The Workspace

Algorithms and Data Structures 2 – Motion Planning and its applications

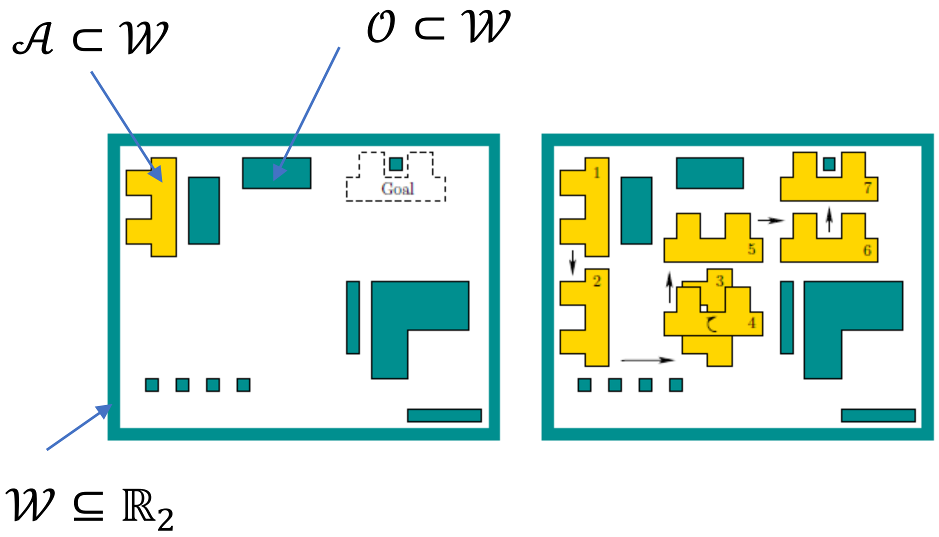
University of Applied Sciences Stuttgart

Dr. Daniel Schneider

What is the workspace?

Definition: Two-dimensional Workspace

Let $\mathcal{W}$ denote the world, which contains a robot and obstacles, both defined by polygons. Let $\mathcal{W} \subseteq \mathbb{R}_2$ denote the set of all obstacles as $\mathcal{O} \subset \mathcal{W}$ and call it forbidden region. Moreover, define $\mathcal{A} \subset \mathcal{W}$ as a robot.



Sources:
Motion Planning: The Essentials – LaValle – http://msl.cs.illinois.edu/~lavalle/papers/Lav11b.pdf

Some Notes on workspace

- The presented workspace definitions are definitions of a continuous workspace. ($\mathcal{W} \subseteq \mathbb{R}_2$).
- In past lectures you have probably already covered some discrete workspaces. Recap will follow.
- In the lecture we will work on **continuous workspaces**.

Important:

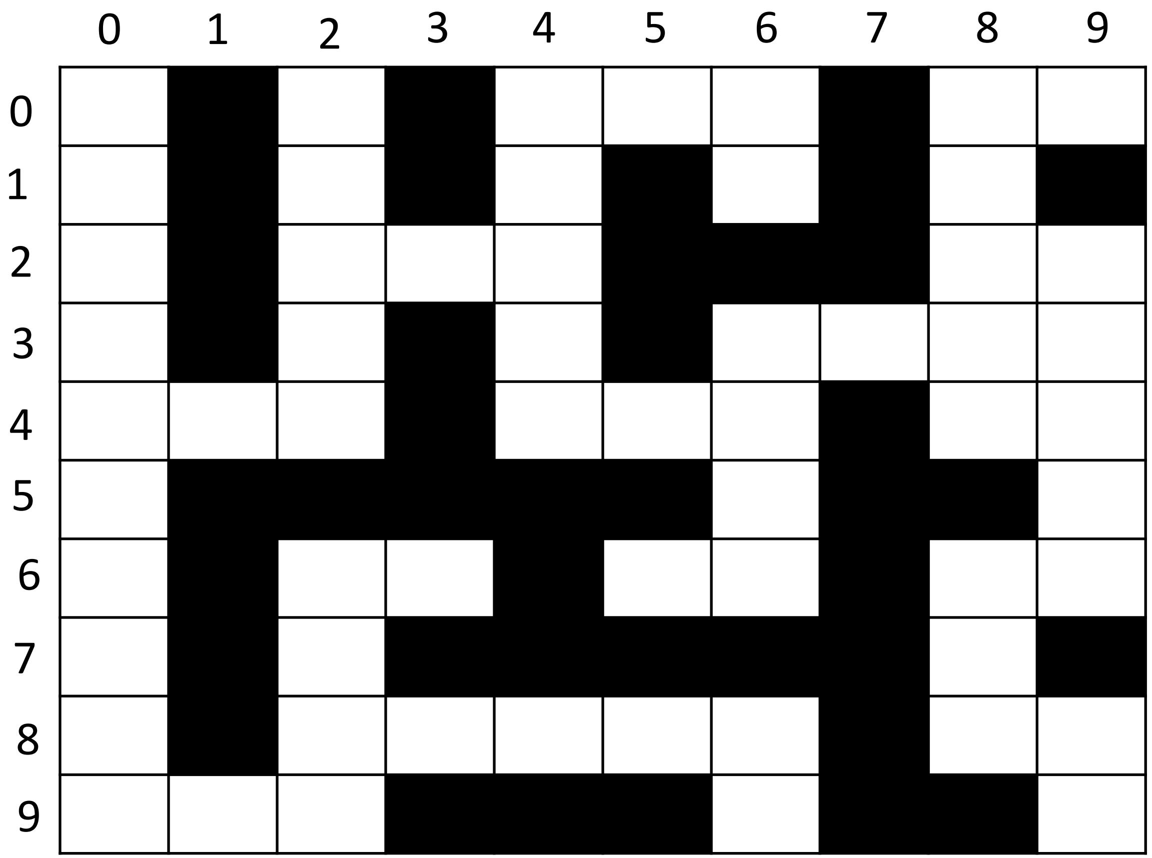
Don’t get confused on the practical work study. There we will use bmp files as input. BMP files are **discrete workspaces** but we assume they are **continuous**. We use BMP files to make you the programming easier for you.

In detail: Collision detection is easy and involves less math and geometry knowledge.

What is a discrete workspace?

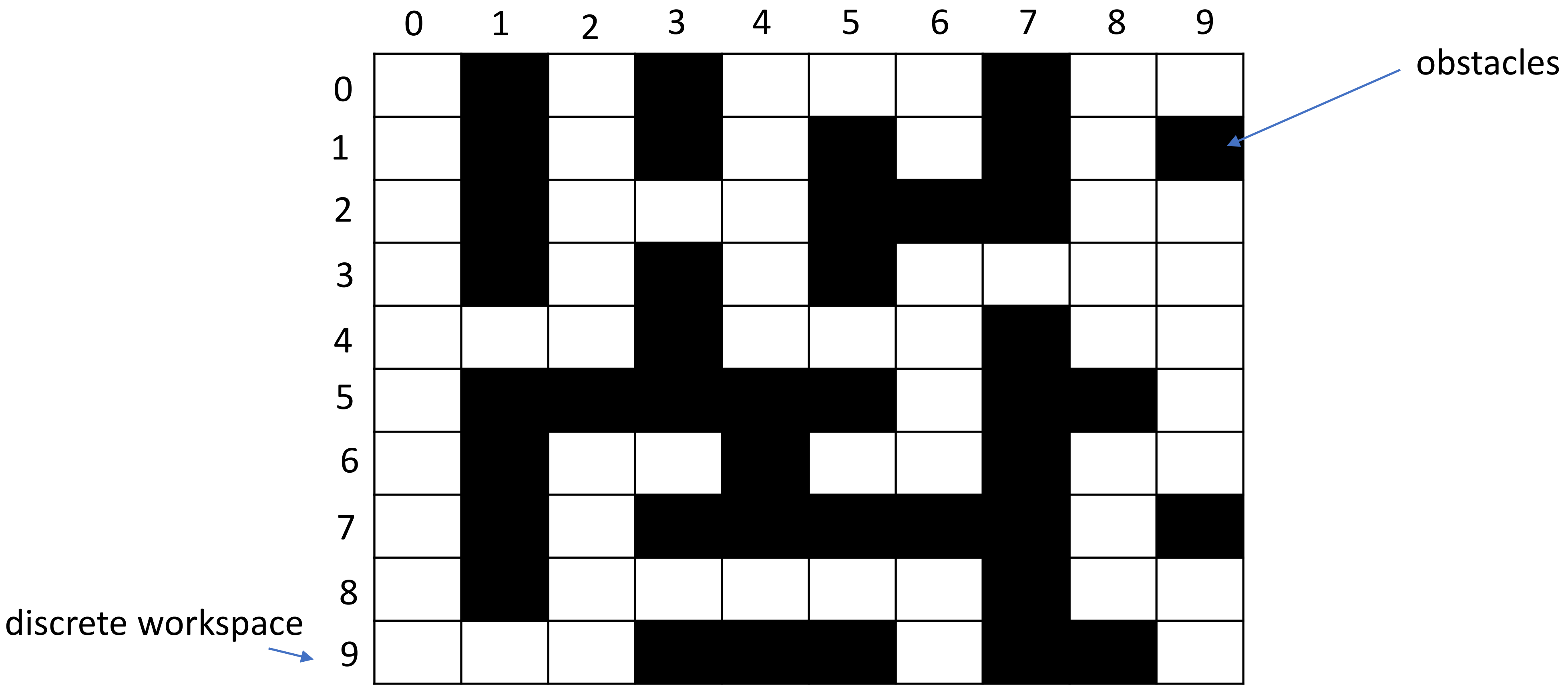
	0	1	2	3	4	5	6	7	8	9
0										
1										
2										
3										
4										
5										
6										
7										
8										
9										

What is a discrete workspace?

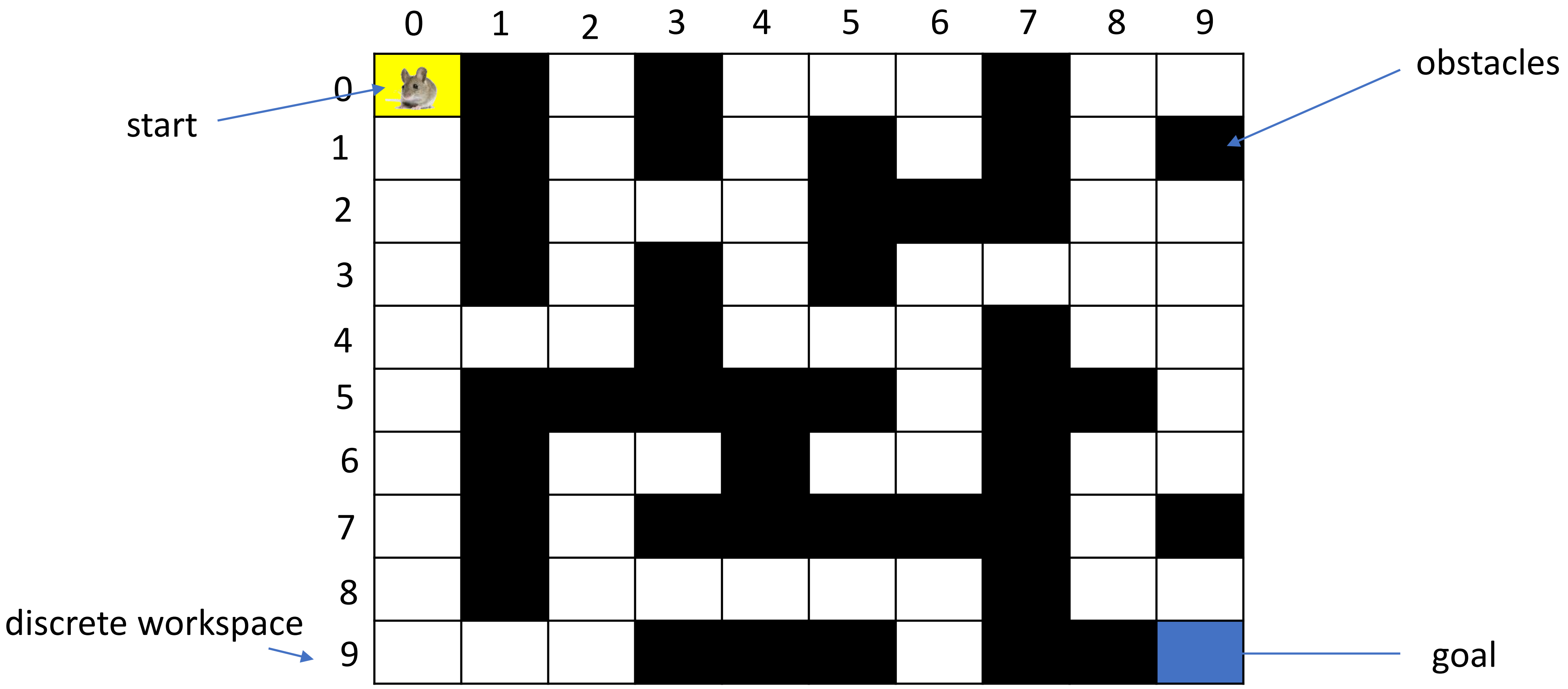


discrete workspace

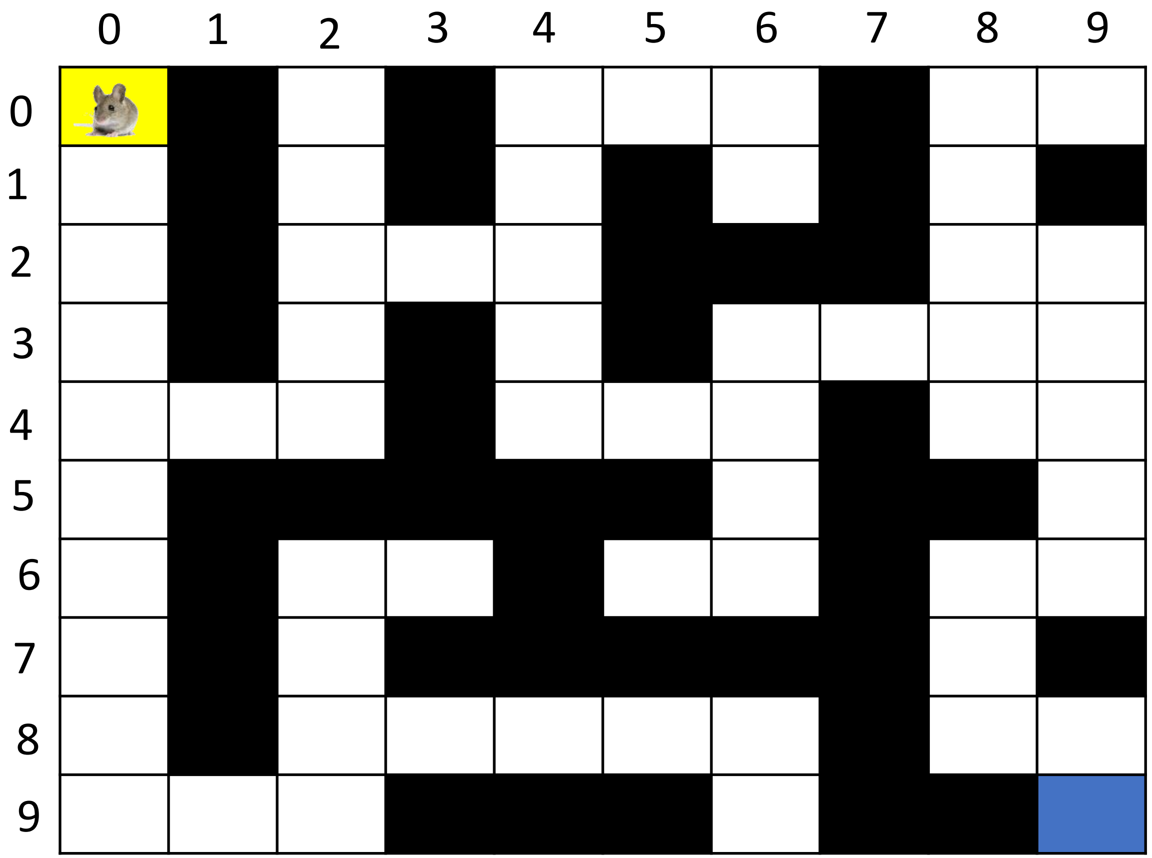
What is a discrete workspace?



What is a discrete workspace?

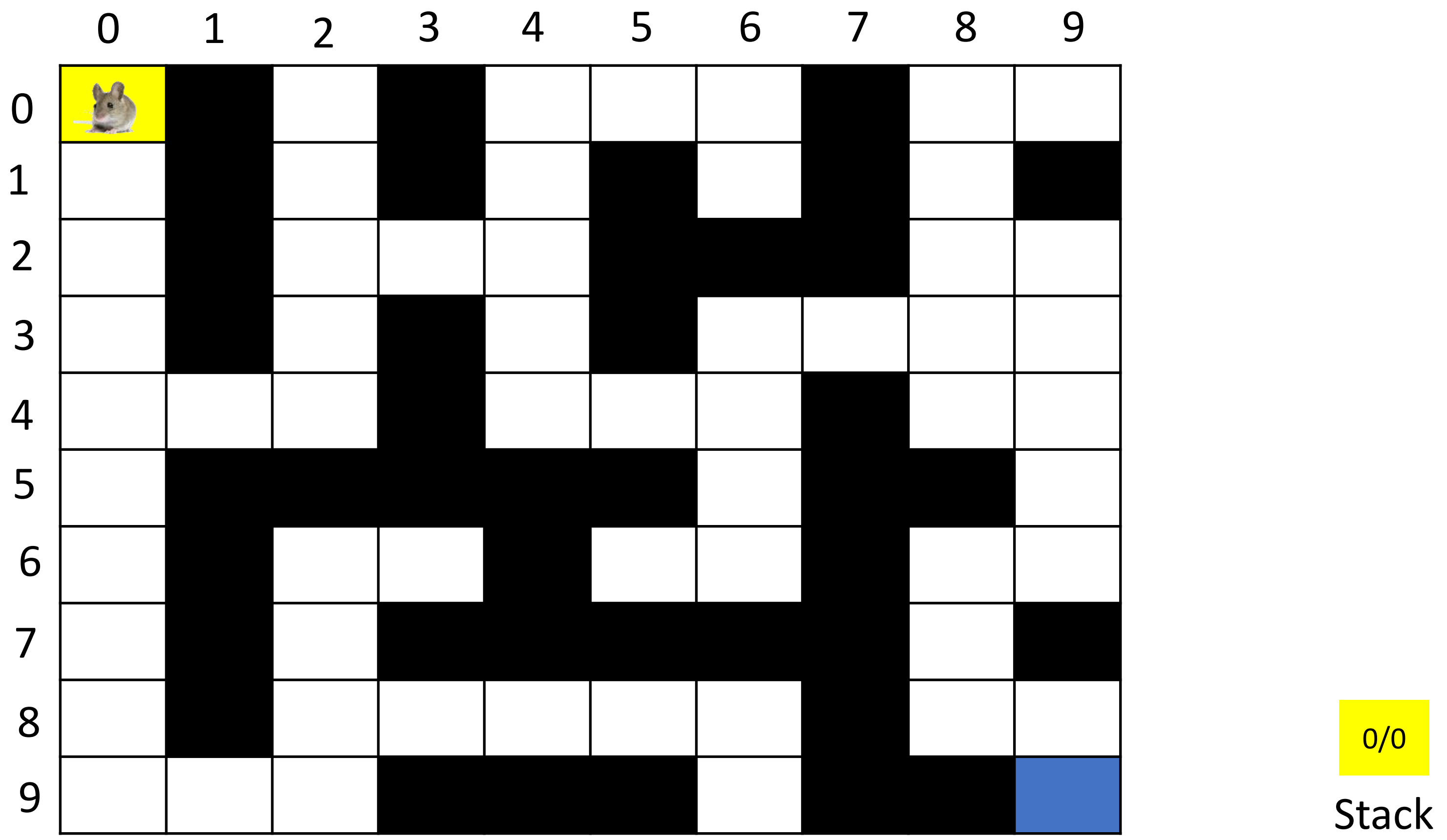


What is a discrete workspace?

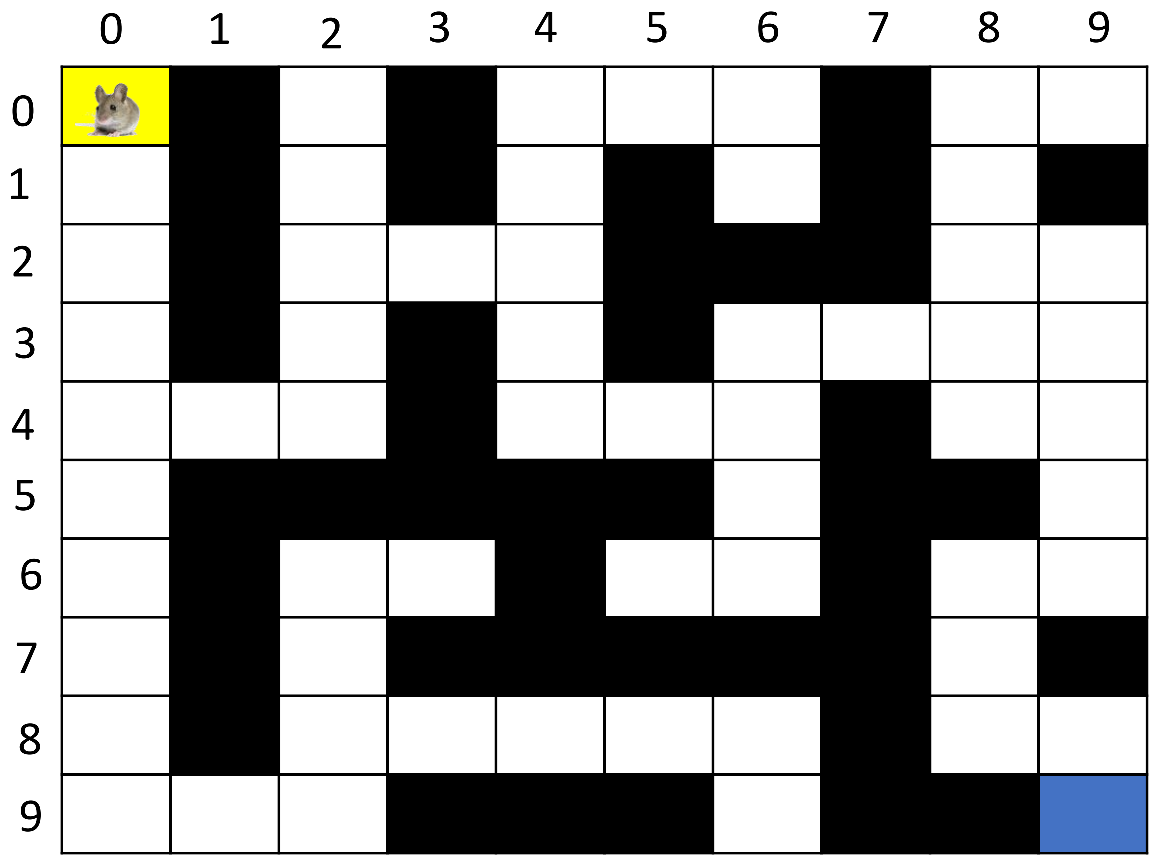


Do you remember what kind of algorithm was used for solving this problem?

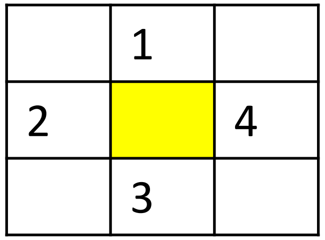
Recap on the backtracking algorithm.



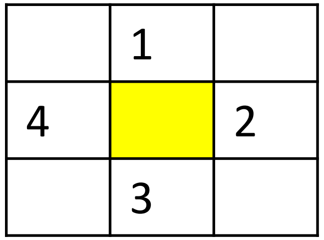
Recap on the backtracking algorithm.



North/West/South/East-Rule



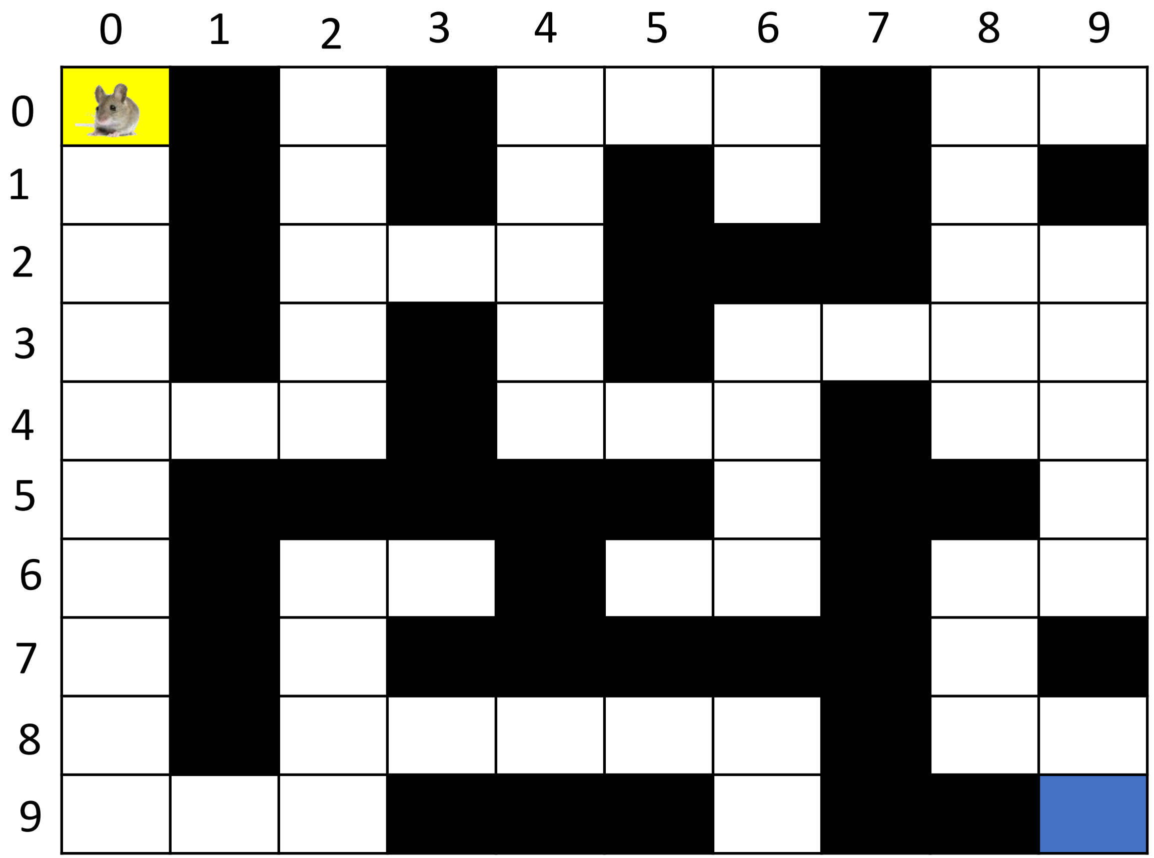
North/East/South/West-Rule



0/0

Stack

Recap on the backtracking algorithm.

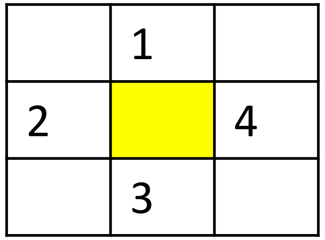


Mouse Goal No way visited

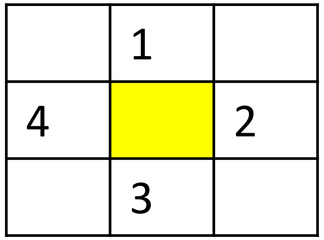
0/0

Stack

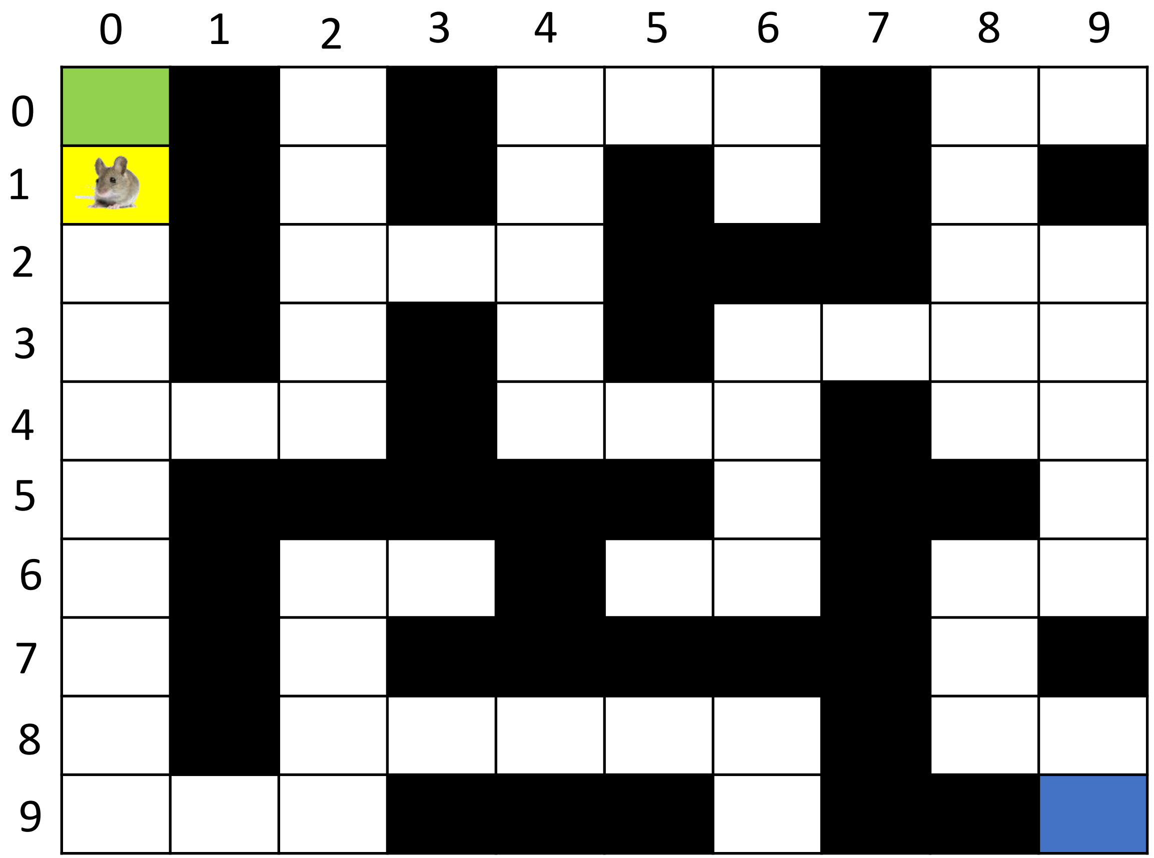
North/West/South/East-Rule



North/East/South/West-Rule

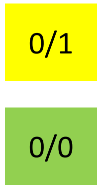


Recap on the backtracking algorithm.



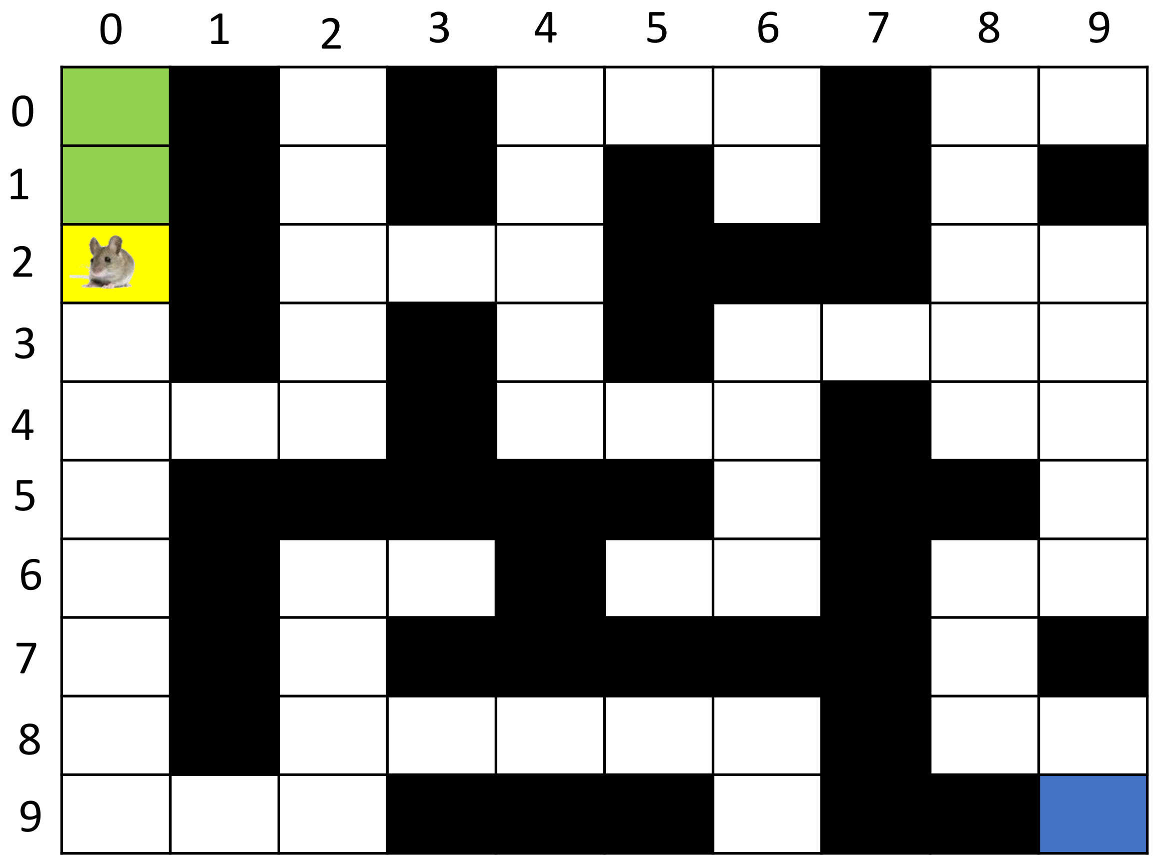
Mouse Goal No way visited

Check what is free with
North/West/South/East-Rule



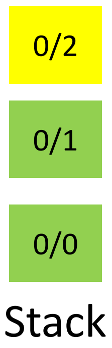
Stack

Recap on the backtracking algorithm.

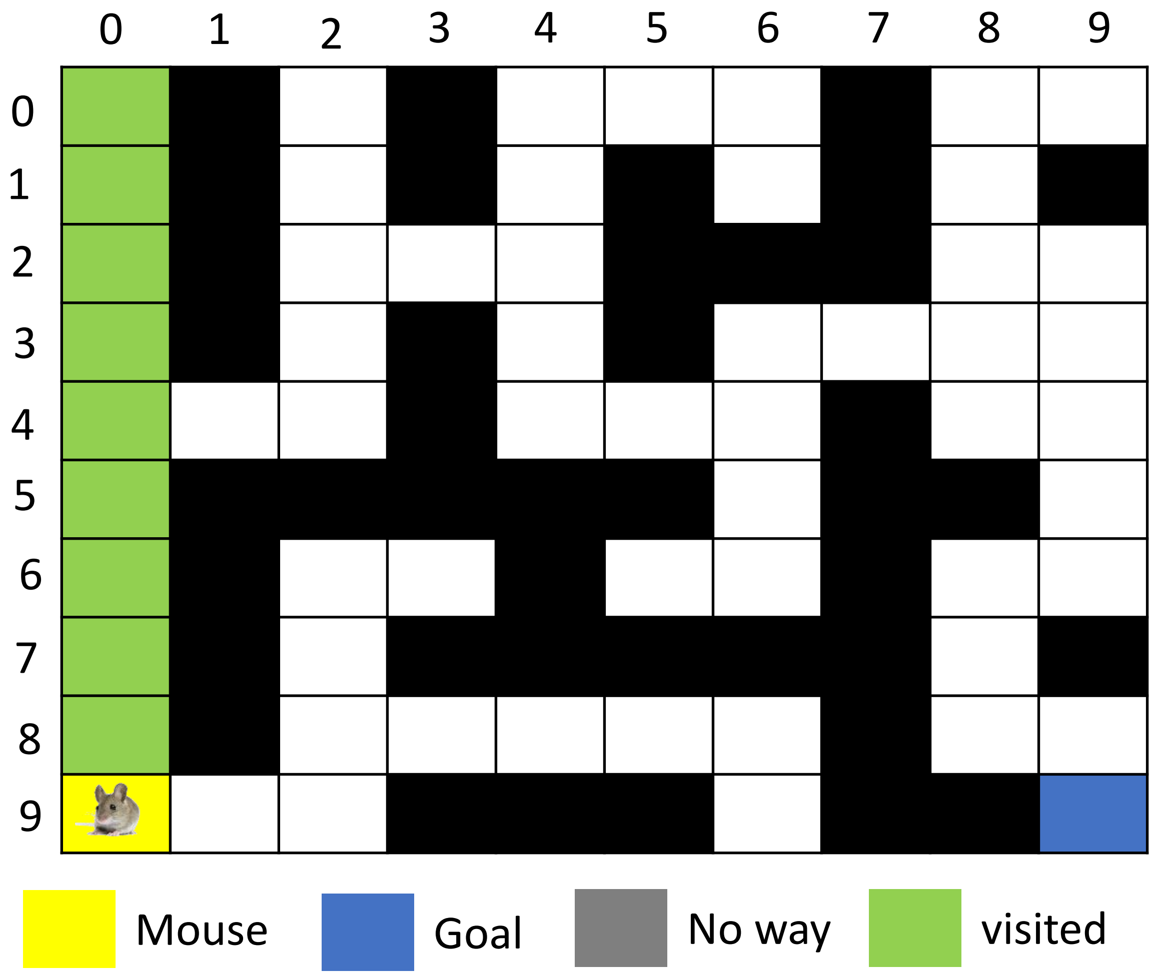


Mouse Goal No way visited

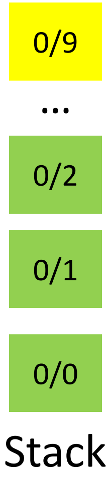
Check what is free with
North/West/South/East-Rule



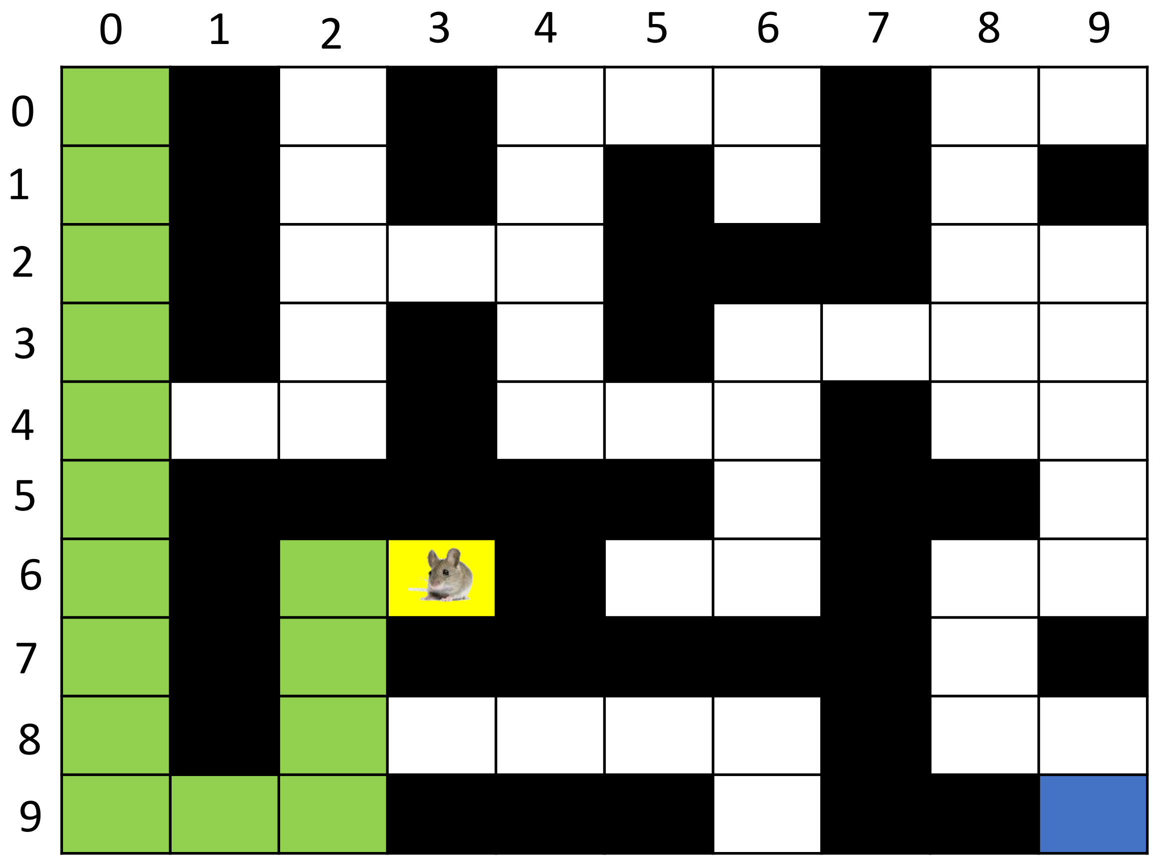
Recap on the backtracking algorithm.



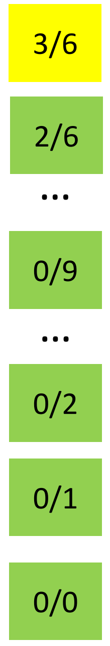
Check what is free with
North/West/South/East-Rule



Recap on the backtracking algorithm.



Mouse Goal No way visited

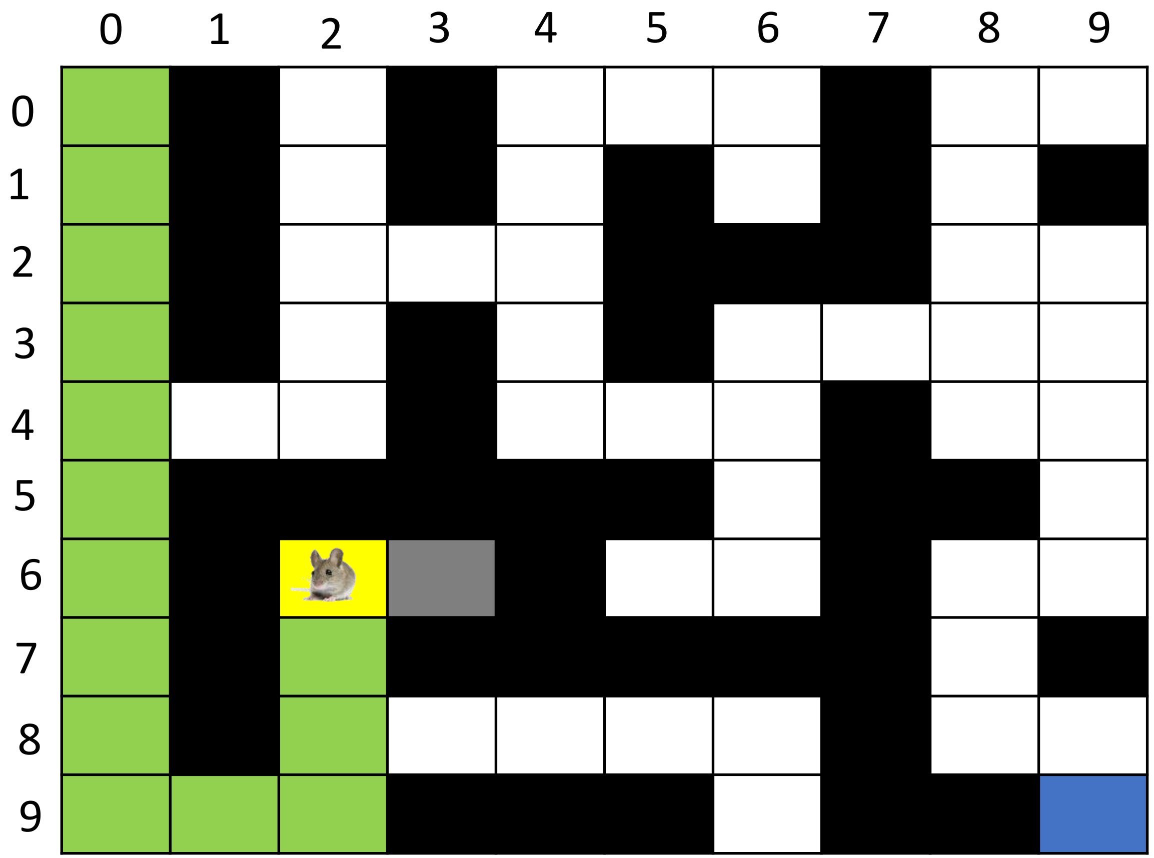


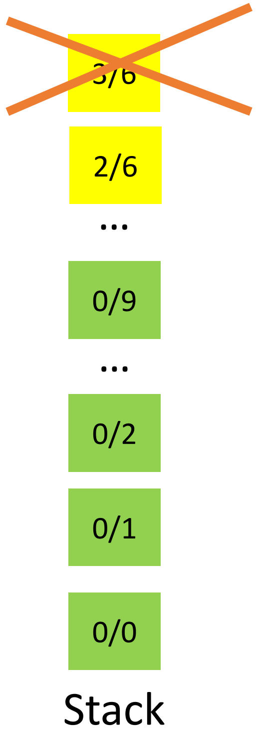
Stack

Check what is free with
North/West/South/East-Rule

No more white field!

Recap on the backtracking algorithm.

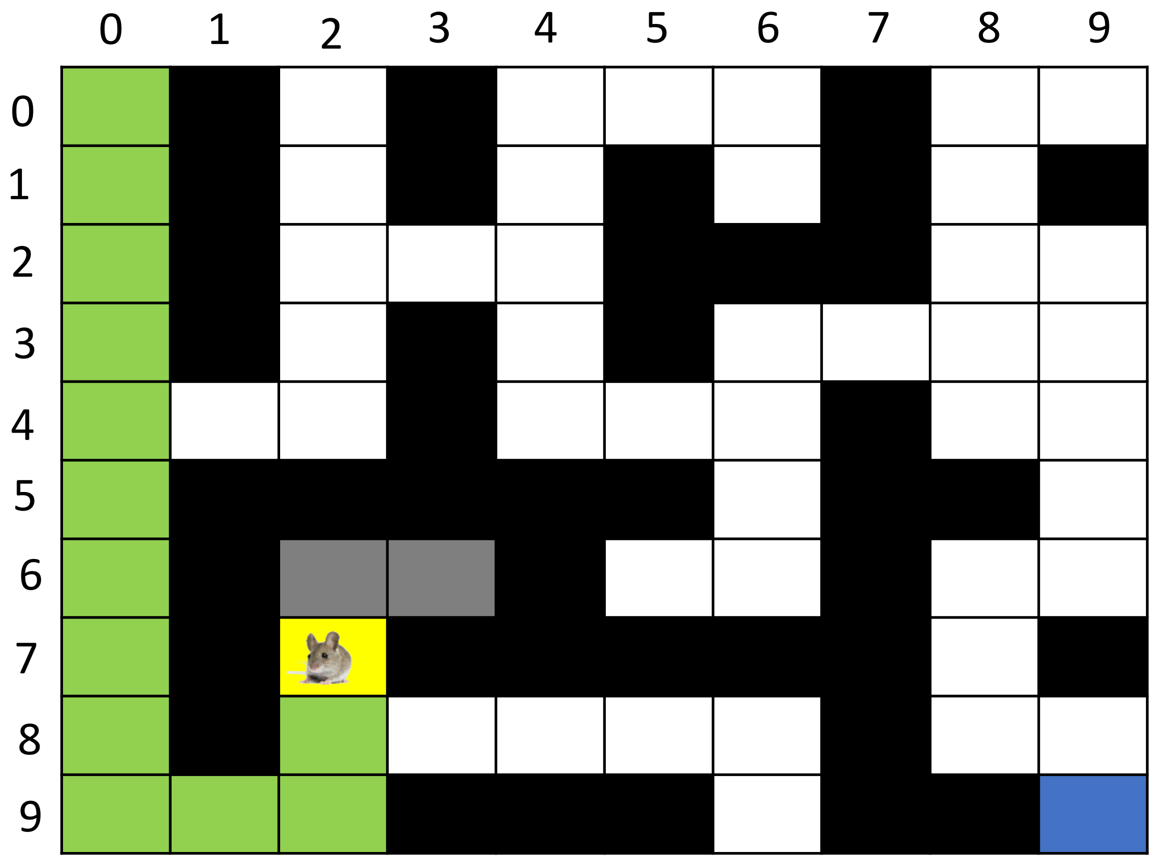




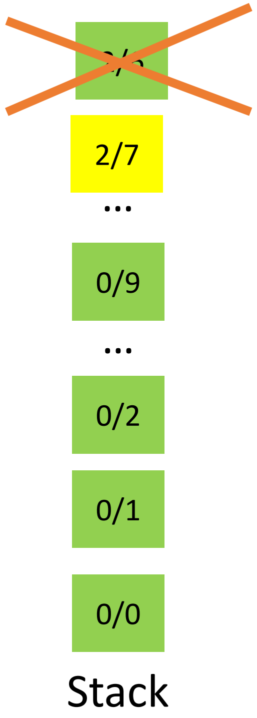
Check what is free with
North/West/South/East-Rule

No more white field!0
→ mark as grey (no way)
→Remove from the stack

Recap on the backtracking algorithm.



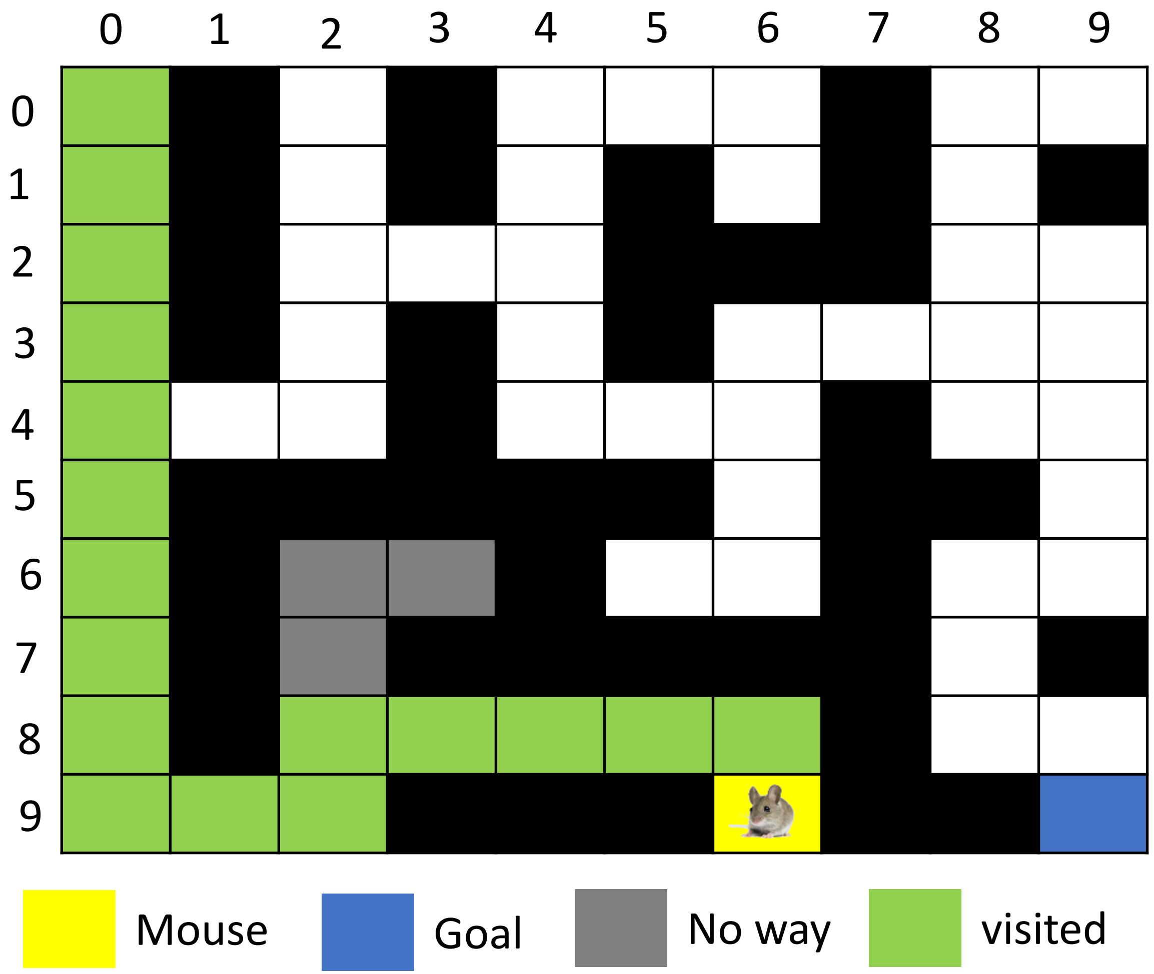
Mouse Goal No way visited

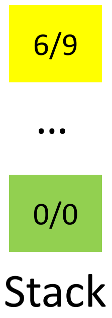


Check what is free with
North/West/South/East-Rule

No more white field!0
→ mark as grey (no way)
→Remove from the stack

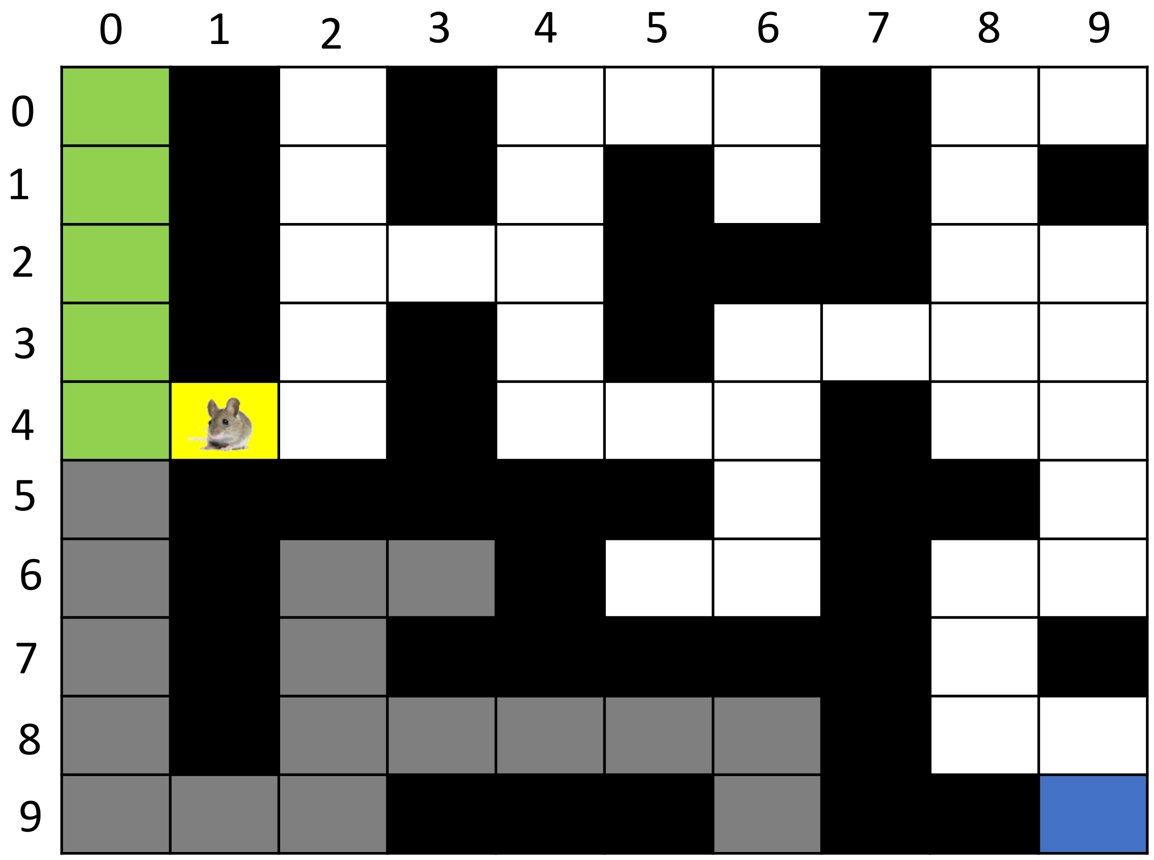
Recap on the backtracking algorithm.



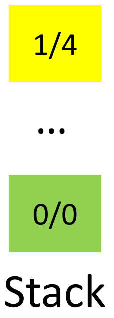


Check what is free with
North/West/South/East-Rule

Recap on the backtracking algorithm.

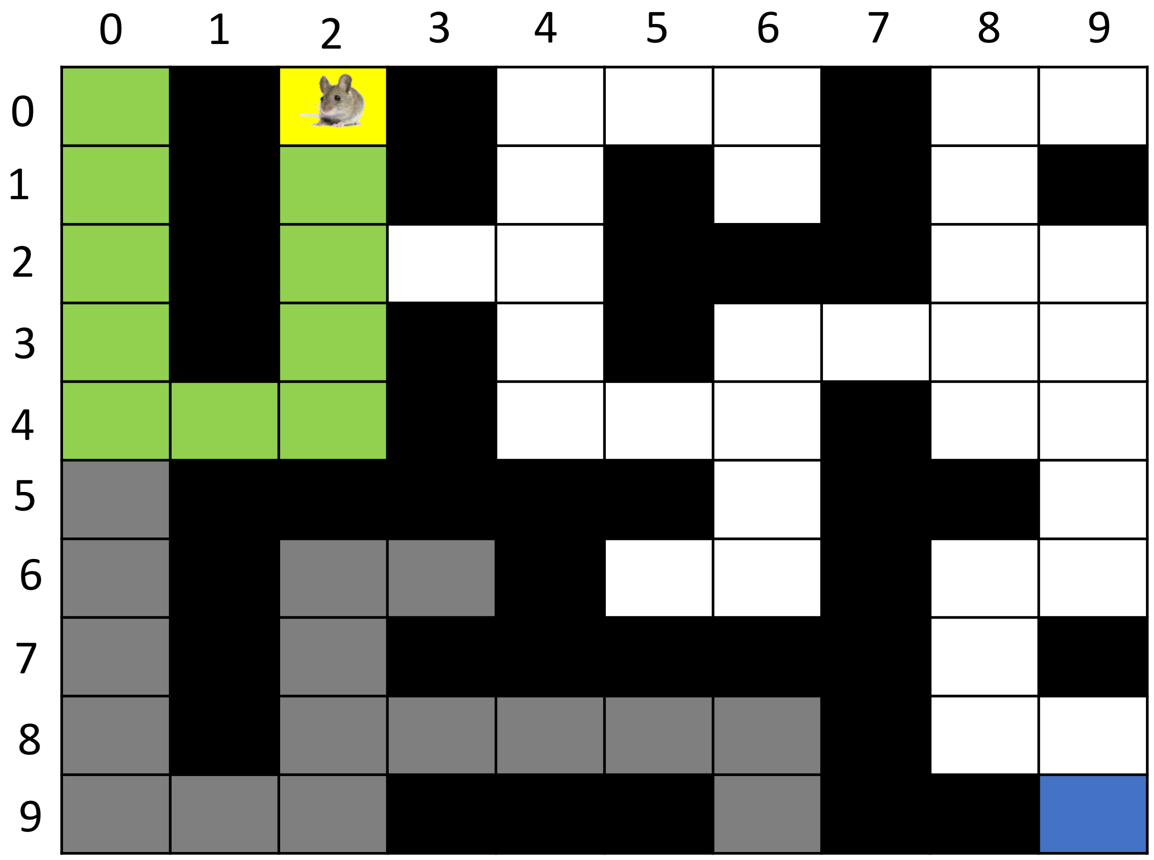


Mouse Goal No way visited



Check what is free with
North/West/South/East-Rule

Recap on the backtracking algorithm.

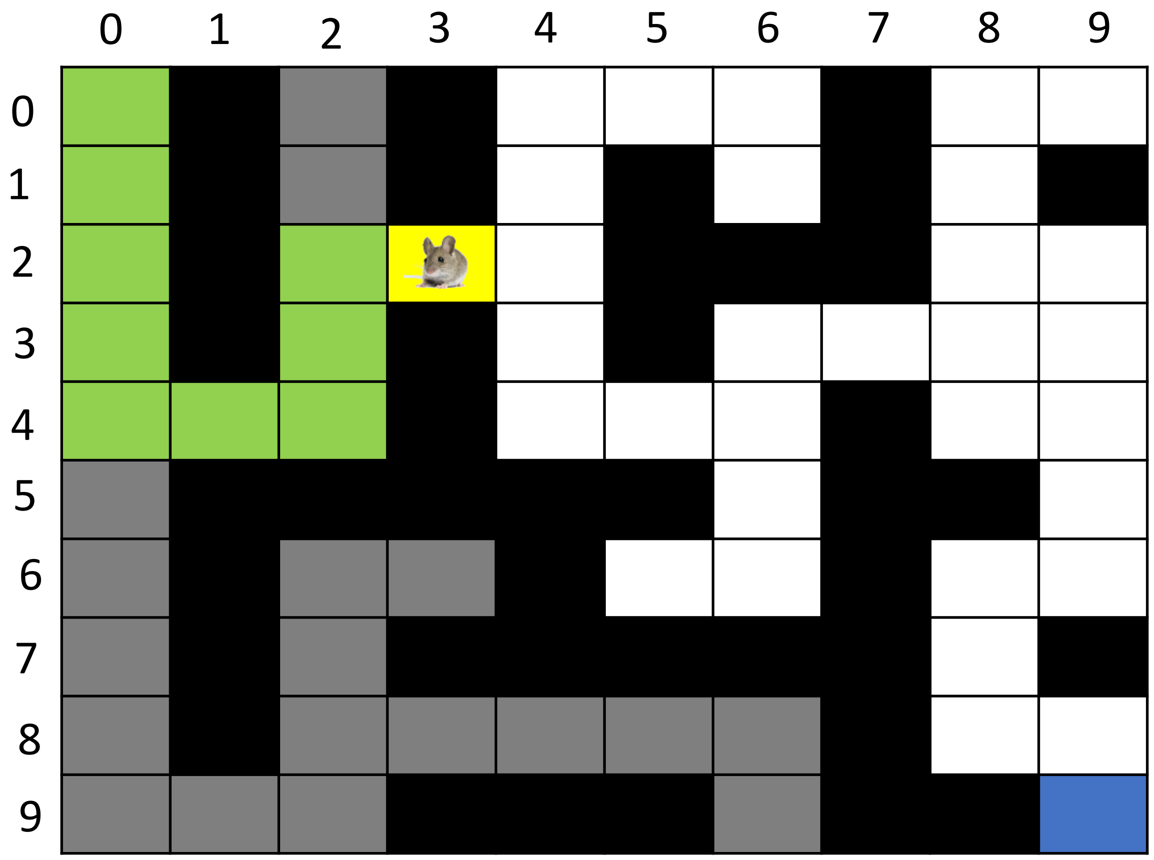


Mouse Goal No way visited

Check what is free with
North/West/South/East-Rule

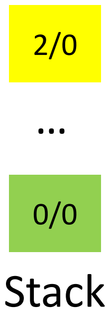
2/0
...
0/0
Stack

Recap on the backtracking algorithm.

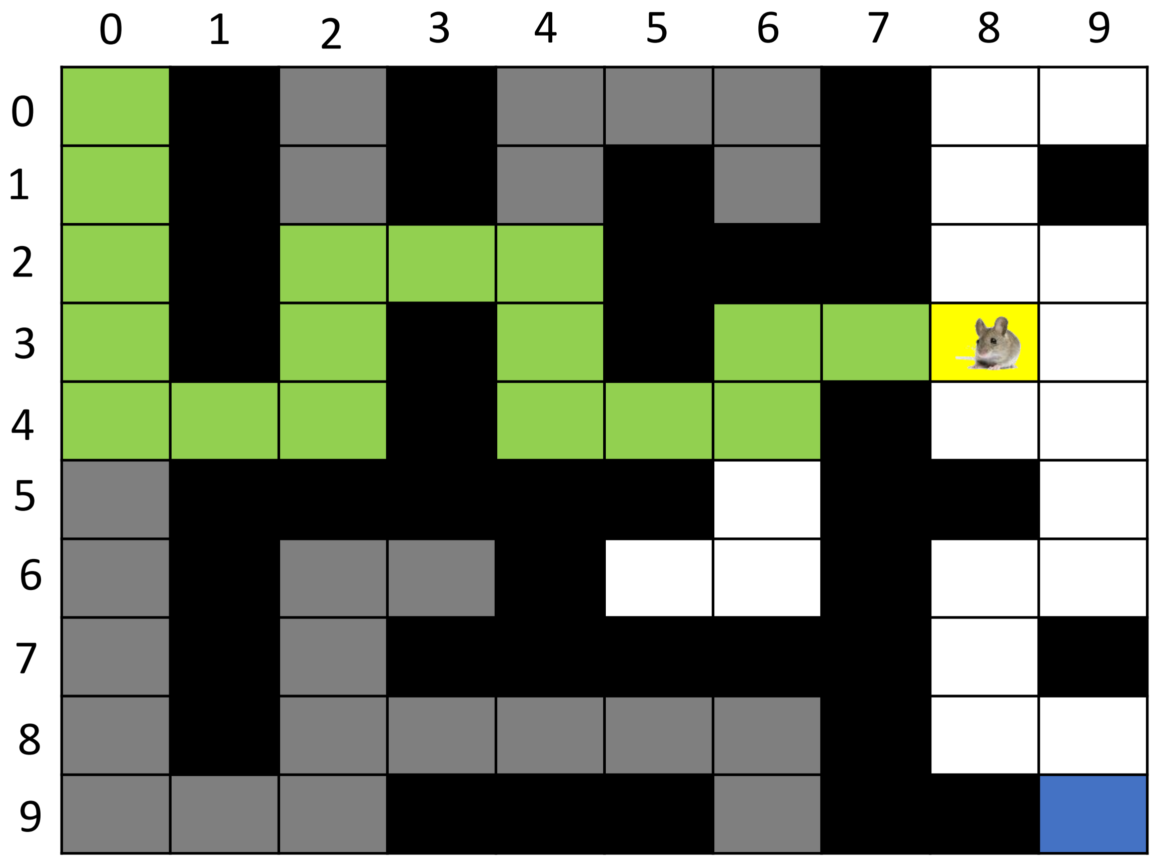


Mouse Goal No way visited

Check what is free with
North/West/South/East-Rule



Recap on the backtracking algorithm.



Mouse Goal No way visited

Check what is free with
North/West/South/East-Rule

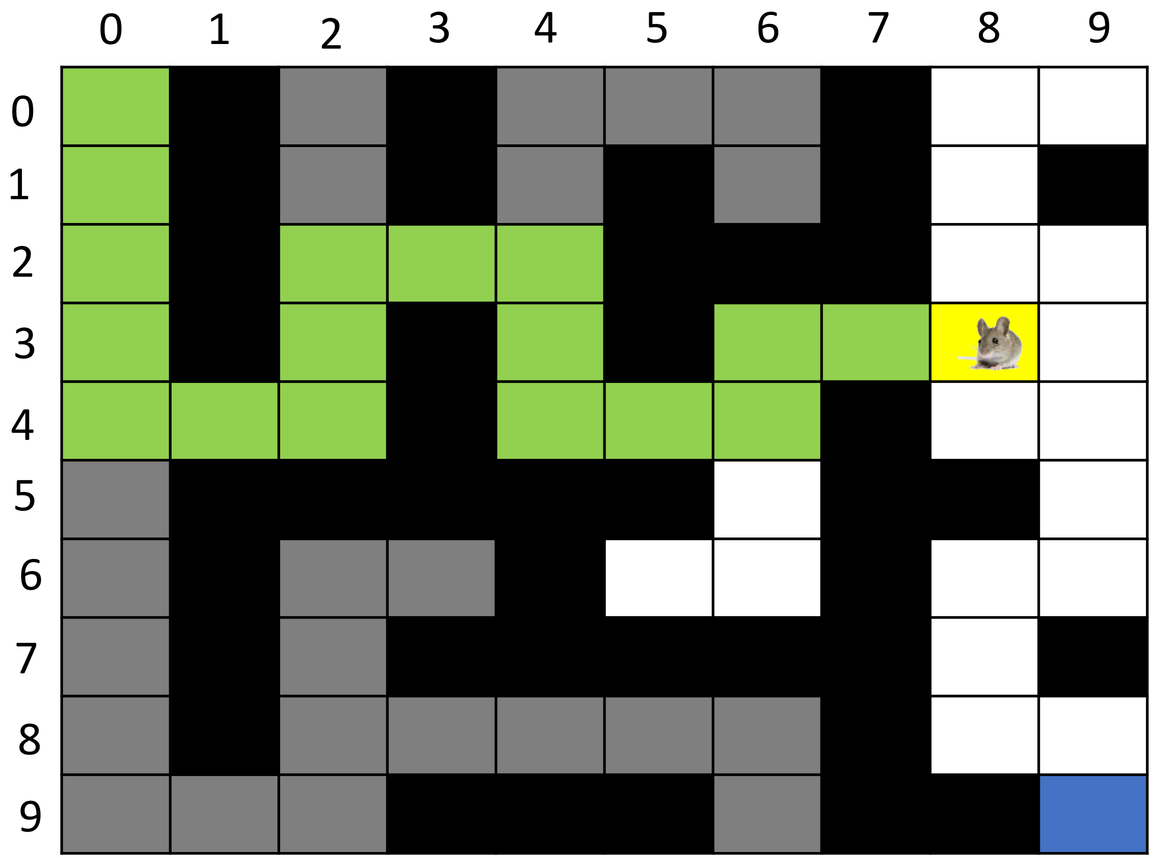
8/3

...

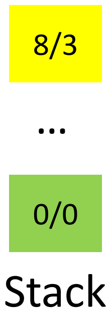
0/0

Stack

Recap on the backtracking algorithm.

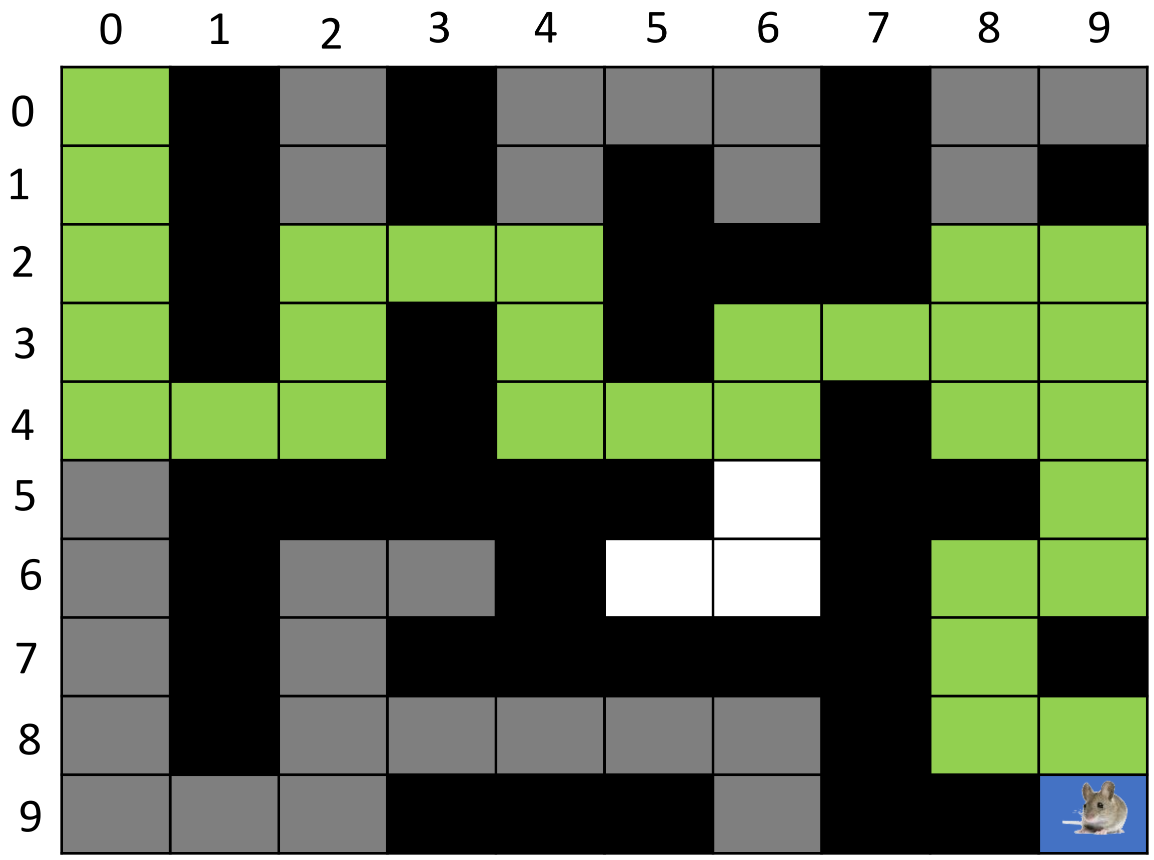


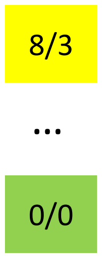
Mouse Goal No way visited



Check what is free with North/West/South/East-Rule

Recap on the backtracking algorithm.



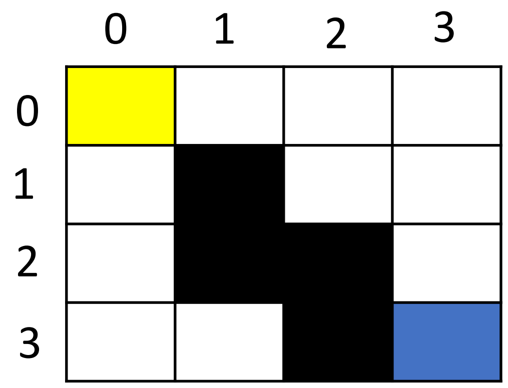


Stack

Mouse Goal No way visited

Check what is free with
North/West/South/East-Rule

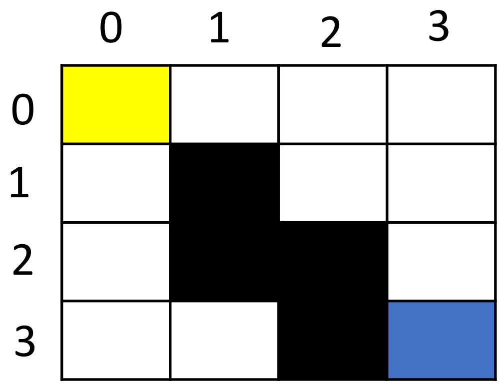
Exercise



Task:

- Write down the stack and with each iteration
- Use the North/West/South/East-Rule

Solution



Task:

- Write down the stack and with each iteration
- Use the North/West/South/East-Rule

6																	3/3
5														3/0		3/2	3/2
4					1/3								3/1	3/1	3/1	3/1	3/1
3				0/3	0/3	0/3						2/1	2/1	2/1	2/1	2/1	2/1
2			0/2	0/2	0/2	0/2	0/2				2/0	2/0	2/0	2/0	2/0	2/0	2/0
1		0/1	0/1	0/1	0/1	0/1	0/1	0/1		1/0	1/0	1/0	1/0	1/0	1/0	1/0	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	0/0	0/0
It.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17

When to use such algorithms?

- When the workspace is only two dimensional
- When the amount of discrete intervals is “small”
- When the robot can only move in two dimensions.
- When storage and performance are not relevant.

Practical applications of such algorithms: little

→ Therefore we address algorithms that can handle many dimensions and do this at high performance.

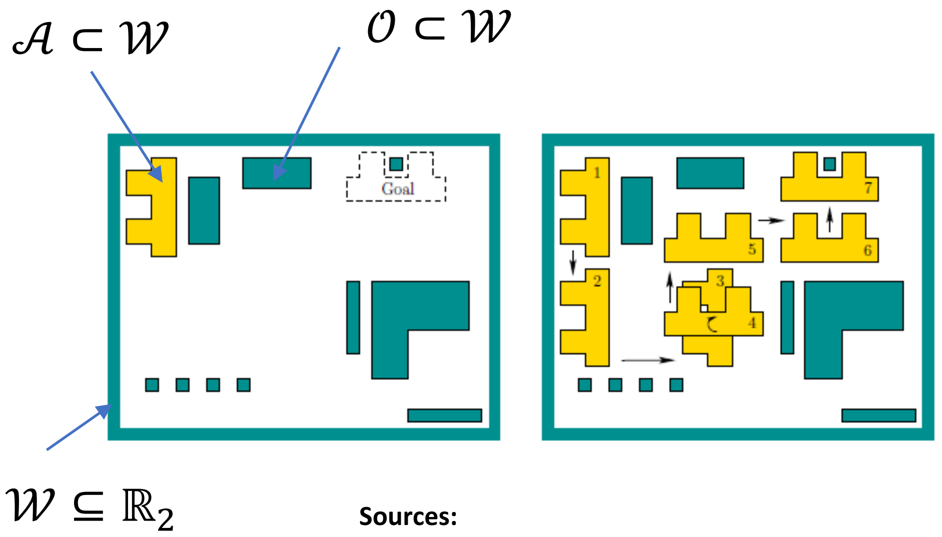
What is the workspace?

Definition: Two-dimensional Workspace

Let $\mathcal{W}$ denote the world, which contains a robot and obstacles, both defined by polygons. Let $\mathcal{W} \subseteq \mathbb{R}_2$ denote the set of all obstacles as $\mathcal{O} \subset \mathcal{W}$ and call it forbidden region. Moreover, define $\mathcal{A} \subset \mathcal{W}$ as a robot.

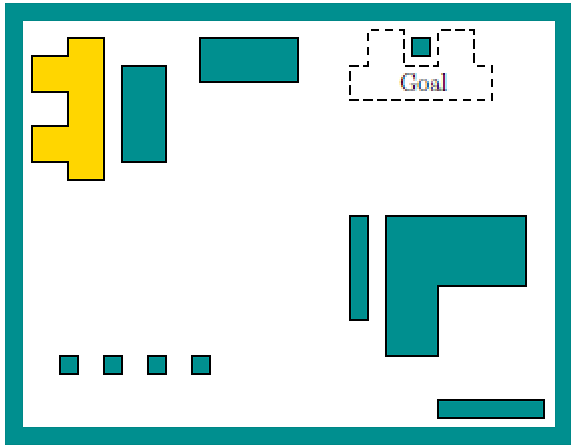
Definition: Three-dimensional Workspace

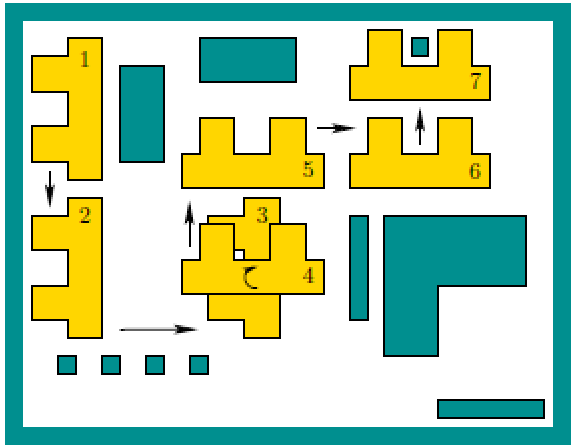
Let $\mathcal{W}$ denote the world, which contains a robot and obstacles, both defined by polyhedra. Let $\mathcal{W} \subseteq \mathbb{R}_3$ denote the set of all obstacles as $\mathcal{O} \subset \mathcal{W}$ and call it forbidden region. Moreover, define $\mathcal{A} \subset \mathcal{W}$ as a robot.



Sources:
Motion Planning: The Essentials – LaValle – http://msl.cs.illinois.edu/~lavalle/papers/Lav11b.pdf

What are the workspaces of our examples?



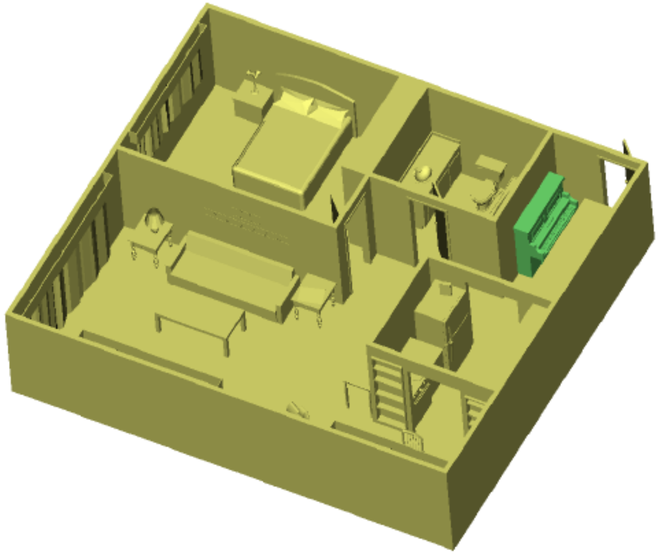


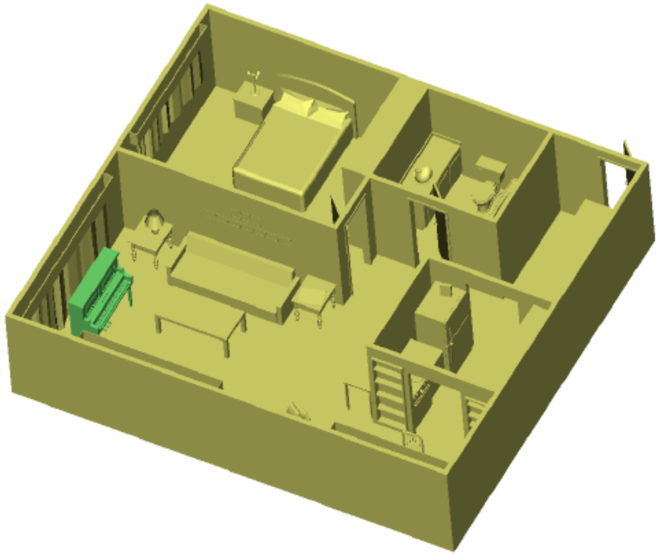
$$\mathcal{W} \subseteq \mathbb{R}_2$$

Sources:
Motion Planning: The Essentials – LaValle - http://msl.cs.illinois.edu/~lavalle/papers/Lav11b.pdf

→ What is the robot?
→ What are the obstacles?

What are the workspaces of our examples?

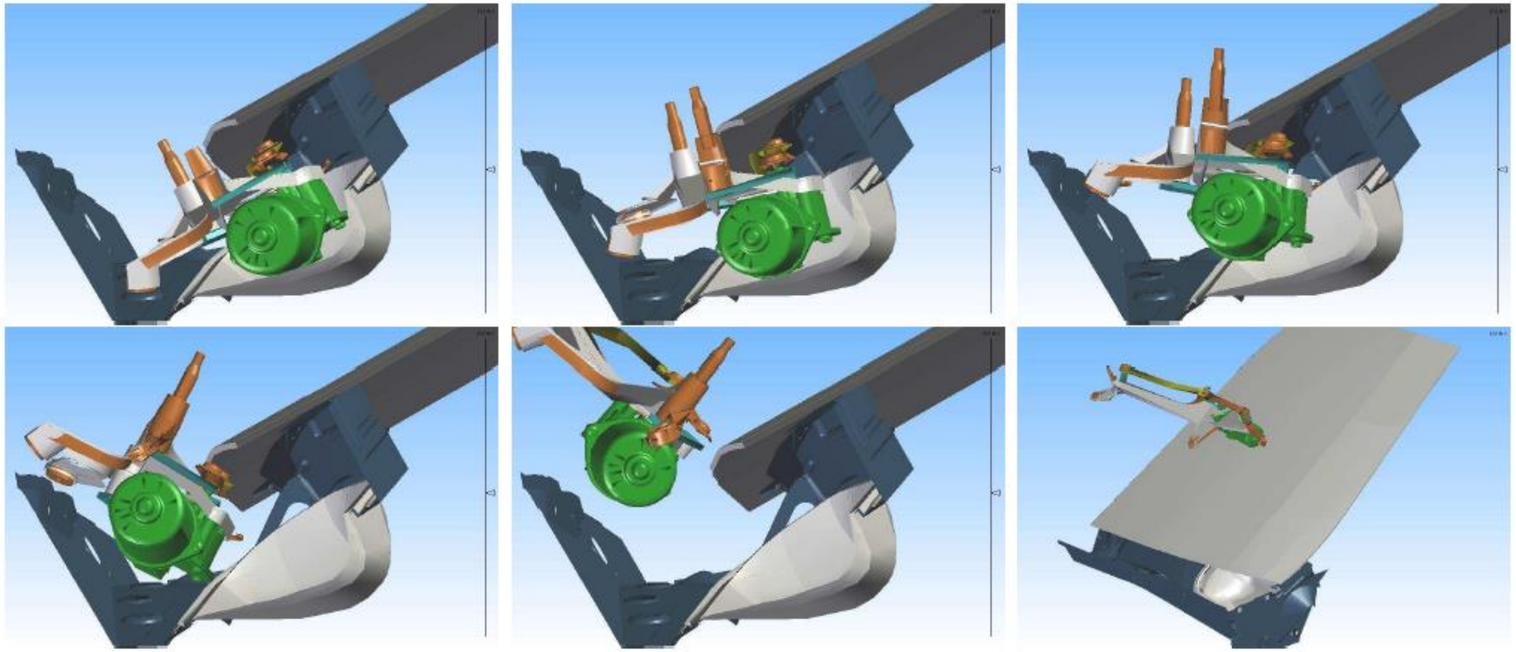




$\mathcal{W} \subseteq \mathbb{R}_3$

→ What is the robot?
→ What are the obstacles?

What are the workspaces of our examples?

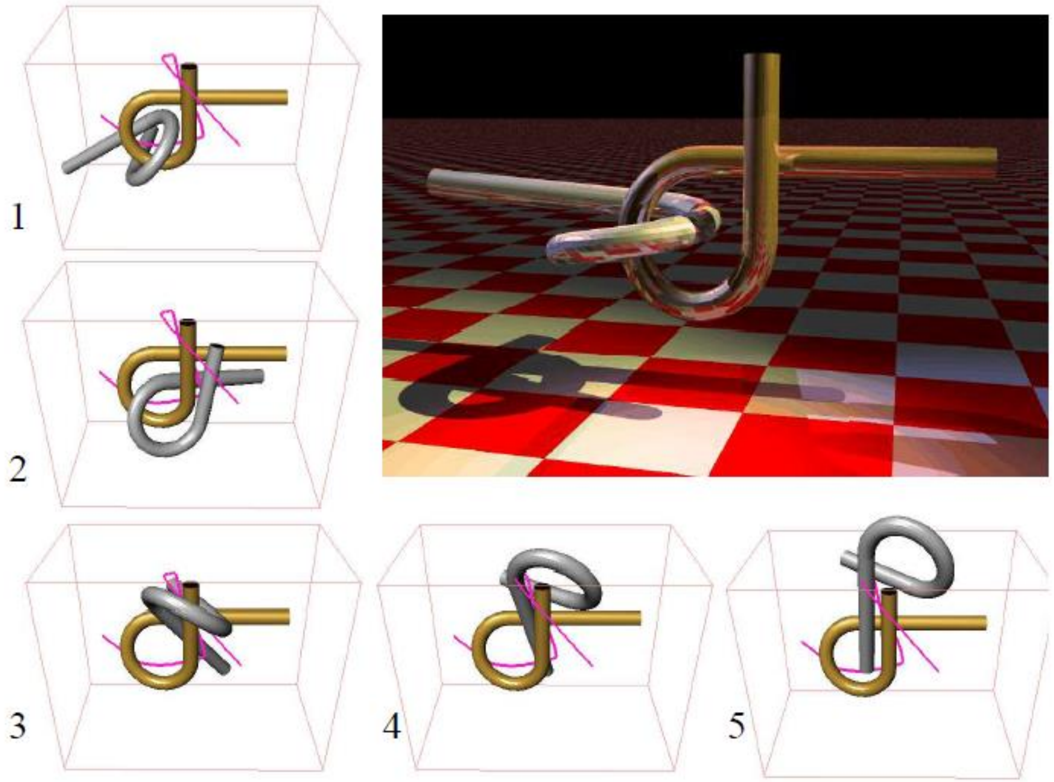


$\mathcal{W} \subseteq \mathbb{R}_3$

Sources:
Planning Algorithms– LaValle - http://planning.cs.uiuc.edu/

→ What is the robot?
→ What are the obstacles?

What are the workspaces of our examples?

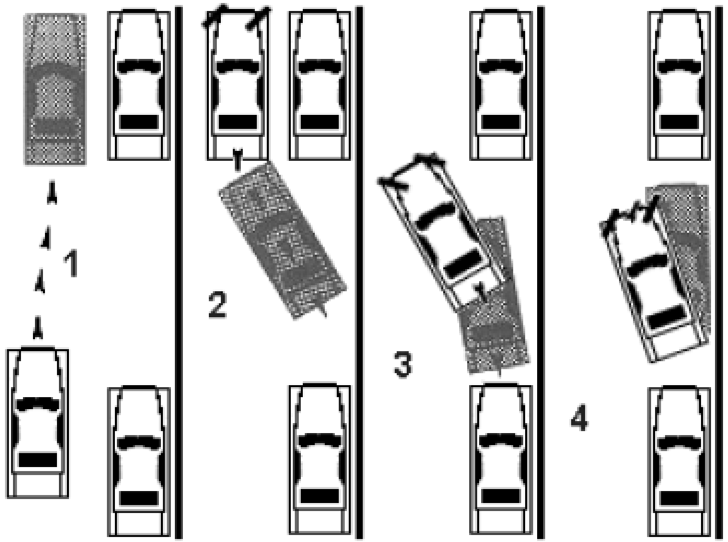


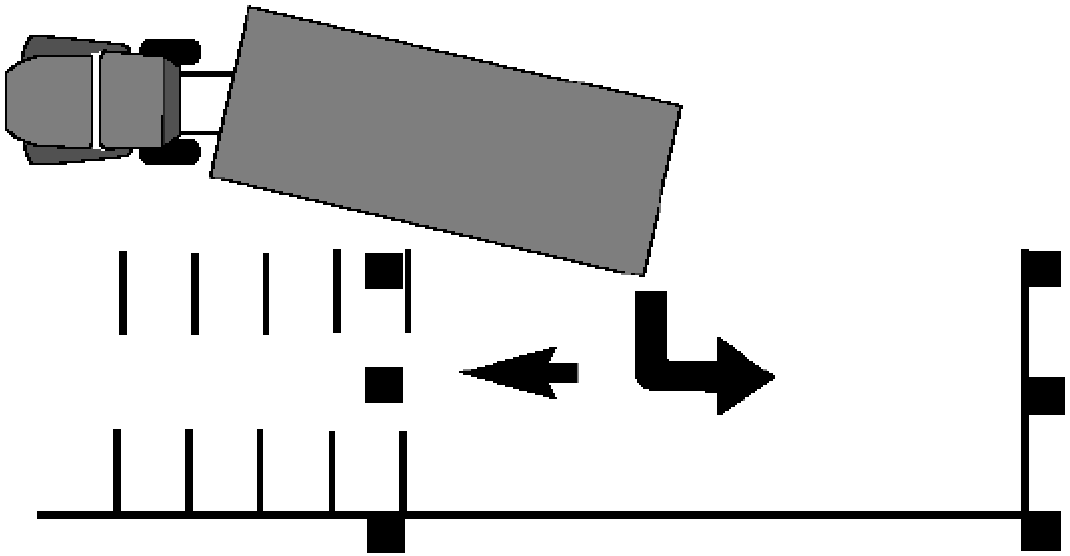
Sources:
Planning Algorithms– LaValle - http://planning.cs.uiuc.edu/

$\mathcal{W} \subseteq \mathbb{R}_3$

→ What is the robot?
→ What are the obstacles?

What are the workspaces of our examples?



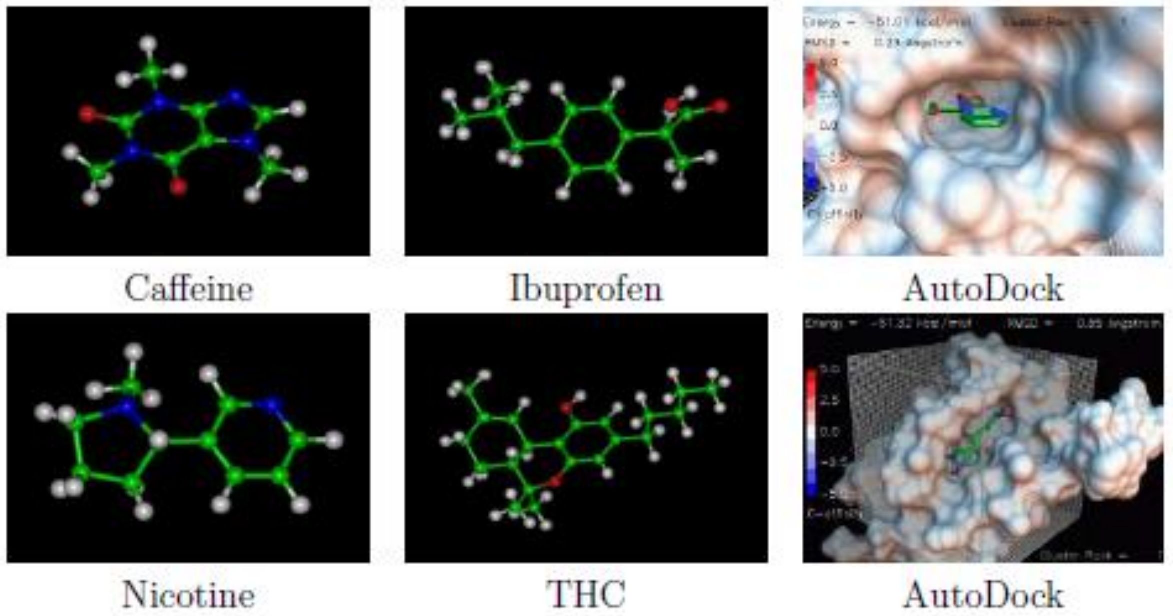


$\mathcal{W} \subseteq \mathbb{R}_2$

Sources:
Planning Algorithms– LaValle - http://planning.cs.uiuc.edu/

→ What is the robot?
→ What are the obstacles?

What are the workspaces of our examples?



Sources:
Planning Algorithms– LaValle - http://planning.cs.uiuc.edu/

$$\mathcal{W} \subseteq \mathbb{R}_3$$

→ What is the robot?
→ What are the obstacles?

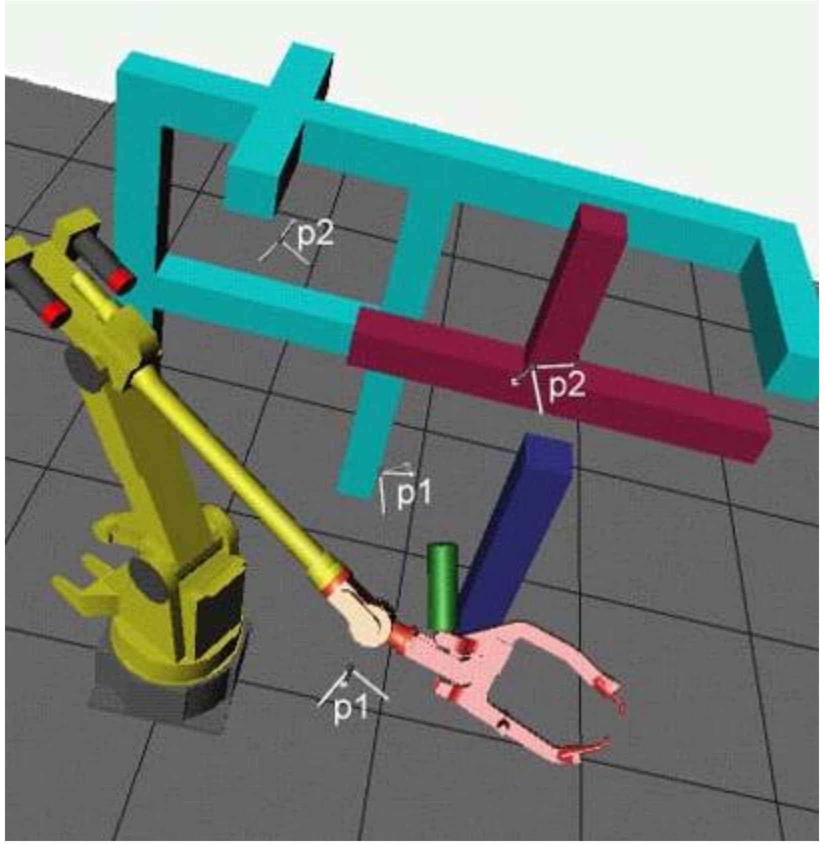
What are the workspaces of our examples?

$\mathcal{W} \subseteq \mathbb{R}_2$

→ What is the robot?
→ What are the obstacles?

Sources:
Robot Motion Planning– Wolfram Burgard et al. – http://ais.informatik.uni-freiburg.de/teaching/ss11/robotics/slides/18-robot-motion-planning.pdf

What are the workspaces of our examples?

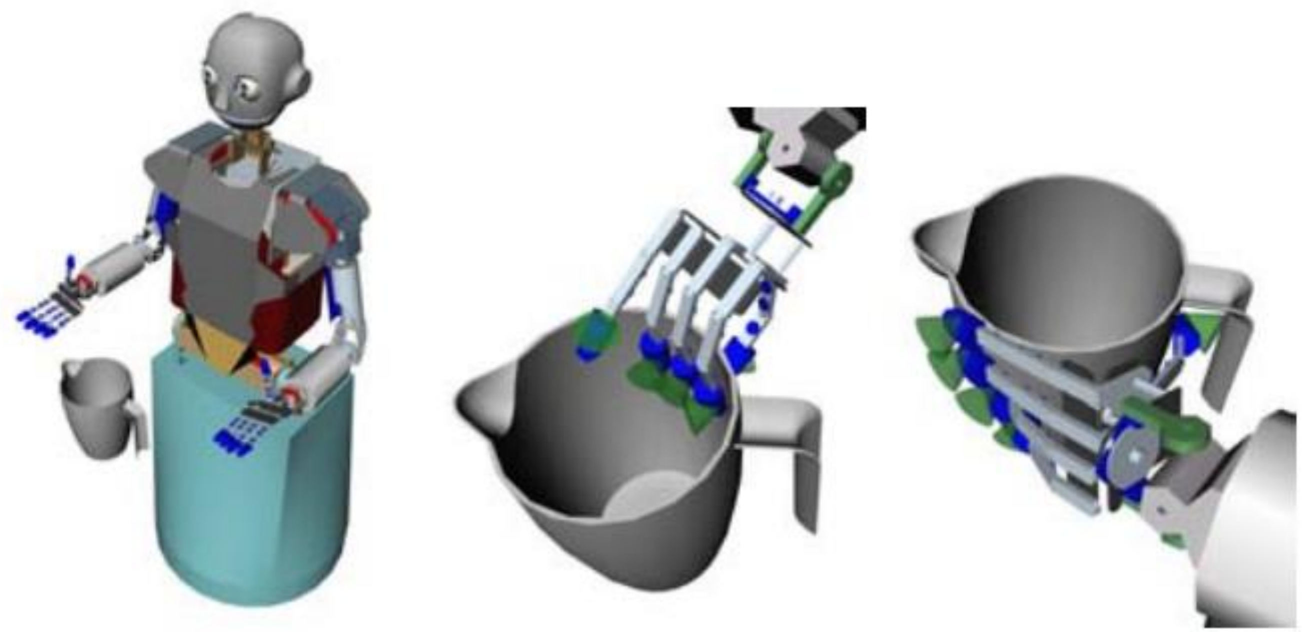


Sources:
A Journey of Robots, Digital Actors, Molecules and Other Artifacts – Jean Claude Latombe–
https://robotics.stanford.edu/~latombe/projects/motion-planning.ppt

$$\mathcal{W} \subseteq \mathbb{R}_3$$

→ What is the robot?
→ What are the obstacles?

What are the workspaces of our examples?



$$\mathcal{W} \subseteq \mathbb{R}_3$$

Sources:
Simultaneous Grasp and Motion Planning – Nikolaus Vahrenkamp et al– https://h2t.anthropomatik.kit.edu/pdf/Vahrenkamp2012.pdf

→ What is the robot?
→ What are the obstacles?

How to model the robot and obstacles?

In 2D:

The robot and obstacles are models with polygons.

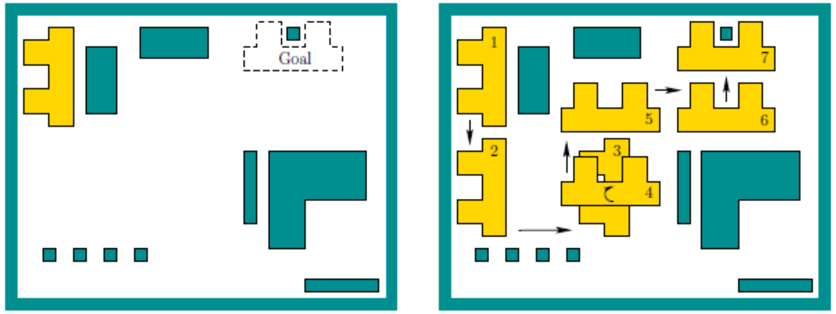
In 3D:

The robot and obstacles are models with polyhedra

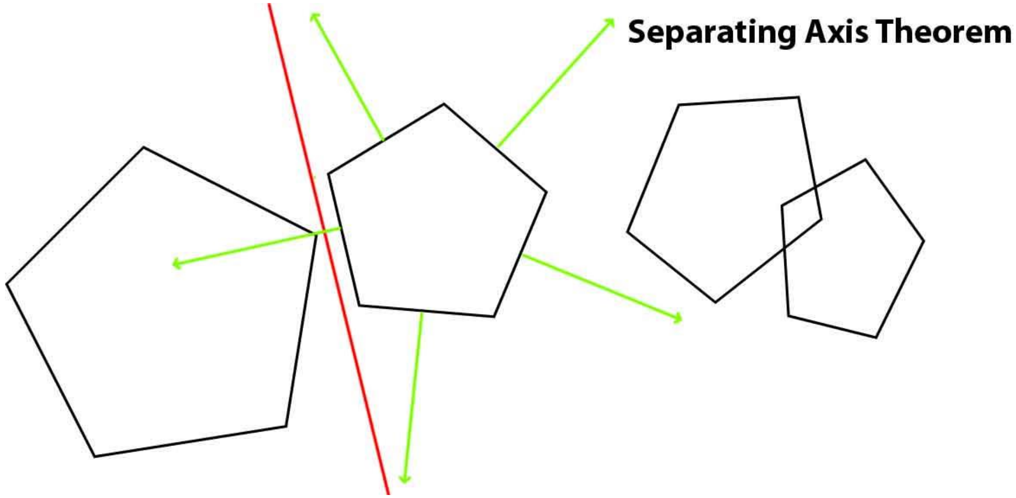
→ In practice: triangle sets.

→ Why? There are fast algorithms to detect collisions for triangle sets.

The most challenging task in the workspace

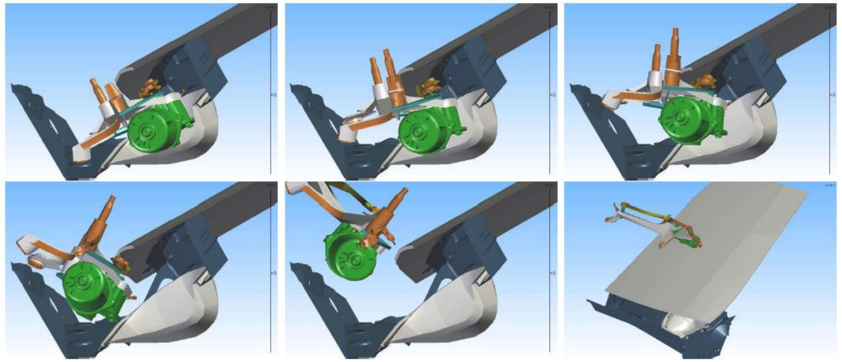


Collision detection 2D



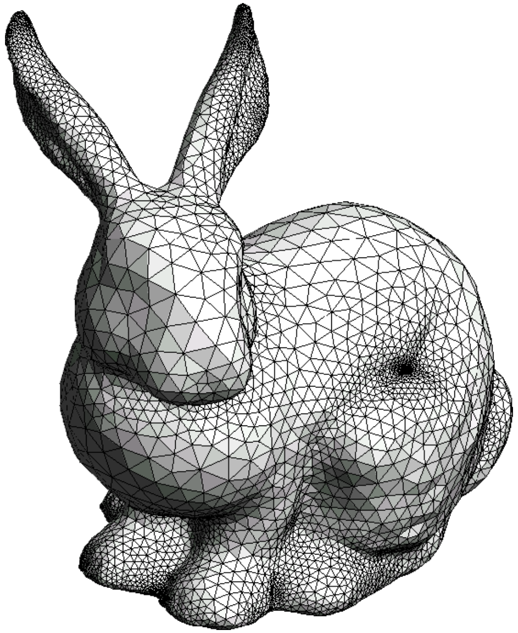
Source: https://www.embed.com/typescript-games/polygon-collision-detection.html

The most challenging task in the workspace



Sources:
Planning Algorithms– LaValle - http://planning.cs.uiuc.edu/

Collision detection 3D – Bounding Volume Hierarchies



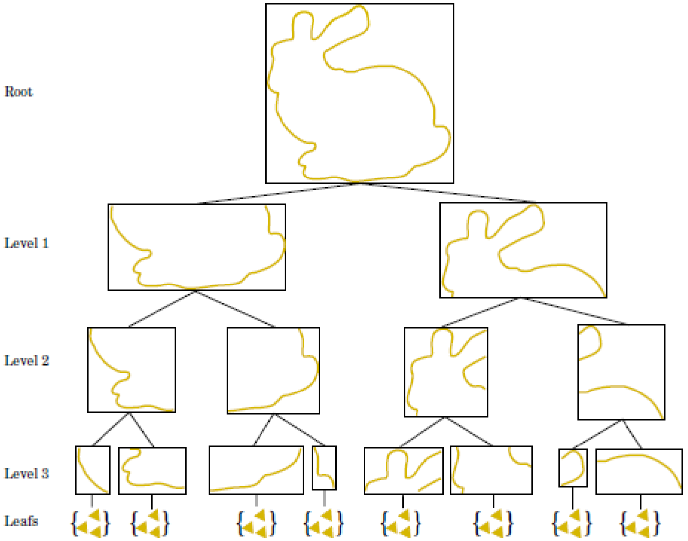


Fig. 15: Example for a Bounding Volume Hierarchy.

→ Topic of Algorithmic Geometry/Later in the course (depending on time)

A more simple but not exact approach

1. Take your obstacles in the workspace 2d/3d and discretize it in a pixel/voxel field.
2. In this field mark all covered pixels with “black”
3. Take your robot. Discretize its border to points.
4. Test for all points (which order?) of the robot whether the point is on a black pixel/voxel.
5. If no point is in “black” voxel” → Robot is free of collision.

→ This algorithm is called Voxmap PointShell Algorithm and is very popular.
Why?

- Easy to implement
- Easy to parallelize
- Precision is sufficient for most applications

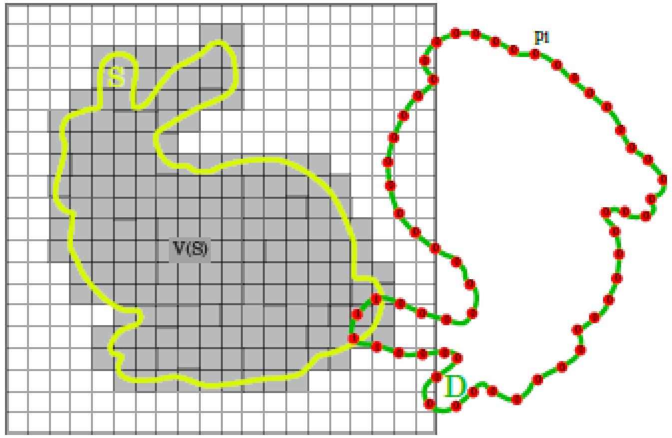


Fig. 14: Example for the Voxmap PointShell^{TM} algorithm.

Note for the practical work study

- By using bmp files we use the Voxmap PointShell algorithm.
- The environment is given by a discrete array.
- The robot as well → you can use the full robot or discretize the border. (Note: the border of the robot is a subset of the whole robot)

But again, still keep in mind:

In reality we are still dealing with continuous motion planning problem.

Exercise

Think about a motion planning example that has not yet been covered yet and define the dimension of the workspace. Moreover note down how you would model the workspace.