

Introduction to Robotics

CSE 461

Lecture 9: Robot Navigation (Mapping and Exploration)

Niloy Irtisam
Lecturer, Dept. of Computer Science and Engineering
Brac University

Robots Navigation

- **Path Planning**: How to I get to my Goal?
- **Localization**: Where am I?
- **Mapping**: Where have I been?
- **Exploration**: Where haven't I been?

Robots Navigation

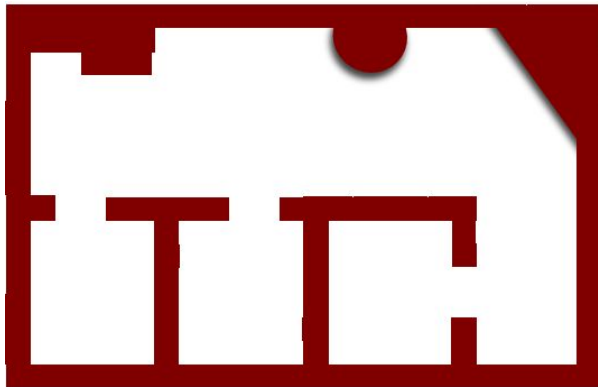
- **Path Planning**: How to I get to my Goal?
- **Localization**: Where am I?
- **Mapping**: Where have I been?
- **Exploration**: Where haven't I been?

Mapping and Exploration

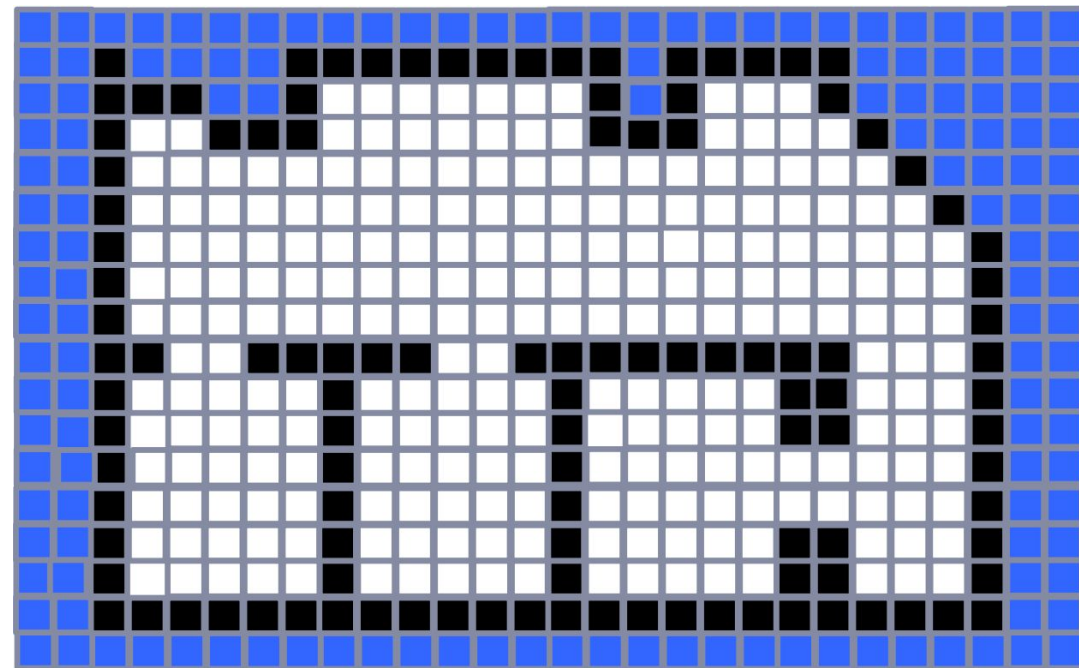
- **Question:**
 - You are roaming around in an unknown space, what can you learn about it?
- Two parts of the problem:
 - **Mapping:** As you roam around the world, how do you build a memory of the shape of the space you have moved through?
 - **Exploration:** Given that you don't know the shape or size of the environment, how to make sure you covered all of it?
- Both have many uses:
 - Returning back to home/charger after some task.
 - Cleaning a new room efficiently; Systematic search for survivors
 - Mapping a collapsed mine or building.
- Mapping and Exploration are also “collections of algorithms”
 - We will focus on “Occupancy Grid” algorithms

What is an Occupancy Grid?

- A way of representing a map as a gridded world where each cell is either “occupied” or “empty” or “unknown”.

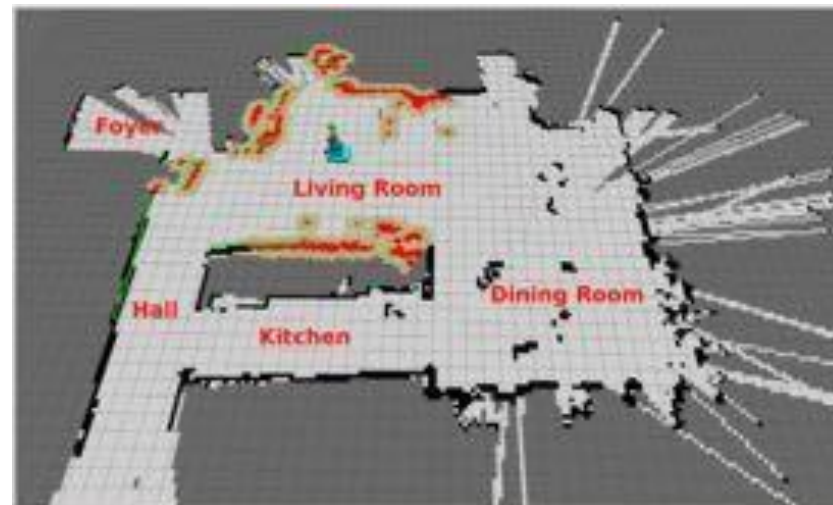
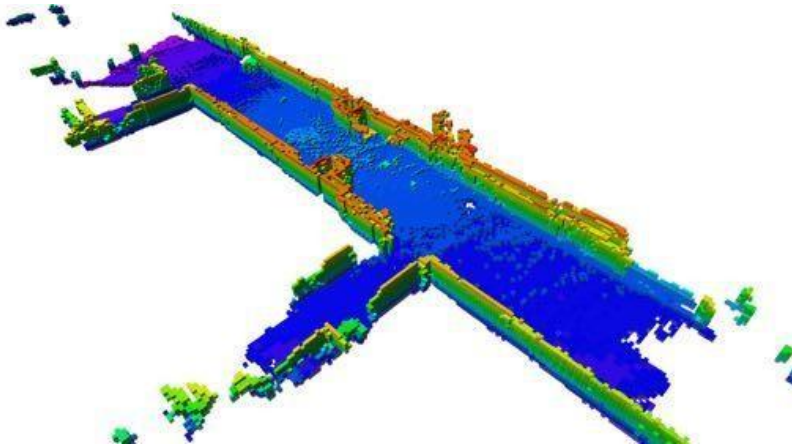
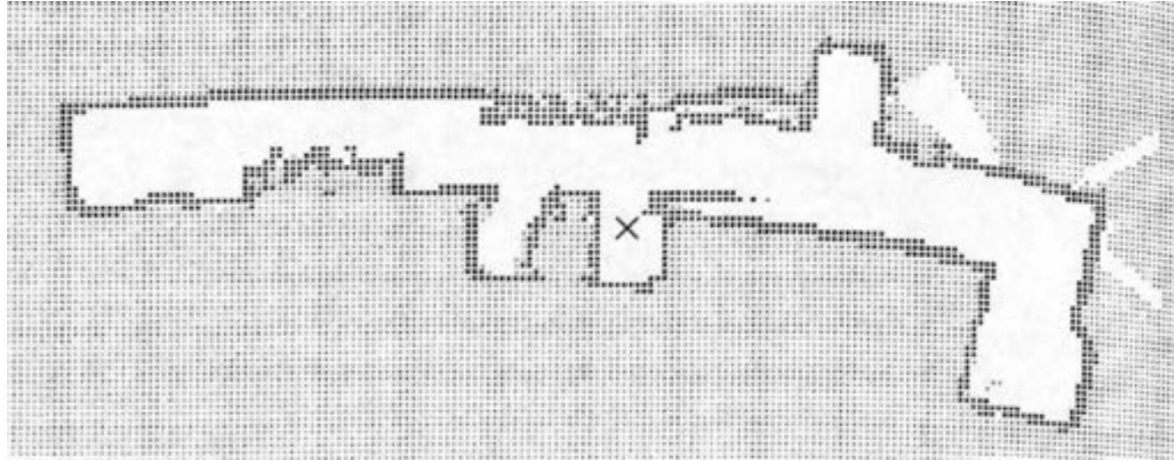


Your World



Grid generated by a Robot => boundary shape

Examples



What is a Sensor Model?

- **Constructing a Sensor Model**

- A sensor measures *raw values* in an environment
- You have to map that into a Grid Cell Value.
- Robots can have very different sensors and configurations
- Examples:
 - LIDAR/Depth Camera
 - Vs. a 360 degree vision/ranging system

Constructing a Sensor Model

- **Example: Depth Sensor Model**

R = maximum range, B = maximum angle

Let say the sensor at point p returns **distance** = " r "

Region 1 ($\text{dist} < r$, grid cell probably empty)

Region 2 ($\text{dist} = r$, grid cell probably obstacle)

Region 3 ($\text{dist} > r$, grid cell unknown/obscured)

- **Simplest Sensor Model**

Where I stand is Empty (white)

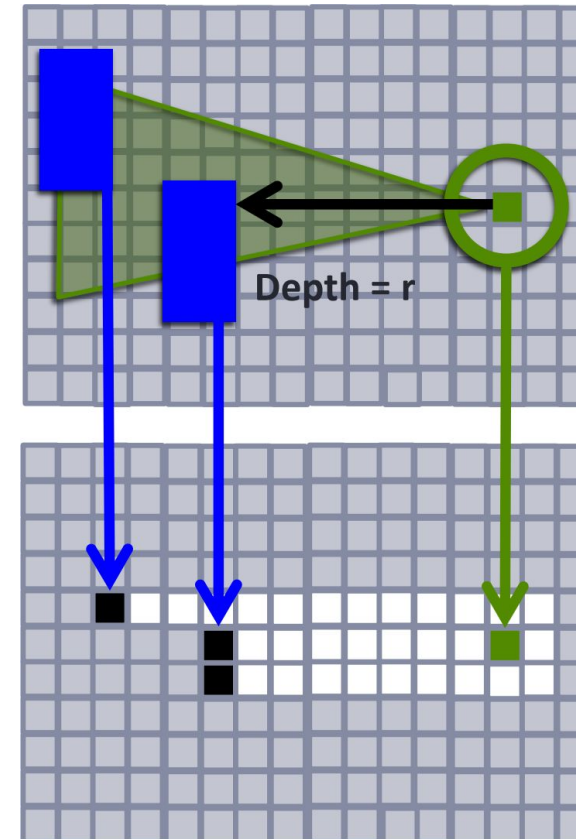
- **A Better Model**

Set Region 1 cells as Empty (white)

Set Region 2 cells as Occupied (black).

Pick a max range/angle where data is reliable

Rest is still Unknown (gray)



A Simple OG Mapping Algorithm

1. Initialize a Grid

- Set all locations as “unknown”, pick a start location and orientation

2. Update the Grid

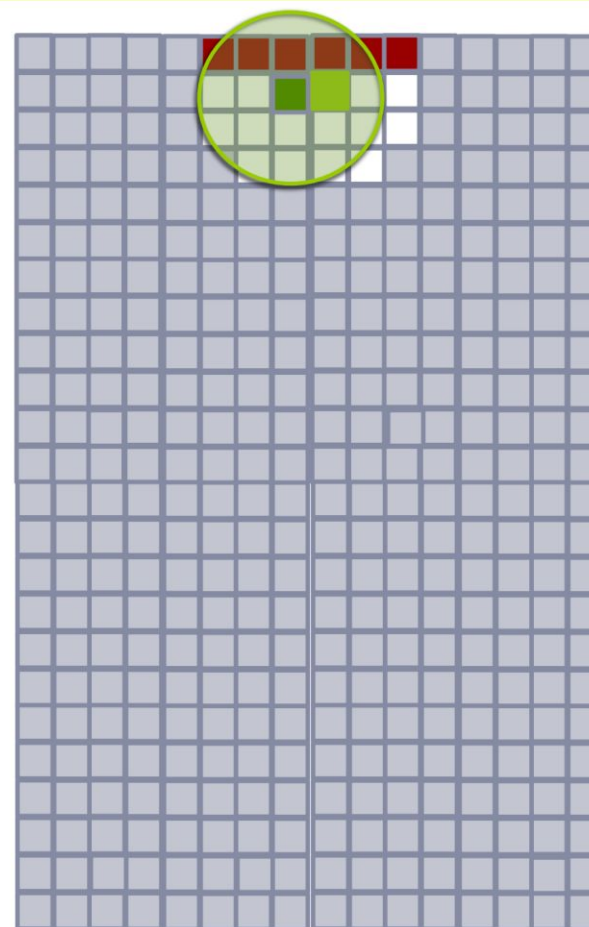
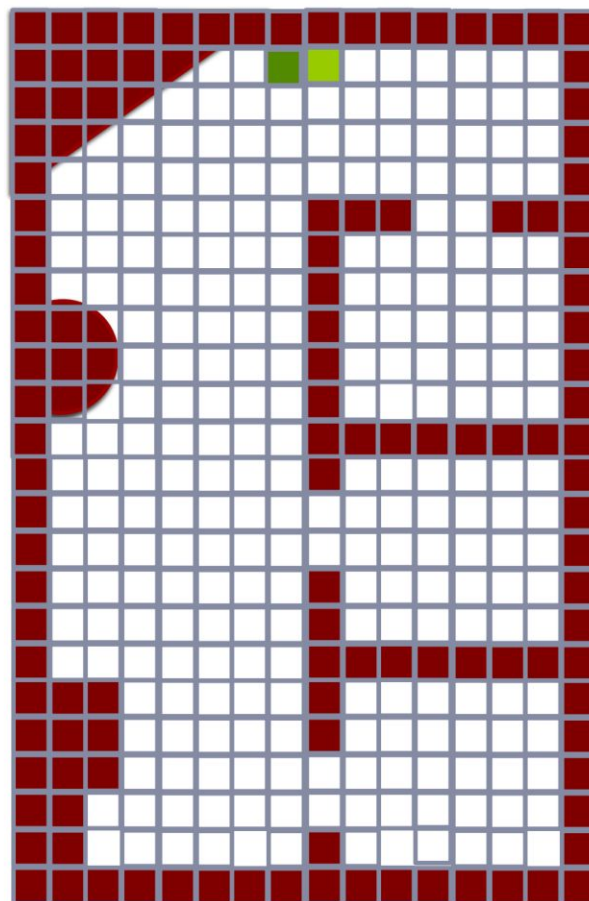
- *Mark your current grid position as “empty”*
- Using your better sensor model,
- *Mark all visible grid locations as “empty” or “occupied”*

3. Pick a Next Move

- Look at neighboring grid positions in your map
- Pick a neighboring grid location that is empty (randomly)
- Move to it and update your current position in the Grid

4. Loop forever

- Keep moving and updating the grid (unless you are “done”)



Exploration

- Basic Concept in Robotics: Navigating a GRID Graph is different
 - DFS works, but will still make a robot retrace steps
 - **Better choice: Frontier Based Exploration**

Exploration in Grid Worlds

- **Frontier Based Exploration**

- A common technique for building maps

- **Key Idea:**

- Identify the “frontiers” between known and unknown

Frontier cell = a unknown cell with at least one empty cell

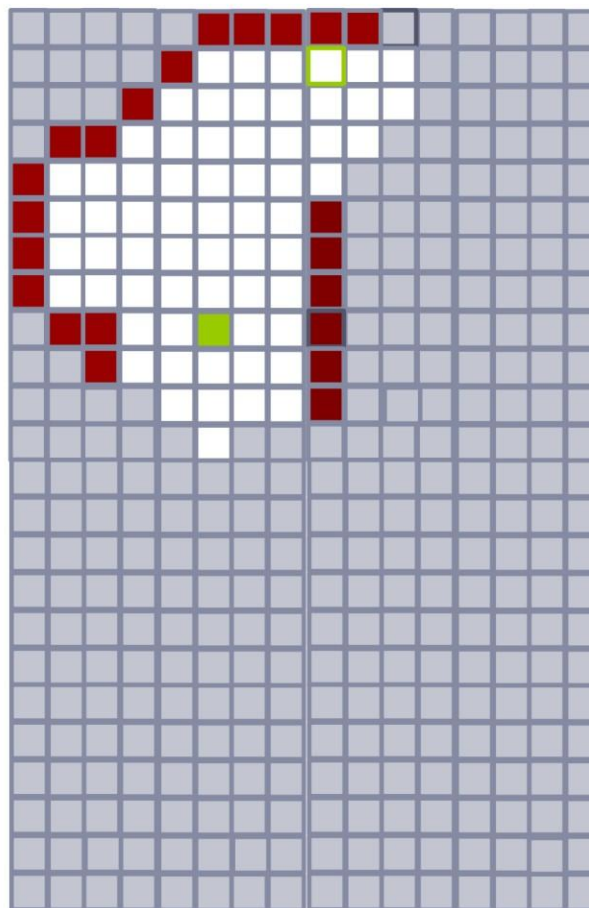
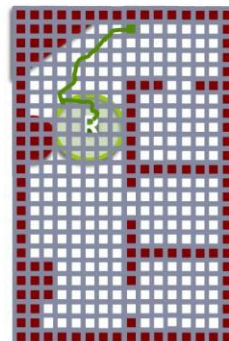
- Pick a frontier cell (e.g. the closest)

Plan a path to go explore it.

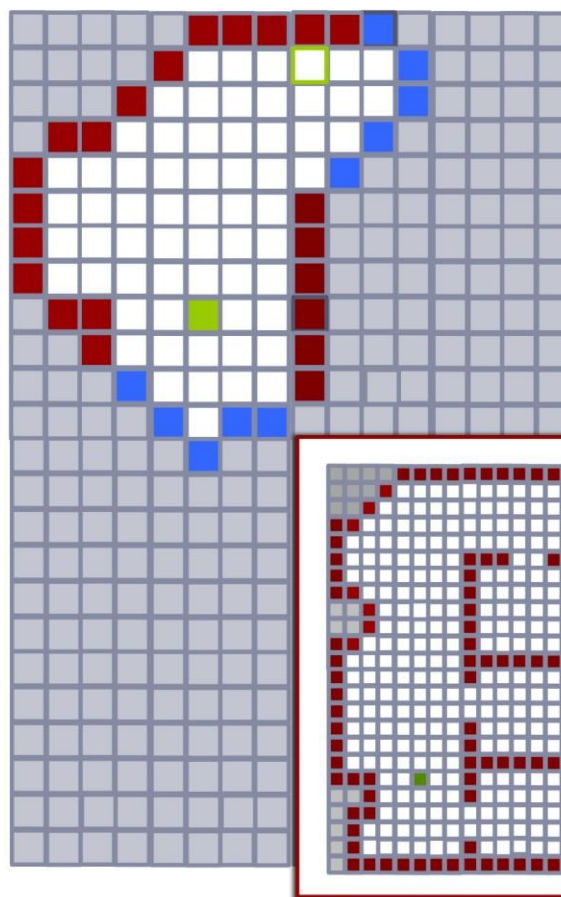
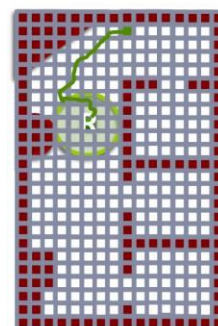
- **Done Condition:**

No more frontier nodes left => your map is Complete!

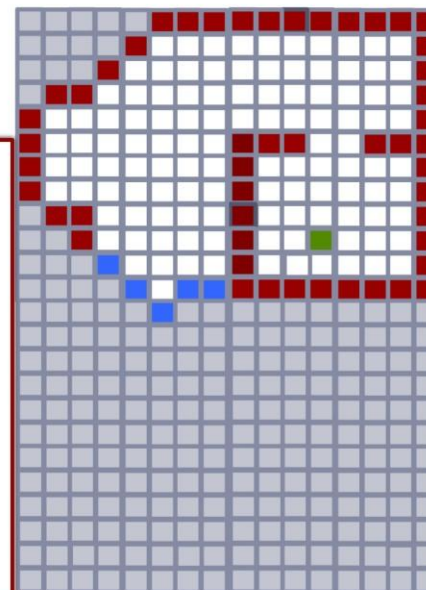
If finite world, then any algorithm that systematically explores frontier nodes is guaranteed to cover the whole world.



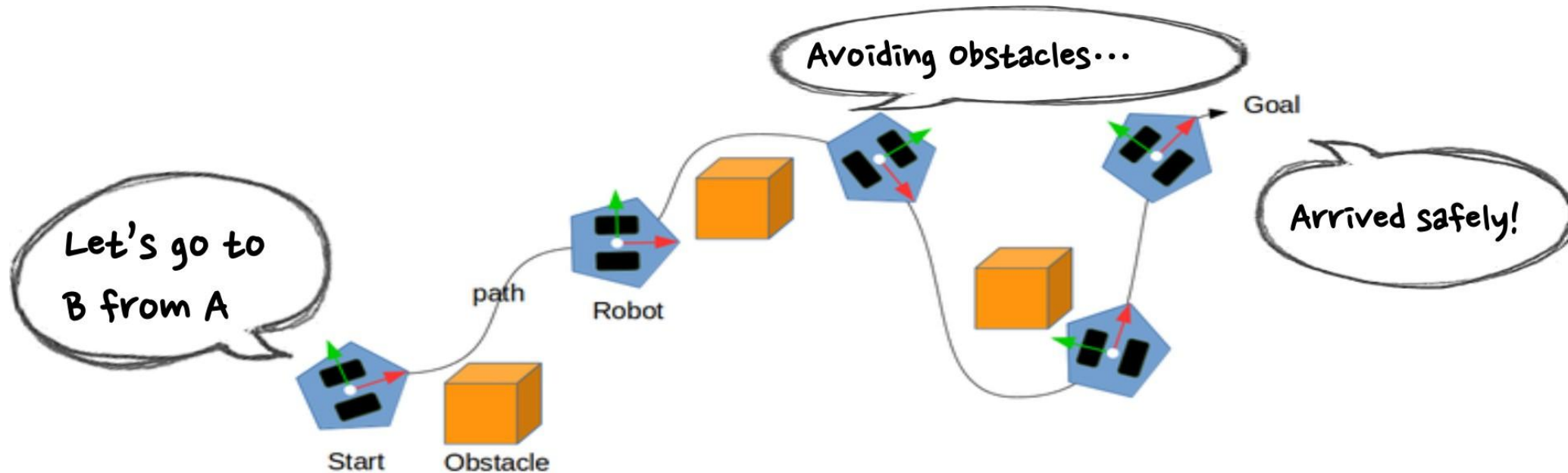
A **Frontier Node** is a
Gray node (Unknown)
next to a
White node (Empty)



A **Frontier Node** is a
Gray node (Unknown)
next to a
White node (Empty)



Summary



- ① **Position**: Measuring/estimating the robot's position
- ② **Sensing**: Measuring obstacles such as walls and objects
- ③ **Map**: Maps with road and obstacle information
- ④ **Path**: Calculate optimal path to the destination and follow the path

Summary

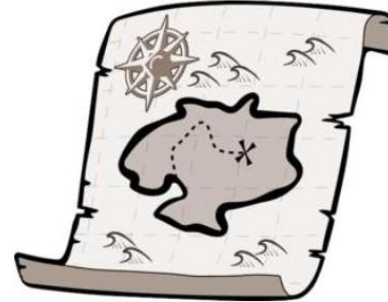
① **Position**



② **Sensing**



③ **Map**



④ **Path**



Position+Sensing → **Map**

SLAM

Simultaneous Localization And Mapping

Position+Sensing+Map → **Path**

Navigation