0 0 0 0 1 1 0 0 0 1;
     1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 1;
     1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 1;
     1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 1;
     1 0 0 0 1 1 1 1 1 0 0 0 0 0 1 1 0 0 0 1;
     1 0 0 0 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 1;
     1 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 1;
     1 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 1;
     1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1;
     1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 1;
     1 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 1;
     1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1;];
```

Note that the obstacle cells are marked with 1 and the free space with 0. The goal position and start position will be indicated as input arguments in the wavefront planner.

2. Brushfire algorithm.

Program in Matlab the Brushfire algorithm to compute the repulsive potential of the occupied cells. Use 8-point connectivity.

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	2	2	2	2	2	1	1	2	2	2	2	2	2	1	1	2	2	2	1
1	2	3	3	3	2	1	1	2	3	3	3	3	2	1	1	2	3	2	1
1	2	3	4	3	2	2	2	2	3	4	4	3	2	1	1	2	3	2	1
1	2	3	3	3	3	3	3	3	3	3	4	3	2	1	1	2	3	2	1
1	2	3	2	2	2	2	2	2	2	3	4	3	2	1	1	2	3	2	1
1	2	3	2	1	1	1	1	1	2	3	4	3	2	1	1	2	3	2	1
1	2	3	2	1	1	1	1	1	2	3	4	3	2	2	2	2	3	2	1
1	2	3	2	2	2	2	1	1	2	3	4	3	3	3	3	3	3	2	1
1	2	3	3	3	3	2	1	1	2	3	3	3	2	2	2	3	3	2	1
1	2	3	4	4	3	2	2	2	2	3	3	2	2	1	2	3	3	2	1
1	2	3	3	3	3	3	3	3	3	3	2	2	1	1	2	3	3	2	1
1	2	2	2	2	2	2	2	2	2	2	2	1	1	1	2	2	2	2	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

You must program a Matlab function with the following input and output parameters:

```
function [value_map]=brushfire(map)
```

Your program, when called with the previous map and the following parameters should generate:

```
[value_map]=wavefront(map)
```

```
value_map =

     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1
     1     2     2     2     2     2     1     1     2     2     2     2     2     2     1     1     2     2     2     1
     1     2     3     3     3     2     1     1     2     3     3     3     3     2     1     1     2     3     2     1
     1     2     3     4     3     2     2     2     2     3     4     4     3     2     1     1     2     3     2     1
     1     2     3     3     3     3     3     3     3     3     3     4     3     2     1     1     2     3     2     1
     1     2     3     2     2     2     2     2     2     2     3     4     3     2     1     1     2     3     2     1
     1     2     3     2     1     1     1     1     1     2     3     4     3     2     1     1     2     3     2     1
     1     2     3     2     1     1     1     1     1     2     3     4     3     2     2     2     2     3     2     1
     1     2     3     2     2     2     2     1     1     2     3     4     3     3     3     3     3     3     2     1
     1     2     3     3     3     3     2     1     1     2     3     3     3     2     2     2     3     3     2     1
     1     2     3     4     4     3     2     2     2     2     3     3     2     2     1     2     3     3     2     1
     1     2     3     3     3     3     3     3     3     3     3     2     2     1     1     2     3     3     2     1
     1     2     2     2     2     2     2     2     2     2     2     2     1     1     1     2     2     2     2     1
     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1
```

It is very important that you generate the repulsive potential map (value_map) using EXACTLY the same format you can see in the previous example. Your “brushfire.m” file will be evaluated exclusively based on the results it generates which must follow the previous format. The algorithm will be tested with different environments.

Representation

In order to correctly show the repulsive potential that your algorithm is generating, you can use the Matlab plotting capabilities to automatically plot the map. Use your own map representation, and include the figure in your laboratory report. The representation should not be done in the “brushfire.m” file, but in the script file that you use.

You can plot the repulsive potentials such as the ones we explained in class, being the higher values the ones containing obstacles and the lowest values, the ones that are further away from obstacles.

3. Wavefront planner.

Program in Matlab the wavefront planner algorithm to compute the optimal path towards the goal. Use 8-point connectivity.

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	21	20	19	18	18	1	1	14	14	14	14	14	14	1	1	3	3	3	1
1	21	20	19	18	17	1	1	14	13	13	13	13	13	1	1	3	2	3	1
1	21	20	19	18	17	16	15	14	13	12	12	12	12	1	1	3	3	3	1
1	21	20	19	18	17	16	15	14	13	12	11	11	11	1	1	4	4	4	1
1	21	20	19	18	17	16	15	14	13	12	11	10	10	1	1	5	5	5	1
1	21	20	19	1	1	1	1	1	13	12	11	10	9	1	1	6	6	6	1
1	21	20	19	1	1	1	1	1	13	12	11	10	9	8	7	7	7	7	1
1	21	20	19	18	17	17	1	1	13	12	11	10	9	8	8	8	8	8	1
1	21	20	19	18	17	16	1	1	13	12	11	10	9	9	9	9	9	9	1
1	21	20	19	18	17	16	15	14	13	12	11	10	10	1	10	10	10	10	1
1	21	20	19	18	17	16	15	14	13	12	11	11	1	1	11	11	11	11	1
1	21	20	19	18	17	16	15	14	13	12	12	1	1	1	12	12	12	12	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

You must program a Matlab function with the following input and output parameters:

```
function [value_map, trajectory]=wavefront(map, [start_row, start_column],
                                           [goal_row, goal_column])
```

Your program, when called with the previous map and the following parameters should generate:

```
[value_map, trajectory]=wavefront(map, [13, 2], [3, 18])
```

```
value_map =

     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1
     1    21    20    19    18    18     1     1    14    14    14    14    14    14     1     1     3     3     3     1
     1    21    20    19    18    17     1     1    14    13    13    13    13    13     1     1     3     2     3     1
     1    21    20    19    18    17    16    15    14    13    12    12    12    12     1     1     3     3     3     1
     1    21    20    19    18    17    16    15    14    13    12    11    11    11     1     1     4     4     4     1
     1    21    20    19    18    17    16    15    14    13    12    11    10    10     1     1     5     5     5     1
     1    21    20    19     1     1     1     1     1    13    12    11    10     9     1     1     6     6     6     1
     1    21    20    19     1     1     1     1     1    13    12    11    10     9     8     7     7     7     7     1
     1    21    20    19    18    17    17     1     1    13    12    11    10     9     8     8     8     8     8     1
     1    21    20    19    18    17    16     1     1    13    12    11    10     9     9     9     9     9     9     1
     1    21    20    19    18    17    16    15    14    13    12    11    10    10     1    10    10    10    10     1
     1    21    20    19    18    17    16    15    14    13    12    11    11     1     1    11    11    11    11     1
     1    21    20    19    18    17    16    15    14    13    12    12     1     1     1    12    12    12    12     1
     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1

trajectory =

    13     2
    13     3
    13     4
    13     5
    13     6
    13     7
    13     8
    13     9
    13    10
    13    11
    12    12
    11    13
    10    14
     9    15
     8    16
     7    17
     6    17
     5    17
     4    17
     3    18
```

It is very important that you generate the wavefront planner map (value_map) and the trajectory using EXACTLY the same format you can see in the previous example. Your “wavefront.m” file will be evaluated exclusively based on the results it generates which must follow the previous format. The algorithm will be tested with different environments.

Note that several trajectories can be obtained, all of them being optimal. Your code must generate one of the optimal trajectories considering an 8-point connectivity. So, the previous trajectory is only one example of several optimal trajectories you could generate.

Note finally that the wavefront planner does not distinguish between up, down, left, right movements and diagonal movements. Since we know that diagonal movements will be longer, you can modify your code to prioritize up, down, left, right movements if possible.

Representation

In order to correctly show the trajectory that your algorithm is generating, you can use the Matlab plotting capabilities to automatically plot the map and the trajectory, such that:

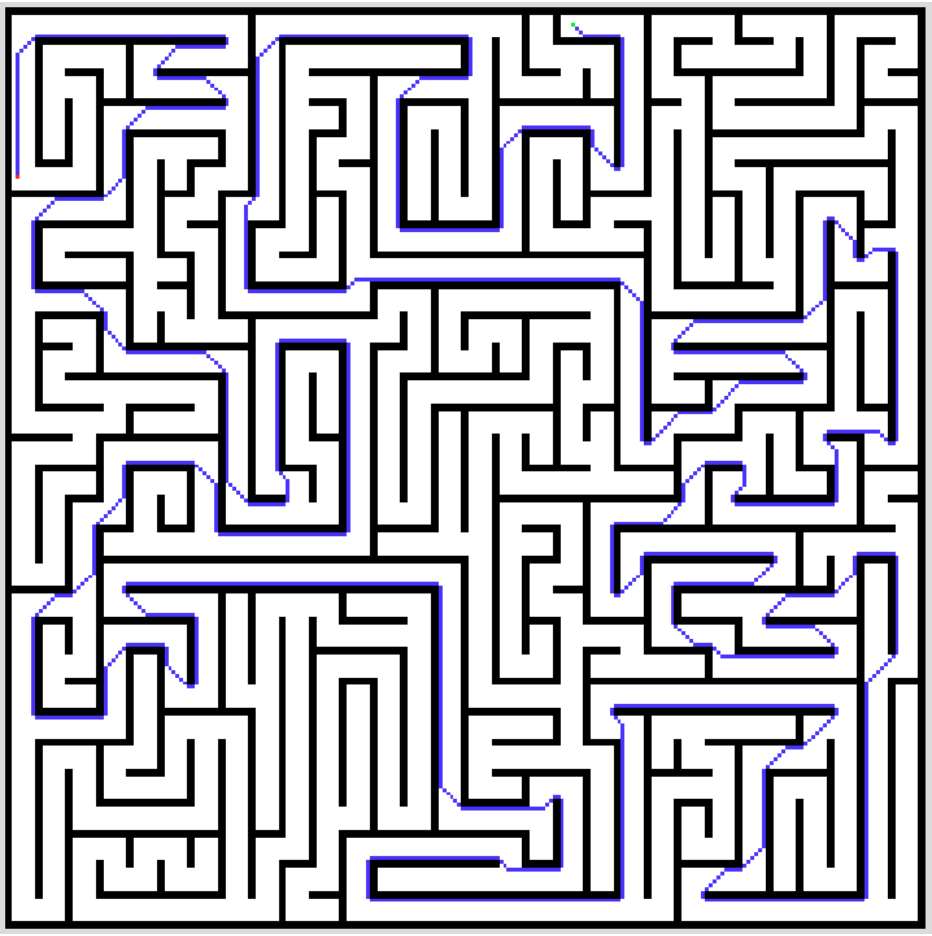
Use your own map and trajectory representation, and include these figures in your laboratory report.

4. Big maze environment

Load the map from the maze.mat file and run your brushfire and wavefront planner algorithms. Represent all results graphically. Use as start and goal positions the ones indicated in the next call:

```
[value_map, trajectory]=wavefront(map, [45, 4] , [5, 150])
```

The map and one of the optimal trajectories is:



5. Submission.

Submit a report in pdf and the following Matlab files: “brushfire.m”, “wavefront.m” and the scripts you used. Explain in detail, in the report, the work done in all the sections. Explain also the problems you found.

Optional: In case you have programmed your algorithms in an efficient way, you might want to test them in bigger environments. Test them using the files “obstaclesBig.mat” and “mazeBig.mat” and report your results.