Autonomous Mobile Robot for Maze Mapping and Navigation with 2D LiDAR
Group16

Group Member:

Telcom+EIE:	Zixin Xiong	IOT+IST:	Taiyu Chen
	Can Chen		Jiayou Zhu
	Xinshuang Liu		Jiayi Zou
	Yuancheng Li		Wenrui Si
	Shengcheng Xia		Chenglin Li
	Ke Lyu		

2025/10/23

Content

Task Allocation

The contribution rates of the software group and the hardware group are calculated separately.

Telcom+EIE		
Zixin Xiong	19%	Coordinate project development and participate in the interface development combining software and hardware. Be responsible for the encapsulation and integration of all hardware modules; mainly responsible for the development of Bluetooth module, PID module and radar module codes, and participate in the connection and testing of pins; be responsible for the writing of documents and PPTs.
Can Chen	19%	Assemble the small vehicle, mainly responsible for pin connections and testing, motor control debugging and decoder data collection and analysis, IMU module data collection, PPT production, and participating in interface testing combined with the software and hardware team.
Xinshuang Liu	25%	Arduino and VScode environment setup, non-blocking code development. Mainly responsible for the development of the interface between software and hardware, PID module, radar module, Bluetooth module, and the collection and analysis of motor decoder data.
Yuancheng Li	19%	Assemble the car, be responsible for the development of the motor part, mainly undertake the testing work of all the codes of the team; design the appearance of the car, search for the library of the radar, and search for available code materials

Task Allocation

Telcom+EIE		
Shengcheng Xia	9%	Assemble the car, collect the materials, and participate in the discussion.
Ke Lyu	9%	Assemble the car, provide data for the radar section, and participate in the discussion.

Task Allocation

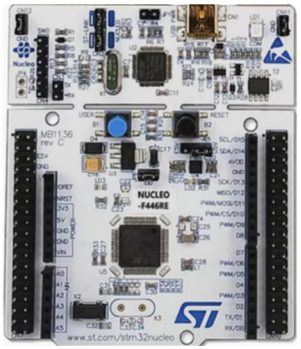
IOT+IST		
Taiyu Chen	40%	Coordinated project development, was responsible for developing navigation and obstacle avoidance algorithms, participated in the radar simulation process. Provided the textual content for the presentation slides of the software team.
Jiayou Zhu	40%	Responsible for the SLAM mapping algorithm, completed the part involving the integration of software and hardware, and wrote the Bluetooth instructions.
Jiayi Zou	6.67%	Participate in the discussion and provide the completion ideas.
Wenrui Si	6.67%	Participate in the discussion and provide the completion ideas.
Chenglin Li	6.67%	Participate in the discussion and provide the completion ideas.

PART 1

Work of Telcom+EIE

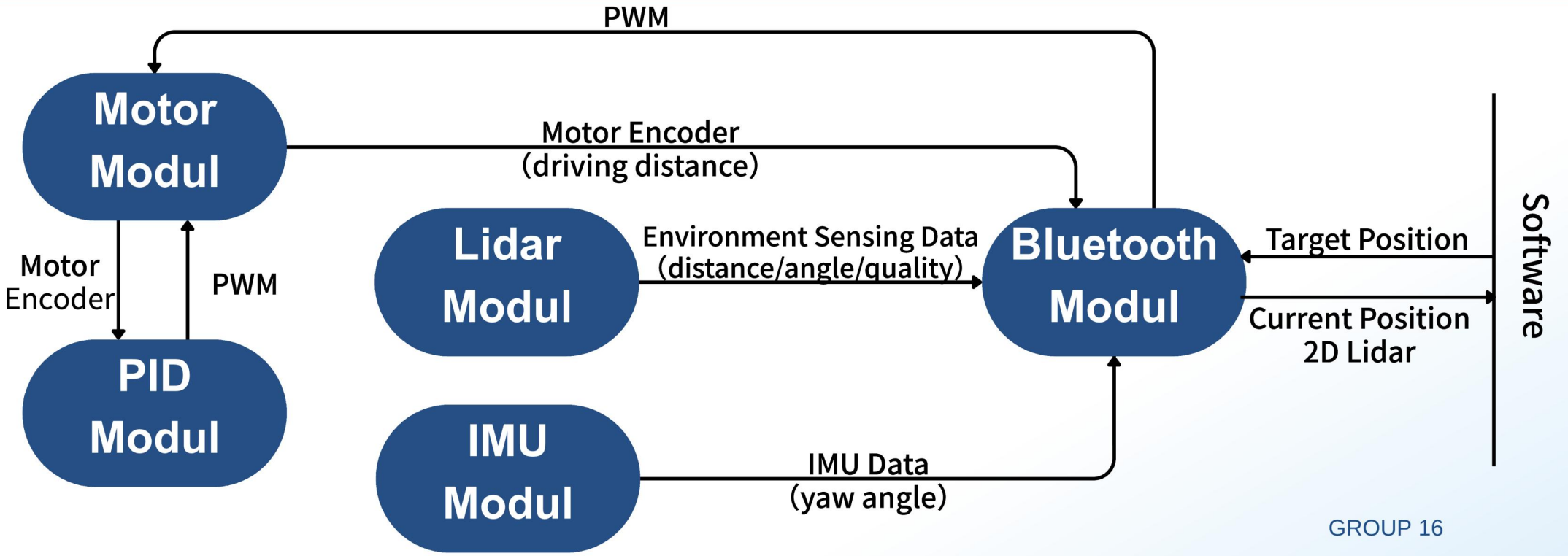
GROUP 16

Hardware connection



GROUP 16

Project Architecture



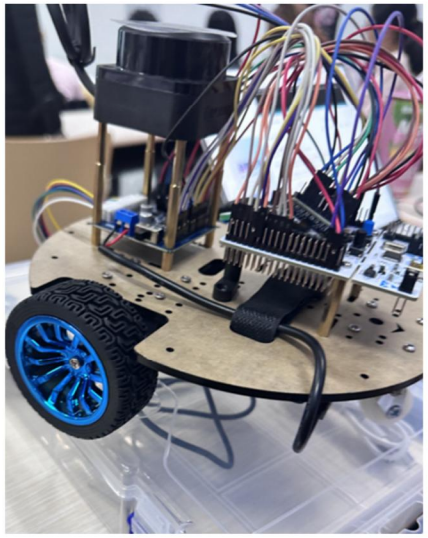
Main

——Integrated function

This is an intelligent main control system for small vehicles based on a non-blocking multi-task architecture. It schedules core tasks such as motor control, radar collection, and IMU processing through hardware timers in a time-sharing manner. It uses a state machine to implement Bluetooth command parsing and autonomous navigation functions, ensuring that all tasks are executed concurrently without any blocking delays. Thus, it achieves real-time and precise control of the vehicle's movement, environmental perception, and pose estimation.

Motor Module

——Drive the car to move

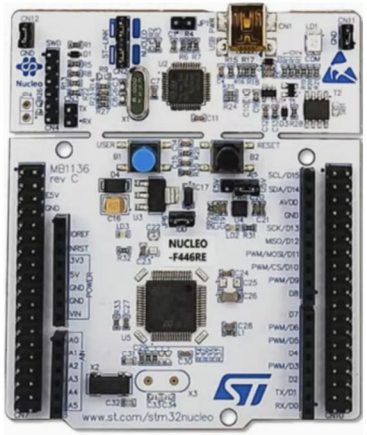


Microcontroller

Motor Driver

Encoder Motor

Motor Module



Microcontroller
Real-time sensor processing
Motor control
Low-level communication with the host PC

Motor driver
Controls the encoder DC motor for precise speed and direction

Encoder DC Motor
Provides locomotion with encoder feedback for speed control and odometry

Function

Control the car to move forward, turn left, turn right, and stop.

- The direction and speed can be controlled by setting the left and right angles and speeds respectively.
- For example, if you want to move forward, then set the speeds of the left and right wheels of the car to be the same.

```
// Motor control pin definitions
#define L_A PC8
#define L_B PB10
#define R_A PC7
#define R_B PB3

// Public function declarations
void Motor_Init(void);
void Motor_Forward(void);
void Motor_Backward(void);
void Motor_Right(void);
void Motor_Left(void);
void Motor_Stop(void);

// New function to set motor PWM and direction
void Motor_Set_Left_PWM(int pwm);
void Motor_Set_Right_PWM(int pwm);
void Motor_Set_Right_Speed(float speed);
void Motor_Set_Left_Speed(float speed);
void Motor_Control(float base_speed, float target_yaw);

// PID and serial logging functions (now part of Motor module)
void Motor_Tick(void);
//void sendToSerialPlot(float target, float actual);


// External declarations for PID variables
extern PID_TypeDef pidLeft, pidRight;
extern PID_TypeDef pidNewParams;
```

Motor.h

IMU Module

——Obtain the yaw angle

MPU6500 IMU
A 6-axis inertial measurement unit for orientation tracking and odometry support

```
Gyro (gx,gy,gz): -443, -863, 622
Total Distance: 17.75 m  Avg Speed: -0.34 m/s
Accel (ax,ay,az): -956, 2212, 3400
Gyro (gx,gy,gz): 79, -1733, -319
Total Distance: 17.87 m  Avg Speed: -0.41 m/s
Accel (ax,ay,az): 4724, -60, -22412
Gyro (gx,gy,gz): -307, 1760, -542
Total Distance: 17.99 m  Avg Speed: -0.37 m/s
Accel (ax,ay,az): 796, 5732, -32768
Gyro (gx,gy,gz): -2357, 1526, -15640
Total Distance: 18.03 m  Avg Speed: 0.00 m/s
Accel (ax,ay,az): 548, 5516, -12708
Gyro (gx,gy,gz): -1903, 2076, -18727
Total Distance: 18.03 m  Avg Speed: 0.00 m/s
Accel (ax,ay,az): 168, 912, -16236
Gyro (gx,gy,gz): -775, 233, -6215
Total Distance: 18.03 m  Avg Speed: 0.00 m/s
```

IMU数据收集

Queen Mary University of London

Function

Provide yaw angle

- Motion data is collected through the MPU6500 sensor, and sensor fusion is performed using the Madgwick filtering algorithm. Finally, the calibrated and coordinate system corrected real-time yaw angle is output, providing precise attitude perception for the motion control system.

```
// 寄存器地址
#define ACCEL_XOUT_H 0x3B
#define GYRO_XOUT_H  0x43
#define PWR_MGMT_1   0x6B
#define ACCEL_CONFIG 0x1C
#define GYRO_CONFIG  0x1B

// 换算系数
#define ACCEL_SCALE_FACTOR 4096.0f // 对于 ±8g 量程
#define GYRO_SCALE_FACTOR  65.5f   // 对于 ±500 °/s 量程
#define G_VALUE            9.80665f // 重力加速度 (m/s^2)

// 使用结构体来存储一个MPU6500传感器实例的所有数据
typedef struct {
  // 存储校准偏移量 (raw)
  float accel_offset_x, accel_offset_y, accel_offset_z;
  float gyro_offset_x, gyro_offset_y, gyro_offset_z;

  // 存储最终处理过的数据
  float ax, ay, az; // 加速度 (g)
  float gx, gy, gz; // 角速度 (°/s)
} MPU6500_t;


// --- 声明与MPU6500操作相关的函数 ---
void mpu6500_begin();
void mpu6500_calibrate(MPU6500_t* mpu);
void mpu6500_read_data(MPU6500_t* mpu);
void MPU6500_Init(); // 新增的初始化函数
void MPU6500_Tick(); // 新增的循环任务函数
float MPU6500_GetYaw();
```

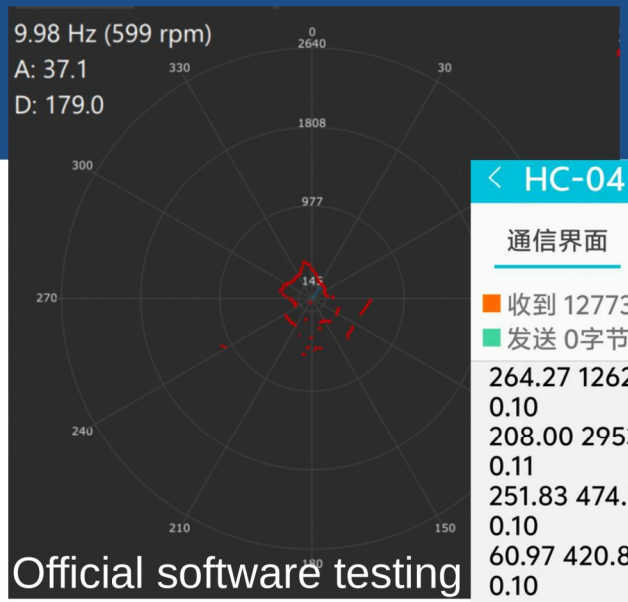
IMU.h

RPLIDAR C1

——Collect environment sensing data

2D LiDAR sensor

Captures angle and distance data from the surrounding environment



Official software testing

Actual data collection

Queen Mary University of London

Function

Control the laser radar to collect environmental data, and then send the processed distance and angle information via Bluetooth.

- Control the RPLIDAR radar sensor through UART communication to control the start and stop of the radar, obtain the health status, and initiate scanning. Real-time parse the point cloud data of the laser radar. After quality filtering and distance screening, the valid angle and distance information is transmitted via Bluetooth. At the same time, provide data statistics and buffer management functions.

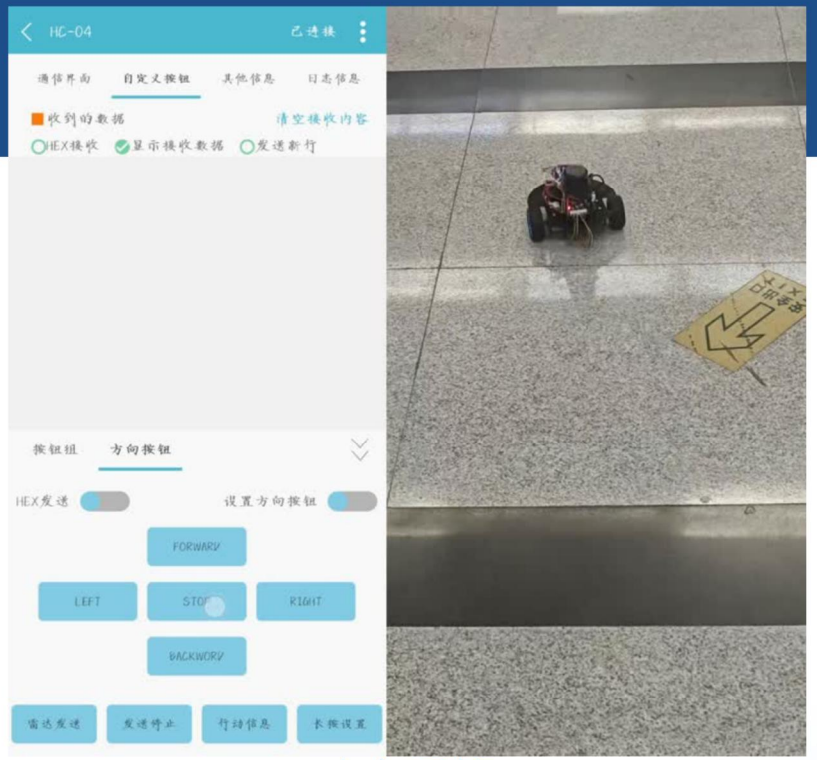
```
/* ===== 过滤阈值（可在编译时通过 -D 覆盖）===== */
#ifndef RADAR_MIN_QUALITY
#define RADAR_MIN_QUALITY 30       // 最小质量
#endif
#ifndef RADAR_MIN_DIST_MM
#define RADAR_MIN_DIST_MM 1.0f     // 最小距离(mm)
#endif
#ifndef RADAR_MAX_DIST_MM
#define RADAR_MAX_DIST_MM 6000.0f  // 最大距离(mm)
#endif

// 全局UART实例在 Radar.cpp 定义（F446RE: UART5 RX=PD2, TX=PC12）
extern HardwareSerial RPSerial;

void Radar_init(void);
// 读取并解析标准采样节点；toSerial=true 时仅打印通过过滤的点
void Radar_read(bool toSerial = true);

// 运行时控制：是否通过蓝牙发送雷达数据
void Radar_EnableTx(bool enable);
bool Radar_IsTxEnabled(void);
```

Radar.h

Bluetooth

——Remote control car, data reception

实现和主机之间的无线通信



小车遥控

数据接收

Queen Mary University of London

Function

Build a two-way wireless communication bridge between the car and the PC/mobile devices

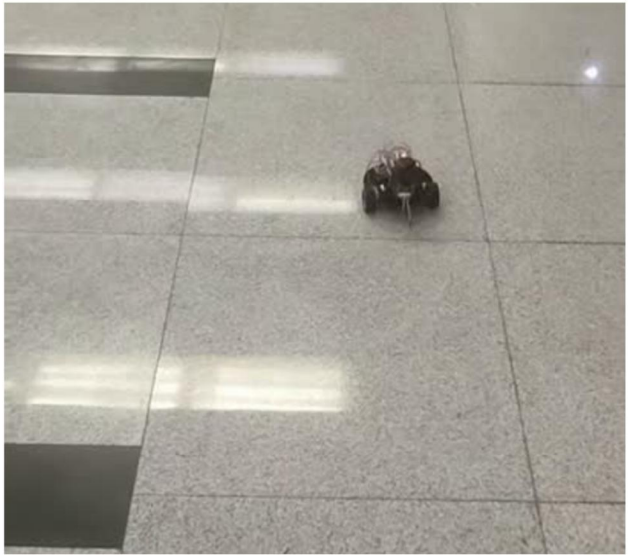
- This Bluetooth module serves as a two-way wireless communication bridge between the vehicle and the PC/mobile terminal. It is responsible for real-time uploading of sensor data such as radar point clouds and IMU orientation to the client for visual display. At the same time, it receives and interprets motion control instructions and system query commands from the client, enabling remote monitoring and precise control of the vehicle's status.

```
void Bluetooth_Init();
bool Bluetooth_Available();
char Bluetooth_Read();
void parseAndProcessData(char* data);
void processBluetoothData();
```

Bluetooth.h

PID

——Precisely control the car



Queen Mary University of London

Function

Precisely control the movement of the car

- The classic position-based PID control algorithm has been implemented. The control quantity is calculated through the three steps of proportion, integration, and differentiation. It can precisely adjust according to the deviation between the target value and the actual measurement value, and has output limiting and parameter safety protection functions. It is used to achieve closed-loop control of systems such as motor speed and cart posture.

```c
// PID 控制器结构体
typedef struct {
    float Kp;       // 比例系数
    float Ki;       // 积分系数
    float Kd;       // 微分系数

    float setpoint;     // 目标值
    float output;       // PID 输出
    float output_max;   // 输出最大值
    float output_min;   // 输出最小值

    // 位置式PID需要的变量
    float integral;     // 积分累积项
    float last_error;   // 上一次误差
} PID_TypeDef;

// PID 初始化
void PID_Init(PID_TypeDef *pid, float Kp, float Ki, float Kd, float output_min, float output_max);

// PID 计算（位置式PID）
float PID_Calculate(PID_TypeDef *pid, float measured);
```

PID.h

PART 2

Work of IOT+IST

GROUP 16

Mission & Success Metrics

Mission Overview

Our mission is to autonomously map, explore, find the exit, and return in an unknown maze. The robot must navigate without human intervention, ensuring full autonomy.

Current Status

We have successfully integrated the communication protocol, SLAM for real-time mapping, exploration loop, homing capability, GUI, and logging. But we need to use the map file.

Key Algorithms

Our key technologies include real-time occupancy grid mapping, frontier-based exploration, A* path planning combined with DWA for local navigation, and replanning on anomaly detection.

System Blueprint & Data Flow

1.End-to-End Pipeline Overview

Data Flow

Sensor data from odometry and LiDAR is processed through SLAM to generate real-time maps and robot pose. These inputs drive the exploration and planning modules.

Planning Module

The planning module uses A* for global path planning and DWA for local obstacle avoidance, ensuring smooth and safe navigation.

Exploration Module

The exploration module identifies frontiers in the map and selects the next best viewpoint for the robot to move towards, ensuring efficient coverage.

Control Module

The control module translates planned paths into velocity commands for the robot's chassis, ensuring precise execution.

System Blueprint & Data Flow

2.Module Boundaries & Responsibility Matrix

Protocol Adapter	SLAM Module	Planning Module
Responsible for communication between the robot and external systems, ensuring reliable data transfer.	Generates real-time maps and robot pose from sensor data, providing the foundation for navigation.	Plans the robot's path using global and local planning algorithms, ensuring efficient and safe navigation.

Communication & Interfacing

——Serial Protocol Specification

01

Up-link Data

Up-link packets carry robot pose (x, y, θ, timestamp) and LiDAR data (angle, distance in mm, quality) to the central system.

02

Down-link Data

Down-link packets transport velocity commands or discrete motion primitives (F, S, Lp, Rp, M<d>, E…) to the robot.

03

Reliability Features

The protocol includes timeout-reconnect, CRC checksum, and anomaly-frame discard to ensure robust communication.

Mapping & Localization Core

Occupancy Grid SLAM Algorithm

1

Grid Update

The occupancy grid is updated using ray-casting from LiDAR scans, with odometry providing a motion prior for accurate mapping.

2

Pose Correction

Scan-to-map matching using distance-field scoring provides pose corrections, ensuring localization accuracy.

Mapping & Localization Core

Unknown Area Handling & Visualization

1 Unknown Area Management

Unknown areas are rendered in grey, ensuring they are not mistaken for free space until confirmed by sensor data.

2 Real-time Visualization

Real-time display layers include point cloud, grid, trajectory, and frontiers, refreshed at 10 Hz for seamless monitoring.

Exploration & Exit Detection

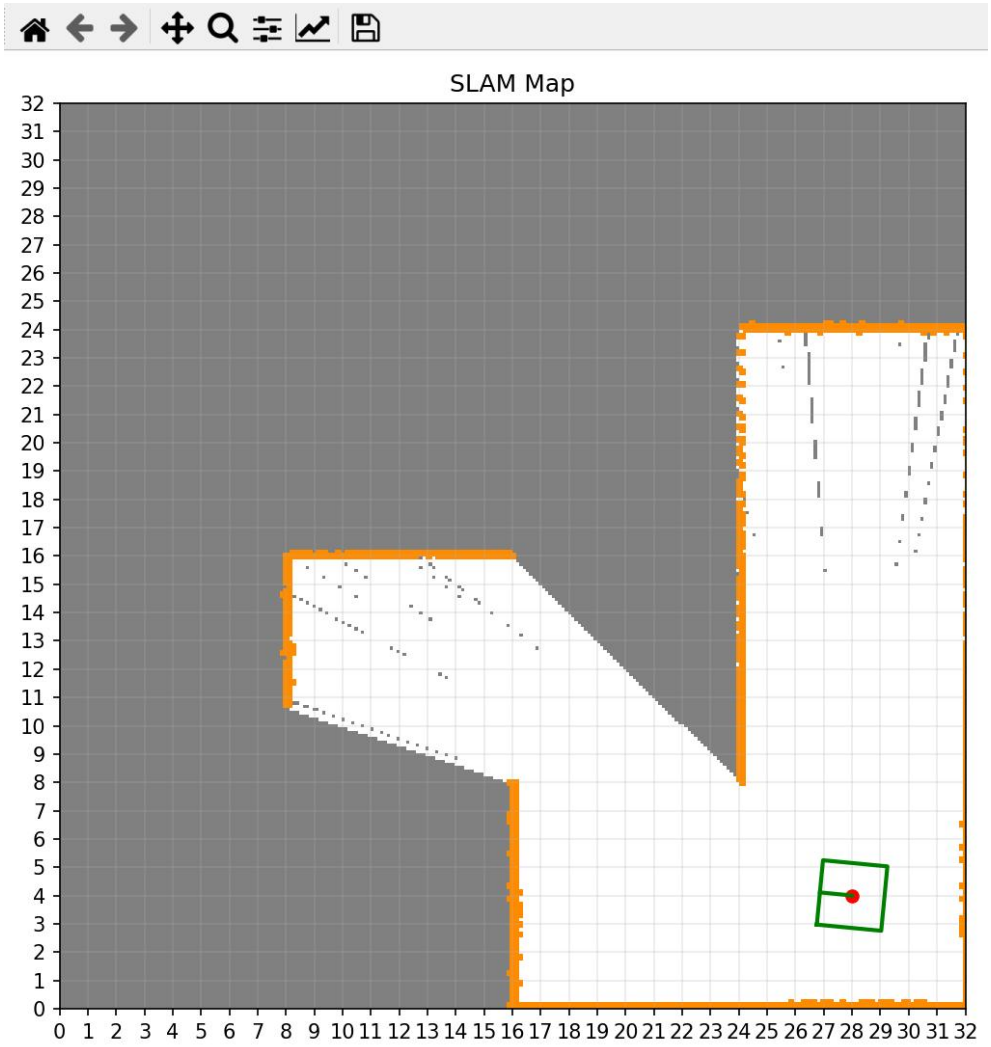
Frontier-Driven Exploration Strategy(Software simulation)

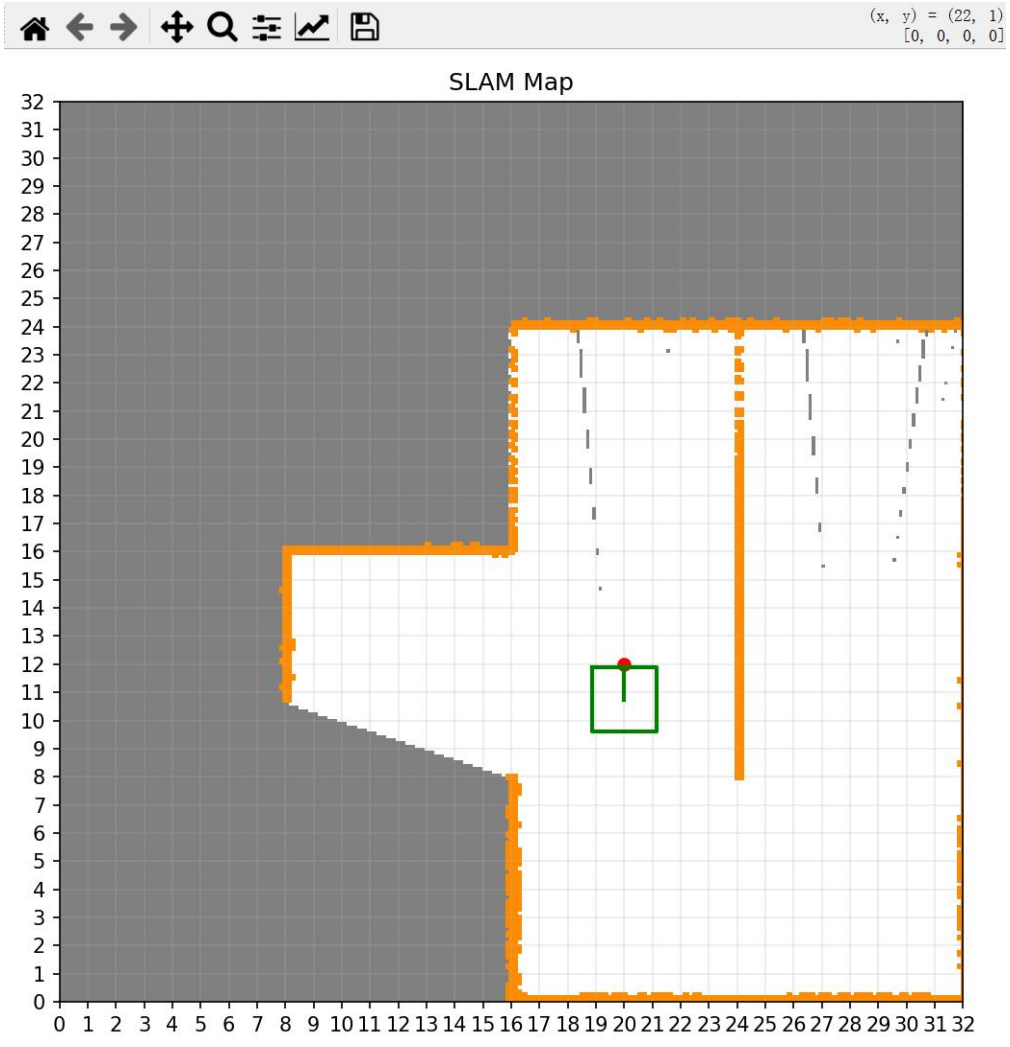
Selection Criteria

Frontiers are scored based on travel cost, expected information gain, and heading change penalty to select the next best viewpoint.

Execution Rhythm

The robot drives 0.5 m, pauses for a 270° scan, and repeats. A 360° spin is triggered if necessary to ensure full coverage.


SLAM Map
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32


(x, y) = (22, 1)
[0, 0, 0, 0]
SLAM Map
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32

Exploration & Exit Detection

Exit Detection & Arrival Logic

Exit Candidate Validation

Exit candidates are validated by checking unknown ratio, line-of-sight reachability, and passage width to ensure reliability.

Motion Planning & Homing

——Global & Local Planner Coupling

Global Planning

The global planner uses A* on an inflated grid to generate a sparse waypoint sequence, ensuring efficient pathfinding.

Local Planning

The local planner uses DWA to select feasible velocity commands, ensuring real-time obstacle avoidance and smooth navigation.

Replanning Mechanism

The system triggers replanning if DWA cannot find a feasible path for 0.5 s, ensuring robustness in dynamic environments.

Motion Planning & Homing

——Global & Local Planner Coupling

Strategy A: Path Reversal

Strategy A replays the driven path in reverse using a LIFO stack of pose snapshots, efficient but sensitive to path blockage.

Strategy B: Replanning

Strategy B treats the final map as static and runs A* from exit to start, immune to dynamic obstacles but computationally costlier.

Validation & Results

Experimental Setup

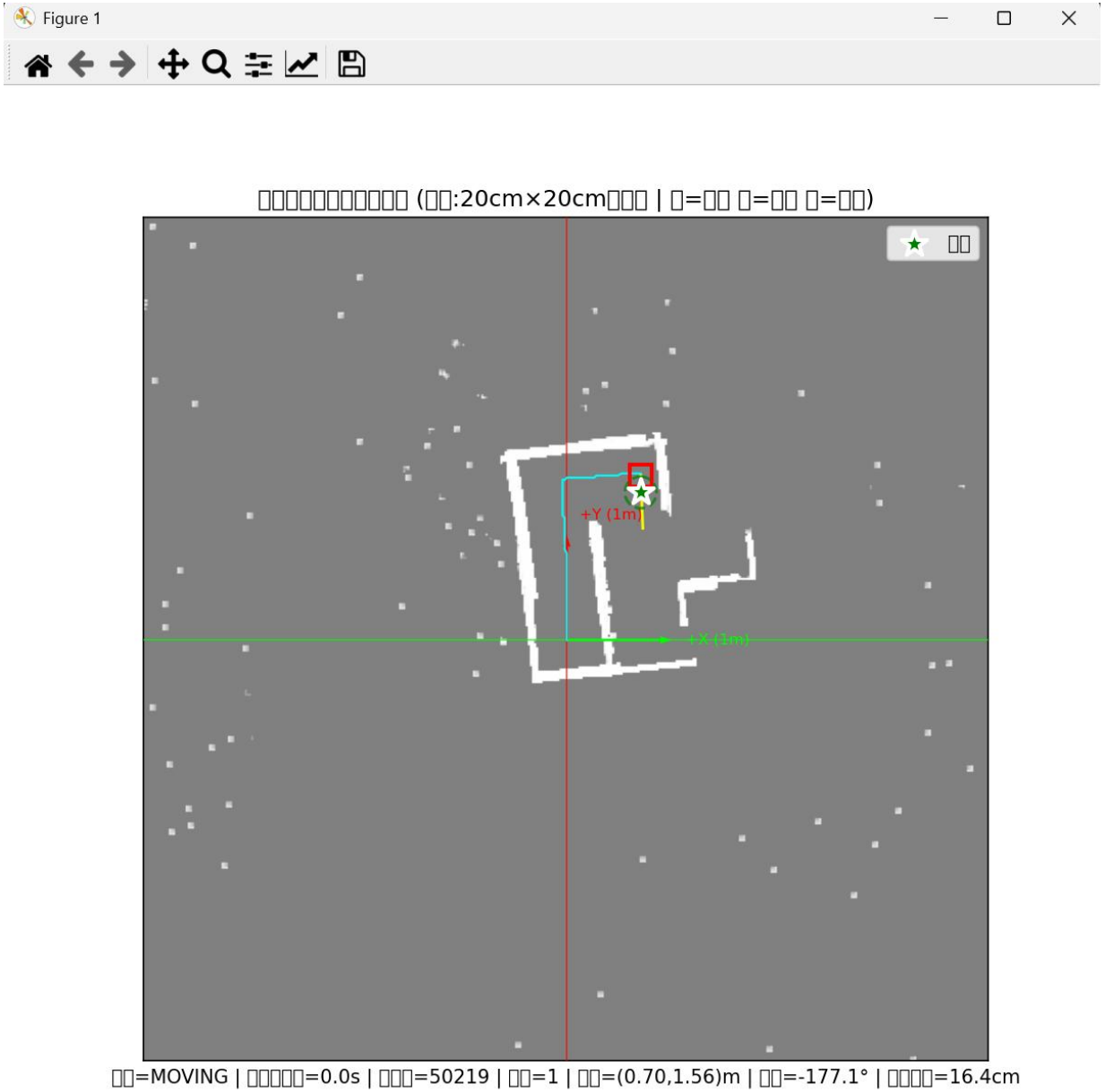
1 Test Environment

The robot was tested in a 2.8 m × 2.8 m wooden maze with corridors wider than the robot diameter plus safety margin.

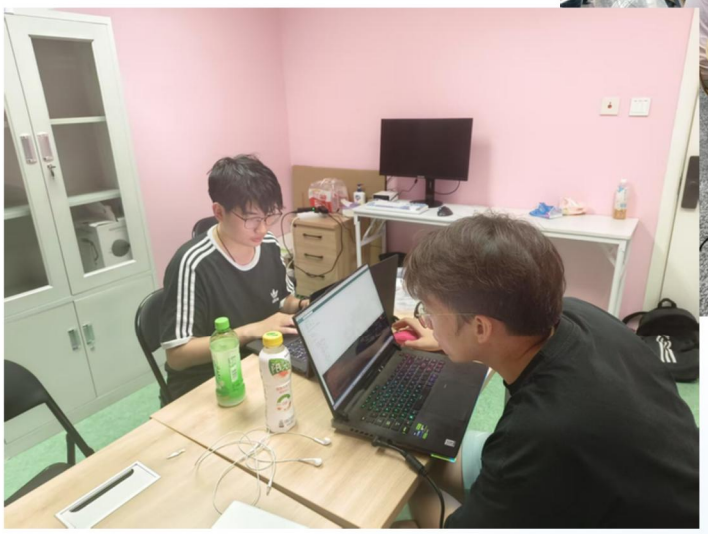
2 Data Collection

Each run followed a scripted protocol, recording raw LiDAR, odometry, map snapshots, and event markers for analysis.

Real SLAM Final Picture



总结



Group 16!

Thanks for watching~

Group Member:

Telcom+EIE:	Zixin Xiong	IOT+IST:	Taiyu Chen
	Can Chen		Jiayou Zhu
	Xinshuang Liu		Jiayi Zou
	Yuancheng Li		Wenrui Si
	Shengcheng Xia		Chenglin Li
	Ke Lyu		