Design & Build

Group 47

Contents

01
Project Objectives

02
Software Section

03
Hardware Section

04
Team Division of Labor

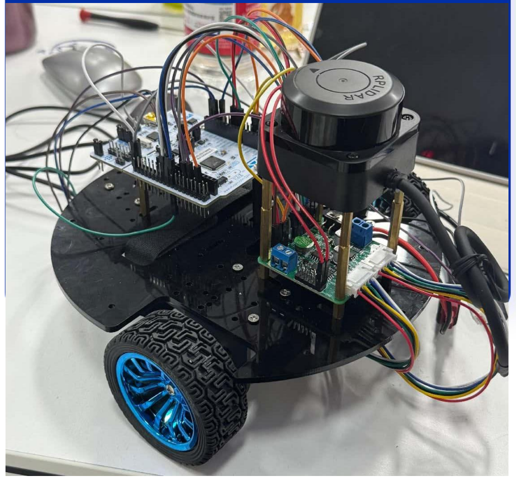
Project Objectives

Define the main goals and capabilities of the project

Primary Goal

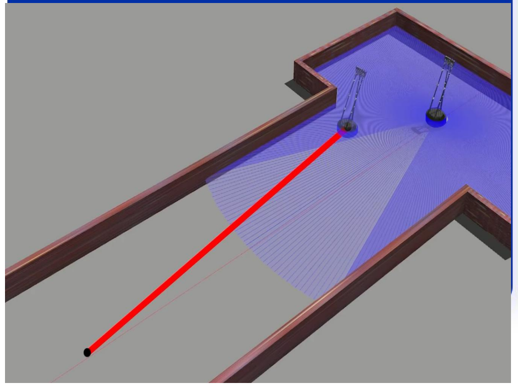
Autonomous Mobile Robot

Design and build a robot that can operate independently in a maze.



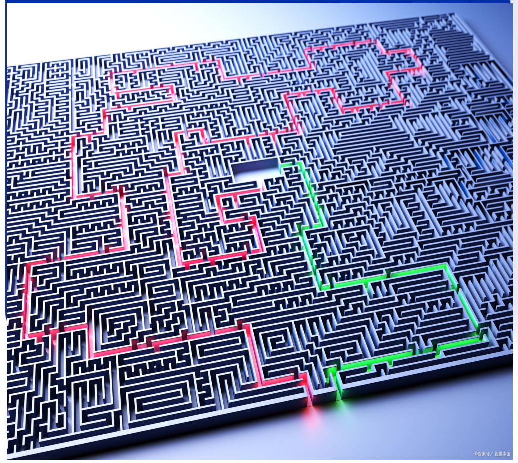
Mapping and Navigation

Enable the robot to map and navigate an unknown maze effectively.



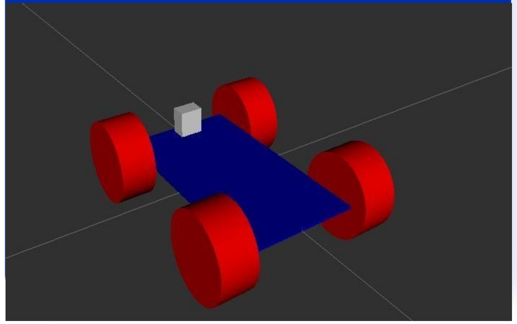
Unknown Environment

The robot should function well in unfamiliar maze settings.

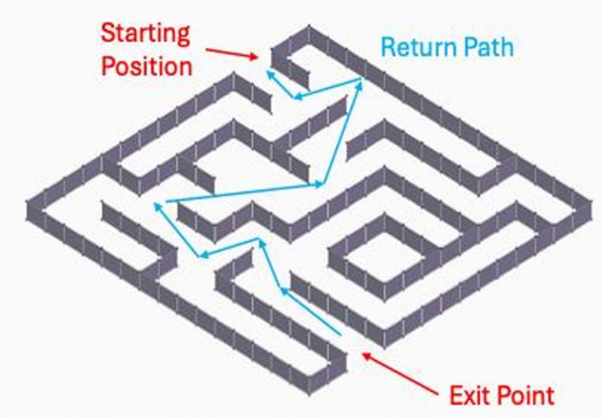


Human Intervention

Operate without any human interaction from start to finish.



Key Capabilities



Real - time Mapping

Use a 2D LiDAR sensor to perform real - time environment mapping.

Path Calculation

Calculate and navigate a path back to the original starting position.

Describe the essential functios of the robot

Autonomous Exploration

Autonomously explore the maze to find the designated exit point.

Full Autonomy

Complete all tasks without external human control or intervention.

Software Section

Explore the software components and algorithms used

Visualization Interface

LiDAR Scan Points

Blue dots represent the LiDAR scan hit points in the environment.

Exploration Target

A pentagram indicates the robot's current exploration target location.

Actual Travel Path

The red curve shows the robot's actual travel path through the maze.

Planned Path Points

Blue line segments are planned path points generated by the A* algorithm.

Frontier Points

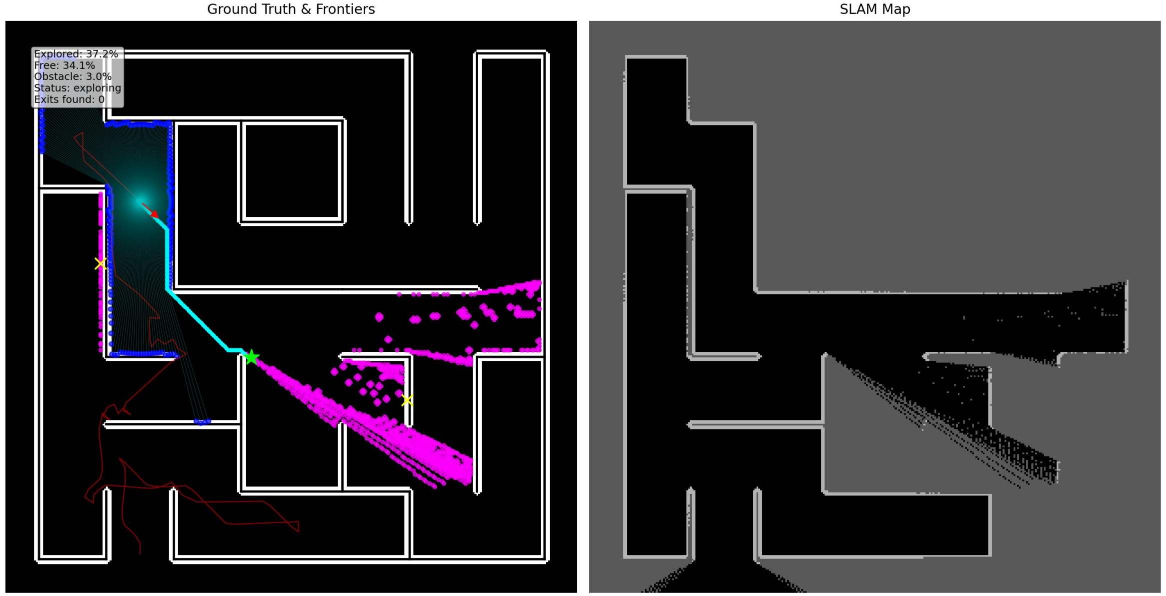
The pink area represents frontier points, indicating unknown regions.

Clustering Points

Yellow cross marks denote clustering points for efficient exploration.

Other Marks

There are also explored, free, obstacle, status, and exits found marks.



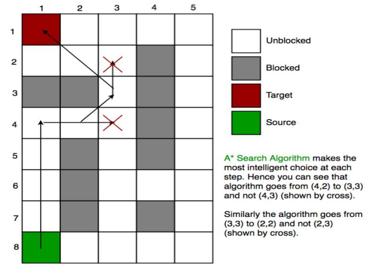
Explored: 37.2%
Free: 34.1%
Obstacle: 3.0%
Status: exploring
Exits found: 0

Software Innovations

Clustering Algorithm

Groups nearby frontier points and selects central points as targets.

A* Algorithm



A* Search Algorithm makes the most intelligent choice at each step. Hence you can see that algorithm goes from (4,2) to (3,3) and not (4,3) (shown by cross).

Similarly the algorithm goes from (3,3) to (2,2) and not (2,3) (shown by cross).

A heuristic pathfinding algorithm combining cost and estimated distance.

Sliding Window

Dynamically maintains a subset of data for efficient real - time computation.

Path Planning and Obstacle Avoidance

Achieve shortest - path navigation and prevent collisions at turns.

Basic Tools

ROS 2

A robust framework for robot software development.

rclpy, actionlib, tf2

Important ROS 2 libraries for various functionalities.

Cartographer

Used for real - time mapping and localization.

PointCloudWithIntensities, Candidate2D, MapLimits

Key data structures in Cartographer.

explore_lite

A tool for maze exploration and related tasks.

ros1_names, Costmap2DClient, map_merge

Components for working with maps and data.

Nav2

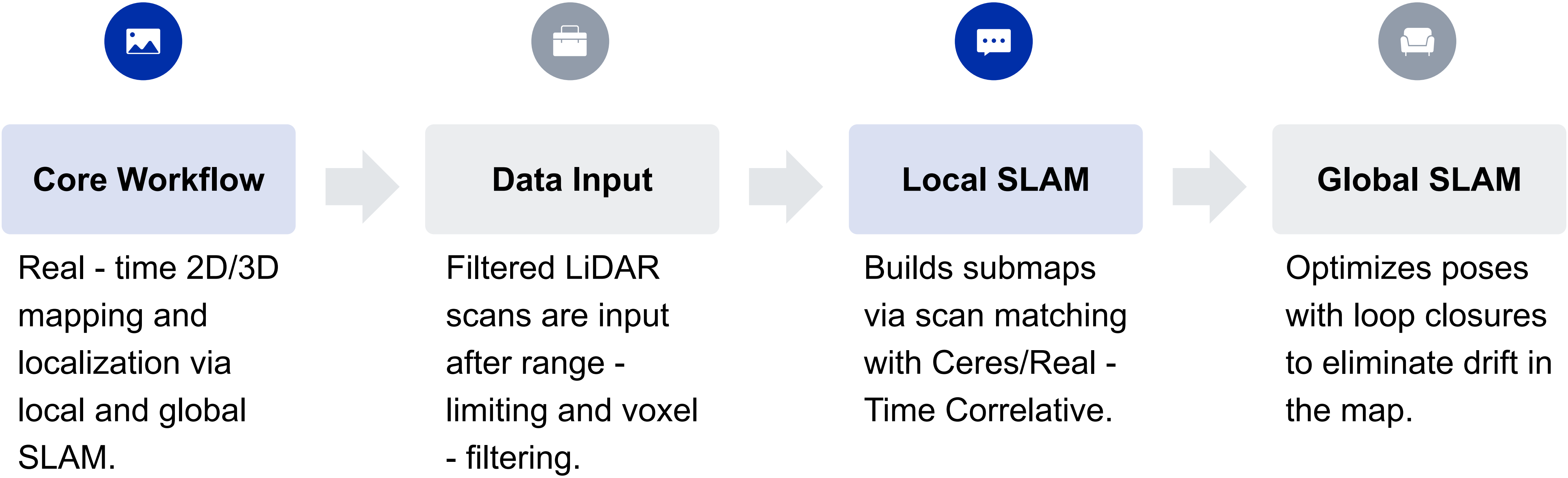
Provides navigation capabilities for the robot.

Others

Includes sensor_msgs, nav_msgs, serial/pyserial, and threading.

SLAM with Cartographer

Detail the SLAM process using Cartographer



Core Workflow

Real - time 2D/3D mapping and localization via local and global SLAM.

Data Input

Filtered LiDAR scans are input after range - limiting and voxel - filtering.

Local SLAM

Builds submaps via scan matching with Ceres/Real - Time Correlative.

Global SLAM

Optimizes poses with loop closures to eliminate drift in the map.

Exit - to - Entrance Return Control Logic

Explain the logic for returning from the exit to the entrance

01

Goal Switch

Triggered when reaching the exit, switches target to entrance coordinates.

02

Path Planning

Uses Nav2's NavigateToPose action to generate the optimal return path.

03

Obstacle Handling

Checks for obstacles in real - time and re - plans if blocked or stuck.

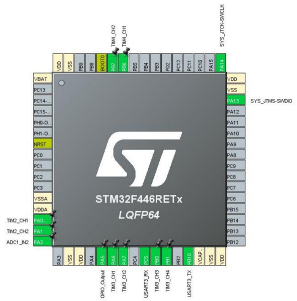
Hardware Section

Present the hardware components and control mechanisms

Hardware Components List

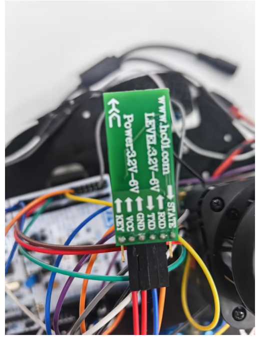
Main Control Chip (MCU)

Uses the STM32F446RE as the central control unit of the project.



Bluetooth Module

The HC - 04 module enables wireless communication with a smartphone.



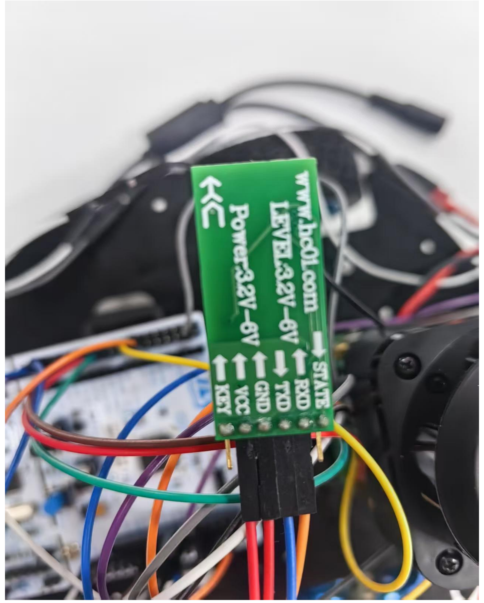
Motor Driver

The AT8236 motor driver controls the robot's propulsion system.

Motor & Encoder

520 geared DC motors with Hall effect encoders for precise speed control.

Core Pin Configuration - Bluetooth Module



Describe the pin connections and functions of the Bluetooth module

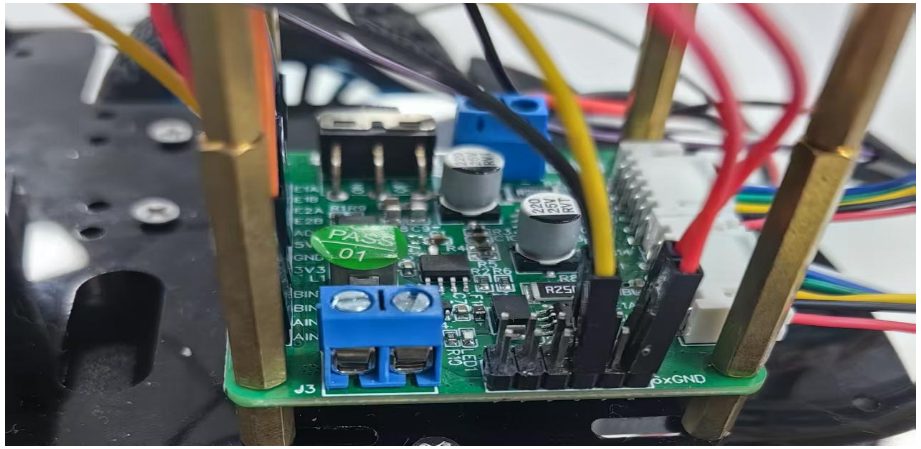
Pin Connection

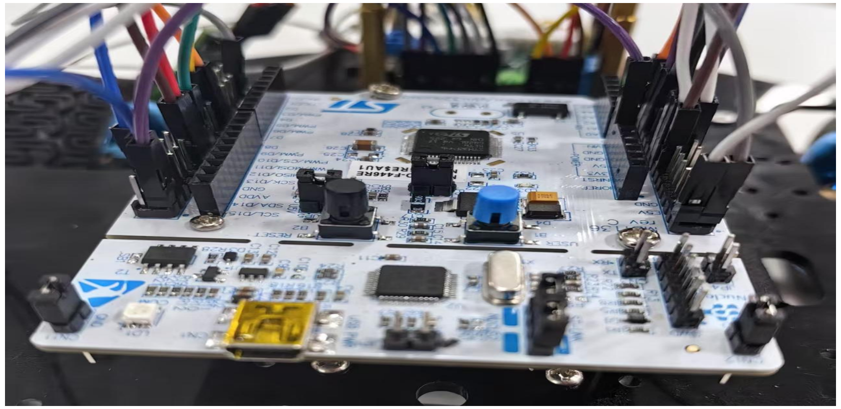
The Bluetooth module interfaces with the STM32 via USART3 (PC5 and PB10).

Communication Function

Receives commands over Bluetooth, like "F" for forward movement.

Core Pin Configuration - 520 Geared Encoder DC Motor





Pin Connection

Left motor connected to TIM3_CH1 (PA6) and CH2 (PA7), right motor to TIM3_CH1 (PB0) and CH2 (PB1).

VS

Function of Calculating Rotational Speed

Uses timer encoder mode to count pulses and calculate motor speed.

Code Modules and Their Functions

Main Logic Module (main.c)

Schedules all modules for a "command → motion" closed loop.

Motor Driver Module (tim.c, adc.c)

Handles motor driving, including rotation and speed calculation.

PID Algorithm Module (pid.c)

Implements PID formula for closed - loop speed control.

Bluetooth Communication Module (usart.c, dma.c)

Parses characters received via USART for command interpretation.

Core Technologies

1 Command Parsing Flow

Initial configuration, serial port transmission, and data reception & processing.

2 Advantages

Reduces CPU load, enhances reliability, and enables efficient data transmission.

3 Core Formula

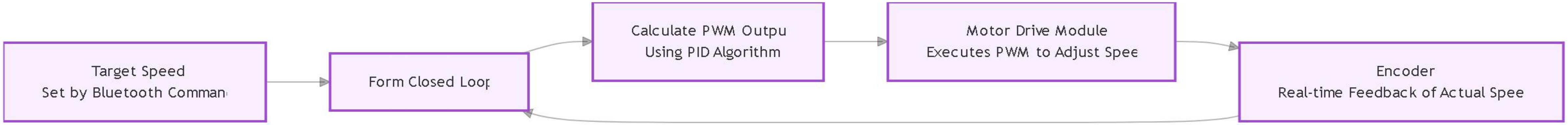
PWM Output = Kp × Error + Ki × ∫Error + Kd × d(Error)/dt for speed control.

4 Speed Closed - Loop Implementation Flow

Determines target speed, calculates error, and uses PID for closed - loop control.

5 Solved Speed Fluctuation Problem

Stabilizes motor speed through closed - loop regulation with PID.



Target Speed
Set by Bluetooth Comman

Form Closed Loop

Calculate PWM Outpu
Using PID Algorithm

Motor Drive Module
Executes PWM to Adjust Spee

Encoder
Real-time Feedback of Actual Spee

Team Division of Labor

Show the division of tasks among team members

Name	Major	Task	Contribution Rate
Zelang Wen(Leader)	Software	Oversees overall software architecture, optimizes Cartographer-based SLAM, implements exit-to-entrance return logic, and coordinates software-hardware compatibility.	15%
Jingping Yan	Software	Develops the visualization interface, implements frontier point clustering, integrates explore_lite, and ensures real-time map data display.	13%
Conghao Li	Software	Codes and optimizes the A* pathfinding algorithm, implements sliding window for efficient computation, and adds collision avoidance at turns.	13%
Jinyu Zhou	Software	Sets up the ROS 2 framework, processes Cartographer data structures, develops serial communication, and integrates multi-threading.	13%
Dingyi Zhang(Leader)	Hardware	Leads hardware design, selects key components (STM32F446RE, HC-04), and coordinates hardware module integration and pin configuration.	13%
Jiahang Li	Hardware	Focuses on motor driver (AT8236) and encoder (520 geared motor) integration, debugs motor speed control, and assists with hardware circuit testing.	13%
Sicheng Meng	Hardware	Testing and adjustment	5%
Tianda Wang	Hardware	Testing and adjustment	5%
Yanlin Zhang	Hardware	Testing and adjustment	5%
Xintong Liu	Hardware	Testing and adjustment	5%

Thanks

2025.10