Artificial Intelligence 2019
Problem Sheet 5

Andreas Birk

Notes

The homework serves as preparation for the exams. It is strongly recommended that you solve them before the given deadline - but you do not need to hand them in. Feel free to work on the problems as a group - this is even recommended.

1 Problem

Given the following very simple elevation map, where each cell value denotes the elevation with respect to some nominal Zero-level:

	0	1	2	3
0	0	1	1	2
1	-1	0	1	3
2	-1	0	2	4
3	0	0	1	2

Figure 1: A very simple elevation map.

Turn it into a cost map, respectively into a roadmap based on the cost. Note: there is a little bit of freedom in the choice of the cost function; just use a reasonable one.

- Can Dijkstra be used on this roadmap to find a shortest path?
- Can A^* be used on this roadmap to find a shortest path? If yes, which evaluation function $f()$ can be used for this?

2 Bonus Problem

Given the map shown in Fig.2. The value "-1" denotes "unknown", i.e., locations for which no sensor data could be gathered. Suppose a Probabilistic Roadmap (PRM) is to be created using straight-line connections through free space based on the Bresenham line drawing algorithm (Fig.3).

Suppose that an intermediate roadmap PRM_2 with the three vertices $v_0 = (1, 7)$, $v_1 = (2, 5)$, and $v_2 = (4, 3)$ plus the three edges (v_0, v_1), (v_0, v_2), and (v_1, v_2) has already been created.

- In the next step the location $v_3 = (6, 4)$ is randomly chosen. How is the PRM extended and what does PRM_3 look like?
- Afterward, the location $v_4 = (6, 7)$ is randomly chosen. How is the PRM further extended and what does PRM_4 look like?

	0	1	2	3	4	5	6	7	8	9
0	1	1	0	0	1	1	1	1	1	1
1	1	0	0	0	0	0	1	0	0	1
2	1	0	0	0	0	0	1	0	0	1
3	1	0	0	0	0	1	1	1	0	1
4	1	0	0	0	0	1	-1	1	0	1
5	1	0	0	0	0	1	1	1	0	1
6	1	0	0	0	0	0	0	0	0	1
7	1	0	0	0	0	0	0	0	0	1
8	1	0	0	1	0	0	0	0	0	1
9	1	1	1	1	1	1	1	1	1	1

Figure 2: A grid map.

```
void line(int x0, int y0, int x1, int y1) {
  int dx =  abs(x1-x0), sx = x0<x1 ? 1 : -1;
  int dy = -abs(y1-y0), sy = y0<y1 ? 1 : -1;
  int err = dx+dy, e2; /* error value e_xy */

  while(1){
    setPixel(x0,y0);
    if (x0==x1 && y0==y1) break;
    e2 = 2*err;
    if (e2 > dy) { err += dy; x0 += sx; } /* e_xy+e_x > 0 */
    if (e2 < dx) { err += dx; y0 += sy; } /* e_xy+e_y < 0 */
  }
}
```

Figure 3: C-code of the Bresenham Line Drawing algorithm.

3 Problem

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	1	0	0	0	0	0	0	1	1	1	1	1	1	1	1
1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1
2	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1
3	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1
4	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1
5	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1
6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1
7	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1
8	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1
9	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1
10	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 4: A grid map.

	-2	-1	0	1	2
-2	0	0	1	0	0
-1	0	1	1	1	0
0	1	1	1	1	0
1	1	1	1	0	0
2	0	0	1	0	0

Figure 5: A ”complex” object.

Given the map in Fig.4 and the ”complex” object shown in Fig.5 - the cells marked with ”1” are part of the object, the others are not; the origin (0,0) is the center of motion of the object.

Use the Minkowski sum to do obstacle growing, i.e., to modify the map such that the object can be treated as a point for (translational) path-planning. You only need to consider the area of the map (i.e., no regions outside of the area of the map are of interest).