CS 369: Introduction to Robotics

Prof. Thao Nguyen

Spring 2026

HAVERFORD
COLLEGE

Outline for today

- Configuration space
- Motion planning
- World representation
- Wavefront planner

Outline for today

- Configuration space
- Motion planning
- World representation
- Wavefront planner

The World consists of...

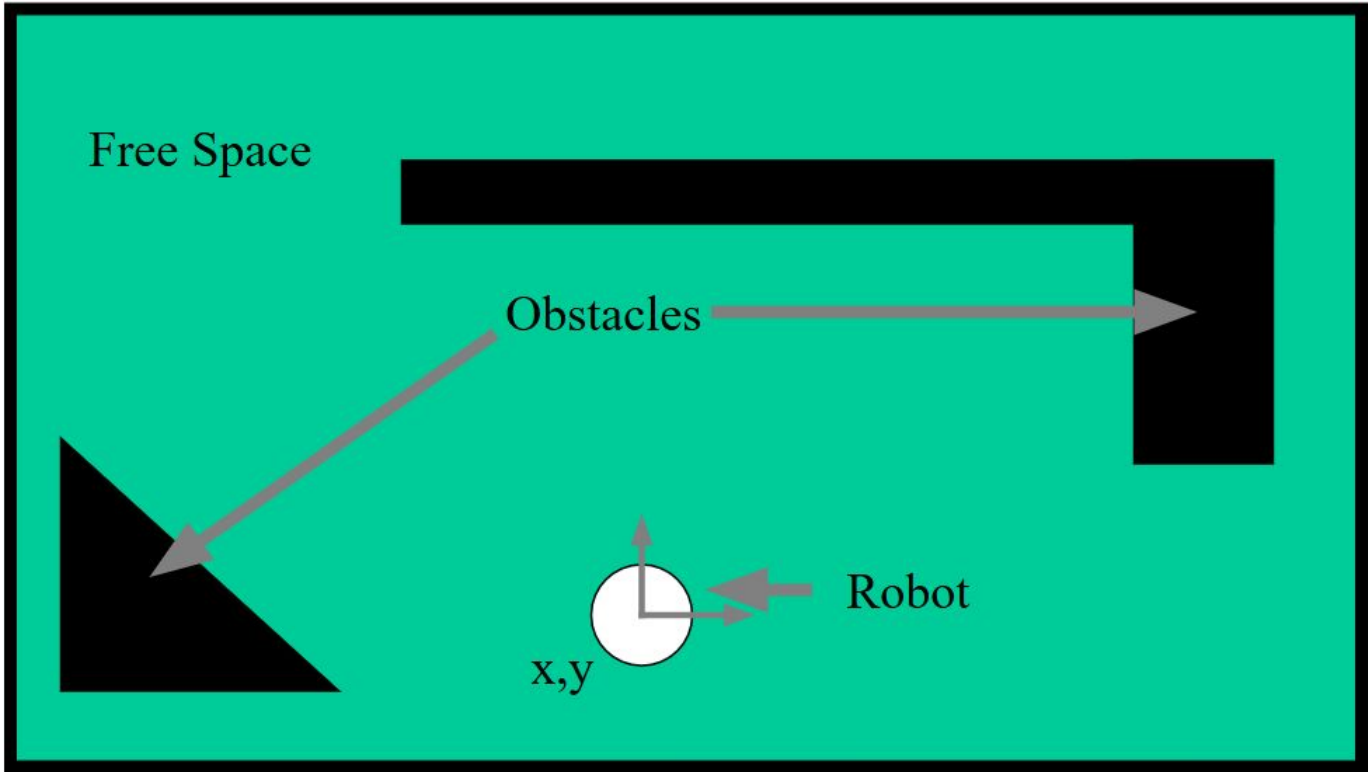
Obstacles

- Already occupied spaces of the world
- In other words, robots can’t go there

Free Space

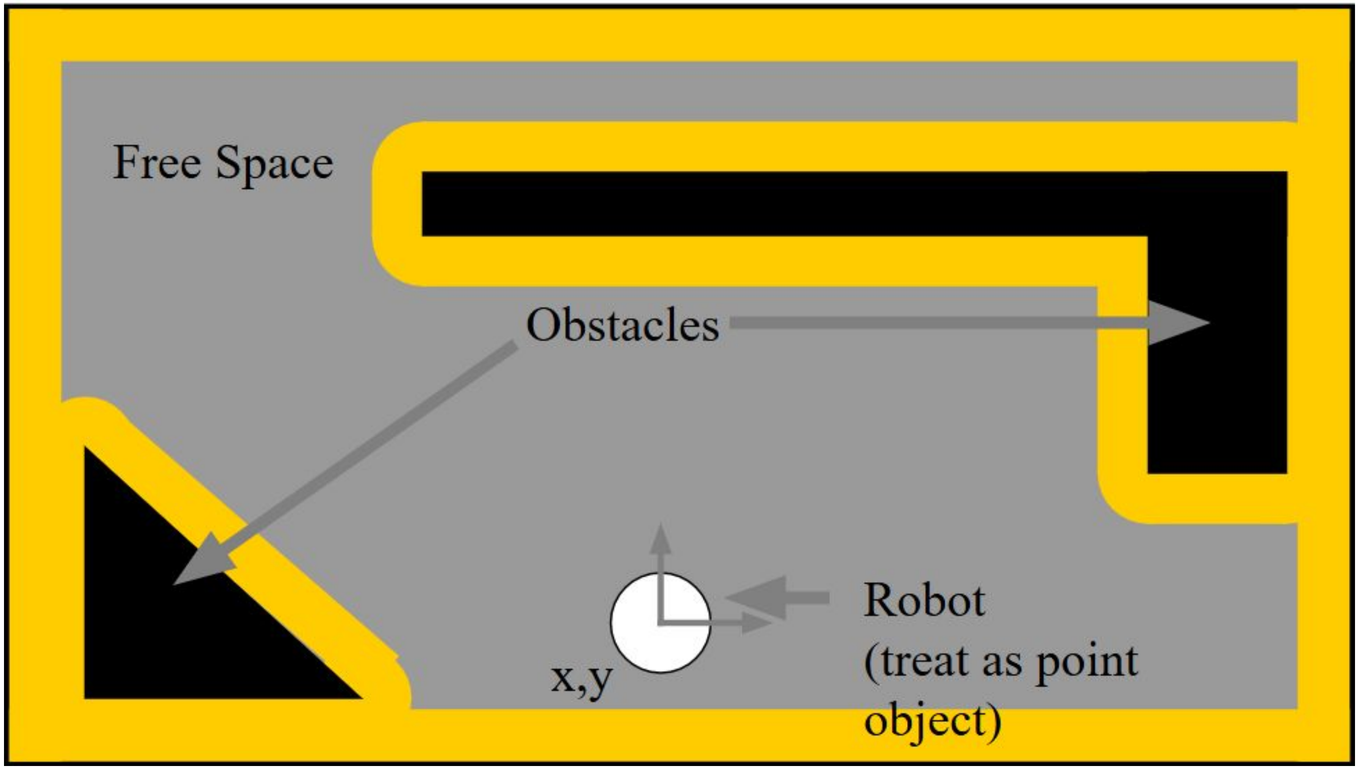
- Unoccupied space within the world
- Robots “might” be able to go here
- To determine where a robot can go, we need to discuss what a *Configuration Space* is

Example of a world (and robot)



Configuration space

accommodate robot size



Configuration space

What it is

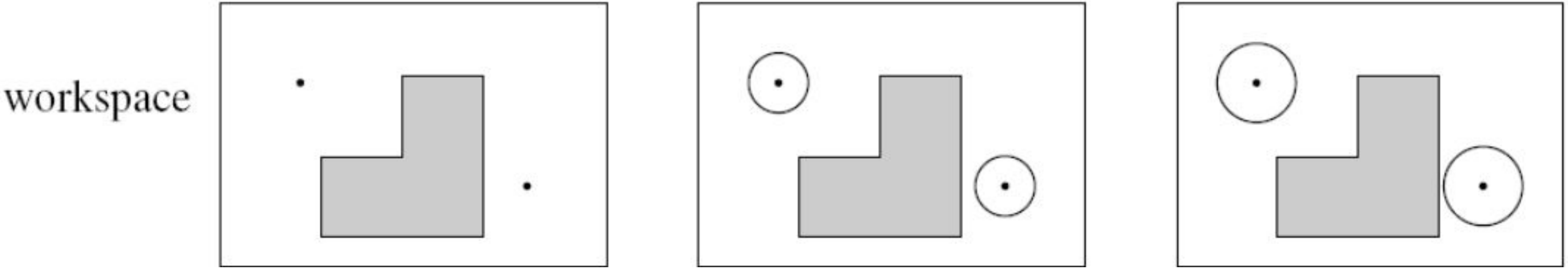
- A set of "reachable" areas constructed from knowledge of both the robot and the world

How to create it

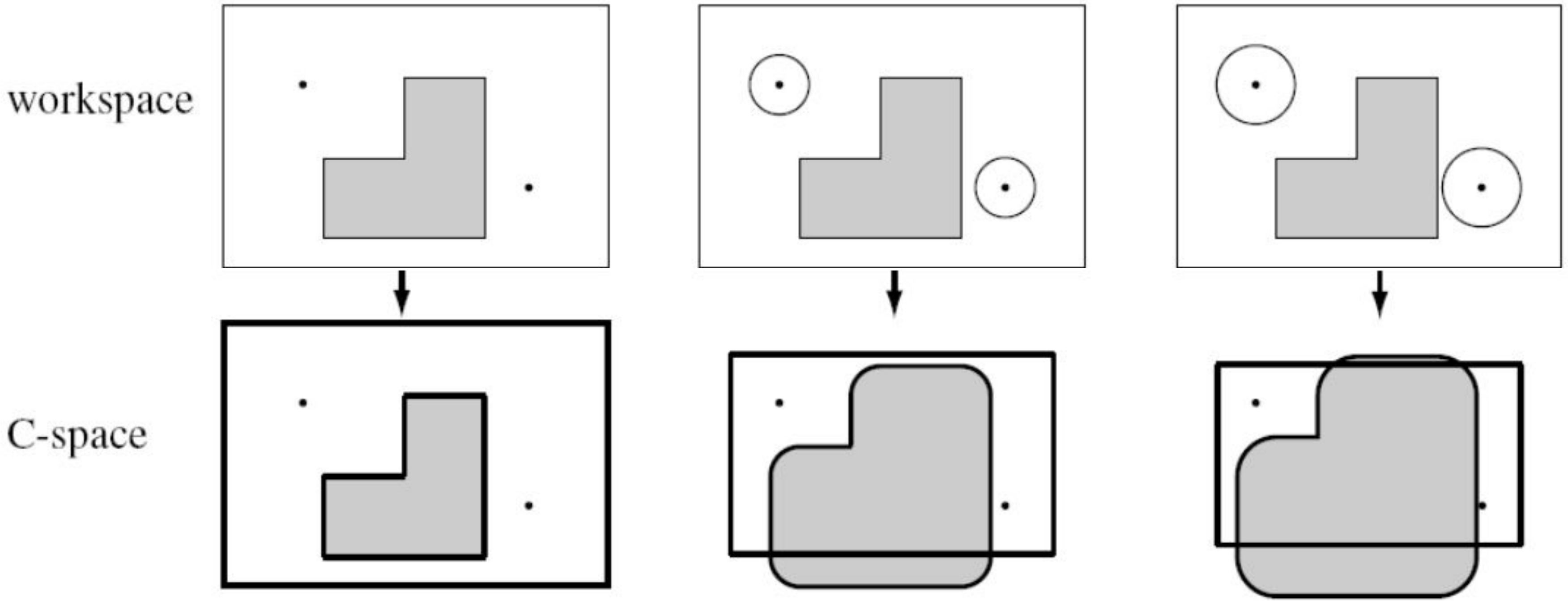
- First abstract the robot as a point object. Then, enlarge the obstacles to account for the robot's footprint and degrees of freedom
- In our example, the robot was circular, so we simply enlarged our obstacles by the robot's radius (*note the curved vertices*)

THE ROBOTICS INSTITUTE

Practice



Practice



Outline for today

- Configuration space
- Motion planning
- World representation
- Wavefront planner

Motion planning

- Process of finding a feasible path from the start to goal configuration

- If $\mathbf{C_{free}}$ denotes the robot's configuration space,

  then a path $\mathbf{p = (c_0, ..., c_n)}$ where $\mathbf{c \in C_{free}}$, $\mathbf{c_0}$ is $\mathbf{q_{start}}$ and $\mathbf{c_n}$ is $\mathbf{q_{goal}}$

Evaluation

- Path feasibility
- Distance from obstacles
- Path length
- Planning time

Distance metrics

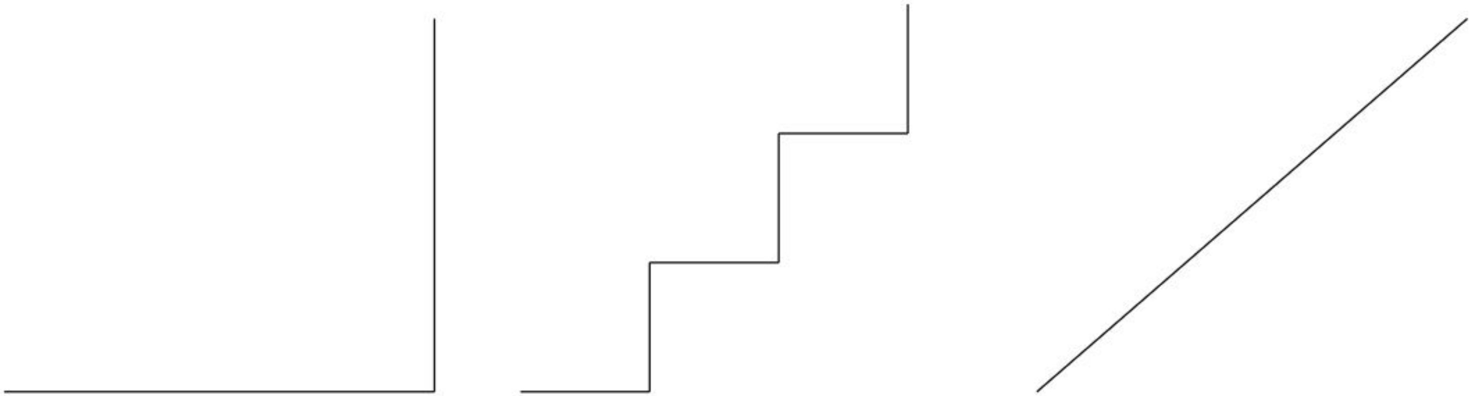
$$d : R^2 \times R^2 \to R$$

L1 Metric $d(a,b) = |a_x - b_x| + |a_y - b_y|$

L2 Metric $d(a,b) = \sqrt{(a_x - b_x)^2 + (a_y - b_y)^2}$

Path length

Which is shortest?



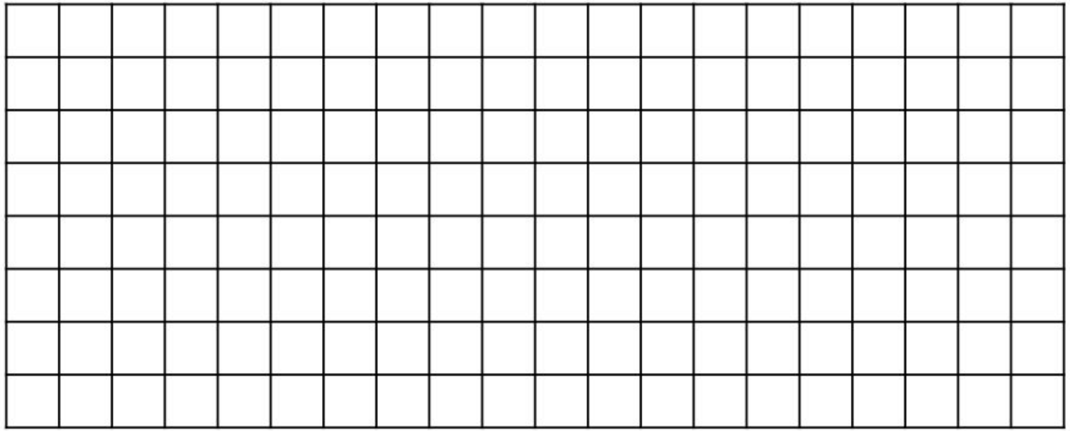
depends on the metric

Outline for today

- Configuration space
- Motion planning
- World representation
- Wavefront planner

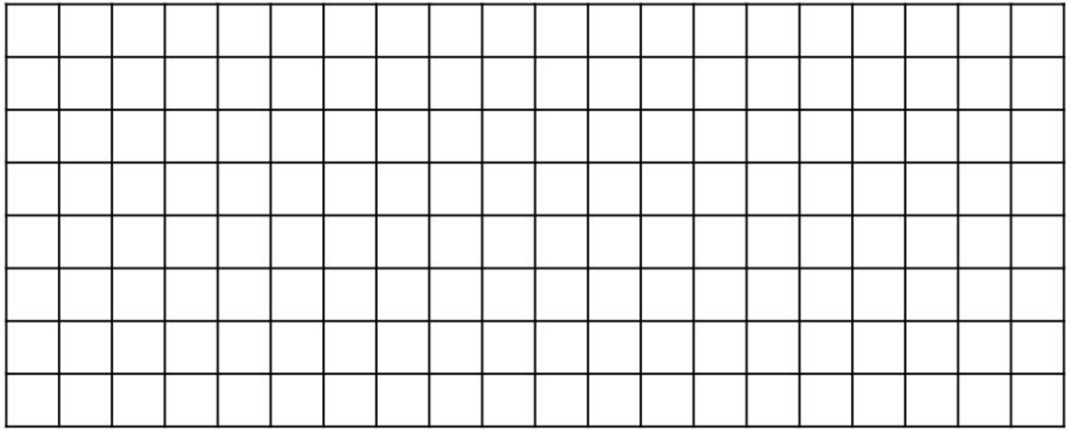
World representation

- Can always use a continuous representation
- For simplicity, use uniform-sized grid cells



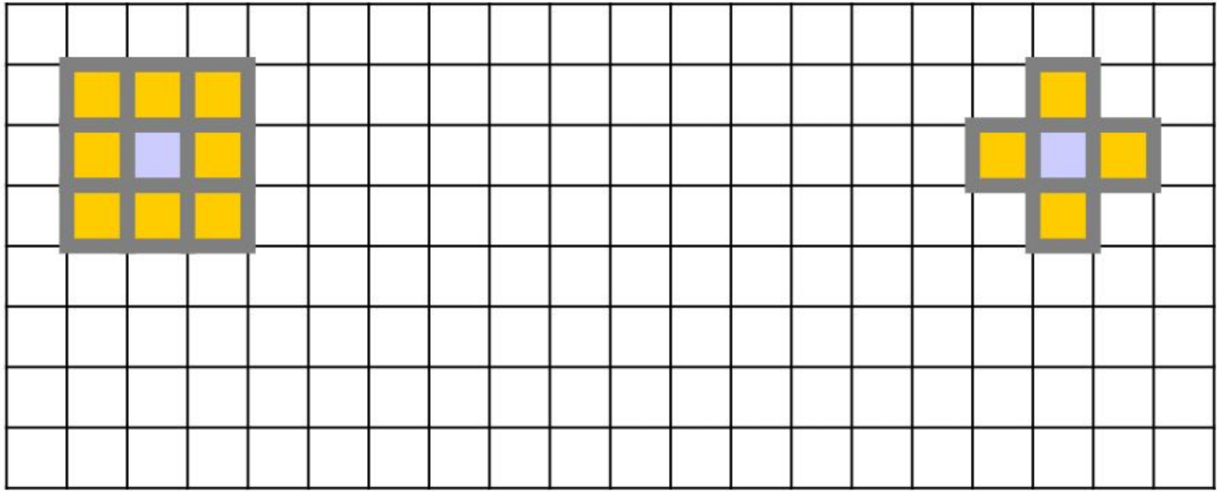
Grid representation

- Distance is reduced to discrete steps
  - what does it mean for the robot to be inside a grid?
- Direction is now limited from one adjacent cell to another



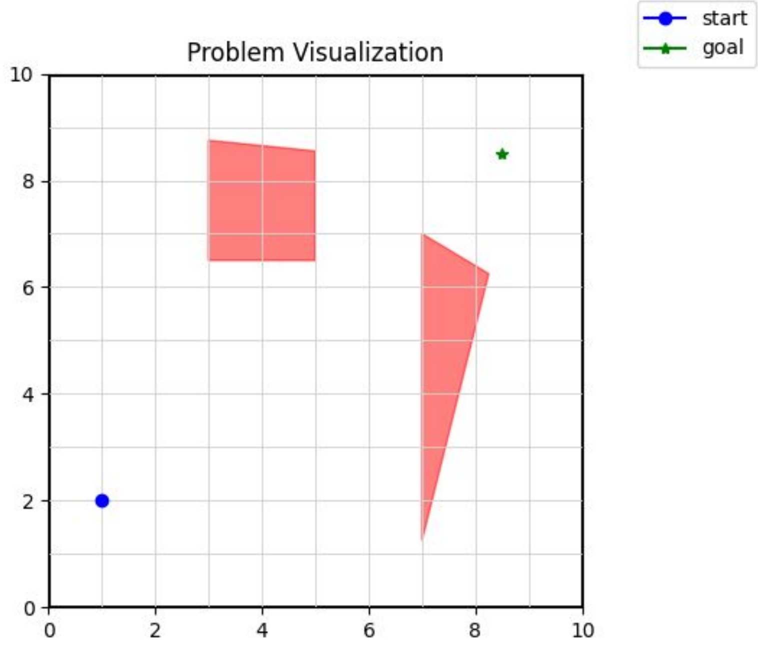
Connectivity

- 8-Point Connectivity
- 4-Point Connectivity
  - *(approximation of the L1 metric)*



Occupancy grid

- A grid where each cell encodes occupancy information
  - Binary: a cell is either occupied (1) or free (0)
  - Probabilistic
- How to determine occupancy?
  - Optimistic
  - Pessimistic



Outline for today

- Configuration space
- Motion planning
- World representation
- Wavefront planner

Wavefront planner

- Common algorithm used to determine the shortest path between two points
- Setup:
  - Label free space with 0
  - Label obstacles with 1
  - Label start as START
  - Label the goal as 2

7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0
3	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Wavefront planner

Starting with the goal, set all adjacent cells with “0” to the current cell value + 1

- this example uses 8-point connectivity

7	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	0	0	0	0
3	0	0	0	0	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	3
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	**2**
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Wavefront planner

Repeat with the modified cells

7	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	0	0	0	0
3	0	0	0	0	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	4	4	4
1	0	0	0	0	0	0	0	0	0	0	0	0	0	4	3	3
0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	3	**2**
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Wavefront planner

Repeat again

7	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	0	0	0	0
3	0	0	0	0	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	5	5	5	5
2	0	0	0	0	0	0	0	0	0	0	0	0	5	4	4	4
1	0	0	0	0	0	0	0	0	0	0	0	0	5	4	3	3
0	0	0	0	0	0	0	0	0	0	0	0	0	5	4	3	**2**
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

And again

7	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	6	6	6	6
3	0	0	0	0	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	5	5	5	5
2	0	0	0	0	0	0	0	0	0	0	0	6	5	4	4	4
1	0	0	0	0	0	0	0	0	0	0	0	6	5	4	3	3
0	0	0	0	0	0	0	0	0	0	0	0	6	5	4	3	**2**
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Wavefront planner

Repeat until no 0’s are adjacent to cells with values >= 2

- 0’s will only remain if unreachable regions exist

7	**18**	17	16	15	14	13	12	11	10	9	9	9	9	9	9	9
6	17	17	16	15	14	13	12	11	10	9	8	8	8	8	8	8
5	17	16	16	15	14	13	12	11	10	9	8	7	7	7	7	7
4	17	16	15	15	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	6	6	6	6
3	17	16	15	14	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	5	5	5	5
2	17	16	15	14	13	12	11	10	9	8	7	6	5	4	4	4
1	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	3
0	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	**2**
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Wavefront planner

To find the shortest path, simply always move toward a cell with a lower number (except value 1)

- The numbers generated by the Wavefront planner are roughly proportional to their distance from the goal

Two possible shortest paths shown