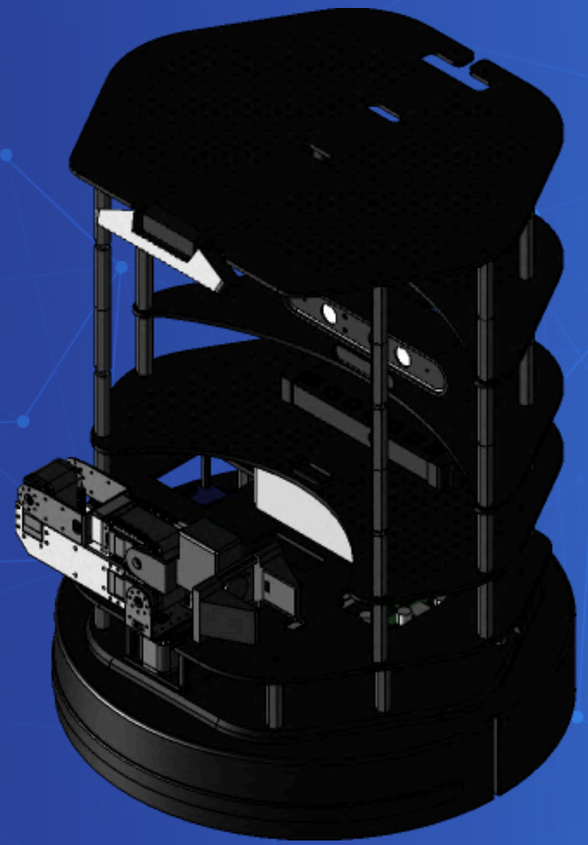
SAKOLRAJWITTAYANUKUL

ROS EDUCATION

Robotic and AI for

Revolutionizing the way we live, work, and think



WHAT IS ROS?

ROS
Robot Operating System

Robot Operating System (ROS) is an open-source, meta-operating system for your robot. It provides the services you would expect from an operating system, including

- Hardware abstraction
- Low-level device control
- Implementation of commonly-used functionality
- Message-passing between processes
- Package management.

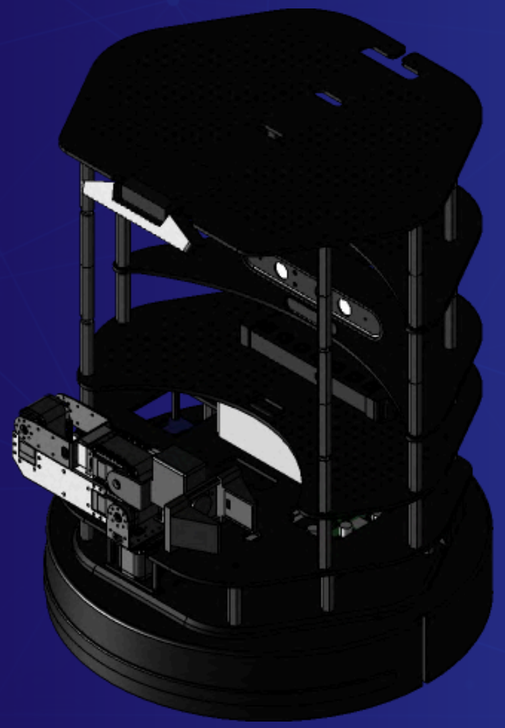
It also provides tools and libraries for obtaining, building, writing, and running code across multiple computers. ROS is similar in some respects to robot frameworks.

- Tools-based

There are many small, generic programs that perform tasks such as visualization, logging, plotting data streams, etc.

- Multi-Lingual

ROS software modules can be written in any language for which a client library has been written. Currently client libraries exist for C++, Python, LISP, Java, JavaScript, MATLAB, Ruby, and more.



PURPOSE why

ROS provides a structured framework where different functions are broken into independent nodes, making it easy to reuse and replace components without affecting the whole system.

- includes many built-in libraries for SLAM, path planning, computer vision, and control, reducing the need to build everything from scratch.
- A large community provides extensive documentation, tutorials, and packages, accelerating development.
- It allows you to work with various robot hardware (sensors, actuators, cameras) using a unified API, reducing compatibility issues.

Our goal is to enhance children's education, support teachers in managing and nurturing students, and improve children's English proficiency.

COMPONENT

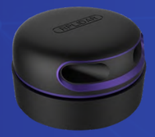
1.Board computer rasberry pi 4

- Navigation and Path Planning
- Processes sensor data from LiDAR, cameras
- Creates real-time maps of environment
- Runs artificial intelligence and machine learning algorithms
- Collects and synthesizes data from multiple sensors

2.Microcontroller arduino mega

- Executes precise motor controls
- Manages low-latency sensor readings
- Handles emergency response mechanisms
- translates commands from board computer into specific hardware actions
- provides feedback to board computer

3.Lidar



- Creates detailed 3D maps of environment
- Helps robot determine its exact position within mapped space
- Obstacle Detection
- Detect human presence

4.Webcam camera

- Visual Monitoring
- Object Recognition
- Verify assembly accuracy

5.microphone

- Receive speech
- order from teacher and student

6.speaker

- speech sound

ABILITY APPLICATION

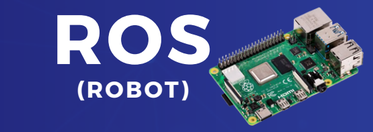
Navigation

- Create real-time 3D environmental maps
- Avoid obstacles dynamically
- Adjust paths in real-time
- Autonomous navigation in complex
- environments
- Multi-sensor integration
- Track robot's precise location continuously

Speak langues

- AI-based Face Recognition → Track kids' emotions and attention during lessons.
- Speech Processing AI → Help children learn English by interacting with a robot.
- Smart Navigation → Assist a teacher’s robot in guiding children safely.
- Object Detection → Recognize school supplies and help kids learn their names in English.

diagram



Display mapping

Depth camera (camera/depth/image_raw)

Sensor → Webcam camera (object_detection_node)

Lidar(scan)

Microcontroller

Mobile control (move_base)

Motor Drive(odom) → Encoder DC motor

HOW NAVIGATION WORK?

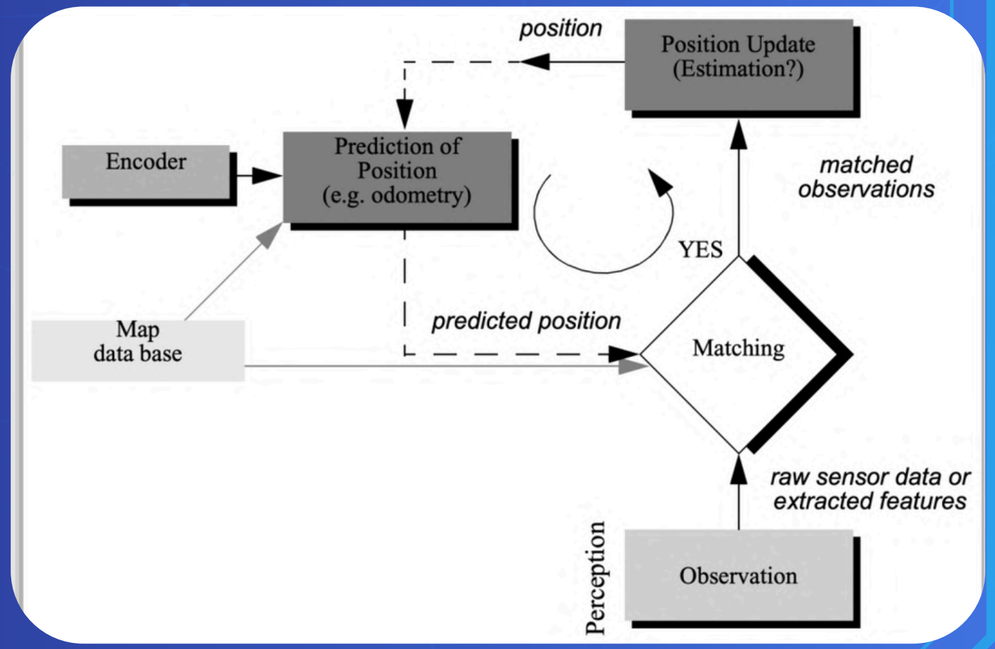
• Navigation is one of the most challenging competences required of a mobile robot.
• Success in navigation requires success at the four building blocks of navigation.

• Perception
The robot must interpret its sensors to extract meaningful data.

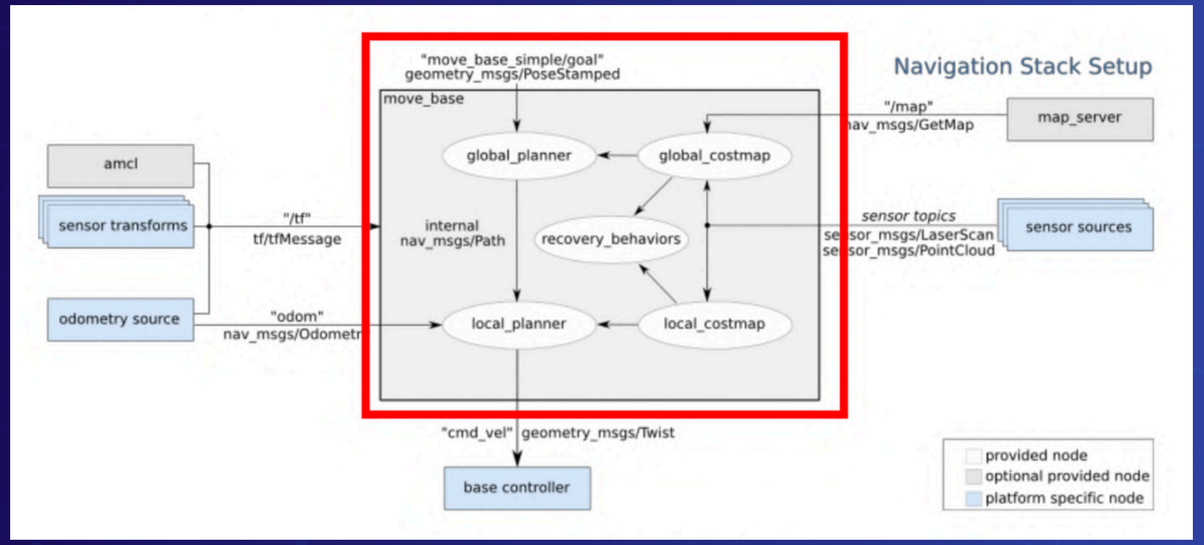
• Localization
The robot must determine its position in the environment.

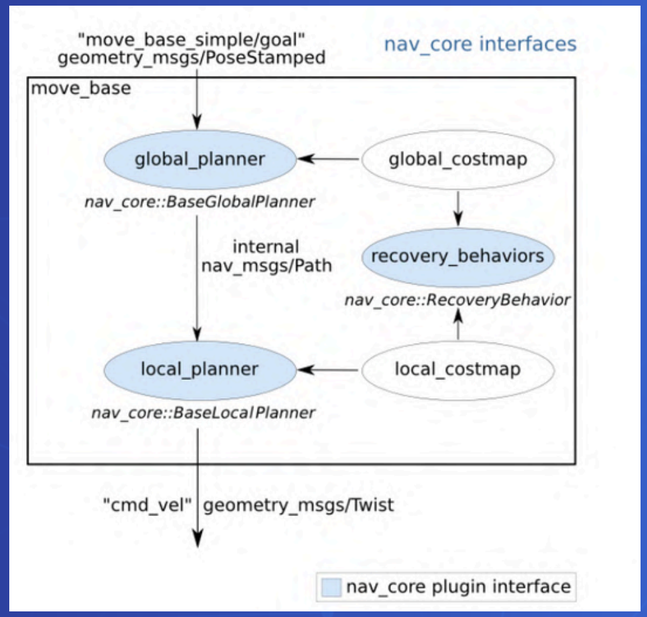
• Cognition
The robot must decide how to act to achieve its goals.

• Motion control
The robot must modulate its motor outputs to achieve the desired trajectory.



ROS NAVIGATION





- Global Planner

The global planner is responsible for generating an overall path from the start position to the goal, considering the entire map of the environment.

- Local Planner

The local planner works in real time to refine the path provided by the global planner. It reacts to dynamic obstacles and ensures smooth, collision-free motion.

- costmap

in ROS represents the environment in a grid format, where each cell has a cost value that indicates how difficult it is for a robot to pass through that area. It helps in path planning, obstacle avoidance, and navigation.

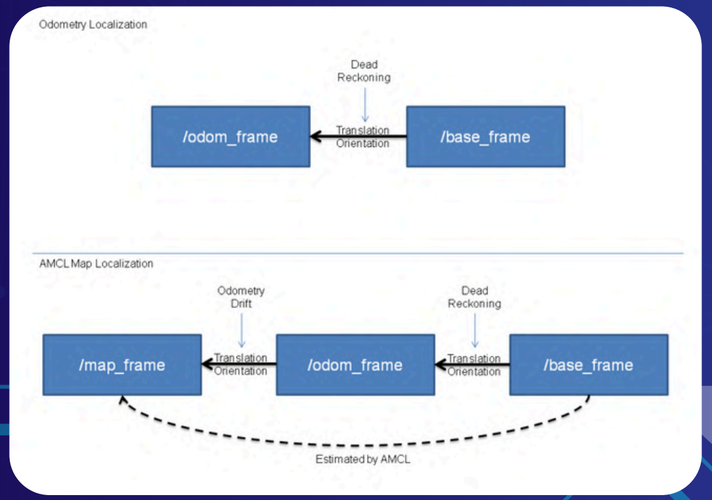
ROS NAVIGATION

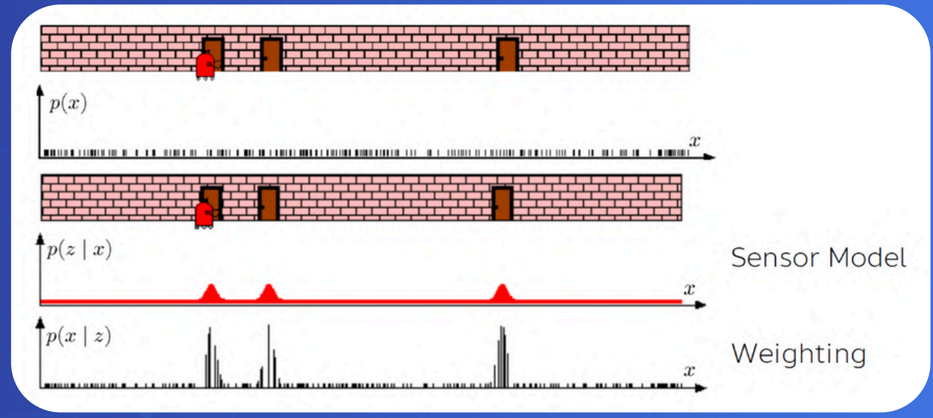
- Adaptive Monte Carlo localization (AMCL)
- Monte Carlo localization (MCL), also known as particle filter localization , is an algorithm for
- robots to localize using a particle filter. Given a map of the environment, the algorithm estimates
- the position and orientation of a robot as it moves and senses the environment.
- AMCL dynamically adjusts the number of particles based on KL distance to ensure that the
- particle distribution converge to the true distribution of robot state based on all past sensor and

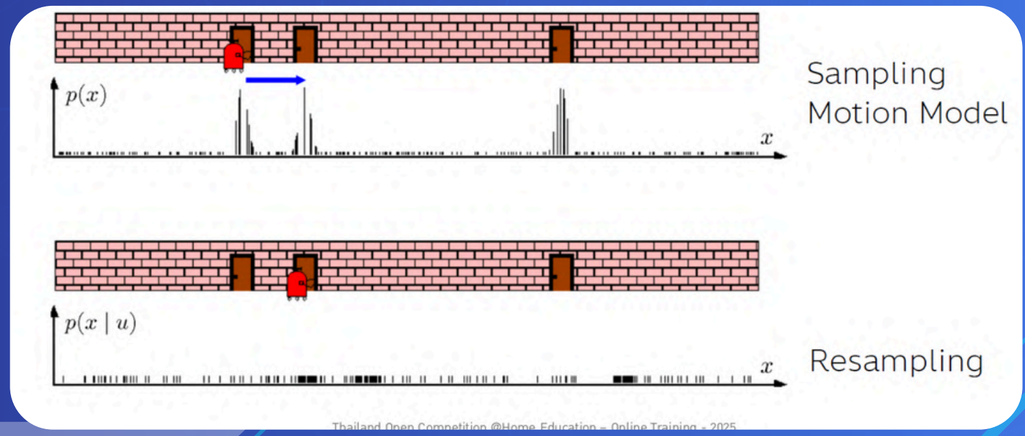
motion measurements with high probability.

Particle Filters (PF)

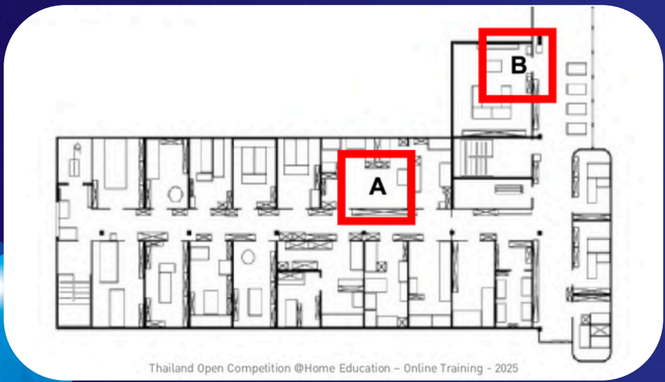
• Sample the next generation for particles using the proposal distribution (Often use the motion model)
• Compute the importance weights
• Resampling : “Replace unlikely samples by more likely”



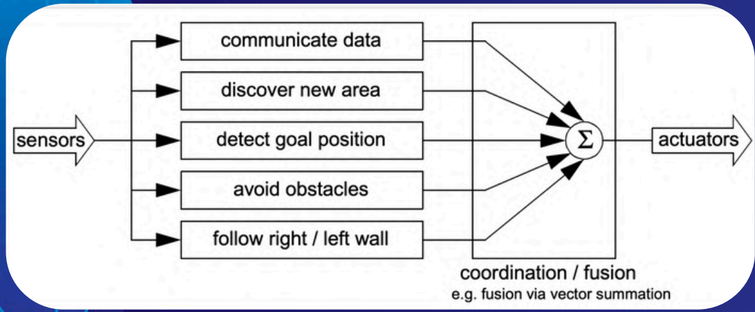




Localization



Localization may seem mandatory in order to successfully navigate between the two rooms. It is through localizing on a map, after all, that the robot can hope to recover its position and detect when it has arrived at the goal location. It is true that, at the least, the robot must have a way of detecting the goal location. However, explicit localization with reference to a map is not the only strategy that qualifies as a goal detector.



Behavior-based navigation
This technique is based on a belief that there exists a procedural solution to the particular navigation problem at hand.
Example
The behavioralist approach to navigating from room A to room B might be to design a left-wall following behavior and a detector for room B that is triggered by some unique queue in room B, such as the color of the carpet. Then the robot can reach room B by engaging the left-wall follower with the room B detector as the termination condition for the program.

Map-based (or model-based) navigation

In map-based navigation , the robot explicitly attempts to localize by collecting sensor data, then updating some belief about its position with respect to a map of the environment.

The advantage

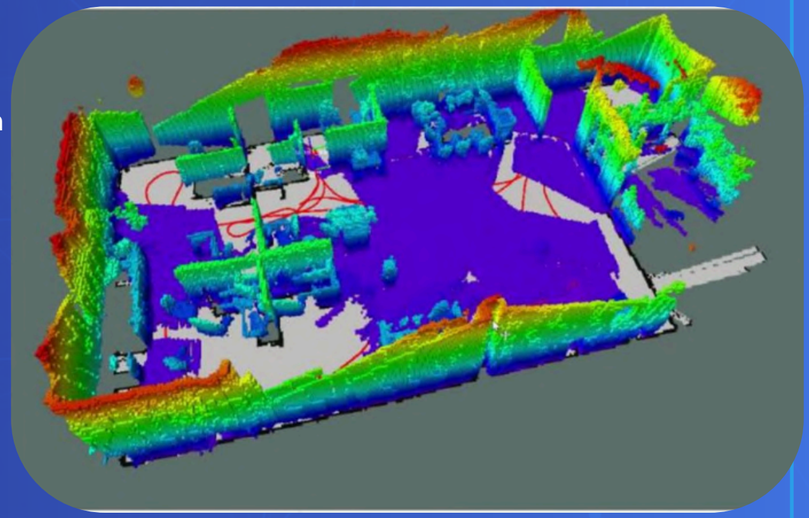
• The explicit, map-based concept of position makes the system's belief about position transparently availableto the human operators.

• The existence of the map itself represents a medium for communication between human and robot: the human can simply give the robot a new map if the robot goes to a new environment.

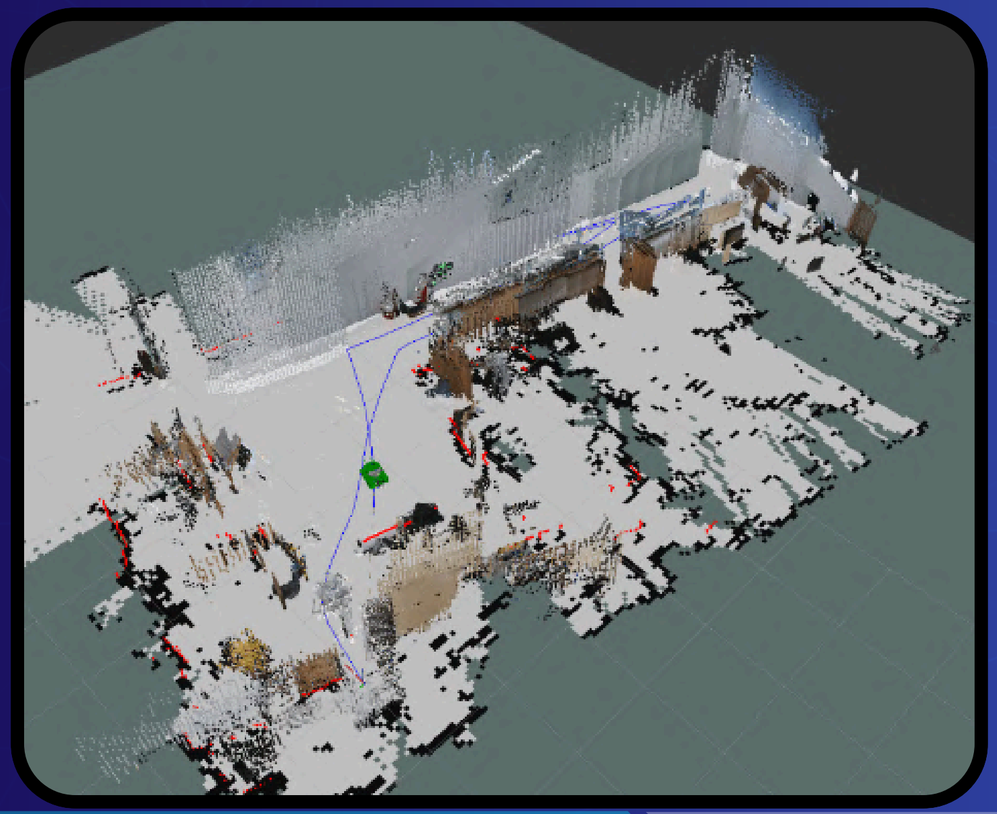
• The map, if created by the robot, can be used by humans as well, achieving two uses.

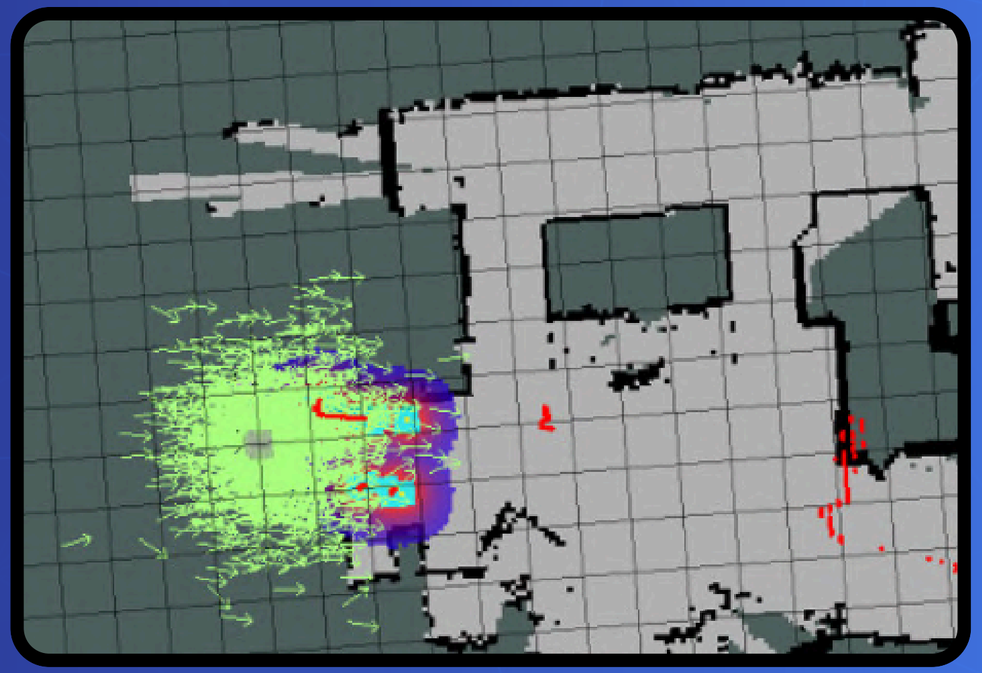
The risk

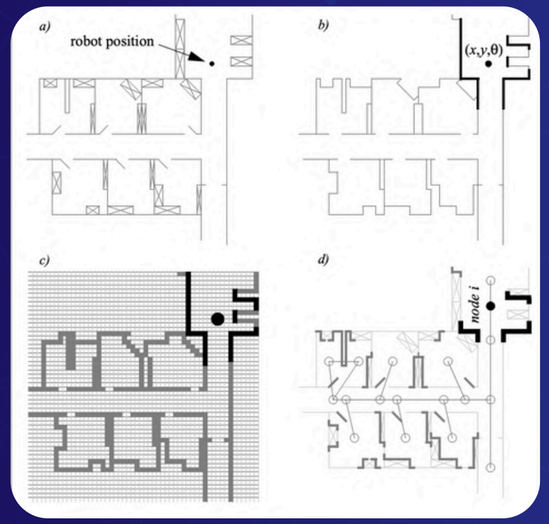
• An internal representation, rather than the real world itself, is being constructed and trusted by the robot. Ifthat model diverges from reality (i.e., if the map is wrong), then the robot's behavior may be undesirable,even if the raw sensor values of the robot are only transiently incorrect

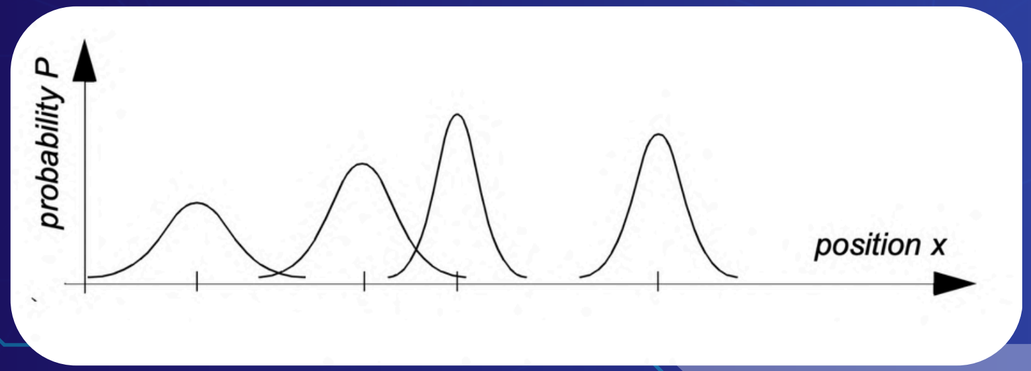


Map-based (or model-based) navigation









Belief Representation

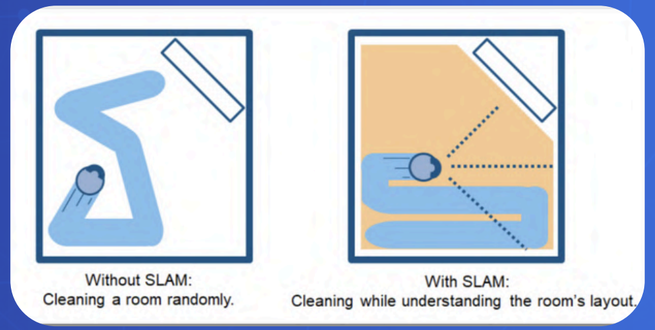
Just as decision-making is facilitated by a single-position hypothesis, so updating the robot's belief regarding position is also facilitated, since the single position must be updated by definition to a new, single position. The challenge with this position update approach, which ultimately is the principal disadvantage of single-hypothesis representation, is that robot motion often induces uncertainty due to effector and sensor noise. Therefore, forcing the position update process to always generate a single hypothesis of position is challenging and, often, impossible.

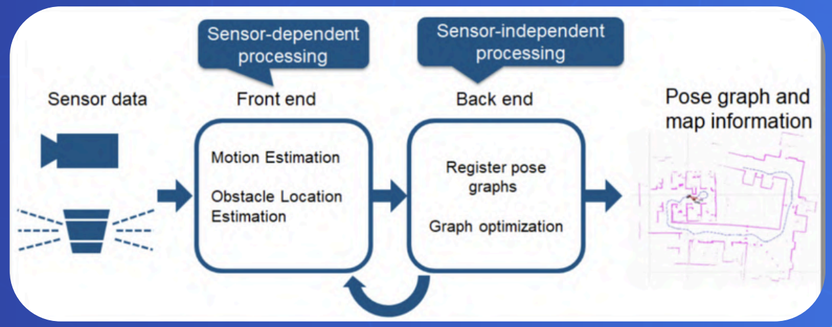
SLAM why important?

SLAM (simultaneous localization and mapping) is a method used for autonomous vehicles that lets you build a map and localize your vehicle in that map at the same time. SLAM algorithms allow the vehicle to map out unknown environments. Engineers use the map information to carry out tasks such as path planning and obstacle avoidance .

Consider a home robot vacuum. Without SLAM, it will just move randomly within a room and may not be able to clean the entire floor surface. In addition, this approach uses excessive power, so the battery will run out more quickly. On the other hand, robots with SLAM can use information such as the number of wheel revolutions and data from cameras and other imaging sensors to determine the amount of movement needed. This is called localization . The robot can also simultaneously use the camera and other sensors to create a map of the obstacles in its surroundings and avoid cleaning the same area twice. This is called mapping .

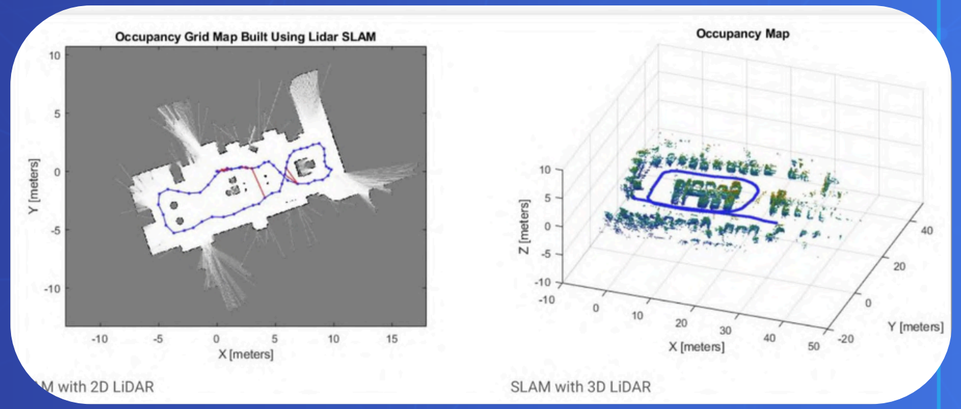
SLAM is useful in many other applications such as navigating a fleet of mobile robots to arrange shelves in a warehouse, parking a self-driving car in an empty spot , or delivering a package by navigating a drone in an unknown environment .





WHY LIDAR?

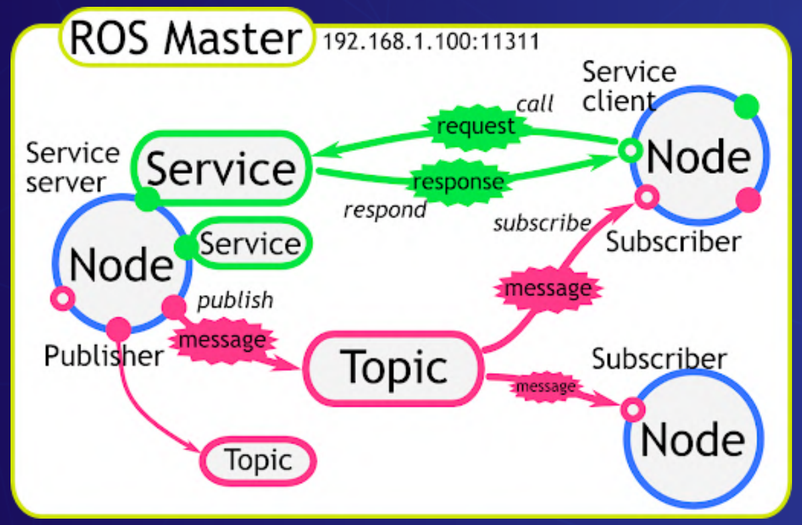
Light Detection And Ranging (LiDAR) is a method that primarily uses a laser sensor (or distance sensor).
Compared to cameras, ToF, and other sensors, lasers are significantly more precise, and are used for applications with high-speed moving vehicles such as self-driving cars and drones. The output values from laser sensors are generally 2D (x, y) or 3D (x, y, z) point cloud data. The laser sensor point cloud provides high-precision distance measurements and works very effectively for map construction with SLAM. Generally, movement is estimated sequentially by matching the point clouds. The calculated movement (traveled distance) is used for localizing the vehicle. For lidar point cloud matching, registration algorithms such as iterative closest point (ICP) and normal distributions transform (NDT) algorithms are used. 2D or 3D point cloud maps can be represented as a grid map or voxel map .



M with 2D LiDAR

SLAM with 3D LiDAR

HOW ROS communication?



communication between different nodes is done using a publish-subscribe model, where:

- A node publishes messages to a topic.
- Another node subscribes to the topic to receive the messages.

This allows different parts of a robot (like sensors, controllers, and actuators) to exchange information efficiently.

ROS topic is a named communication channel used to transfer messages between nodes.

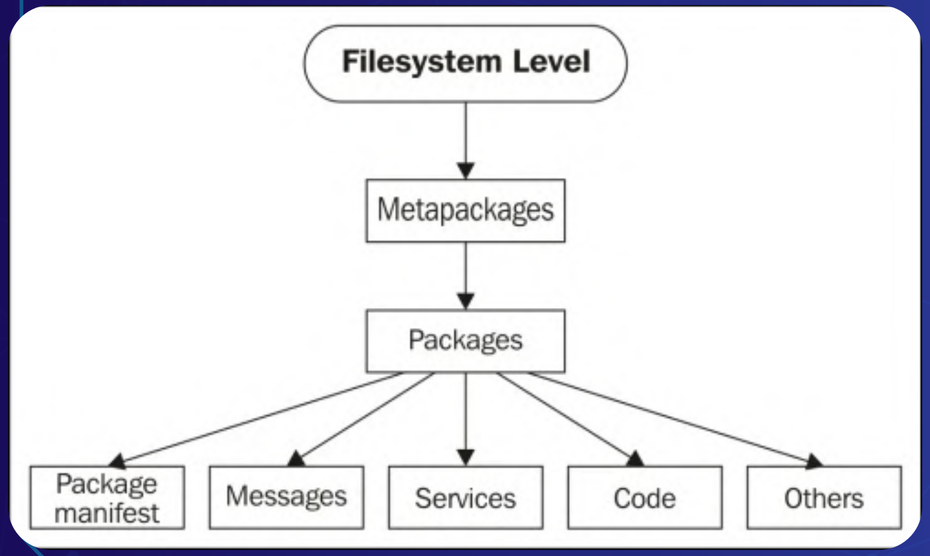
- Topics are asynchronous, meaning they don't wait for a response.
- Each topic has a message type (e.g., sensor data, velocity commands).
- Multiple nodes can publish or subscribe to the same topic.

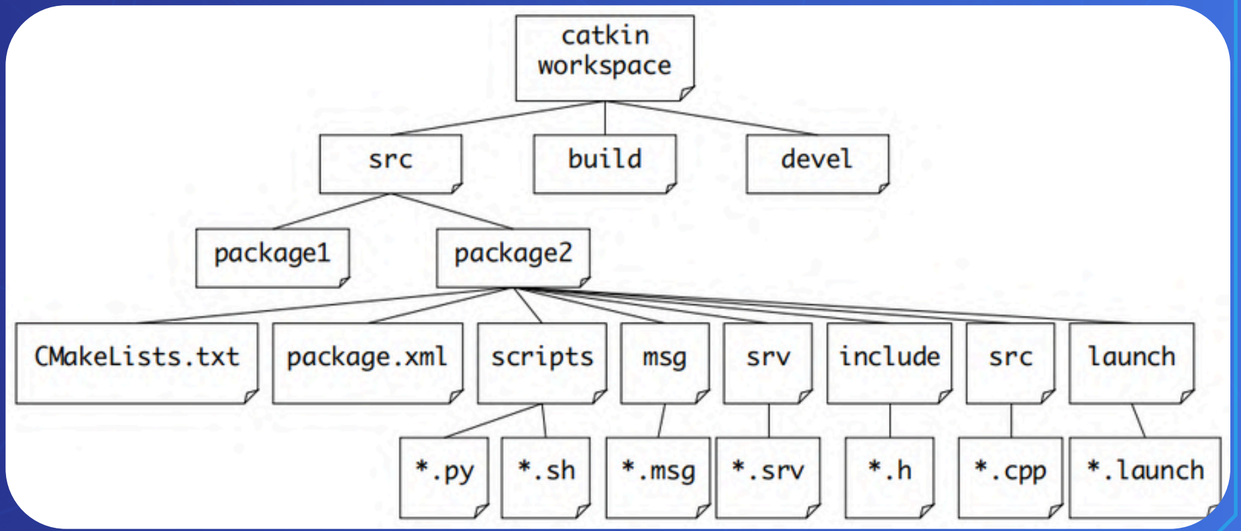
publisher node sends data to a topic.

- A node can publish sensor data, motor commands, or status updates.
- Multiple nodes can publish to the same topic.

subscriber node listens to a topic and processes incoming messages.

HOW ROS communication?





ROS
AI INTEGRATION

ROS provides a framework that allows AI algorithms to be integrated into robotic systems. AI can enhance perception, decision-making, control, and interaction in robotics.

Computer Vision & Perception

- AI-based vision models help robots recognize objects, people, and environments.
- Used for navigation, grasping, and autonomous movement.
- Example: Using YOLO or OpenCV with ROS to detect objects.

Deep Learning for Motion & Control

- AI can optimize motion planning and robotic control.
- Used for dynamic obstacle avoidance and grasping objects.
- Example: Reinforcement Learning (RL) for self-learning movement strategies.

TensorFlow/PyTorch with ROS

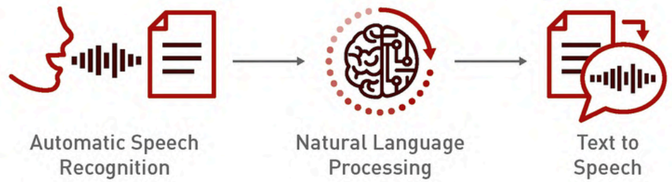
- TensorFlow and PyTorch provide deep learning capabilities for image recognition, speech processing, and decision-making.
- AI models are loaded into ROS nodes to process real-time data.

ROS
AI INTEGRATION

- NLP & Chatbots for Voice Interaction

AI models like GPT or Rasa can enable speech-based interaction in ROS robots.
Works with text-to-speech (TTS) and speech recognition (STT) systems.

PIPER



SAKOLRAJWITTAYANUKUL

Thank You.

FOR YOUR ATTENTION