
电子科技大学
格拉斯哥学院
Glasgow College, UESTC

# Team Design Project and Skills (2022-23)

## Final Report

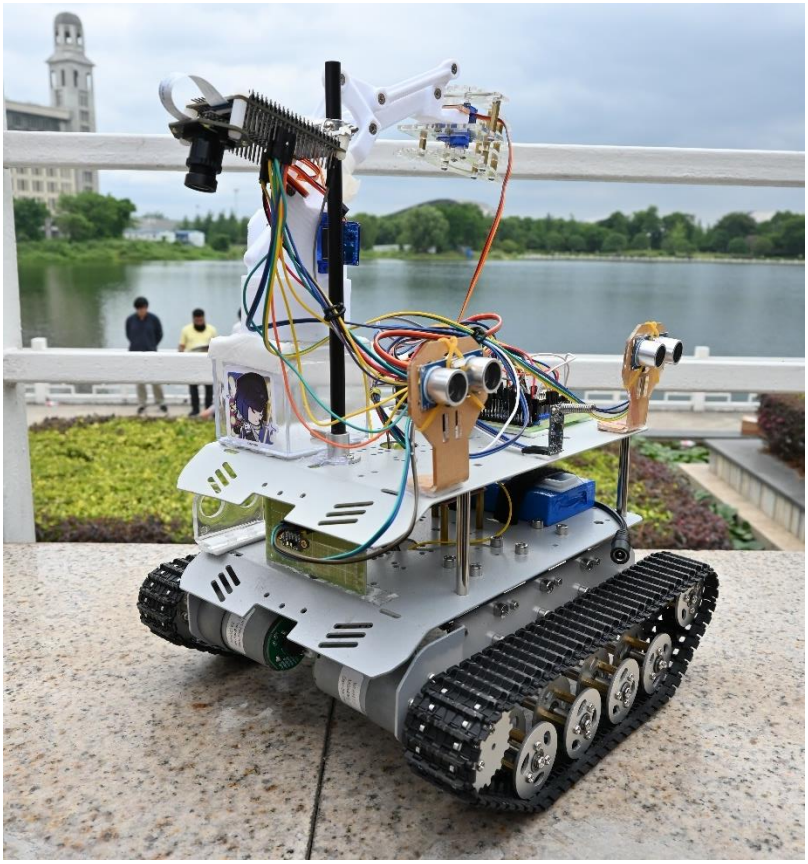
## Team 44: FastFurious

Team Members

| Gao Yinuo | Jin Shuyi | Li Enze | Liu Ruitong | Xu Xiuyi |
|---|---|---|---|---|
| 2614165G | 2614217J | 2614229L | 2614239L | 2614220X |
| 2020190907038 | 2020190501014 | 2020190501026 | 2020190501036 | 2020190501017 |
| Xiao Ziyi | Yuan Zijun | Zhang Jiaqi | Zhou Yuchen | Zhao Zixun |
| 2614232X | 2614147Y | 2614204Z | 2614216Z | 2614146Z |
| 2020190501029 | 2020190907020 | 2020190501001 | 2020190501013 | 2020190907019 |

[In order of Chinese Pinyin]

**All members contributed equally in this team project.**

# Abstract

Machine vision and automatic control are highly significant and widely explored domains in the field of electronic engineering. They have emerged as crucial components for numerous applications across various industries. This project presents the design of an autonomous line-following and obstacle-avoiding rover, utilizing the OpenMV as the main control chip for visual recognition and autonomous driving, complemented by the STM32 (L432K) microcontroller to enable a series of distinctive behaviors. The rover is equipped with modules such as ultrasonic sensors, laser rangefinders, motor drivers, and wireless communication, enabling a wide range of intelligent functionalities. The robot integrates path tracking, obstacle avoidance, shape recognition, wireless communication, and projectile capabilities into a compact PCB-based hardware platform. It exhibits high integration, stability, small form factor, reliability, strong anti-interference ability, and user-friendly operation. The hardware design includes the integration of these modules and the software design encompasses the development of algorithms for vision processing, motion control, and communication. This paper provides a detailed description of the hardware and software design processes, highlighting the unique advantages of the system. Experimental results demonstrate the effectiveness and performance of the developed autonomous rover.

# Content

# List of Figures

# List of Tables

# 1. Introduction

There were two patios, totally six tasks for the car to complete, where the cost needed to be limited to 1,000 RMB. In patio 1, the task 1 required the car to keep line-tracking until arriving at the bridge. In the second task, the car was required to go across the bridge. Task 3 occurred after going down the bridge, the car needed to find the line and pass through the wooden gate placed next to the stairs. In patio 2, the first task required the car to recognize the shape of the arrow and then moved to the corresponding direction, keeping moving until arriving at the table tennis ball basket. The second task should be performed then, which meant to release the table tennis ball to the basket. Task 3 was related to communication. This meant the car should stop after entering the area beside the planter and transmit a specified message to the laptop at the same time.

# 2. Top System Design

## 2.1 Task Analysis

### 2.1.1 Patio 1

In patio 1, the rover's primary task is to follow a distinct lane with a unique texture and color compared to the surrounding ground. Once it completes tracking the lane, the rover needs to make a right turn and traverse a bridge that has a ramp inclined at 14 degrees. After crossing the bridge, the rover will turn left again and resume tracking. The final objective in this patio is to navigate through a designated gate, which has a height and width of 0.5 meters. Overall, the key functions of this patio involve lane tracking and maneuvering at specific turning points.

Based on our analysis, in order to achieve the aforementioned tasks, our robot needs to possess the following capabilities:

1. Lane Detection: The robot should be equipped with a reliable vision system capable of detecting and tracking the specific lane based on its distinct texture and color.
2. Object Detection: The robot should be able to detect and recognize the bridge, allowing it to approach and cross it safely. It should also be capable of identifying the gate and determining its position and dimensions accurately.
3. Path Planning and Navigation: The robot should have advanced path planning algorithms to determine the optimal route for following the lane, making turns, and reaching the designated gate.

4. Maneuvering and Control: The robot should have precise control over its movements to execute accurate line tracking, turns, maintain stability while traversing the bridge's ramp, and navigate through the narrow gate without collisions.
5. Real-time Monitoring: The robot should continuously monitor its environment, including the lane, bridge, and gate, to adapt its movements accordingly and make any necessary adjustments to successfully complete the tasks.

### 2.1.2 Patio 2

In patio 2, the rover's initial objective is to locate a card measuring 10 cm x 10 cm and identify the shape of the pattern displayed on it. The subsequent steps depend on the detected shape, which could be an up arrow, right arrow, or left arrow. For each shape, the rover will proceed towards a specific vertex of a diamond and engage in lowering a sign.

Once the assigned task is accomplished, the rover will proceed autonomously to a predetermined location to release a ball into a basket. Subsequently, it will navigate towards a designated area to establish a Bluetooth connection with a computer operating at a frequency of 433MHz. The rover will transmit a message containing the required information to the computer.

In order to accomplish the assigned tasks in patio 2, the following additional capabilities should be incorporated into our rover:

1. Shape Analysis and Recognition: The robot should possess sophisticated algorithms to analyze the pattern on the card and accurately recognize the shape, whether it is an up arrow, right arrow, or left arrow.
2. Manipulation and Grasping: The vehicle needs manipulation and grasping abilities to pull down signs and release the ball into the basket. It should have the necessary mechanical components or robotic arms to perform these actions effectively.
3. Wireless Communication and Bluetooth Connectivity: The robot should be equipped with wireless communication modules and the capability to establish a Bluetooth connection operating at a frequency of 433MHz. This allows it to transmit the required information to the computer.

## 2.2 Top System Design

The top-level systems for the two patios are depicted in Figure 1 and the modules connection is shown in Figure 2. Different modules are employed to accomplish various tasks, and the

diagram illustrates the task designs for each module and the working principles of the entire system.

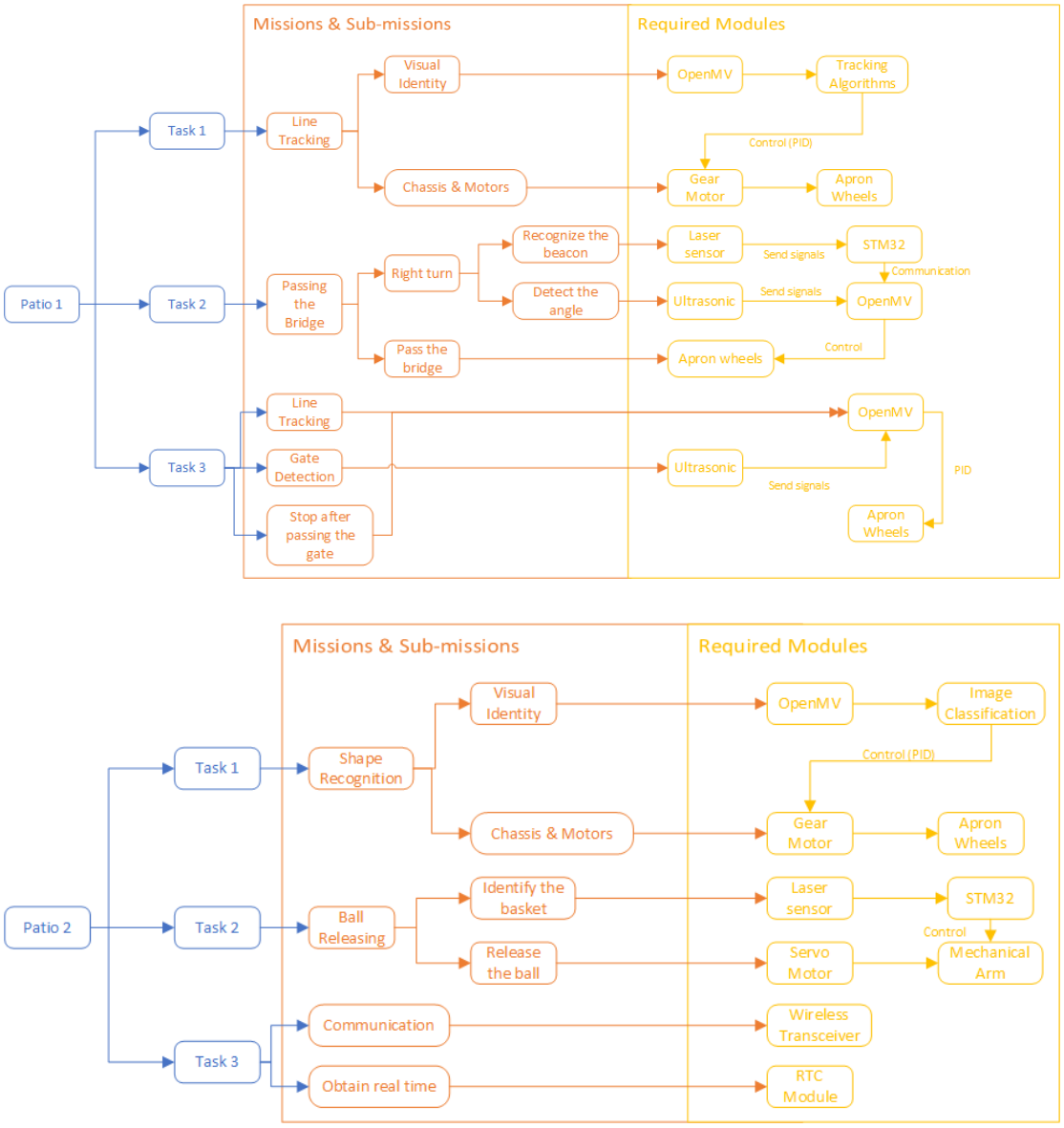



Figure 1. The top-level systems of two patios

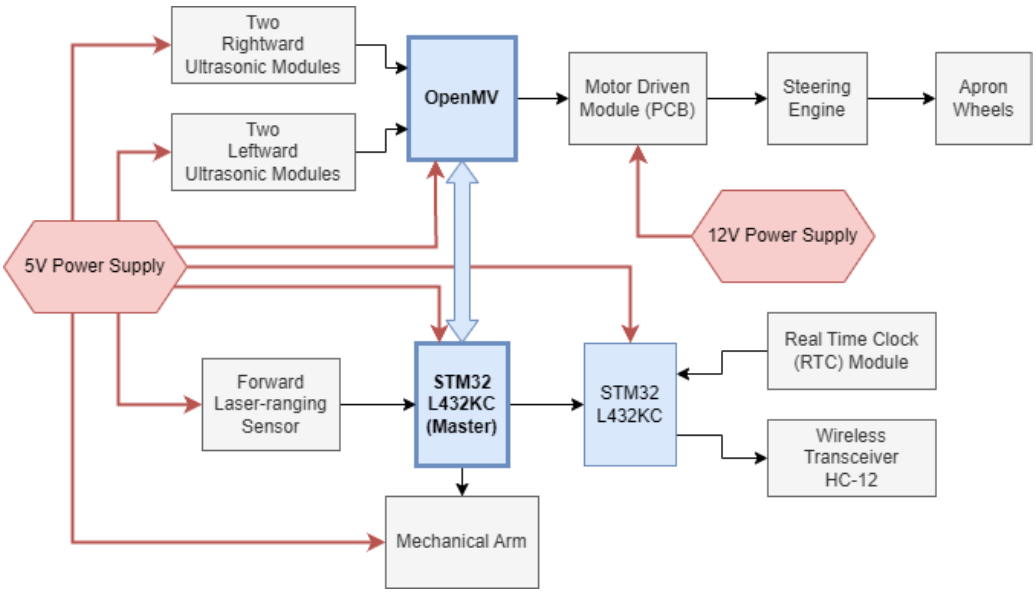



Figure 2. The block diagrams of modules connection

# 3. Subsystem Design

## 3.1 Hardware: Vehicle Structure

### 3.1.1 Chassis and Wheel Selection

| **Section Author:** | **Zhao Zixun** |
|---|---|
| | **UESTC ID: 2020190907019, GUID: 2614146Z** |

The main body of the car was composed of metal parts, which not only improved the stability and reliability of the car structure, but also increased the weight of the car and thus increased the friction with the ground. While ensuring that the car is not easy to be damaged, it can better prevent slipping on the road surface and the slope of the bridge, and is less affected by factors such as wet ground. Figure 3 showed the appearance of the chassis.

Figure 3. Appearance of the Metal Chassis

As for the wheels, apron wheels were chosen instead of ordinary wheels and Mecanum wheels. The appearance of three kind of wheels were shown in Figure 4. Given the practical considerations of rough terrain, using ordinary wheel might not have enough friction. The bridge's surface must also be considered, as small wheels might become trapped in the gaps; however, apron wheels effectively circumvent this issue.

If Mecanum wheels were chosen, although it had the advantage of omnidirectional movement, which could better complete the turning work in line tracking, the quality of Mecanum wheels might not strong enough to support the heavy metal chassis. In addition, Mecanum wheel required high ground smoothness and was easy to wear, but the road condition of the test site was not good. Using Macanum wheels could only improve limited steering ability as well as increasing failure rate and cost of the vehicle.

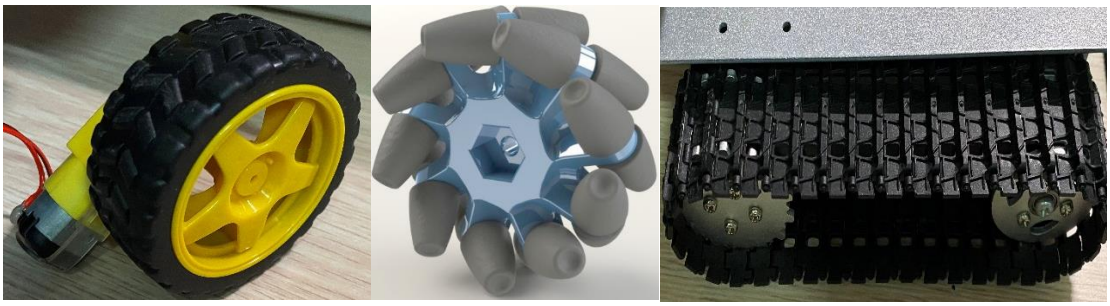


Figure 4 Comparison of Three kinds of Wheels: Ordinary Wheels (Left), Macanum Wheels (Middle), Apron Wheels (Right)

### 3.1.2 Motor Selection

| **Section Author:** | **Li Enze** |
|---|---|
| | **UESTC ID: 2020190501026, GUID: 2614229L** |

There were two MG540 DC motors with Hull encoder to drive the car and read the actual speed of each track. MG540 DC high-torque deduction gear motor was an ordinary motor with a reduction box, shown in Figure 5. This kind of reduction box with gears reduced the speed and increased the torque. There were three reasons for us to choose MG540 motors.

Figure 5. MG540 High-Torque Deduction Gear Motor

Firstly, for a system, a uniform voltage of power supply was significant, which meant there was no need for more power source with another voltage or one extra transformer. In our project, most of the power supply was 5 volts and the most common power supply on the market was 12 volts. Therefore, MG540 motor was used at a 12V operating voltage and 5V Vcc of its Hull encoder.

Secondly, high torque was needed for the motor driven our car. Our car was a complex system with a large number of components, which led to a great weight. Hence, high torque was needed to drive the ‘heavy car’ for running and climbing. Moreover, another feature of MG540 high-torque deduction gear motor, low speed, was not a shortage for our project, as the total time of the task was so generous that there was no requirement for a high speed.

Thirdly, a Hull encoder was helpful working as a real-time speed sensor. When the Hull encoder was powered, it would output the signal which contained the information of speed. This signal could be decoded with orthogonal decoding. Although this function was not used at the end, it was a function that attracts us.

Figure 6 gave the 6-pin interface of MG540 DC motor. Pin 1 and Pin 6 were two inputs connected to the motor driven module, which would output two DC driven voltage with changeable value controlled by the duty cycle of PWM. The RMS value of input controlled the rotation speed as well as direction of the motor. Additionally, as the Hull encoder was not

used in our project, Pin 2 to Pin 5, which was related to sensor, are not used.

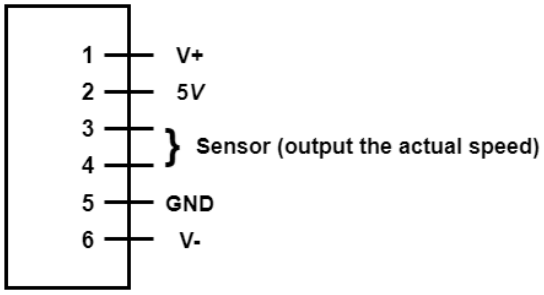



Figure 6. The Interface of MG540 DC Motor with Hull Encoder

## 3.2 Hardware: Motor Driven Module

| Section Author: | Xiao Ziyi |
|---|---|
| | UESTC ID: 2020190501029, GUID: 2614232X |

### 3.2.1 Required Function Analysis

Section 3.1.2 showed that the motor required a voltage input of 12V and the battery chosen was also 12V. So the main requirements of the PCB were to accept and output 12V respectively. Another important function was that PCB could receive control signals from OpenMV and then control the speed of the motors. This required the PCB to convert TTL signals into different voltage levels. Moreover, multiple ground terminals should also be created on the PCB so that the ground pins of different modules could be connected together for convenience. This was to ensure that all the modules had same voltage reference. In conclusion, the PCB should have the following functions:

- Could receive 12V input from the battery.
- Could power two 12V motor simultaneously.
- Could receive control signals from OpenMV and process the signals.
- Had terminals that could connect all ground terminals together.

## 3.2.2 Chip Selection

### 3.2.2.1 Full-Bridge Driver L298n

To power the motors, the dual full-bridge driver L298n was selected as the main chip to convert the 12V supply voltage to varied output voltage through TTL levels. L298n was an integrated monolithic circuit that was widely used in motor driven modules, which had advantages such as high voltage and high current capability [1]. In this PCB the Multiwatt15 package was chosen for the convenience of soldering. The Multiwatt15 package of L298n was shown in Figure 7.

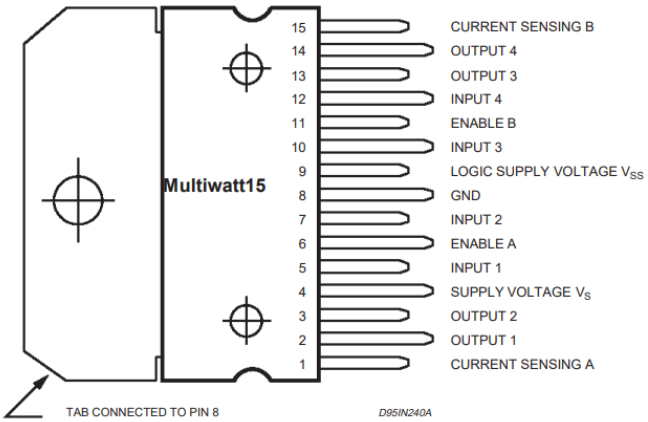



Figure 7. Multiwaltt15 Package of L298n [1]

Another reason for the selection of L298n was that it accepted simple TTL logic levels and could control the state of motors easily. The function of each pin was introduced as follows [1].

**Voltage Pins (Pin 4, Pin 9)**

There were two voltage pins, one was the supply voltage (Pin 4) and the other one was the logic supply voltage (Pin 9). The supply voltage should be connected to the 12V battery which served as the overall power supply of the module. The logic supply voltage should be connected to 5V which powered the internal logic circuits of L298n.

**Enable Pins (Pin 6, Pin 11)**

By inputting different PWM waves to these two pins, L298n was able to control the speed of the motor. The duty cycle of the PWM wave could decide the speed of the motor.

**Input Pins (Pin 5, 7, 10, 12)**

By inputting a high or low level into these pins, it was possible to control the operating states of the motor, such as positive rotation, negative rotation, start and stop.

**Output Pins (Pin 2, 3, 13, 14)**

Enable A, Input 1 and Input 2 was responsible for the control of Output 1 and Output 2, while Enable B, Input 3 and Input 4 was responsible for the control of Output 3 and Output 4. Output 1 and 2 should be connected to the positive and negative terminals of Motor 1, Output 3 and 4 should be connected to the positive and negative terminals of Motor 2. The full logic table of L298n was shown in Table 1.

| Input 1 | Input 2 | Enable A | Output 1 & Output 2 |
|---|---|---|---|
| 0 | 0 | x | Brake |
| 1 | 1 | x | Float |
| 1 | 0 | PWM | Positive Speed Control |
| 0 | 1 | PWM | Negative Speed Control |
| 1 | 0 | 1 | Full Speed (Forward) |
| 0 | 1 | 1 | Full Speed (Backward) |

Table 1. Logic Table of L298n

## 3.2.3 PCB Design

### 3.2.3.1 Important Considerations

During the design process of PCB, the following rules and considerations should be taken into account [2].

- The signal wires should be as short as possible and should be away from power lines.
- Ceramic decoupling capacitor should be connected to the battery to perform filtering. The capacitor should also be adjacent to the battery as much as possible.
- The turn or corner of wires should be 45° but not 90°, which could avoid the jam of signals.
- The wires for the conduction of input and output voltage, ground signals should be wider than common wires, which could reduce the heat generated and impedance.
- A large area of copper should be covered on the PCB to connect all the volage input and ground terminals respectively, which could reduce the overall heat generated and the impedance of the PCB.

### 3.2.3.2 L298n Circuit Design

A typical L298n circuit could be obtained from the datasheet and was shown in Figure 8 [1]. From Figure 8 an important consideration could be drawn that diodes should be added between output pin of L298n and the motor. This was designed to protect the chip from being broken down by current surge from motors. Motor was inductive component. When the state of motor suddenly, for example from stop to positive rotation, it might generate larger current and went back to the chip. The unidirectional property of diode could perfectly avoid this situation.

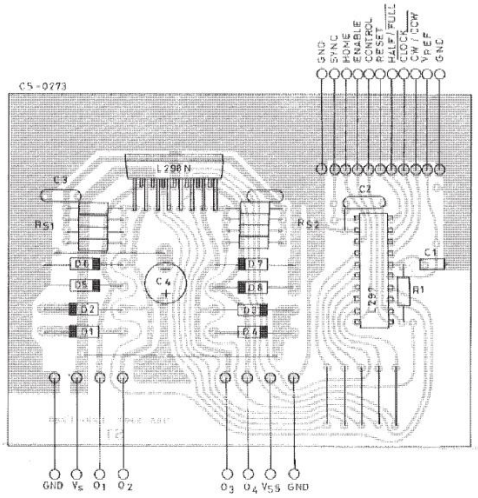



Figure 8. A suggested PCB Layout for L298n [1]

However, during practical testing our group found out that using one L298n existed some problems. The first was that one L298n could not power two motors perfectly at the same time. The large current flowing in the circuit might break down the chip. Hence our group purpose a dual L298n structure to power the motors. One L298n was responsible for one motor. Both L298n shared the same voltage supply at the same time. The circuit now became:

First L298n: Only use Enable A, Input 1, Input 2, Output 1 and Output 2

Second L298n: Only use Enable A, Input 3, Input 4, Output 3 and Output 4

As for the DC-DC buck circuit, a typical circuit for L78M05 was shown in Figure 9.

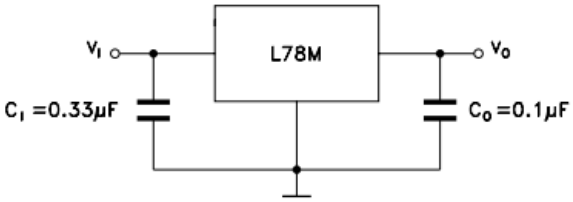



Figure 9. A Typical Circuit for L78M05 which works as DC-DC Buck Circuit [3]

$V_I$ was the input voltage which should be 12V and $V_O$ was the output which should be 5V.

The final circuit design of PCB was shown in Figure 10.

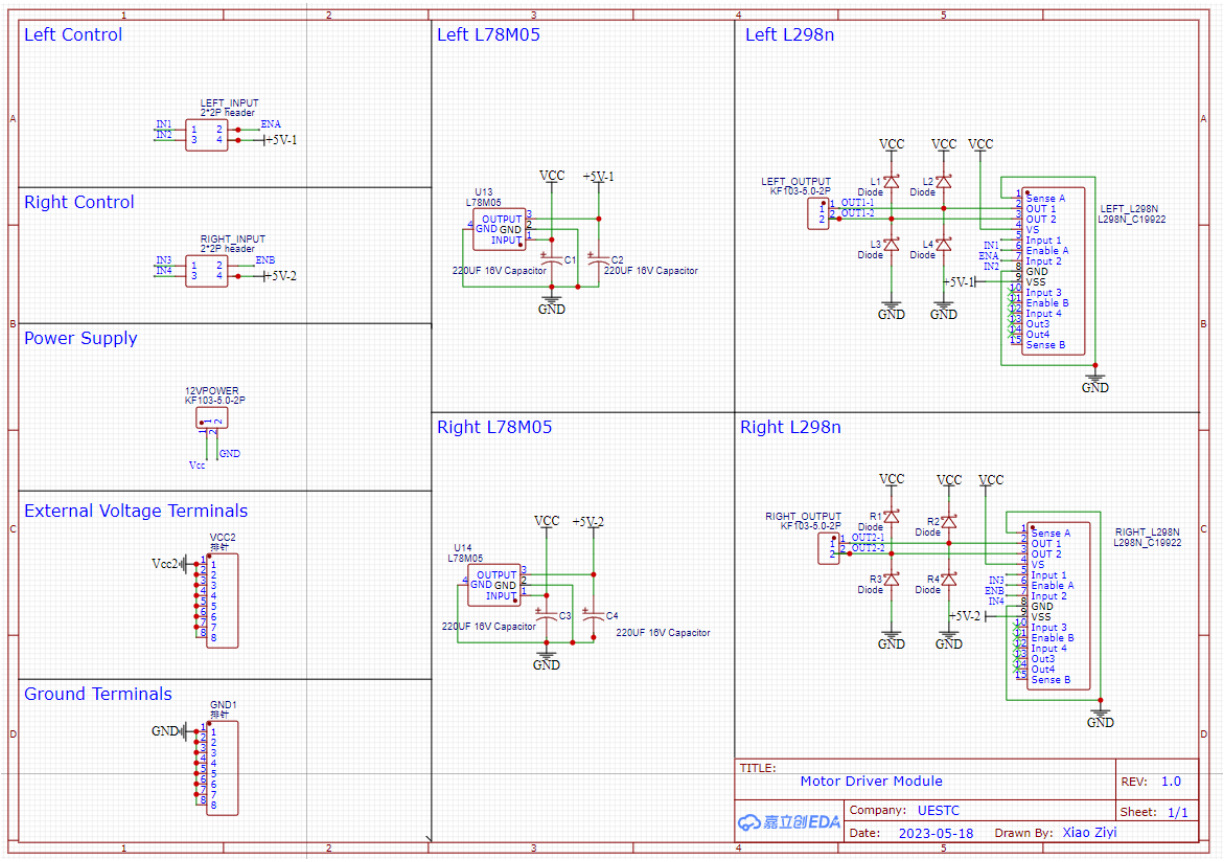



Figure 10. Final Circuit Design of PCB

### 3.2.3.3 PCB Layout Design and Routing

Electrical components were placed on the PCB and connected according to following specifications:

• Top: Right and Left L298n

• Center: Right and Left L78m05, Capacitors, Diodes

• Bottom Terminals: Left and Right Control Ports, Power Supply Terminals

• Right Terminals: Output of Right L298n, Ground Terminals

• Left Terminals: Output of Left L298n, External Voltage Supply Terminals

• All the terminals and ports should be placed on the edge of PCB for convenient connection among modules

• Board Size: 87.49mm × 61.65mm

• Routing Width:

| Rules | Line Width (mm) | Interval (mm) | Hole External Length (mm) | Hole Internal Length (mm) |
|---|---|---|---|---|
| Default | 0.254 | 0.152 | 0.61 | 0.305 |
| GND | 0.762 | 0.152 | 0.61 | 0.305 |
| VCC | 1.016 | 0.152 | 0.61 | 0.305 |

Table 2. Routing Rules of PCB

• All the PCB should be covered with copper to reduce heat production and impedance

The final PCB layout and routing was implemented in software JiaLiChuang and was shown in Figure 11.

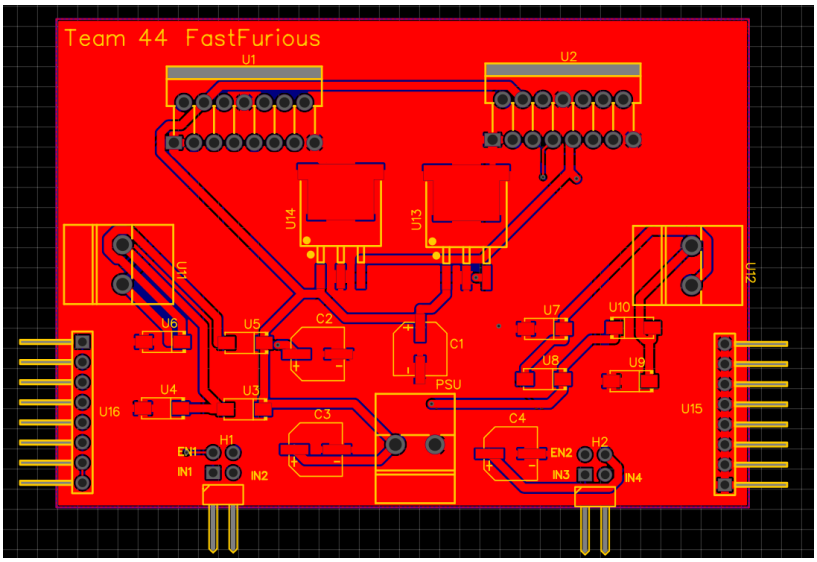



Figure 11. PCB Layout and Routing in Software JiaLiChuang

#### 3.2.3.4 PCB Manufacture and Soldering

Manufacturer JiaLiChuang was chosen to make the PCB. JiaLiChuang was founded in Shenzhen in 2006 and was known for its high-quality products and advanced manufacturing capabilities, specialized in small size PCB production. With a strong reputation in the industry, Jialichuang offers a range of features and advantages to its clients such as highly customization options and short production time.

As for the parameters of manufacturing, the thickness of PCB was 1.6mm and had two layers made of FR-4 material.

All components were manually soldered to the PCB. The final practical PCB was shown in Figure 12.

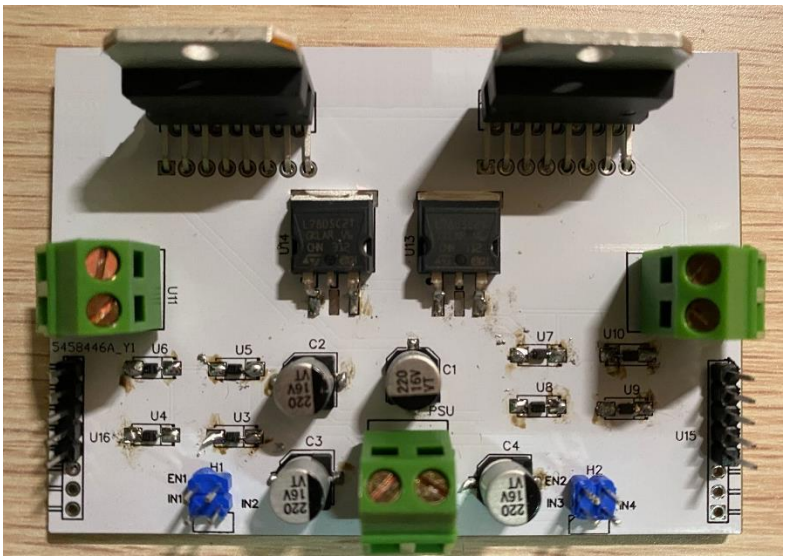



Figure 12. Practical PCB with all components soldered

## 3.3 Hardware: Mechanical Arm

| Section Author: | Xiao Ziyi |
|---|---|
| | UESTC ID: 2020190501029, GUID: 2614232X |

### 3.3.1 Support of Mechanical Arm

The support of mechanical arm was built through 3D printing. The arm was designed to have 4 axes and 3 degrees of freedom. It was produced using printer (HORI)Z300 and the material used was polylactic acid. As for the structure of the arm, it had a clip height of 12cm, chassis diameter of 62mm and a shoulder height of 16cm. The structure of the support was shown in

the following figure.

Figure 13. Structure of the mechanical arm

### 3.3.2 Claw of Mechanical Arm

The claw of mechanical arm was shown in the following figure. The claw was controlled by servo motor SG90 which would be introduced in latter sections. The claw was made of plastic and had enough friction to grab the table tennis ball tightly.

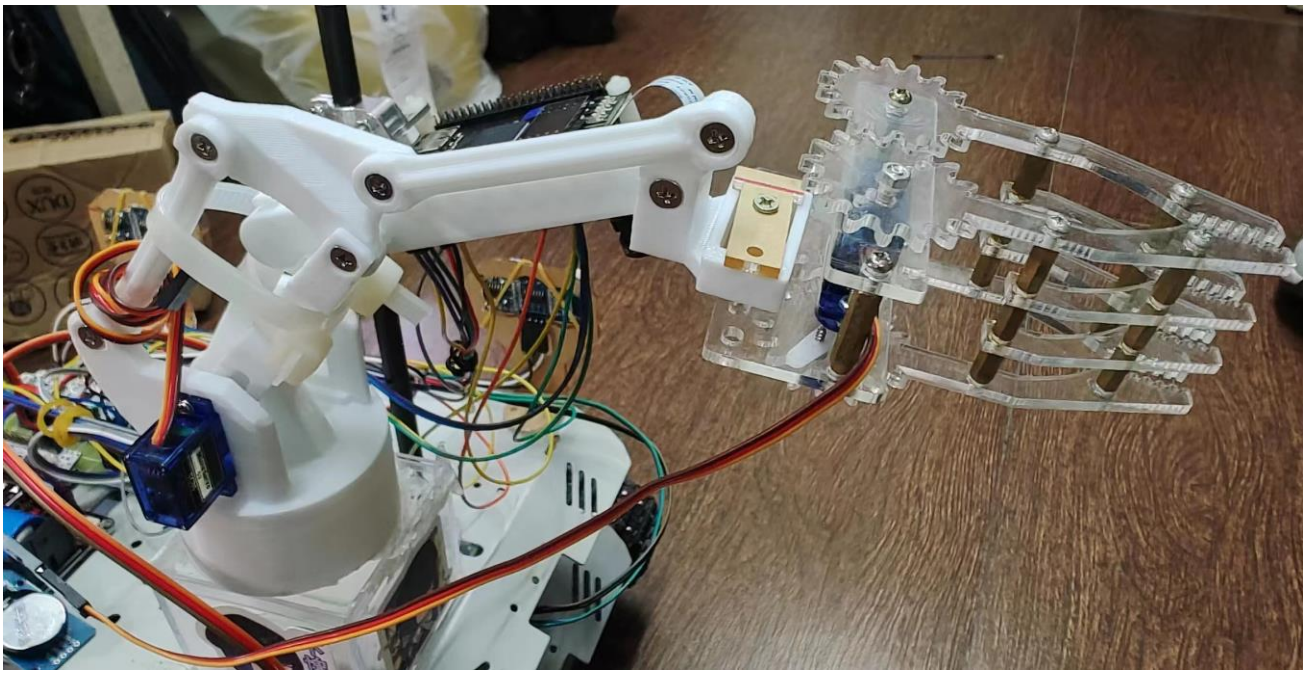


Figure 14. Claw of Mechnical Arm

# 3.4 Distance Obtain

## 3.4.1 Ultrasonic Module

| Section Author: | Li Enze |
|---|---|
| | UESTC ID: 2020190501026, GUID: 2614229L |

The HC-SR04 ultrasonic distance measurement module provided non-contact distance sensing from 2cm to 400cm with the accuracy up to 3mm. Module included ultrasonic transmitter, receiver, control circuit. The transmitter of module emitted ultrasonic waves, which then bounced off an object and were detected by the receiver. By measuring the time that took for the ultrasonic wave traveling, the distance could be calculated with the speed of sound wave.

In Figure 15 input and output pins were clear. Vcc was the power supply with a 5V input DC voltage and GND was connected to the ground. Trig was an input pin that when there was a high-level voltage input more than 10us, the module would be triggered. Then, the module would test itself with eight square waves at 40kHz. If the test was finished, there would be output at pin Echo. The time of high-level output was the time in which the ultrasonic waves travel.


Figure 15. The Ultrasonic Module with 4 Pins

In our project, two ultrasonic modules were on the left side and the other two ultrasonic modules were on the right side. They were used for adjusting direction and angle of our car. High frequency of detection, high detection accuracy and relatively low price were the reasons why we chose to use HC-SR04 ultrasonic module.

## 3.4.2 Laser Ranging Sensor

| Section Author: | Li Enze |
|---|---|

| | |
|---|---|
| | **UESTC ID: 2020190501026, GUID: 2614229L** |
| **Section Author:** | **Gao Yinuo** |
| | **UESTC ID: 2020190907038, GUID: 2614165G** |

### 3.4.2.1 Introduction

The VL53L1X laser-ranging sensor was a time-of-flight (ToF) distance measurement module based on the VL53L1X, with an accurate range up to four meters, fast range frequency up to 50 Hz, I2C interface communication, and low power consumption. Relying on optical elements, distance measurement was always precise to different color and reflectivity of target.

In Figure 16, there were 6 pins in VL53L1X laser-ranging sensor. Vcc and GND were the 5V power supply and grounded line. SCL and SDA were serial clock line (SCL) and the serial data line (SDA) in I2C, respectively. With SCL and SDA lines, the laser-ranging sensor could achieve the bidirectional data transmission and timing control with MCU. The remaining two pins GPI01 and XSHUT were not used in this project.

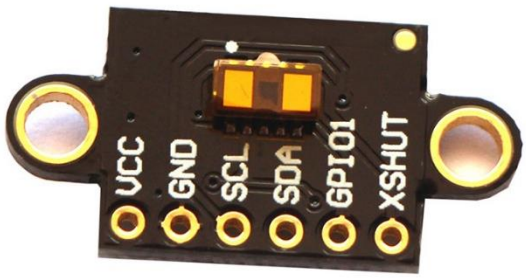



Figure 16. The Laser-ranging Senser with 6 Pins

Belonging to the two-wire serial bus, Inter Integrated Circuit (I2C) is a one-master-multi-slave bus structure. Each device on the bus has a specific address to distinguish from other devices. The physical I2C interface has two bidirectional lines. One is called the serial clock line (SCL) and another is the serial data line (SDA), which can be used to send and receive data. The communication is initiated by the master, and the slave device responds passively to achieve the transmission of data.

There are two methods to implement the I2C communication. The first one is using the I2C peripherals on the chip. The second one is using software simulation. Since there are dedicated I2C pins on the chip, the micro controller can automatically implement the

transmission of data according to the I2C communication protocol, which means the first method is chosen.

## 3.5 Hardware: Power Supply

| Section Author: | Xiao Ziyi |
|---|---|
| | UESTC ID: 2020190501029, GUID: 2614232X |

### 3.5.1 12V Main Power Supply

A 12V 6000maH lithium battery was used as the main power supply to power the motors. This voltage output allowed it to power 2 motors simultaneously and the battery capacity was enough for completing two patios.

### 3.5.2 5V Power Supply

Although L298n could output a 5V on PCB, we still chose another 5V power supply to power all other peripherals such as Mbed, OpenMV, laser-ranging module and ultrasonic module. This was because using the 5V on L298n would cause the voltage on both motors not equal, making the vehicle not go straightly. It used a very common lithium battery 18650 which had a typical output voltage of 3.7V. To power other peripherals, a step-down and boost circuit were added to the PCB to convert the voltage to 3.3V and 5V.

The module also had multiple ground terminals and output terminals which was very convenient for connecting other modules

Figure 17. 5V Power Supply

## 3.6 Software: Line Tracking Algorithm

| Section Author: | Zhang Jiaqi |
|---|---|
| | UESTC ID: 2020190501001, GUID: 2614204Z |

### 3.6.1 Problem Analysis

In Task 1 of Patio 1, our challenge involves guiding the rover to navigate to the bridge following a fixed route. Figure 18 illustrates the distinctive characteristics of this road segment.

The objective of this module is to develop a control system that enables the rover to identify and navigate along the gravel path. This system is designed as a digital closed-loop system, allowing the rover to continuously adjust its position relative to the center of the road. The system comprises three primary submodules: a OpenMV controller, the rover motors, and a camera. The camera module of OpenMV can be utilized to capture the road conditions in front of the rover. Simultaneously, we will employ specialized algorithms to process the images and extract relevant information for navigation. This system is implemented using various hardware components, including the OpenMV controller, the motors, and the corresponding interfaces. Figure 19 illustrates the block diagram representation of the system.

Figure 18. Sample Image of Route Analyzed and Tracked

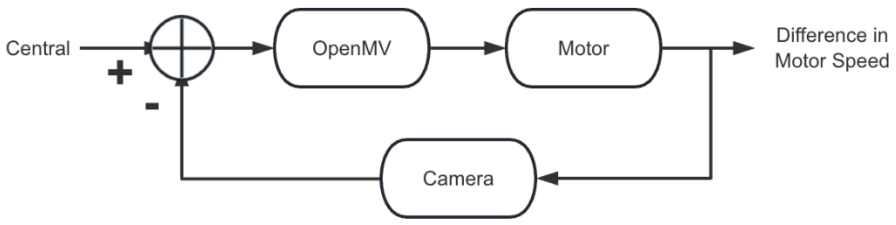



Figure 19. Block Diagram of Line Tracking Feedback Control System

### 3.6.2 Working Principle

The operating principle of this system involves the initial capturing of an image of the gravel section in front by the sensor. Subsequently, the captured image undergoes pixel-level edge

detection to identify and categorize the gravel. Using this information, the controller calculates the deviation of the rover from the center of the road. The computed deviation value is then utilized to adjust the speed difference between the two motors, enabling the rover to execute a turn while moving forward. The OpenMV controller commands the operation of this system.

#### 3.6.2.1 Path Recognition

After conducting field surveys, we found that the road surface of the fixed route we need to travel on has richer edge features compared to the surrounding road surfaces. Therefore, we decided to use edge detection methods to process the image information. Simultaneously, while applying the edge detection algorithm, we need to convert the captured images into grayscale to facilitate subsequent algorithmic processing. A sample photo captured in RGB565 format, as well as the grayscale image after edge detection processing, are shown in Figure 20.

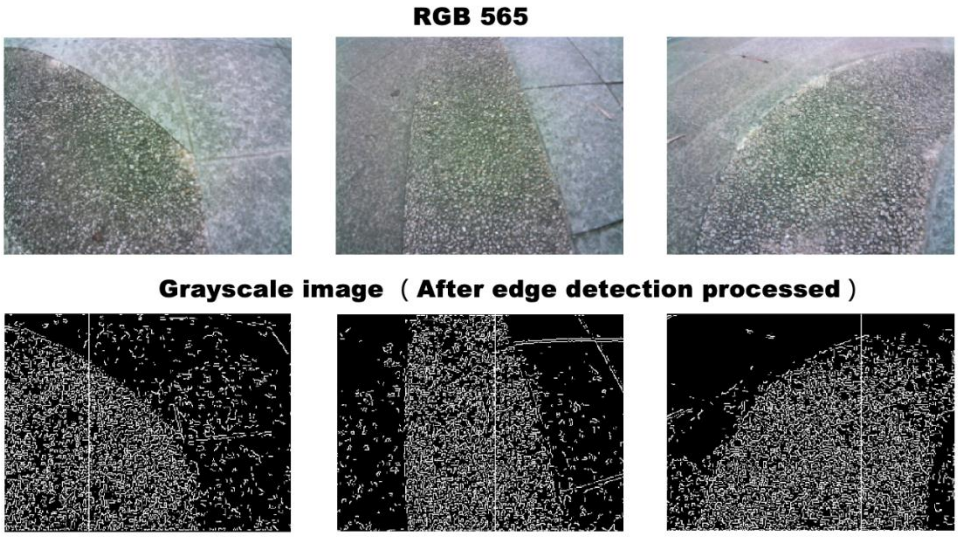



Figure 20. Original Image and Processed Image

In the line-following algorithm, after obtaining the grayscale image processed by the edge detection algorithm, we divide the resulting 320x240 grayscale image into 40 smaller regions by dividing it horizontally into segments of every eight pixels. This gives us 40 smaller grayscale images of size 8x240. We calculate the average grayscale value for each smaller grayscale image, resulting in an array of 40 average values. We then sum up these 40 average values.

Next, we perform a cumulative sum of the average values from left to right, tracking the current region's average value. When this cumulative sum exceeds half of the total sum, we consider that we have found the direction in which the rover needs to move forward. This

information is reflected in the control of the left and right motors, ultimately determining the differential speed between the rover's left and right wheels.

As shown in Figure 20, when we need to make a left or right turn, the rich edge features on the target road cause the centroid of the white pixels in our image to shift. Therefore, by marking the horizontal position of the centroid, as illustrated by the vertical white line in Figure 20, we can correct the rover's direction based on the difference between the horizontal position of the white line and the geometric center of the image.

#### 3.6.2.2 Noise-Reduction Filter Design

During the algorithm's execution, we can incorporate filtering algorithms to more accurately determine the direction in which the rover needs to move. In the aforementioned line-tracking algorithm, we prioritize the target road with richer edge features. However, among the 40 divided regions, there are several regions with lower average grayscale values that correspond to the adjacent road surfaces. These regions can introduce errors in the algorithm when we accumulate the average values.

To address this, we have developed an effective filtering algorithm. In the array obtained from the 40 average values, we manually adjust the smaller 15-20 values to 0. This is because these values often correspond to regions that are not part of our target road. By doing this, we can better focus our algorithm's results on the target road, ensuring that the direction of movement is aligned with the target road.

## 3.7 Software: Arrow Recognition Algorithm

| **Section Author:** | **Xu Xiuyi** |
|---|---|
| | **UESTC ID: 2020190501017, GUID: 2614220X** |

### 3.7.1 Problem Analysis

Template matching and arrow recognition are fundamental computer vision techniques used in a wide variety of applications, including object detection, tracking, robotics, and augmented reality. Among the existing algorithms for identifying arrows through template matching, the existing algorithms include image recognition based on convolutional neural network (CNN) [4], template matching based on double boundary algorithm [5], template matching based on Normalized Cross-Correlation (NCC) [6], pixel point statistics, feature point extraction, etc. In this part of the article, we will introduce the multi-template matching implemented by NCC algorithm to achieve the result of arrow recognition [6].

The whole task 1 could be divided into: starting from the designated location and arriving at

the recognition point; arrow recognition; positioning based on the shape of the recognized arrow and pushing down the sign. To facilitate a clearer understanding, we had divided this task into four distinct phases: pre-recognition processing, arrow recognition and post-recognition processing. The target arrows were shown in Figure 21.

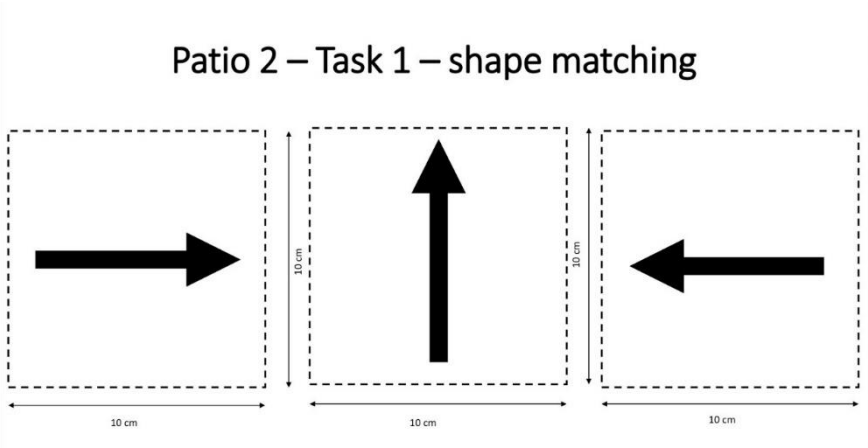



Figure 21. Image of Three Target Arrows

### 3.7.2 Pre-recognition processing

At the beginning of the task, both L432KC and OpenMV were initialized simultaneously. OpenMV initiated the execution of the pre-set code, while the rover set off from the starting point, straight to the recognition point. Additionally, OpenMV directly controlled the forward laser ranging module using the I2C protocol. It continuously monitored the distance ahead and brought the rover to a halt once the distance reaches a predetermined threshold (Yellow point in Figure 22). The forward laser module was employed for distance measurement and transmits the obtained distance back to OpenMV through the I2C protocol, enabling effective control. The determination of the distance threshold was derived from extensive experimentation to identify the optimal solution.

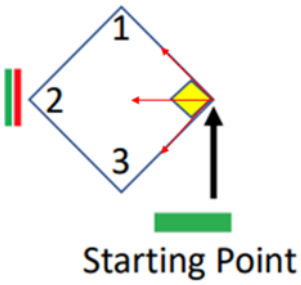



Figure 22. Route for the Vehicle in Patio 2 Task 1

### 3.7.3 Arrow recognition

When the rover stopped at yellow point, the task entered the arrow recognition stage, OpenMV deactivated the forward laser and start arrow recognition. We applied the multi-template matching algorithm based on NCC algorithm, and based on this algorithm, we added a counter for three different arrows in the task to increase the robustness of the algorithm.

#### 3.7.3.1 Introduction of NCC algorithm

The basic template matching algorithm consists of computing at each location in the image being examined a distortion function that measures the similarity between the template and the image [6]. Then, the minimum distortion or maximum correlation position is used to localize the template into the inspection image [6, 7]. The NCC algorithm measures the similarity between the template and different regions of the search image by calculating the cross-correlation coefficient. It provides a robust similarity measure between images, allowing for accurate template matching even in the presence of noise, variations in lighting conditions, and partial occlusions.

As for the cross-correlation coefficient, it was calculated by comparing the difference between template and corresponding region of the search image. This involves summing the element-wise product of pixel intensities in the template and the search image region, and then dividing it by the product of their standard deviations. Next, normalize the coefficients to ensure that the cross-correlation values fall within the range of -1 to 1. A value of 1 represents a perfect match between the template and the search image. Finally identify the position(s) in the search image where the normalized cross-correlation coefficient is highest. These locations indicate the regions in the search image that are most similar to the template image.

#### 3.7.3.2 Implementation of NCC Algorithm

Before importing the arrow template, ensure that it is a smaller image representing what we aim to locate within a larger search image. Compute the mean and standard deviation of pixel intensities for both the template and search image. Select a window size that aligns with the dimensions of the template image. Slide this window across the search image, examining the template at each position of the window.

#### 3.7.3.3 Overall Process

Figure 23 illustrates the complete process of arrow recognition. The workflow can be summarized as follows: Firstly, a set of template images is stored in the flash memory of

OpenMV prior to the recognition task. When the camera captures arrow information, template matching is initiated. As depicted in Figure 23, shape1, shape2, and shape3 counters are initialized to zero, and the number of matches is set to "i" times. Each match result increments the respective counter by 1. Once the number of matches exceeds 10, the shape with the highest count is selected, and its corresponding shape value is outputted.

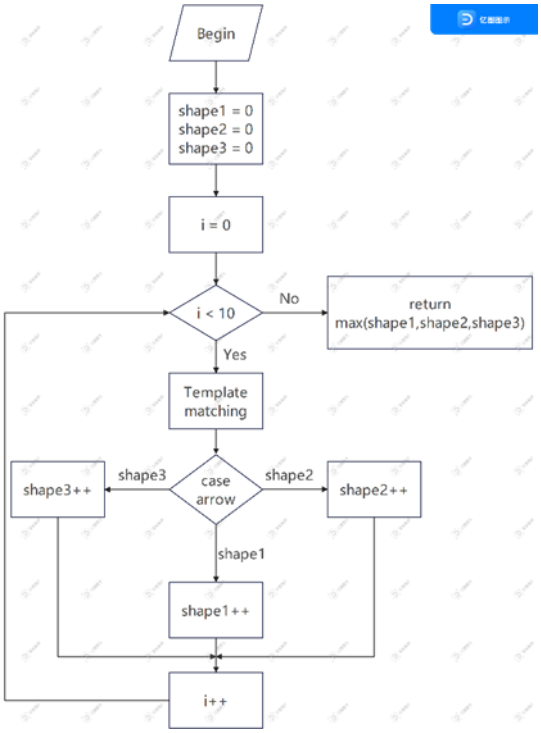



Figure 23. Flow Chart of Arrow Recognition Algorithm

## 3.7.4 Action after Arrow Recognition

### 3.7.4.1 Left Arrow

When the result was a left arrow, OpenMV applied line tracking algorithm and moved to position 3 in the Figure?? and hit the sign.

### 3.7.4.2 Straight Arrow

When the result was a straight arrow, Openmv controlled the rover to turn left, walk straight by a preset speed and direction to the position 2 in the figure4, and hit the sign.

### 3.7.4.3 Right Arrow

When the result was a left arrow, Openmv applied line tracking algorithm and moved to the position 3 in the Figure ?? and hit the sign.

### 3.7.5 Results

In this section, we presented the results of a car successfully completing the recognition of three different types of arrows to perform various tasks. The car's ability to accurately identify and respond to these arrows played a crucial role in executing tasks with precision and efficiency.

The car successfully completed the initial segment of the straight-line task by reaching the arrow recognition point with precision. The recognition system effectively distinguished between arrows pointing left, right, and straight, providing reliable information for task execution. The accuracy of arrow recognition was consistently high across multiple test scenarios and road conditions. The recognition process was optimized for speed and responsiveness, enabling the car to smoothly transition between tasks based on the recognized arrows. This efficient execution contributed to minimized delays and improved overall task completion time. Figure 24 showed the process that the vehicle was going to the point for arrow recognition

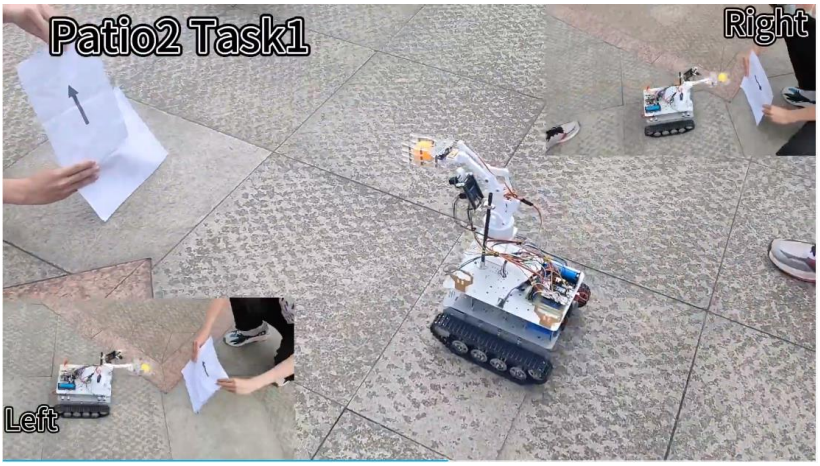



Figure 24. Arrow Recognition Process

After recognition, the car reliably navigated to three different target points, as depicted in figure 25, based on the specific instructions provided by each arrow, demonstrating a high level of precision throughout the task execution.

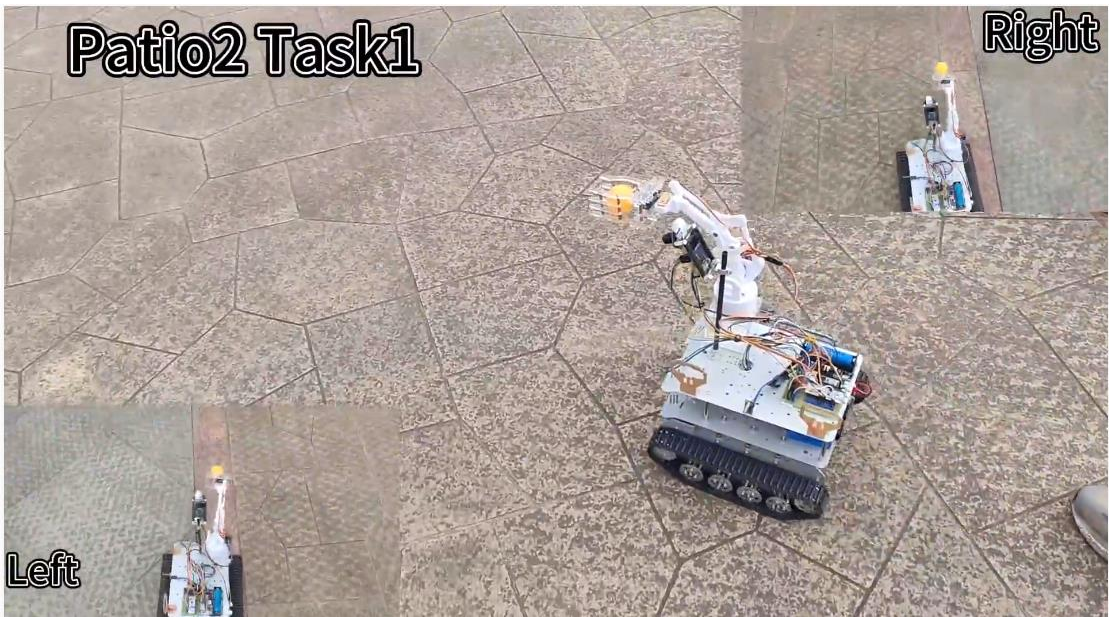



Figure 25. Go to Specific Directions

Finally, the car successfully and managed to push down the signs at each location, shown in Figure 26.


Figure 26. Knock down the Arrow

## 3.8 Software: PID Control and Steering Logic

| Section Author: | Jin Shuyi |
|---|---|
| | UESTC ID: 2020190501014, GUID: 2614217J |

### 3.8.1 Moving System

The moving system consists of 3 aspects: the Driving system, the Adaptive dynamic system and the Steering system.

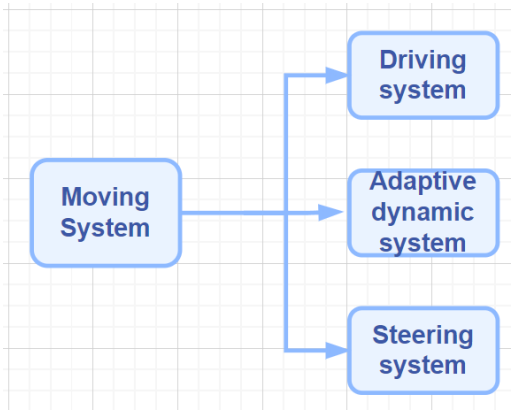



Figure 27. Three Categories of Moving System

#### 3.8.1.1 Driving System

The driving systems includes several parts: the motor controller, motor driver, and motor, which help the car to finish the action of moving forward and backward, climbing ramps and steering.

(1) Driving system hardware configuration

The car utilized a pair of MG540_12V motors for propulsion, with precise control facilitated by the A4950 driver chip. The driver chip's output power was contingent upon the duty cycle of the PWM signal it receives. To regulate the motor speed effectively, we employed the MBED (L432KC) microcontroller, which generated and output the necessary PWM signals to command the driver chip.

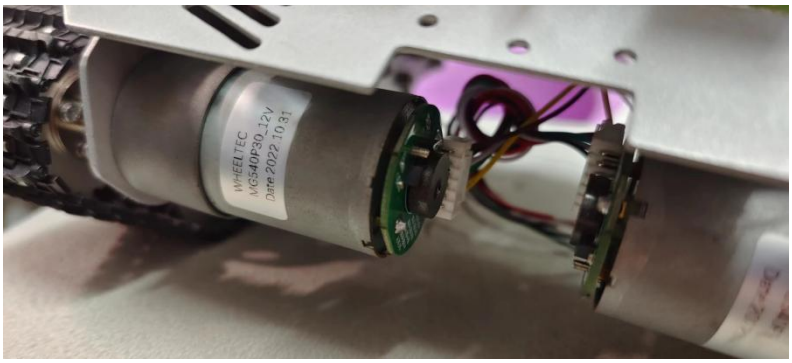



Figure 28. Motor Installed below the Rover

2) Driving system performance

Considering that the speed curve is derived from a fully charged battery, field tests have revealed that the speed curve undergoes alterations as the battery power diminishes. As a result, relying solely on PWM duty cycle control proves inadequate for maintaining optimal

system stability. To address this challenge, we will implement the Adaptive Dynamic System, which will be elaborated upon in Section 3.5.1.2. Furthermore, additional information regarding the rover's powertrain performance can be found in Table 3.

The driving system performance of the rover encompasses several key parameters. It is important to note the following details:

1)Maximum Adjustment Frequency: This parameter measures the shortest adjustment time in which the rover exhibits significant swinging when the speed difference between the two tracks is set to ±50%. It indicates the responsiveness and agility of the rover's driving system in adapting to changes in speed and direction.

2)Maximum Forward Speed: This test determines the highest attainable speed of the rover when moving forward on a concrete road. It reflects the overall propulsion capability and efficiency of the rover in a typical operating environment.

3)Maximum Climb Angle: This measurement involves assessing the car's capability to ascend inclined surfaces. The maximum climb angle is determined by testing the car's performance on a ramp with a dry wood surface. It showcases the car's traction and climbing ability under challenging conditions.

| Test Items | Test performances |
|---|---|
| Maximum Adjustment Frequency | 10Hz |
| Maximum Forward Speed | 1.4m/s |
| Maximum Climbing Angle | 25° |

Table 3. Performance of the Vehicles Driving System

This driving system exhibits promising feasibility for successfully accomplishing the tasks based on the following aspects:

1.The maximum speed allows the car to finish both tasks in 5 minutes.

2.The maximum climbing angle allows the car to climb the bridge in patio 1 task 2 whose angle is about 14°.

3.The maximum adjusting frequency allows the car to adjust its speed for both tracks frequently.

### 3.8.1.2 Adaptive dynamic system

#### 3.8.1.2.1 Reference PWM Determination

When we tested rover, it was observed that the battery power level had an impact on the rover's ability to maintain a straight trajectory and execute precise steering. To address this issue, an adaptive approach is employed to modify the reference Pulse Width Modulation (PWM) signals. This method involves generating a series of optimized PWM signals that ensure the left track's speed matches the right track's speed. These optimized PWM signals serve as the initial values for Proportional-Integral-Derivative (PID) control.

Furthermore, the steering function also utilizes this set of optimized values to ensure that both tracks rotate at the same speed. This synchronization prevents any shift in the center of rotation during steering maneuvers. By maintaining equal track speeds during steering, the rover achieves improved steering accuracy and stability.

The principle of this method is to set the PWM signal output for the left track's motor to 0.5 and then adjust the PWM signal output for the right track's motor until the speed difference between the left and right crawlers is less than 2%. This state needs to be maintained for 5 seconds to satisfy the convergence condition.

Figure 29 illustrates the operational principle behind the determination of adaptive reference PWM in this system.

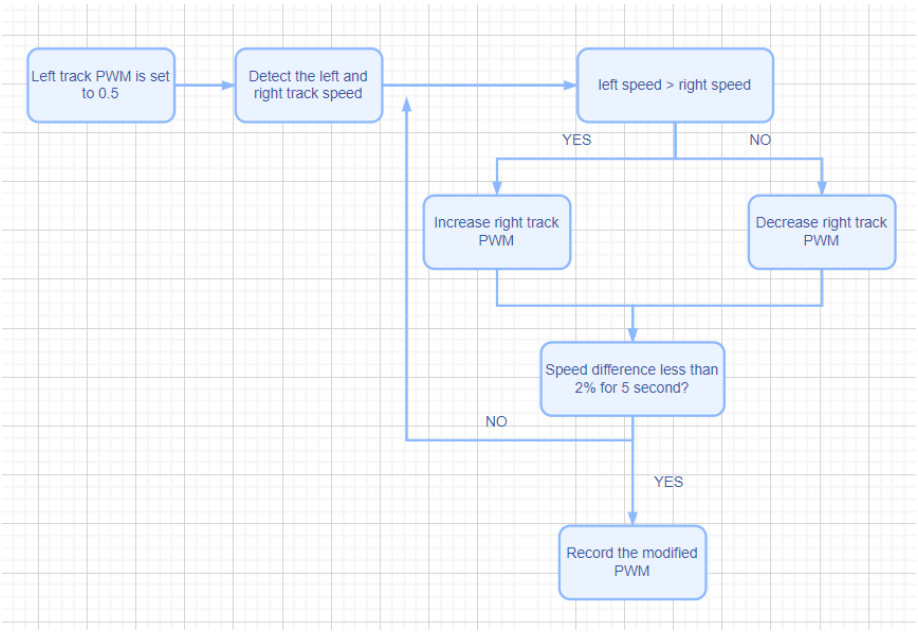



Figure 29. Reference PWM determination working mechanism

#### 3.8.1.2.2 PID Dynamic System

**(1) Background and Introduction**

Since the PWM duty cycle-speed curve is subject to change over time, utilizing PWM signals to control the track speed proves to be unstable. Furthermore, when supplying 8 Volts (0.75 for PWM Duty Cycle) to both motors, discrepancies in rotation speed are observed, with the potential speed difference reaching up to 30%. This discrepancy arises due to variations in friction forces across different gears, shafts, or junctions. Consequently, the introduction of PID speed control becomes necessary as it allows for precise motor speed regulation by dynamically managing the output power of each motor driver. This implementation aims to enhance the overall performance of the driving system.

PID stands for Proportional-Integral-Derivative, which is a popular control algorithm used in various control systems. It is designed to adjust and maintain a desired output by continuously analyzing and adjusting the system's input or control signals.

The PID controller takes into account three key factors: proportional, integral, and derivative. The proportional term calculates an output response based on the current error (the difference between the desired setpoint and the actual value). The integral term accounts for the accumulation of past errors over time, helping to eliminate steady-state errors. The derivative term considers the rate of change of the error, enabling the controller to anticipate and respond to rapid changes in the system.

By combining these three terms, the PID controller dynamically adjusts the control output to minimize the difference between the desired setpoint and the actual value. It provides a balance between responsiveness and stability, making it widely used in applications such as robotics, process control, temperature regulation, and motion control.

The specific tuning of PID parameters (proportional gain, integral gain, and derivative gain) is crucial to achieve optimal control performance, ensuring stability, fast response, and minimal overshoot or oscillation.

The PID control sketch map is demonstrated below

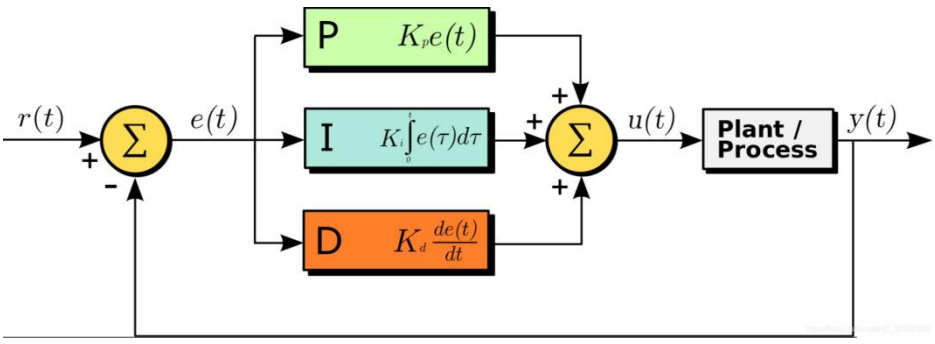



Figure 30. PID Control Diagram

### (2) PID Speed Controlling Algorithm

In this system, the program sets the reference speed, while the hall sensor (motor encoder) provides real-time feedback on the shaft's rotation speed. Acting as the controller, the OpenMV calculates the speed error by determining the difference between the reference speed and the actual speed. The controller employs PID calculations to process this speed error and generate a modified PWM signal.

The motor driver utilizes the modified PWM signal to output the corresponding control voltage to the motor. As a result, the shaft speed gradually converges towards the set reference speed with the assistance of PID control. Figure 31 illustrates the control mechanism of PID in the rover.

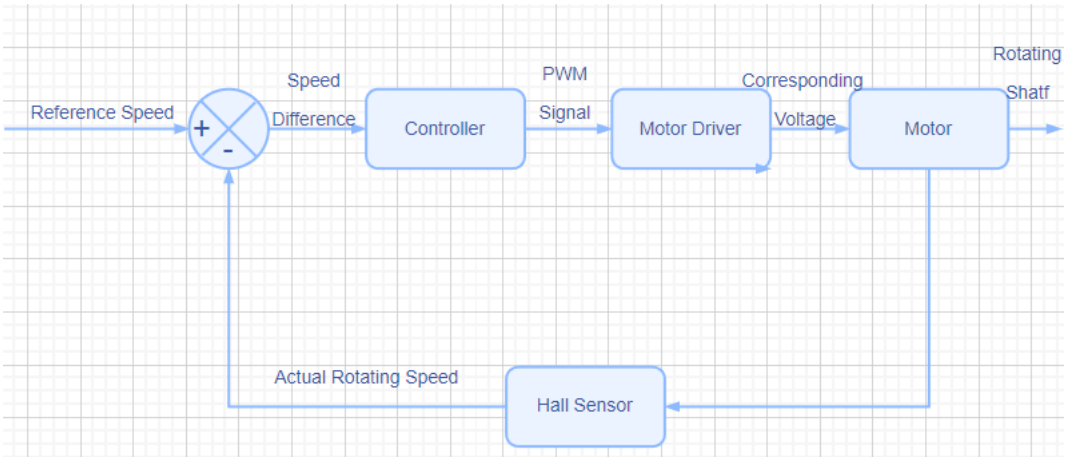



Figure 31. The control mechanism of PID control

The expression for the PID control can be expressed as

$$PWM_{current} = PWM_{previous} + K_p \times (V_{reference} - V_{actual}) + K_D \times (V_{Error} - V_{lastError}) + K_1 \times (\sum_{n=0}^{current} V_{error})$$

In the above equation, P control is used to increase the PWM signal if the current speed is less than the reference speed, and vice versa. D control is implemented to help the system reach the set speed as quickly as possible by reducing oscillation. I control is employed to eliminate the static error of the system.

The debugging of PID parameters is a comprehensive process in which each parameter affects each other. Many attempts in the actual debugging process are very important and necessary. In adjusting process, there were three parameters Kp, Ki and Kd should be adjusted. It is found PID algorithm would have an excellent performance if the three

parameters were adjusted separately. As a result, in the adjusting process, the parameter Kp at the beginning was adjusted and then Ki and Kd. Eventually, some small changes of the values of three parameters were conducted.

Parameter Kp reflected the magnitude of data volatility. The larger the value of Kp is, the larger the fluctuation of the data will be. In real practice, the value of Kp is needed, which ensures the data reach the reference value in the shortest time.

In addition, the derivative parameter Kd anticipates the future trend of the error based on its rate of change. Adding the D parameter improves the system's ability to react to sudden changes and reduces overshoot and oscillations. It essentially provides damping to the control system. Nevertheless, an overly high D value can introduce noise amplification and make the system more sensitive to measurement noise.

Finally, the integral parameter Ki takes into account the accumulated past errors over time and helps eliminate steady-state errors in the system. By increasing the I parameter, the control system becomes more effective at correcting long-term errors. However, an excessively high I value can introduce instability and cause overshoots and oscillations.

The specific comparisons we tested are shown in Table 4 below.

| Kp | Ki | Kd | Time(s) |
|---|---|---|---|
| 400 | 0 | 0 | 4.9 |
| 800 | 0 | 0 | 2.3 |
| 1200 | 0 | 0 | 1.4 |
| 1200 | 20 | 0 | 1.6 |
| 1200 | 20 | 5 | 1.2 |
| 1200 | 18 | 15 | 0.7 |

Table 4. PID Speed Control System Performance

#### 3.8.1.3 Steering System

##### 3.8.1.3.1 Typical Steering System:

To accomplish both tasks effectively, the steering system requires a high level of stability and precision. Various methods such as timers and compasses have been suggested for steering control. However, relying on a timer for steering accuracy is problematic since it is heavily reliant on battery power, resulting in inaccuracies when the battery level is low. Similarly, the stability of the steering system can be greatly affected by the ambient electromagnetic field when using a compass.

Therefore, we have introduced yaw control for the rover by utilizing ultrasonic measurements of the distance between the rover and a beacon. By measuring the distances between the front and rear of the rover and the beacon, we can obtain the yaw angle between the rover and the beacon. The introduction of PID control allows the rover to align itself parallel to the beacon, enabling precise steering. The detailed implementation of the specific algorithm will be discussed in Section 3.8.2.4.

| Types of methods | Steering angle(degree) | Steering error(degree) |
|---|---|---|
| Timers | 74 | -16 |
| Compasses | 98 | +9 |
| Theta PID | 93 | +3 |

Table 5. Turning accuracy comparison between different methods

From the table, we can know that the accuracy of PID control method is acceptable and the stability is relatively high.

#### 3.8.1.3.2 Filter System

Due to environmental factors, the laser detector installed in front of the car and the ultrasonic detector on the right are easily affected, such as the flying of small insects or interference diffraction of waves. The detector is prone to suddenly detecting a very small number of abnormal values. These will have a significant impact on our steering operations, leading to early or delayed turns. Therefore, we add a filtering system to eliminate the interference of Outlier. The principle is that it limits the magnitude to be larger than a predefined value. Meanwhile, the numbers nearby will also help to detect the outliers. This effectively improves the robustness of the system. The performances are shown below.

| Detect distances/cm | Performances |
|---|---|
| 320 | ok |
| 318 | ok |
| 315 | ok |
| 66 | Outlier，Removed! |
| 310 | ok |
| 308 | ok |
| 306 | ok |

Table 6. The performances of the filter system

# 3.9 Software: Distance Filtering Algorithm

## 3.9.1 Filtering of Ultrasonic

| Section Author: | Zhou Yuchen |
|---|---|
| | UESTC ID: 2020190501013, GUID: 2614216Z |

In the experiment, we have made an irritating discovery regarding the performance of the ultrasonic detectors. It has come to our attention that there are instances where the ultrasonic detector receives erroneous signals. Specifically, the detector has shown a tendency to receive signals that are either excessively high or extremely low. These irregular readings pose a significant challenge as they can lead to erroneous operations within our system.

For instance, during ultrasonic edge detection, if the detector prematurely detects an excessively high signal, it can trigger premature turning actions for our rover, causing the entire system to lose control. Moreover, we have also observed a notable increase in the likelihood of the detectors returning inaccurate information when two ultrasonic detectors are installed on the same side. This occurrence can be attributed to interference between the detectors themselves.

Given these circumstances, it is imperative for us to develop a program that effectively filters out the erroneous data returned by the ultrasonic detectors. By doing so, we aim to enhance the overall reliability and accuracy of our system's operations.

We found that the occurrence of erroneous signals received by the ultrasonic sensors is often sporadic and shows significant variation compared to adjacent detection results. In light of this, we have considered an approach based on detecting neighboring values to eliminate these erroneous signals.

The flow diagram of our filter code is shown in Figure 32.

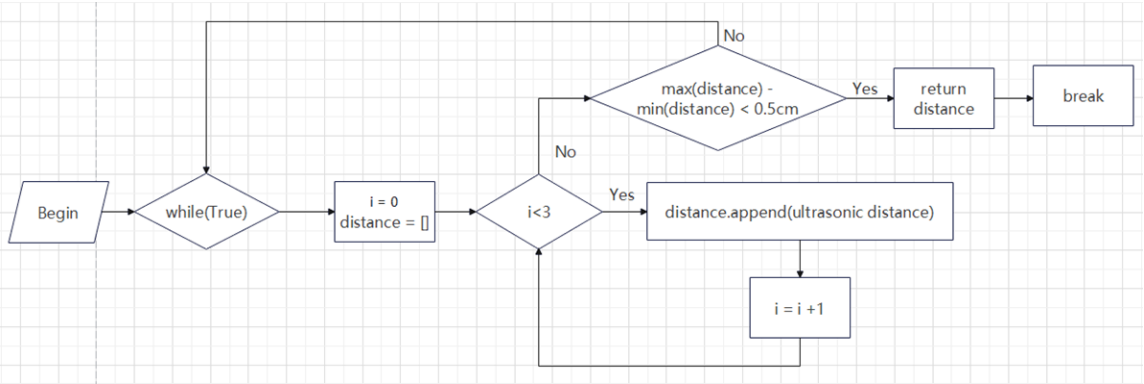



Figure 32. The flow diagram of the ultrasonic sensor detection filter

Our fundamental idea is to eliminate abnormal signals by comparing the difference between each detection result and its adjacent results. In program design, we take three measurements as one cycle and judge the measurement results within each cycle. If we find a result in a certain cycle that has a significantly larger difference compared to the other two results in same cycle, we discard the measurement results of that cycle and continue to detect until the ultrasonic detector detects a result with a smaller difference.

In our program, the threshold for the difference is set at 5mm. This means that if the difference between the results of the three measurements exceeds 5mm, those three results are discarded, and the measurements are repeated until the difference between the results is less than 5mm.

With the implementation of the filtering program, the probability of the ultrasonic detector returning erroneous signals is significantly reduced, leading to a notable enhancement in the overall robustness of our system. This crucial improvement allows for more accurate and reliable detection, minimizing false positives or false negatives that could potentially compromise the system's performance. By effectively mitigating the occurrence of erroneous signals, we ensure a higher level of precision and confidence in the functionality of our ultrasonic detector, ultimately bolstering the system's ability to operate flawlessly in diverse environments and challenging conditions.

### 3.9.2 Filtering of Laser Ranging Sensor

| **Section Author:** | **Zhao Zixun** |
|---|---|
| | **UESTC ID: 2020190907019, GUID: 2614146Z** |

In actual tracking tests, errors or misleading signals may be generated due to the accuracy and reliability of laser-ranging sensor devices and surrounding uncertain people or objects. These signals caused the microcontroller to make a wrong judgment, end present execution process in advance, and executed the next instruction in the wrong position. In order to eliminate the influence of similar problems, a laser-ranging sensor filtering system is designed.

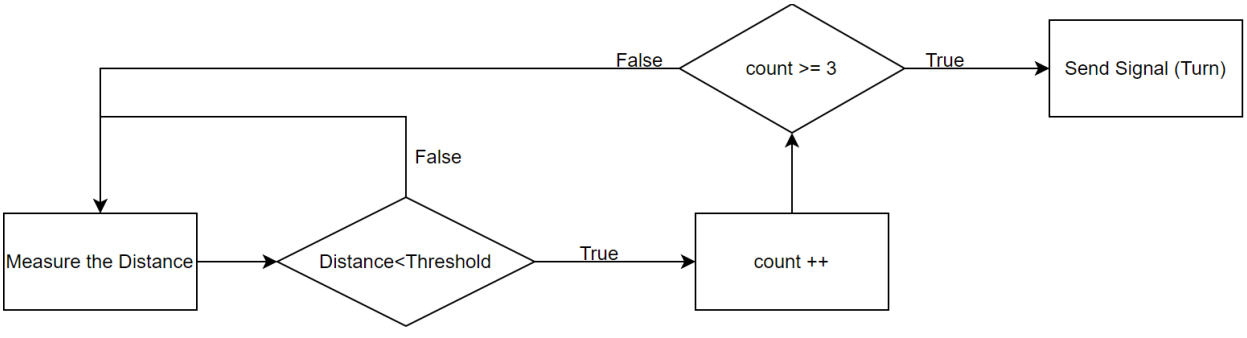


Figure 33. Flow Logic Diagram of The Laser-Ranging Sensor Filter System

Referring to the flow chart above, the laser ranging sensor on the front of the car would continuously perform the "measuring distance" task. When a distance value less than the set threshold was received, a count operation was performed. When the received distance value was greater than the threshold, it returns to the execution of the "Measure distance" task without performing the count operation. At the same time, once a return distance value was greater than the threshold, the completed count would be cleared to zero. After counting, the results were compared. If the number of consecutive counts reaches three times, the next operation is carried out, and the signal of executing the turn was sent to OpenMV through Mbed. If the number of counts was less than three, the execution of the measurement task was returned until the number of consecutive counts reached three.

## 3.10 Software: Bridge Crossing Algorithm

| **Section Author:** | **Gao Yinuo** |
|---|---|
| | **UESTC ID: 2020190907038, GUID: 2614165G** |

### 3.10.1 Task Analysis

The objective of this task was to go on the bridge, then going through the bridge surface which was composed of wire mesh, and finally off the bridge.

### 3.10.2 Experimental Design

The real-time speed of the robot could not be obtained because the quadrature decoder was not used, which means that the speed of going on the bridge need to be measured in the field. After inspection, it was found that the speed of the line tracking was too small to go on the bridge, so it is necessary to increase the vehicle’s speed before going on the bridge. The vehicle kept moving at a constant speed on the bridge surface.

Two points needed to be paid attention to when the vehicle was going to get off the bridge. One was that the speed of the robot should be decreased before going off the ramp to avoid a rollover caused by a high speed. Another was that the laser-ranging sensor in the front of the vehicle should be turned off before crossing the bridge. As shown in Figure 34 below, the distance between the laser-ranging sensor and the ground was likely to be less than the threshold which was preset to be 30cm because this sensor was facing down when going down the bridge. In order to prevent triggering this laser-ranging sensor incorrectly, it was necessary to close it before going on the bridge, which also meant to delay the activation of

this sensor.

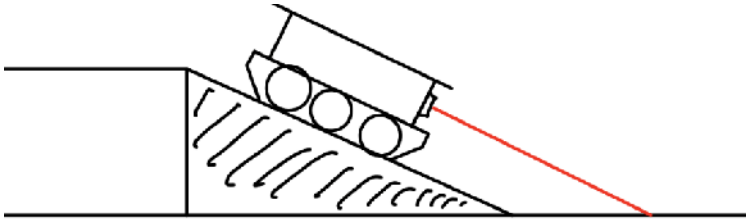


Figure 34. Sketch Map when the Vehicle was going off the bridge

### 3.10.3 Important Considerations

1. As for why the quadrature decoder is not used, the main reason was that calling the function of the quadrature decoder in OpenMV would make the related code and wiring more complicated, while debugging the vehicle 's speed was easier and more straightforward. OpenMV had no built-in function of quadrature decoder [8] and we had to write by ourselves.

2. As for why the laser-ranging sensor was used to detect the beacon instead of using the ultrasonic module, the main reason was that there have been already four ultrasonic modules in the two sides of the robot. This means that there existed interference among them [9]. If another ultrasonic module was added, the degree of interference would increase, resulting an inaccurate result.

3. Tests showed that the apron wheels could pass smoothly through the wire mesh on the bridge surface, which also confirmed that it was appropriate to choose the apron wheels.

### 3.10.4 Results

By adjusting the speed of going on the bridge, on the bridge surface and off the bridge, the robot can cross the bridge successfully.

## 3.11 Software: Fence Tracing Algorithm

| **Section Author:** | **Jin Shuyi** |
|---|---|
| | **UESTC ID: 2020190501014, GUID: 2614217J** |

### 3.11.1 Objectives

The objective of this task is to complete the transition from the prismatic region to the basket and throw the ball. In terms of our group, we commanded the rover to go along the handrails of the orange line just as shown in Figure 35 below.

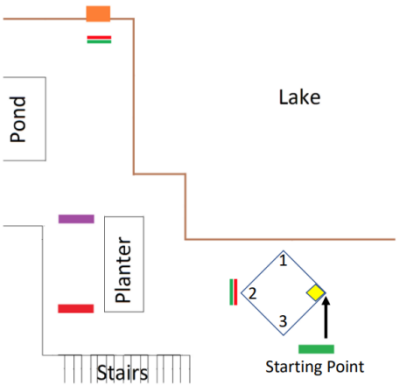



Figure 35. The transition in Patio 2

### 3.11.2 Fence Tracking Feedback System Working Mechanism

Due to the stationary nature of the handrails, we decided to employ an ultrasonic sensor instead of beacons for fence tracking. This approach enhances the rover's stability. Examining the predetermined route for the rover's transition from task 1 to task 2, we can utilize the working mechanism illustrated in Figure 36 to guarantee the rover's movement.

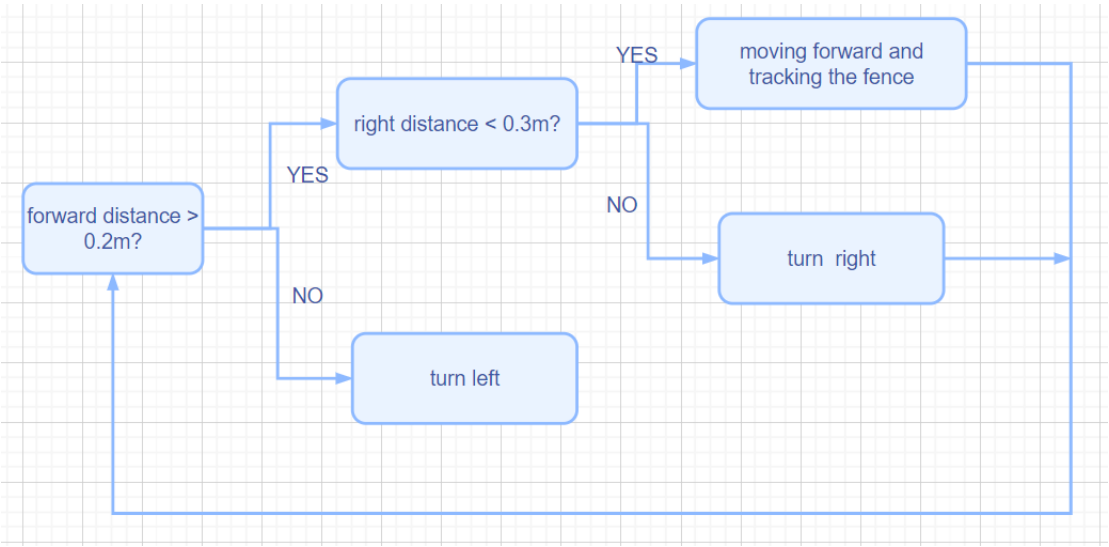



Figure 36. The fence tracking flow chart for transition

## 3.11.3 Ultrasonic Sensor Fence Tracking Module

### 3.11.3.1 The Ultrasonic Sensors Location and performances

The HC-SR04 ultrasonic Sensor is used and its specific location is shown in Figure below.

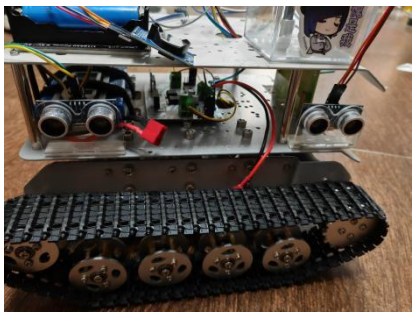


Figure 37. The Ultrasonic Sensors

The specific performances are shown below.

| Test items | Performances |
|---|---|
| Maximal detecting distance | 0.51m |
| Minimal detecting distance | 0.02m |
| Sampling frequency | More than 50Hz |
| Detecting accuracy | Less than 1cm |
| Maximal incidence angle | 28° |

Table 7. Specific performances of ultrasonic Sensor

These above characteristics verifies the ability of the rover from 3 aspects: the detecting range, the detecting accuracy and the detecting frequency. These abilities enables the rover to successfully finish the task.

### 3.11.3.2 P-P Control Mechanism for Controlling the Distance

When the rover is experiencing the phase of tracking the fence and moving forward, the 2 ultrasonic sensors are utilized and the relevant mechanism is shown in Figure.5.

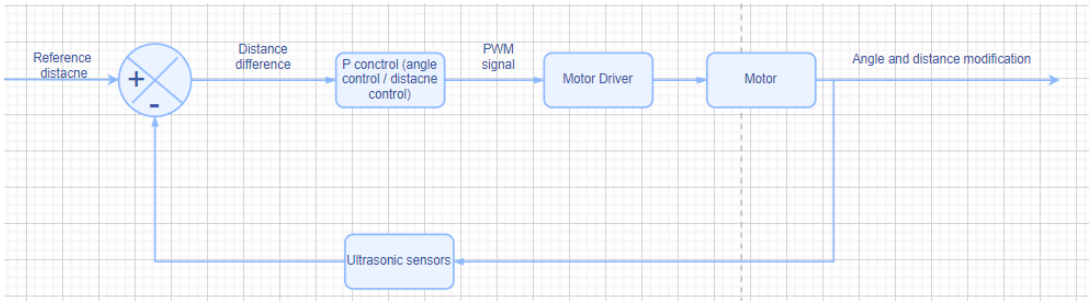



Figure 38. The fence tracking flow chart for transition

In Figure 38, the motor control of the OpenMv is dynamically governed by the data acquired from the ultrasonic sensors. This setup incorporates two proportional controls, one for the rover's yaw angle and another for the distance between the rover and the fence. While the

method presented in following part can theoretically handle simultaneous adjustment of both distance and angle, field tests have revealed that the distance adjustment falls short of our expectations. Consequently, an additional distance control has been introduced to enhance the distance adjustment performance. Both proportional controls operate concurrently, maximizing the system's real-time adjustment capabilities.

### 3.11.4 PID control of Theta error and Distance error

In real tests, we discovered that only the single PID control is hard to make the car smoothly track the fence. Therefore, we add 2 PID for both the Theta error and Distance error, just as shown in Figure 39, which help us complete this task together.

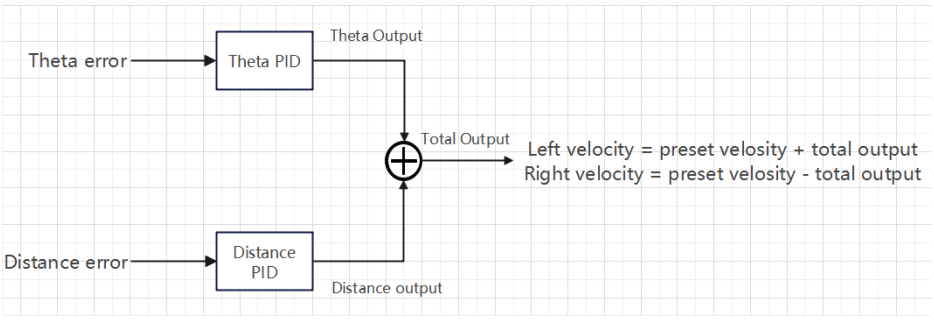



Figure 39. Flow chart of Theta PID and Distance PID

The principles are shown in Figure 40 and Figure 41.

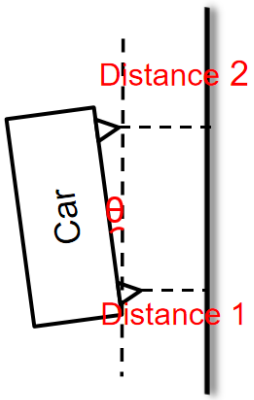



Figure 40. Theta Error

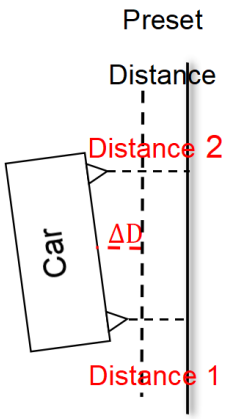



Figure 41. Distance Error

### 3.11.4.1 Distance P control for Distance error

The distance control is to control the distance between the rover and the fence. The distance between the rover and the fence can be expressed by

$$\text{current distance} = \frac{\text{Distance 1} + \text{Distance 2}}{2}$$

When the rover is moving forward and the current distance exceeds the desired distance, it should veer to the right. This means that the speed of the left track should increase while the speed of the right track should decrease, and vice versa. This behavior can be achieved using the following proportional control equation:

$$PWM_{left-previous} = PWM_{left-current} + K_P * (current\_distance - reference\_distance)$$

$$PWM_{right-previous} = PWM_{right-current} - K_P * (current\_distance - reference\_distance)$$

This is the first kind of PID which effectively avoid the rover to rushing to the fence or deviating the fence too much.

### 3.11.4.2 Theta P control for Theta error

In order to reduce the oscillation and better control the rover, another PID was also utilized by us—the Theta control. The core content of this PID control is to calibrate the yaw angle of the car. To be more precise, we introduce the geometric Figure 42 :

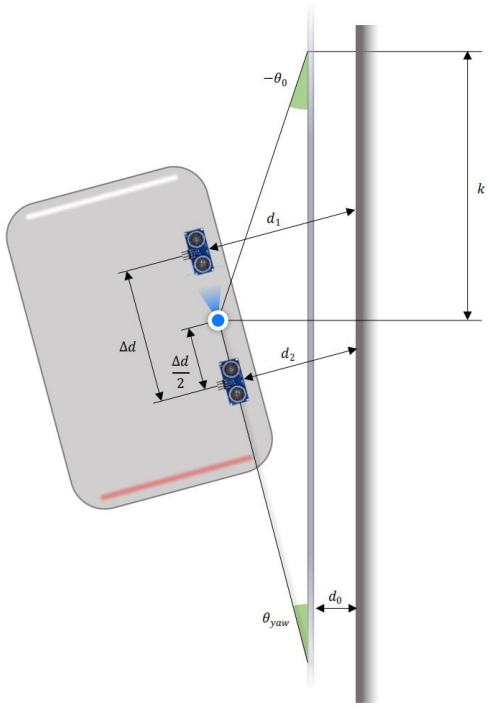



Figure 42. The geometric chart of the Theta control

In this figure, the average distance can be obtained by the equation

$$d_{avg} = 0.5(d1 + d2)$$

Also, it is obvious that there exists a deviation of the rover by$\theta_{\mathrm{yaw}}$ between the the rover and the reference axis. The angle can be obtained by

$$\theta_{\mathrm{yaw}} = \arctan\frac{\mathrm{d2} - \mathrm{d1}}{\Delta \mathrm{d}}$$

Meanwhile, to achieve the objective of having the rover return to and intersect with the reference axis at a distance of k, the yaw direction of the rover should be adjusted to an angle $\theta_0$ relative to the negative direction of the reference axis. This angle can be determined using the following geometric relationship:

$$\theta_0 = \arctan\frac{\mathrm{d} - \mathrm{d}_0}{\mathrm{k}}$$

Finally, the angle which the rover needs to calibrate is shown below

$$\theta_{\mathrm{calibrate}} = \theta_{\mathrm{yaw}} - \theta_0$$

By dividing the value by $\pi$, we can normalize it, resulting in an angular proportional gain ranging from -1 to 1. This normalized value will be utilized to regulate the discrepancy in motor speed.

# 3.12 Software: Basket Detection System and Ball Releasing System

| Section Author: | Xiao Ziyi |
|---|---|
| | UESTC ID: 2020190501029, GUID: 2614232X |

## 3.12.1 Principle

To detect the basket, a laser ranging module was used in front of the vehicle. When the forward distance obtained by the laser was smaller than a threshold, it would send a signal to the OpenMv to stop the vehicle.

As for the releasing of the table tennis ball, two function in Mbed were written, the grab function and release function.

The principle of SG90 was using PWM wave to control the angle of the servo, the relationship between pulse width of the PWM and the angle of SG90 was shown as follows. A typical period of PWM was 20ms.

| Pulse Width | Angle |
|---|---|
| 0.5ms | 0° |
| 1ms | 45° |
| 1.5ms | 90° |
| 2.0ms | 135° |
| 2.5ms | 180° |

Table 8. Relationship of Pulse Width and Angle in SG90

## 3.12.2 Grab Function

The grab function was to ensure the ball would not fall from the mechanical arm when the vehicle was operating. This function was set to run at the beginning of the patio. The pulse width was set to be 0.5ms which could grab the ball very tightly.

## 3.12.3 Release Function

The release function was to release the ball when the vehicle moves to the basket. The pulse width was set to be 2.0ms. This pulse width should be large enough to release the ball completely. When Mbed received the signal from OpenMv that the basket was in range, this

function would be run.

## 3.13 Software: Connection between OpenMV and Mbed

| | |
|---|---|
| **Section Author:** | **Zhang Jiaqi** |
| | **UESTC ID: 2020190501001, GUID: 2614204Z** |
| **Assisted by:** | **Zhao Zixun** |
| | **UESTC ID: 2020190907019, GUID: 2614146Z** |

### 3.13.1 Task Analysis

In order to alleviate the issue of insufficient pin numbers on the OpenMV, we utilized the STM32L432KC to control the mechanical arm, the real-time clock module, and the HC-12 wireless transceiver module. Additionally, we leveraged the STM32L432KC to connect with the laser ranging module to obtain the distance between the car and the obstacle ahead, and timely communicate this information to the OpenMV to determine whether a response is necessary. As a result, we designed an effective inter-microcontroller communication scheme for this course project. This scheme only requires two GPIO interfaces from each microcontroller for implementation and does not demand any additional communication protocol support from the pins.

At the beginning of the design, four communication modes were considered, which were serial communication, I2C communication, SPI communication and USB communication.

First, if serial communication was used, OpenMV could provide serial port data transmission function through the UART interface, and the Mbed can also achieve data sending and receiving through its own UART interface. Through this communication mode, data transmission can be realized through a simple communication protocol, but the reliability was poor and easy to be interfered.

The second was I2C communication, which could ensure that OpenMV as a host can well control multiple slaves and communication with an address, device addressing is simple. Although the car uses one OpenMV and two Mbed, one of the Mbed did not need to communicate with OpenMV, so there was no multi-master-slave situation, so 12C communication was not used

The third SPI communication, the speed was fast, and the parallel transmission could avoid communication conflicts. However, the same clock frequency was required, and additional clock lines need to be reserved in some applications. SPI communication protocol needs the

same clock frequency or even extra clock line in some cases, which increases the complexity of the structure, so the scheme was abandoned.

Finally, USB communication, you can use the USB interface on the OpenMV board, you can realize the communication between the Mbed through the USB interface, support hot plug, plug and play. Because OpenMV's USB port often needs to be connected to a computer to regulate parameters during testing, so it is abandoned.

As for the serial communication mode, after comprehensive consideration, since the communication between OpenMV and Mbed only occured when the turning operation was required, Mbed would send a one-way turning signal to OpenMV, so in order to further simplify the structure and prevent OpenMV from dropping frames when too much data needs to be processed, the serial communication mode was changed to Mbed sending high or low level signals representing 1 or 0 to OpenMV through pins to realize the transmission and communication of instruction signals. When the pin low signal was converted to a high level signal, it means that the turning signal was generated.

The same communication logic was also used in releasing the table tennis which would be mentioned in Section 3.12.

### 3.13.2 Working Principle

#### 3.13.2.1 Interface Definition

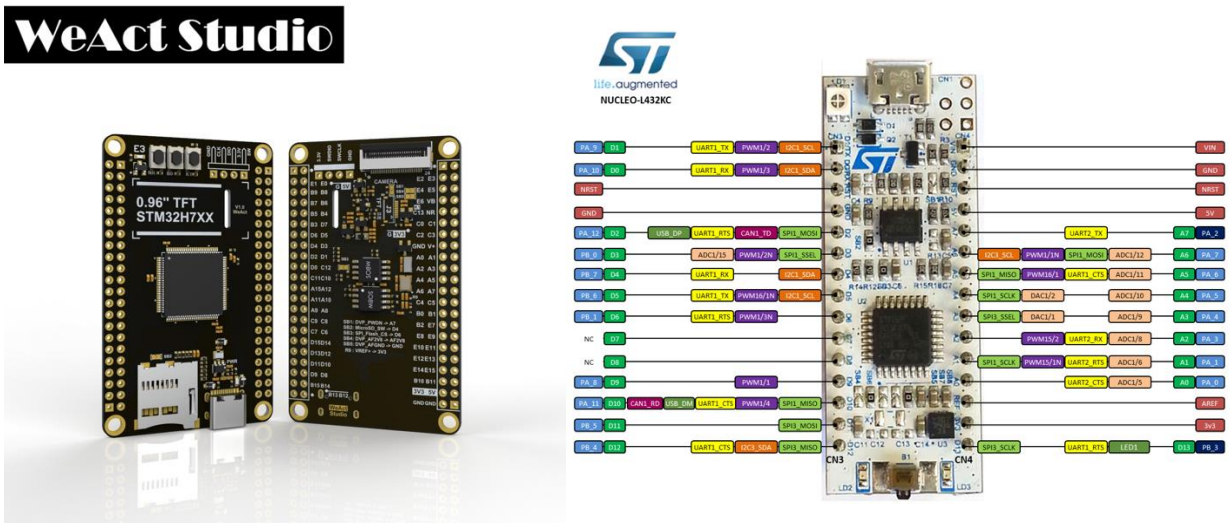



Figure 43. Interface Definition of OpenMv and Mbed

| Pin | Type of Pin | Pin on the MBED |
|---|---|---|

| | | |
|---|---|---|
| Communication between MBED and OpenMV(comm_out) | Digital Out | A3 |
| Communication between MBED and OpenMV(comm_in) | Digital In | A4 |

Table 9. Communication Pin Guideline for MBED

| Pin | Type of Pin | Pin on the OpenMV |
|---|---|---|
| Communication between MBED and OpenMV(comm_out) | Digital Out | P6(PA5) |
| Communication between MBED and OpenMV(comm_in) | Digital In | P9(PD14) |

Table 10. Communication Pin Guideline for OpenMV

#### 3.13.2.2 Signal transmission

##### 3.13.2.2.1 Patio 1

In Patio1, after completing Task1, we need to turn and ascend a bridge. We placed a beacon next to the bridge, hence, the need to activate the front-facing laser distance sensor to detect the beacon. When the vehicle is less than 30 cm from the beacon, it will stop and turn to face the bridge squarely.

During this process, the OpenMV first needs to raise the voltage of the **comm_out** pin. Concurrently, the **comm_in** pin of the STM32L432KC detects this high-level input, triggering the laser to measure the forward distance. Once the vehicle is less than 30 cm from the beacon, the **comm_out** pin of the STM32L432KC raises to high voltage level, causing the **comm_in** pin of the OpenMV to detect this high level. Consequently, the OpenMV is informed that it should stop and turn.

Subsequently, both the OpenMV and STM32L432KC set the **comm_out** pin to low voltage level, signifying the completion of an inter-microcontroller communication session.

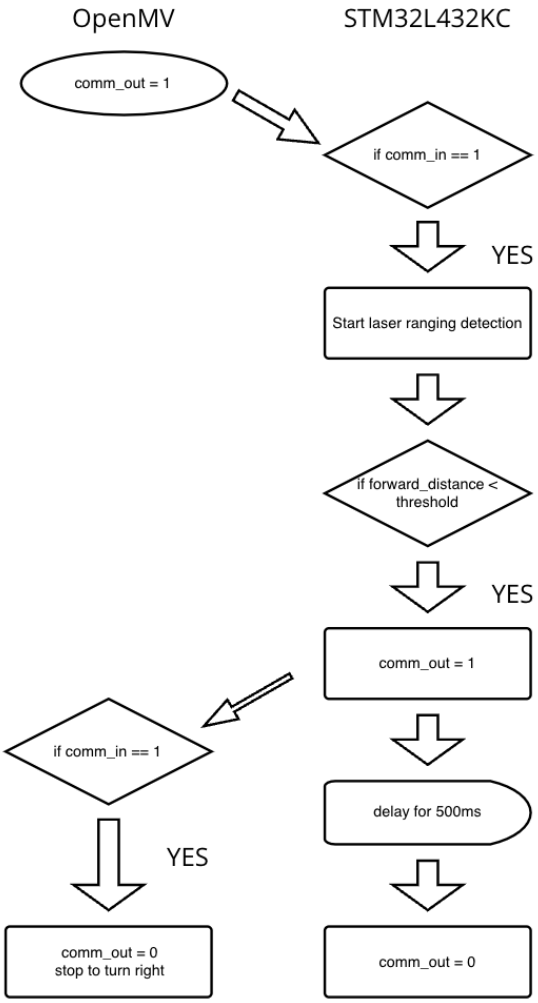



Figure 44. Communication working principle for patio1

### 3.13.2.2.2 Patio 2

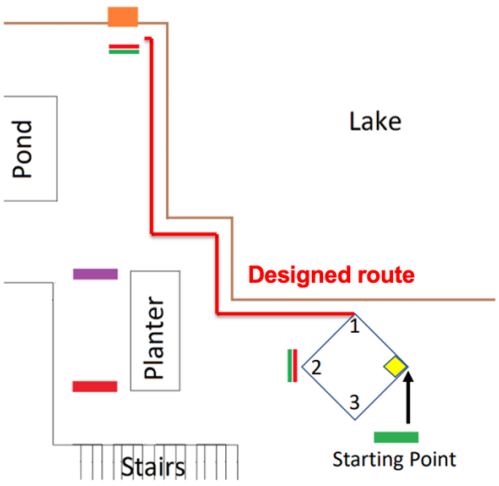



Figure 45. Designed route for Patio 2

In Task2 of Patio2, we need to follow the designed route to reach the front of the basket. The second and fourth turns require us to activate the laser ranging sensor located at the front of the car to detect the distance between the car and the front fence, to determine whether a turn is necessary. The interaction logic here is roughly the same as the one used in Patio1 for beacon detection.

Simultaneously, the front-facing laser ranging sensor will also be used for basket detection to judge whether the car has arrived at the basket's location, thereby executing the ball-releasing operation with the mechanical arm.

Throughout the whole process, the number of communications between OpenMV and STM32L432KC is fixed, and the purpose of each communication is clear and distinct. As a result, we can program the STM32L432KC to determine its behavior when it receives signals from OpenMV for a certain number of times.

For instance, in Task2 of Patio2, when the **comm_in** pin of STM32L432KC receives a high-level signal for the first time, it should activate the front-facing laser ranging sensor to alert the car to turn in time. After the completion of the turning operation, both OpenMV and STM32L432KC should simultaneously set the **comm_out** pin to a low level, marking the completion of a communication session.

When the **comm_in** pin of STM32L432KC receives a high-level signal for the second time, it should perform the same operation as before, that is, making the fourth turn in the designed route and heading towards the basket.

When the **comm_in** pin of STM32L432KC receives a high-level signal for the third time, it should activate the front-facing laser ranging sensor to alert the car whether it has reached the front of the basket. Once the car is in front of the basket, it should stop, and the STM32L432KC should control the mechanical arm to release the ping-pong ball. With this, Task2 of Patio2 is completed.

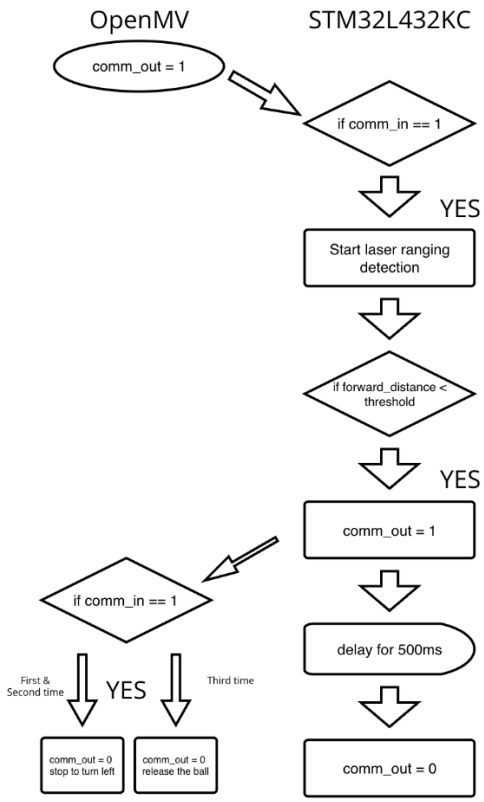



Figure 46. Communication working principle for patio2

In Task 3 of Patio 2, we need to drive the car to a specified location and use the HC-12 module to send messages to the laptop. The OpenMV is responsible for controlling the car's movement, while the STM32L432KC is responsible for driving the HC-12 to send messages to the laptop through the UART communication protocol.

After completing Task 2, our car will first drive to the designated spot and stop. Then, the OpenMV will raise the voltage level of the **comm_out** pin. Upon receiving the high-level signal at the **comm_in** pin, the STM32L432KC immediately drives the HC-12 module to send messages.

### 3.13.3 Results

During the testing and demonstration process in Patio1 and Patio2, this inter-microcontroller communication scheme has effectively and efficiently accomplished the tasks required. OpenMV can receive signals and make turns when needed, and STM32L432KC can also complete its tasks when it needs to release the ping-pong ball and drive the HC-12 module to send messages.

Furthermore, after multiple tests, the stability of this communication system has been validated.

# 3.14 Software: Real Time Obtain and Wireless Transmission

| Section Author: | Liu Ruitong |
|---|---|
| | UESTC ID: 2020190501036, GUID: 2614239L |

## 3.14.1 Problem Analysis

The problem of task3 in patio2 required the car stop in a determined area and send the team number, team member names and Time of date (24-hour clock) in a radio signal at 433MHz. In this task, we have three tasks to accomplish:

Obtain real-time time using a clock module: This task involves selecting and integrating a suitable clock module and writing code to retrieve and parse the time information from the module. It is important to ensure the accuracy and stability of the clock module to provide precise real-time time when required.

Send radio signals through a Bluetooth module: This task requires selecting and integrating a Bluetooth module to transmit data wirelessly to other devices. Code needs to be written to establish communication with the Bluetooth module and send the desired signals.

Receive signals on a PC: This task involves setting up the PC to receive the signals sent through the Bluetooth module. It requires configuring the PC's Bluetooth settings and developing code or using software to receive and process the incoming signals.

## 3.14.2 Module Selection

### 3.14.2.1 Wireless Communication module

In this part, we have chosen the HC-12 Bluetooth module as the device for transmitting information. The HC-12 wireless serial communication module is a next-generation, multi-channel embedded wireless data transmission module.

Figure 47. HC-12 Module

The reasons for choosing the HC-12 module are:

1. Low power consumption: The HC-12 module operates with low power consumption, which is crucial for projects that need to run for extended periods. It can be configured with different power levels to meet the requirements of various applications and maximize battery life.
2. Cost-effective: The HC-12 module is cost-effective, making it a budget-friendly choice. It enables reliable data transmission without significantly increasing project costs.
3. Suitable frequency band: As stated in the requirements, the message needs to be transmitted as a radio signal at 433 MHz, and the HC-12 module's wireless operating frequency range falls within 433.4 to 473.0 MHz, making it a suitable choice that aligns with the project's specifications.

#### 3.14.2.2 Clock Module

For the clock module, we have opted for the DS3231 module as our choice. The DS3231 is a highly accurate real-time clock (RTC) module.

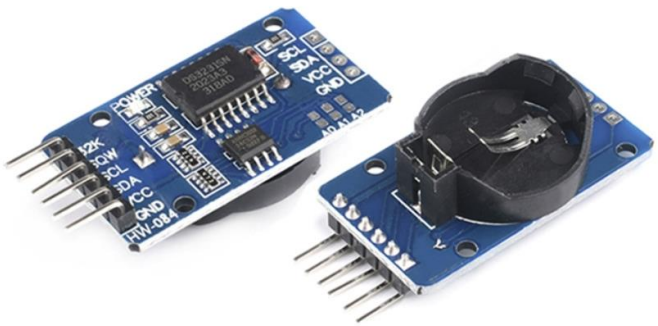


Figure 48. DS3231 Module

The reasons for choosing the DS3231 are:

1. High Accuracy: The DS3231 provides precise timekeeping with a deviation of only a few seconds per month.
2. Backup Battery Support: The DS3231 module includes a backup battery that allows it to retain timekeeping data during power outages or when the main power source is disconnected. This feature ensures that the module can continue operating without losing track of time.
3. I2C Communication: The DS3231 module communicates with the microcontroller using the I2C protocol, which is a widely used standard for serial communication. This makes it easy to interface the module with various microcontrollers and integrate it into different projects.
4. Low Power Consumption: The module is designed to operate with low power consumption, making it suitable for battery-powered applications or situations where energy efficiency is crucial.

#### 3.14.2.3 Control Board

In this task, a STM32L432KC is chosen as the control board. The reason for choosing DS3231 is because L432 has both I2C and UART interfaces, allowing simultaneous communication with both the Bluetooth module and the clock module. Additionally, the L432 communicates with the OpenMV module. When the car reaches the destination, the OpenMV module will send a signal to the L432 to initiate the transmission of information.

#### 3.14.2.4 Receiving Device

Here, we have chosen the HC-12-USB as the receiving device. This product consists of the core Bluetooth module HC-12t and the USB driver module CP2104. The greatest advantage of HC-12-USB is that you only need to plug the device into the computer, install the driver, and use a serial port debugging assistant to establish communication between the PC and the HC12 module connected to the microcontroller.

## 3.14.3 System Block Diagram

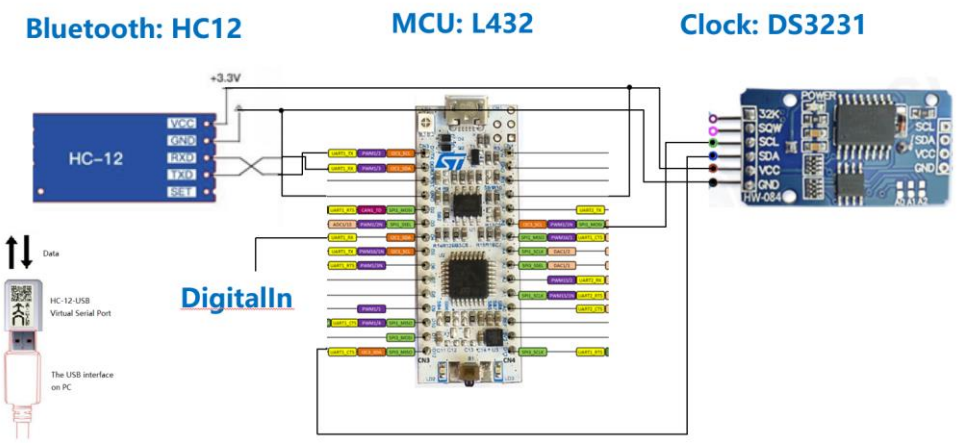



Figure 49. System Design of Real Time Obtain and Wireless Transmission

## 3.14.4 Software Implementation

### 3.14.4.1 Real Time Obtain

The communication method used by the DS3231 module is I2C (Inter-Integrated Circuit), where the L432 communicates with the DS3231 module by reading and writing to its registers. This allows for the initialization and retrieval of date information. The timing diagram of data transfer on I2C Serial Bus is as follows:

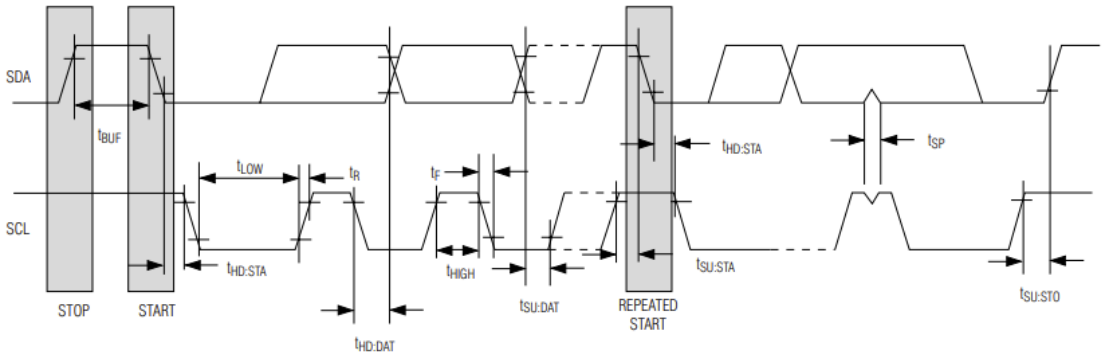



Figure 50. I2C Communication of DS3221

From the timing diagram, it can be observed that the start signal occurs when SCL is high and SDA transitions from high to low. The stop signal occurs when SCL is low and SDA transitions from low to high. After the start signal, SCL should be pulled low to prepare for data transmission. Once the SDA level changes, SCL is pulled high to send the data (starting with the most significant bit) in a loop for 8 times (one byte). During the first 8 SCLK cycles,

the write command byte is inputted, and the data byte is inputted on the rising edge of the following 8 SCLK cycles. The data input starts with bit 0. According to the DS3231 data sheet, the address of DS3231 is 1101000, so the write address for the host is 0xD1, and the read address is 0xD0. The procedure for reading the register pointer is as follows: start signal – write device address – write register address – start signal – receive data – stop signal. The method to modify or read the date or time is by modifying or reading the values stored in the registers. For the first time using, we need to write the real-time into DS3231. After the initial setup, there is no need to set the time again unless there is a power failure. The read function follows a similar process, but the procedure is reversed. In this case, the SDA line is used for input, and the host acts as the receiver. The following image shows the clock and calendar registers of DS3231:

| ADDRESS | BIT 7 MSB | BIT 6 | BIT 5 | BIT 4 | BIT 3 | BIT 2 | BIT 1 | BIT 0 LSB | FUNCTION | RANGE |
|---|---|---|---|---|---|---|---|---|---|---|
| 00h | 0 | 10 Seconds | | | Seconds | | | | Seconds | 00–59 |
| 01h | 0 | 10 Minutes | | | Minutes | | | | Minutes | 00–59 |
| 02h | 0 | 12/$\overline{24}$ | $\overline{AM}$/PM<br>20 Hours | 10 Hours | Hour | | | | Hours | 1–12 + $\overline{AM}$/PM<br>00–23 |
| 03h | 0 | 0 | 0 | 0 | 0 | Day | | | Day | 1–7 |
| 04h | 0 | 0 | 10 Date | | Date | | | | Date | 01–31 |
| 05h | Century | 0 | 0 | 10 Month | Month | | | | Month/Century | 01–12 + Century |
| 06h | 10 Year | | | | Year | | | | Year | 00–99 |

Figure 51. Address of DS3221

### 3.14.4.2 Send-time

First, we need L432 to receive a signal from Openmv. I set up a digitalout pin on Openmv and a digitalin pin on L432 to receive the transmitted signal. When the car reaches the destination, Openmv will set the pin pull-up to drive L432 and send the information.

For the Bluetooth module, HC-12 offers four transparent transmission modes and 127 communication channels that can be configured based on our requirements. In accordance with the task requirement, the default CH001 (433.4MHz) channel was selected. For the transmission mode, considering the length of the message to be sent, we selected the FU3 mode for transmission.

As the HC-12 combines the N32S032 microcontroller in a large package, data can be transmitted directly by UART commands.

The data in UART serial communication is organized into blocks called packets or frames, block of the framework includes:

The Start bit: the start bit is the synchronization bit added before the actual data. The start bit marks the beginning of the packet. Usually, the idle data line, that is, when the data

transmission line does not transmit any data, it remains at the high voltage level (1).

To start data transmission, send UART to pull the data line from high level to low level (from 1 to 0). The receiving UART detects this change from high to low on the data line and starts reading the actual data. Usually, there is only one start bit.

Stop bit: the stop bit, as the name suggests, marks the end of the packet. It is usually two bits long, but usually only one bit is used. To end the transmission, UART keeps the data line at high voltage (1).

Parity bit: parity allows the receiver to check whether the received data is correct. Parity is a low-level error checking system, which has two types: even check and odd check. Parity bit is optional, but it is not widely used.

Data bits: data bits are the actual data transmitted from the sender to the receiver. The length of the data frame can be between 5 and 9 (9 bits if parity is not used, and only 8 bits if parity is used). Generally, LSB is the first bit of data to be transmitted (unless otherwise specified).

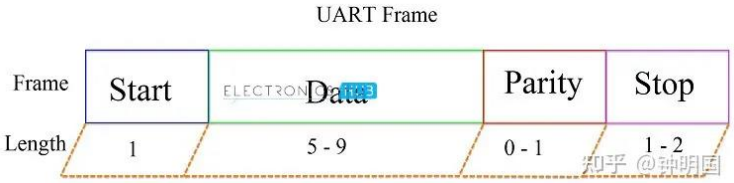



Figure 52. Frame of UART

In order to make the two parts communicate smoothly, initialize the UART communication by opening a UART communication port on microcontroller or device. The UART protocol is a half-duplex communication protocol. In half-duplex communication, data can be transmitted in two directions, but only one direction at a time. Since the HC-12 only need to send message, UART is available in this task. To establish UART communication between L432 and the HC-12 module, the baud rate, data bits, parity, and stop bits of L432 needed to be confirmed to match the settings of the HC-12 module. The default baud rate for HC-12 is usually 9600 bps. Here we chose the default mode and use HC-12-USB to receive information. In addition, the baud rate of the serial port debugging assistant should also be set to match the baud rate setting of the HC-12 module.

Once the UART communication is set up, data can be sent to HC-12 module by writing the data to the UART transmit buffer of your microcontroller or device. The data will be transmitted serially through the UART interface to the HC-12 module. Similarly, to receive data from the

HC-12 module, the data can be read from the UART receive buffer of L432. The received data will be transmitted serially through the UART interface by the HC-12 module. It's important to ensure that both the microcontroller or device and the HC-12 module have compatible UART settings for successful communication. Additionally, make sure to connect the HC-12 module to the appropriate UART pins (TX and RX) of L432. After sending a message, the LED indicator of HC-12-USB shows its status. If the LED flashes yellow, it indicates that the USB has received the information. Serial port debugging assistant can be used to view the transmitted content.

#### 3.14.4.3 Result

When the car arrives, we use Serial debugger assistance to check the message received by HC-12-USB. From the diagram, it can be seen that the displayed group number, member names, and time are all correct.

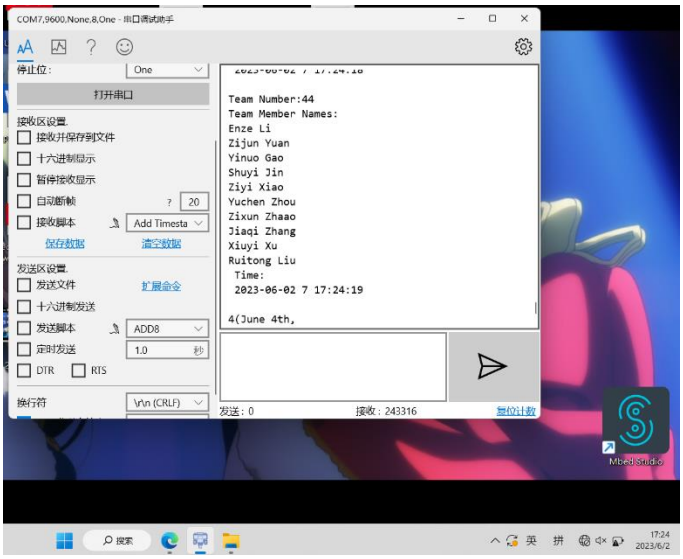



Figure 53. Date Receive on Computer

## 3.15 The Movement from the Basket to the Destination in Task 3

| Section Author: | Li Enze |
|---|---|
| | UESTC ID: 2020190501026, GUID: 2614229L |

After putting the ping pong ball into the basket, the vehicle would go back for a small distance and then rotated 90 degrees to the left. This was to prevent the car from colliding with the basket when turning to left. Next, the car should go to the area for task 3 wireless communication beside the planter. However, due to the rough and uneven gravel pavement,

there would be three situations, in Figure 54, 55 and 56.

1. The car went straight and went to the area directly, shown in Figure 54. In this situation, ultrasonic module on both sides would detect nothing (too far away). Then, by setting a stop time at 50 second, the car would stop automatically in the area of task 3.

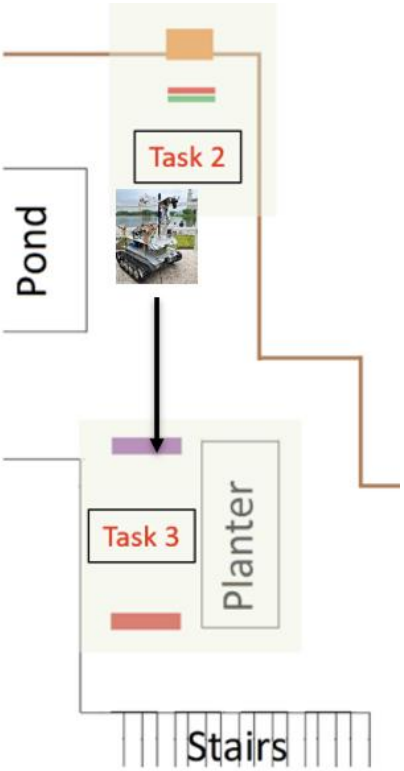



Figure 54. Situation 1: the Movement from the Basket to the Destination in Task 3

2. The car went on left direction and the ultrasonic module on the right side would detect the planter, shown in Figure 55. Therefore, the car should turn right for 135 degrees and adjust its direction to move along the side of planter with PID control. Then, the car would turn left and went into the area until it stopped.

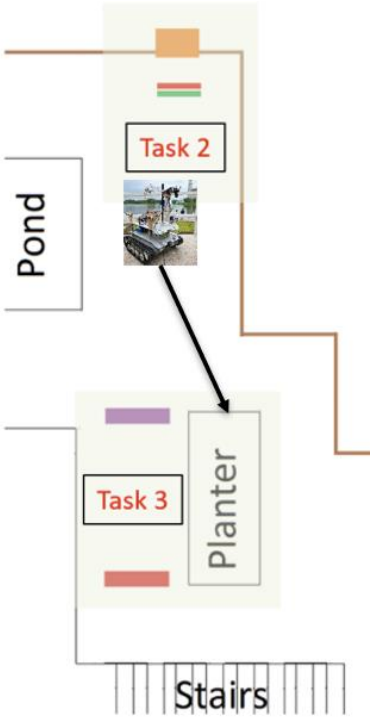



Figure 55. Situation 2: the Movement from the Basket to the Destination in Task 3

3. The car went on right direction and the ultrasonic module on the right side would detect the pond, shown in Figure 56. Therefore, the car should turn left for 45 degrees and adjust its direction to move along the side of pond with PID control. Then, the car would go straight until it went into the area and stopped.

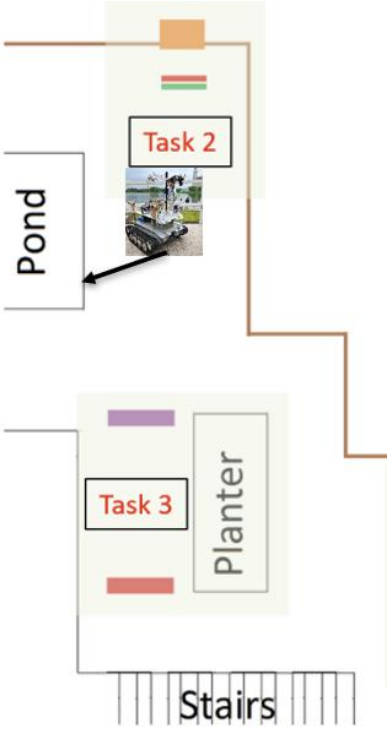


Figure 56. Situation 3: the Movement from the Basket to the Destination in Task 3

Finally, after the car went into the area and stopped, the STM32 would pull up comm-out pin to do the wireless communication as task 3.

# 4 Backup System Design

## 4.1 Mechanical Arm

| Section Author: | Yuan Zijun |
|---|---|
| | UESTC ID: 2020190907020, GUID: 2614147Y |

### 4.1.1 Introduction

In ping-pong delivery task, the rover should put a ping-pong ball into the basket. There are many methods can achieve this goal, such as robotic arm, slope structure, etc. In these choices, the first scheme we chose was a vertical delivery structure created by plastics bottle, which is shown in **Figure 57**.

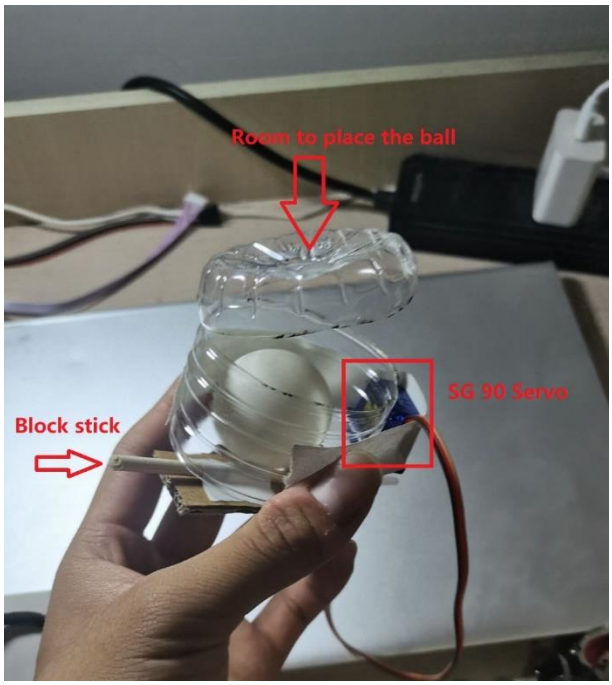



Figure 57. Structure of ping-pong delivery system

### 4.1.2 Shortcomings

However, this structure has a series of shortcomings. For instance, the height of the basket is about 30 cm. This means this structure must be higher than that value. In addition, it should be put in the front of the rover, which will conflict with Openmv, which is more important and irreplaceable. Another reason of abandoning it is the low compatibility between rover and the structure. To make the structure higher than the basket, a bracket should be design for the structure. However, if the bracket is made by another plastic bottle or cardboard, it will be unreliable. Considering the bridge task of Patio 1, the center of gravity is important. When go on and go down the bridge, the rover will shake back and forth in large amplitude. If the structure is not robust and stable enough, it may separate from the rover or even make the rover rollover. As a result, the structure with screws and nuts is a better choice.

## 4.2 Line Tracking Algorithm based on Neural Network

| **Section Author:** | **Zhou Yuchen** |
|---|---|
| | **UESTC ID: 2020190501013, GUID: 2614216Z** |

### 4.2.1 Introduction

Line recognition is a process that involves analyzing the color or grayscale variations in an image to extract key line features. By detecting these features, such as the edges or distinct patterns, a computer system can generate a fitted straight line that serves as a reference for guiding a robot along a desired path. This approach enables the robot to navigate autonomously while adhering to the prescribed route.

The Fully Convolutional Neural Network (FCN) is a deep learning model used for image segmentation and semantic segmentation tasks. Image segmentation involves classifying each pixel in an image and assigning it to different objects or regions, enabling detailed analysis of the image [10, 11]. Unlike traditional Convolutional Neural Networks (CNNs), FCN removes fully connected layers and introduces transpose convolutional layers to progressively restore low-resolution feature maps to the resolution of the input image, enabling dense pixel-level predictions [10, 11]. In the context of autonomous driving, FCN can be used to segment and label roads, accurately distinguishing different components such as lane lines, road surfaces, and sidewalks. This is crucial for lane keeping, navigation, and safety in autonomous vehicles [10-12].

In our design, we plan to train a shallow fully convolutional neural network (FCN) specifically

designed for the task of semantic segmentation. Our goal is to accurately segment and identify gravel road paths in images, allowing us to extract the necessary line information for further analysis and decision-making. By leveraging the power of convolutional layers and the end-to-end learning capability of the FCN architecture, we aim to create a model that can effectively capture the intricate details and nuances of gravel road paths, enabling us to navigate and interact with the environment more autonomously and intelligently.

### 4.2.2 Data Collection and Preprocessing

We have gathered a dataset consisting of 74 road images, captured using a smartphone. Each image has dimensions of 3072 x 4096 pixels and is formatted as an 8-bit RGB image. Additionally, we have meticulously annotated each image with a binary semantic mask, as shown in figure 1. In this annotation, the road areas are delineated using white pixels, while the remaining background regions are represented by black pixels.

Figure 58. Image collected with its binary semantic mask

After annotating the images, we extracted 10 photographs as a validation test set, forming a training set of 64 images and a validation test set of 10 images. The preprocessing of the images involved converting them to grayscale, resizing them to 112x112 pixels, randomly rotating both the images and their corresponding masks, and normalizing the pixel values. These preprocessing steps were undertaken to ensure uniformity and facilitate further analysis and experimentation.

### 4.2.3 Model construction:

Taking into account the limited computational power of OpenMV, the FCN is designed as a shallow network consisting of only 5 convolutional layers and 1 transpose convolutional layer to restore the image to its original size. The network structure, as shown in Table 11.

| Layer (type) | Output Shape | Param # |
|---|---|---|
| input_17 (InputLayer) | [(None, 112, 112, 1)] | 0 |
| conv2d_95 (Conv2D) | (None, 112, 112, 16) | 160 |
| max_pooling2d_63 (MaxPooling | (None, 56, 56, 16) | 0 |
| conv2d_96 (Conv2D) | (None, 56, 56, 32) | 4640 |
| max_pooling2d_64 (MaxPooling | (None, 28, 28, 32) | 0 |
| conv2d_97 (Conv2D) | (None, 28, 28, 64) | 18496 |
| max_pooling2d_65 (MaxPooling | (None, 14, 14, 64) | 0 |
| conv2d_98 (Conv2D) | (None, 14, 14, 64) | 36928 |
| max_pooling2d_66 (MaxPooling | (None, 7, 7, 64) | 0 |
| conv2d_99 (Conv2D) | (None, 7, 7, 128) | 73856 |
| conv2d_transpose_15 (Conv2DT | (None, 112, 112, 128) | 147584 |
| conv2d_100 (Conv2D) | (None, 112, 112, 1) | 129 |

Total params: 281,793
Trainable params: 281,793
Non-trainable params: 0

Table 11. The network structure and output of each layer

Although the network architecture consists of merely six layers, it surpasses a staggering 280,000 trainable parameters, rendering it impractical for deployment on OpenMv. In order to address this limitation, it becomes imperative to refine the model's structure and reduce the parameter count. Notably, within the 280,000 parameters, the transpose layer alone encompasses more than 140,000 parameters, constituting approximately fifty percent of the total. Consequently, an avenue worth exploring is the potential elimination of the transpose layer to achieve a significant reduction in the model's training parameters.

To address this problem, we considered a neural network architecture called U-Net. U-Net is a deep learning architecture originally proposed by Olaf Ronneberger, Philipp Fischer, and Thomas Brox in 2015 for image segmentation tasks [13]. The U-Net structure consists of two parts: an encoder and a decoder.

The encoder is composed of a series of convolutional layers and pooling layers, used to gradually reduce the size of the feature maps and extract abstract features [13]. The decoder, on the other hand, is composed of a series of upsampling layers and convolutional layers, used to restore the size of the feature maps to the original image size and gradually generate segmentation results.

At each layer in the decoder, there is a corresponding connection to the feature map of the corresponding layer in the encoder. This is done to utilize skip connections and fuse more

detailed information.

U-Net plays a crucial role in reducing the number of model parameters. The skip connections between the encoder and decoder in U-Net allow the feature maps from the encoder to be directly passed to the decoder, reducing the redundancy of parameters. As a result, many layers in the encoder and decoder can share the same weights, reducing the overall parameter count of the network.

Additionally, U-Net replaces fully connected layers with upsampling operations. Traditional fully connected layers introduce a large number of parameters when transforming low-dimensional features to high-dimensional features. In contrast, U-Net uses upsampling layers such as transpose convolution or bilinear interpolation to gradually restore the size of the feature maps without introducing extra parameters. This approach effectively reduces the parameter count while maintaining resolution accuracy.

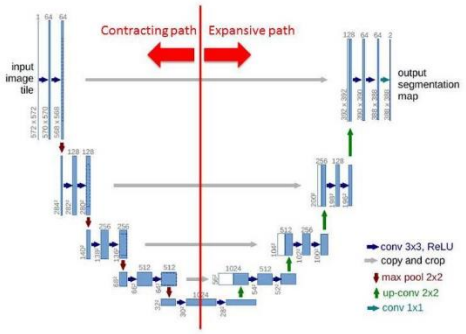



Figure 59. The structure chart of U-net[13]

A typical U-Net structure, as shown in Figure 59, indicates that the network still has a significant depth. To address this, we designed a shallow U-Net network consisting of 3 encoder layers and 3 decoder layers, with the input image size for this network is 112x112. The network structure is shown in table 12.

| Layer (type) | Output Shape | Param # |
|---|---|---|
| input_10 (InputLayer) | [(None, 112, 112, 1)] | 0 |
| conv2d_144 (Conv2D) | (None, 112, 112, 16) | 80 |
| conv2d_145 (Conv2D) | (None, 112, 112, 16) | 1040 |
| max_pooling2d_28 (MaxPooling | (None, 56, 56, 16) | 0 |
| conv2d_146 (Conv2D) | (None, 56, 56, 32) | 2080 |
| conv2d_147 (Conv2D) | (None, 56, 56, 32) | 4128 |
| max_pooling2d_29 (MaxPooling | (None, 28, 28, 32) | 0 |
| conv2d_148 (Conv2D) | (None, 28, 28, 64) | 8256 |
| conv2d_149 (Conv2D) | (None, 28, 28, 64) | 16448 |
| up_sampling2d_23 (UpSampling | (None, 56, 56, 64) | 0 |
| conv2d_150 (Conv2D) | (None, 56, 56, 32) | 8224 |
| conv2d_151 (Conv2D) | (None, 56, 56, 32) | 4128 |
| conv2d_152 (Conv2D) | (None, 56, 56, 32) | 4128 |
| up_sampling2d_24 (UpSampling | (None, 112, 112, 32) | 0 |
| conv2d_153 (Conv2D) | (None, 112, 112, 16) | 2064 |
| conv2d_154 (Conv2D) | (None, 112, 112, 16) | 1040 |
| conv2d_155 (Conv2D) | (None, 112, 112, 16) | 1040 |
| conv2d_156 (Conv2D) | (None, 112, 112, 1) | 17 |

Total params: 52,673
Trainable params: 52,673
Non-trainable params: 0

Table 12. The network structure of our U-net

It is evident that the model's parameters have been reduced to approximately 50,000, indicating that our model is capable of being deployed on OpenMV.

We performed 50 epochs of training on the model, and the training results are shown in Figure 60.

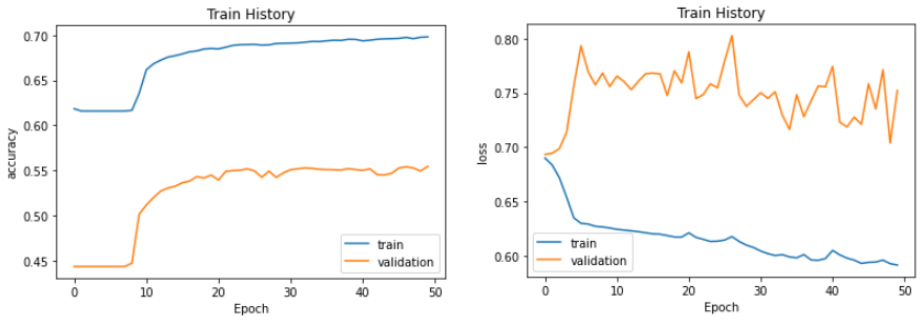



Figure 60. The training results of our model

(a)The accuracy of the model (b)The loss of the model.

It is evident that the model tends to converge after 50 training iterations. However, the highest accuracy achieved on the training set is only 70%, and the accuracy on the validation set is only 55%. This indicates that the model is still in an underfitting state, where it hasn't fully captured the patterns and complexities of the data.

The predicted result of the model is shown in Figure 61.

Figure 61. Prediction result of the model

(a)Original figure (b)Manually marked mask (c)Model generation

It seems that the masks generated by the model suffer from issues such as blurred edges, a relatively high number of false positives, and a lower true positive rate. Blurred edges can occur when the model fails to accurately capture the precise boundaries of objects in the images, resulting in a less precise segmentation. This can lead to a loss of important details and affect the overall quality of the segmentation results.

To address this problem without increasing the depth of the model, we plan to modify the loss function.

In our original design, training utilized the binary cross-entropy loss function, which calculated the overall image loss by computing the cross-entropy loss per pixel. The formula for this loss function is as follows:

$$\text{Binary crossentropy} = -\frac{1}{N}\sum_{i=1}^{N} y_i \log(p_i) + (1 - y_i)\log(1 - p_i)$$

Where $y_i$ is the true label of the $i$-th sample, $p_i$ is the predicted label of the $i$-th sample, and $N$ is the total number of samples [14].

It can be observed that in semantic segmentation tasks, the cross-entropy loss function has some limitations. The main issue is that it disregards the spatial correlation between pixels and the balance between classes. The cross-entropy loss function treats each pixel independently when calculating the loss, overlooking the spatial correlation between pixels. However, in semantic segmentation, the spatial relationships between pixels are crucial as neighboring pixels often belong to the same class. Therefore, neglecting spatial correlation can lead to blurry predictions in boundary regions of the segmentation.

To address these limitations, we have introduced the Structural Similarity Index (SSIM) loss

function and the Intersection over Union (IoU) loss function. The formulas for both loss functions are shown as follows:

$$\text{SSIM}(x, y) = \frac{(2\mu_x\mu_y + C_1) \cdot (2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1) \cdot (\sigma_x^2 + \sigma_y^2 + C_2)}$$

Where $x$ and $y$ represent the input images, while $C_1$ and $C_2$ are small constants added to stabilize the division. The terms $\mu_x$, $\mu_y$, $\sigma_x$, $\sigma_y$, and $\sigma_{xy}$ represent the mean and standard deviation values calculated from $x$ and $y$ [15].

$$\text{IoU}(A, B) = \frac{A \cap B}{A \cup B} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

Where, $A$ and $B$ represent the predicted box and the real box respectively. The value of an IOU ranges from 0 to 1. A larger value indicates a higher degree of overlap between the predicted box and the real box [16].

The SSIM loss function measures the structural similarity between the predicted segmentation map and the ground truth map. It takes into account the luminance, contrast, and structural similarities between corresponding pixels. By incorporating spatial information and pixel relationships, the SSIM loss function encourages the model to generate segmentation maps that preserve the structural details of the objects [15, 17].

On the other hand, the IoU loss function calculates the intersection over union between the predicted segmentation and the ground truth. It provides a measure of how well the predicted segmentation aligns with the true segmentation. By optimizing for a higher IoU, the model learns to produce more accurate and precise segmentation maps [16, 17].

By incorporating both the SSIM and IoU loss functions into the training process, we aim to overcome the shortcomings of the cross-entropy loss function and improve the performance of the model in terms of preserving spatial details and achieving accurate segmentations.

Finally, the total loss function of the model is the sum of the three individual loss functions. Therefore, the total loss function can be expressed as:

$$TotalLoss = \text{Binary crossentropy} + (1 - SSIM) + (1 - IoU)$$

The training results of the model with new loss function are shown in Figure 62.

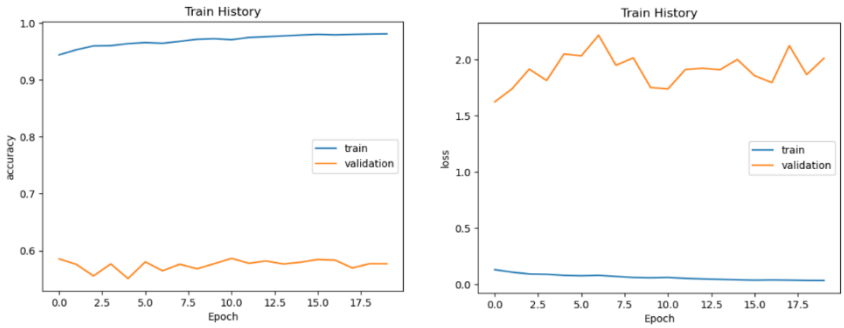



Figure 62. The training results with new loss function

(a)The accuracy of the model. (b)The loss of the model.

From Figure 63, it can be observed that the model has achieved a high accuracy of 0.95 on the training set, indicating a strong ability to fit the training data. Although the accuracy on the validation set is relatively lower at 0.6, it still represents a substantial improvement compared to the original model. This suggests that the modified loss function has effectively enhanced the model's generalization capabilities, allowing it to perform better on unseen data.

The predicted result of the model is shown in Figure 63.

Figure 63. Prediction result of the modified model

(a)Original figure (b)Manually marked mask (c)Model generation

It can be observed that with the help of the new loss function, the model produces segmentation boundaries that are more distinct. Additionally, there is a decrease in false positives and an increase in true positives. The current model has largely met our requirements and expectations.

### 4.2.4 Quantification and deployment

The final step in deploying our model on OpenMv is to perform quantization, specifically converting it to int8 data type, and then converting it to the tflite format. Model quantization is the process of converting a floating-point model into a fixed-point representation, thereby reducing the model's memory consumption and computational requirements. In our case, we

have quantized both the model's weights and activation values, transforming them into int8 format [9, 10]. This quantization allows for a significant reduction in the model's memory footprint and computational workload, making it more suitable for deployment on resource-constrained devices like OpenMv.

## 4.3 Ultrasonic Module based on Keil

| Section Author: | Yuan Zijun |
|---|---|
| | UESTC ID: 2020190907020, GUID: 2614147Y |

### 4.3.1 Introduction

Ultrasonic detector is an important block to measure the distance between the rover and the obstacle by measuring the time between the emission and reception of the ultrasonic signal. To realize these goal, HC-SR04 was chosen. 4 pins, which were Vcc, Tri, Gnd, Echo, were used to drive this device. Internal crystal was used for timing. In our design, the ultrasonic detector is used as “eyes” of the rover. By measuring the distance, the rover can judge whether the rover go towards the right direction and whether it go alone the balustrade. This device has high accuracy. In addition, it is easy to use. However, it has shortcomings including the 4-meter limitation of distance measuring and unexpected huge error. Furthermore, it can detect a large area because of the divergence of the sonic, which may cause misjudgment of obstacles.

### 4.3.2 Principle

The principle of measuring the distance between obstacle and the HC-SR04 is that the device firstly emits ultrasonic signal. The signal will move forward with the velocity at about 340 m/s, which is the spread velocity of the sonic in the air. When the ultrasonic signal hit the obstacle, it will be reflected back at the same velocity as before. After that, the signal will be received by HC-SR04. If the time between emission and reception is figured out, the distance between the device and obstacle can be calculated by calculation formula:

$$Distance = \frac{v * t}{2}$$

In the formula, “v” is 340 m/s, the spread velocity of sonic. “t” is the time between emission and reception. The product of v and t is the distance that sonic spread to the obstacle and then go back. As a result, the distance between the obstacle and device should be divided by Figure 64 shows the principle of measuring distance by ultrasonic.

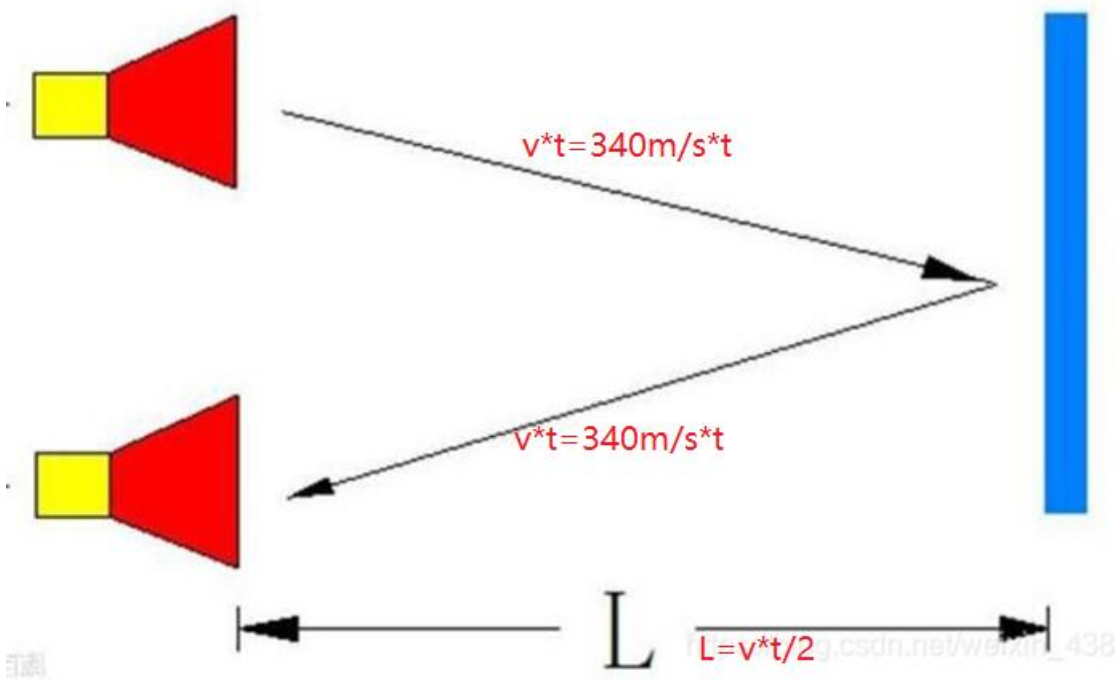



Figure 64. Principle of distance measuring by sonic signal

HC-SR04 can output and receive the ultrasonic signal. Figure 64 demonstrates the principle of HC-SR04.

There are 4 pins in HC-SR04, Vcc, Trig, Echo and Gnd. Vcc is the pin to supply power for HC-SR04 while Gnd is ground reference voltage. Trig pin is used to activate the device. When Trig pin receive a 10 us (or longer) pulse send by STM32, it will emit ultrasonic signal (8 period with 40kHz). Meanwhile, the Echo pin will pull up the output voltage. When the ultrasonic hits the obstacle and go back to the device, Echo will pull down the voltage, which means the time that the high voltage of the Echo lasts is the time that the ultrasonic signal travel in the air. Figure 65 shows the waveform of HC-SR04 when it is working.

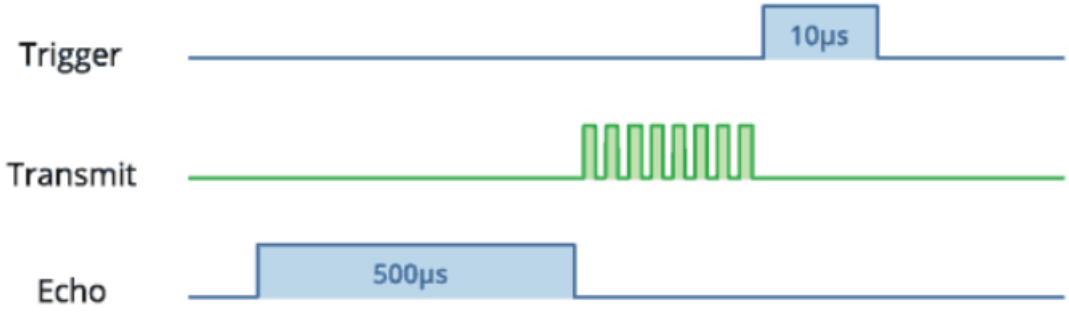



Figure 65. Working waveform of HC-SR04

To drive the HC-SR04, 4 pins of STM32 should be used. 3.3V and Gnd pins are connected to Vcc and Gnd pins of the HC-SR04 respectively to provide power supply. Another pin, connected to Trig pin, should output a pulse lasts more than 10us to activate the ultrasonic block. Finally, the pin which is connected to the Echo pin must have function of timer. The Echo high voltage time should be recorded to calculate the distance. Figure 66 shows the configuration of the STM32 pin. In this picture, PB3 is used to output pulse. PA0 is the timer pin, which is connected to Echo to record the time. Other pins are used to activate system and internal crystal clock of the STM32.

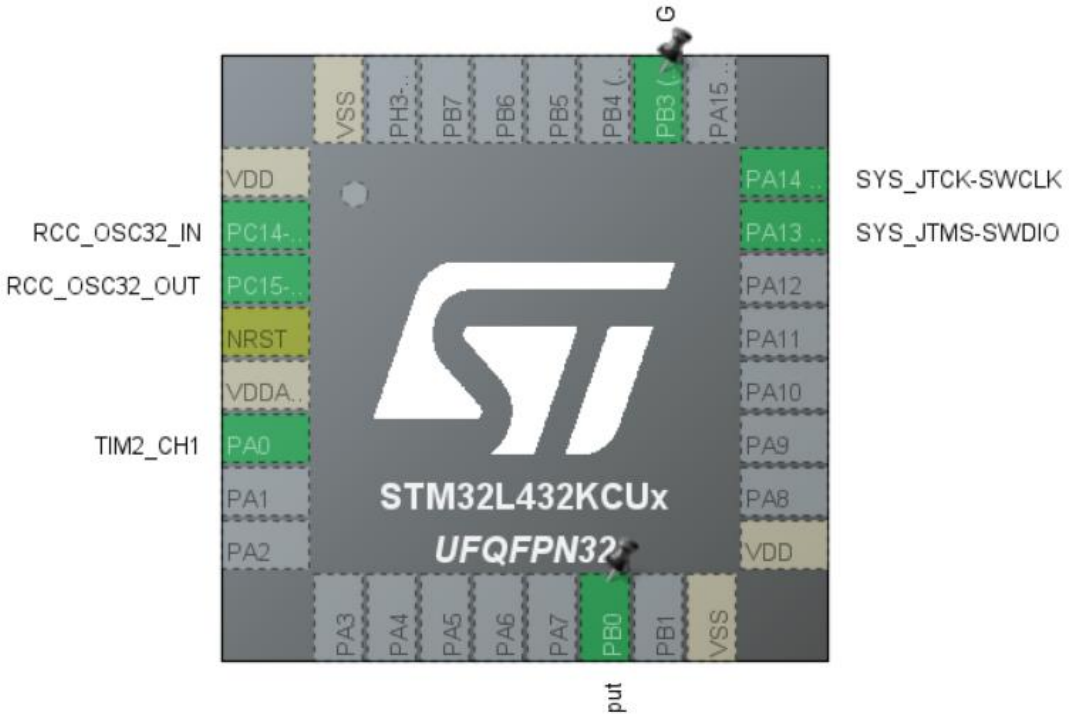



Figure 66. Pin configuration of the STM32

As for the code, a simple function of getting the value of distance is created. First, the pulse is output. As long as the Echo pin pull up the voltage, the timer will record the time. When the Echo pin pull down the voltage, the lasting time is cleared. In order to save the resource of STM32, a callback function was implemented to record the time. The timer was realized by configuring the internal crystal clock of the STM32. After that, the distance can be calculated by function:

$$Distance = \frac{340 * t}{2}$$

To get a more reasonable result, the units of these parameters should be changed. The distance is shown by centimeter and the accuracy is 0.1 centimeter. At the same time, for testing the block conveniently, UART was used to show the distance on the computer in real

time.

# 5. System Integration, Results and Discussion

## 5.1 Design of Working Flow

The working flow of Patio 1 is depicted in Figure 67. Initially, the rover initiates tracking along the road, continuously monitoring its position until the laser ranging sensor detects the presence of a beacon by measuring the distance. Subsequently, the rover executes a 90° right turn. To ensure safe traversal onto the bridge without veering off its edges, two ultrasonic sensors are employed in conjunction with a reference beacon. Upon successfully crossing the bridge, the rover proceeds forward until it detects the second beacon located on the opposite side. Following this, the rover executes a left turn to track along the final segment of the path until preset run time over.

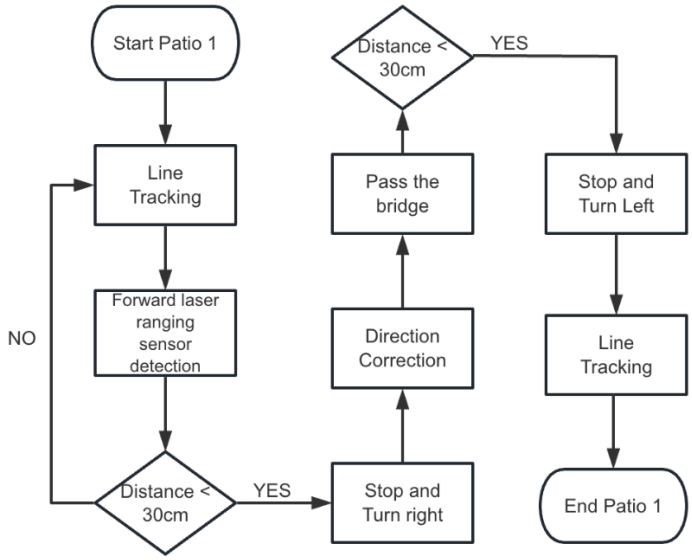



Figure 67. Working flow of Patio 1

The working flow of Patio 2 is depicted in Figure 68.

The rover first drives to the location where the arrow needs to be determined. The direction of the given arrow is then identified. After determining the arrow's direction, the rover moves towards the position corresponding to the arrow and knocks down the target object. The rover then turns and proceeds towards the railing. It drives alongside the railing, turn to left when the distance detected by the front laser ranging sensor is less than 30cm, and to the right when the distance to the railing on the right side is too far. Throughout the entire driving

process, a PID algorithm is applied to correct the rover's direction of movement. After completing a fixed number of turns, the front laser ranging sensor is used to detect the basket. Once the basket is detected, the rover stops and releases a ping pong ball, indicating the completion of the first two tasks. Finally, the rover turns and drives for a specific duration to reach the designated location. It then sends a message, which states all task finished.

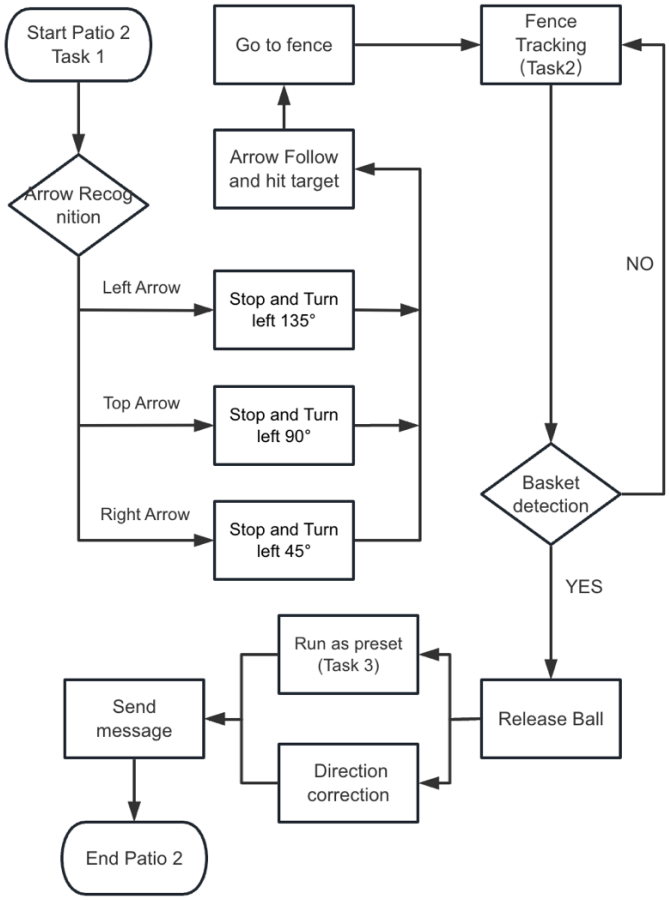



Figure 68. Working flow of Patio 2

## 5.2 Integration of Main Controllers and Components

The wire connection and components integration are shown in Figure 69.

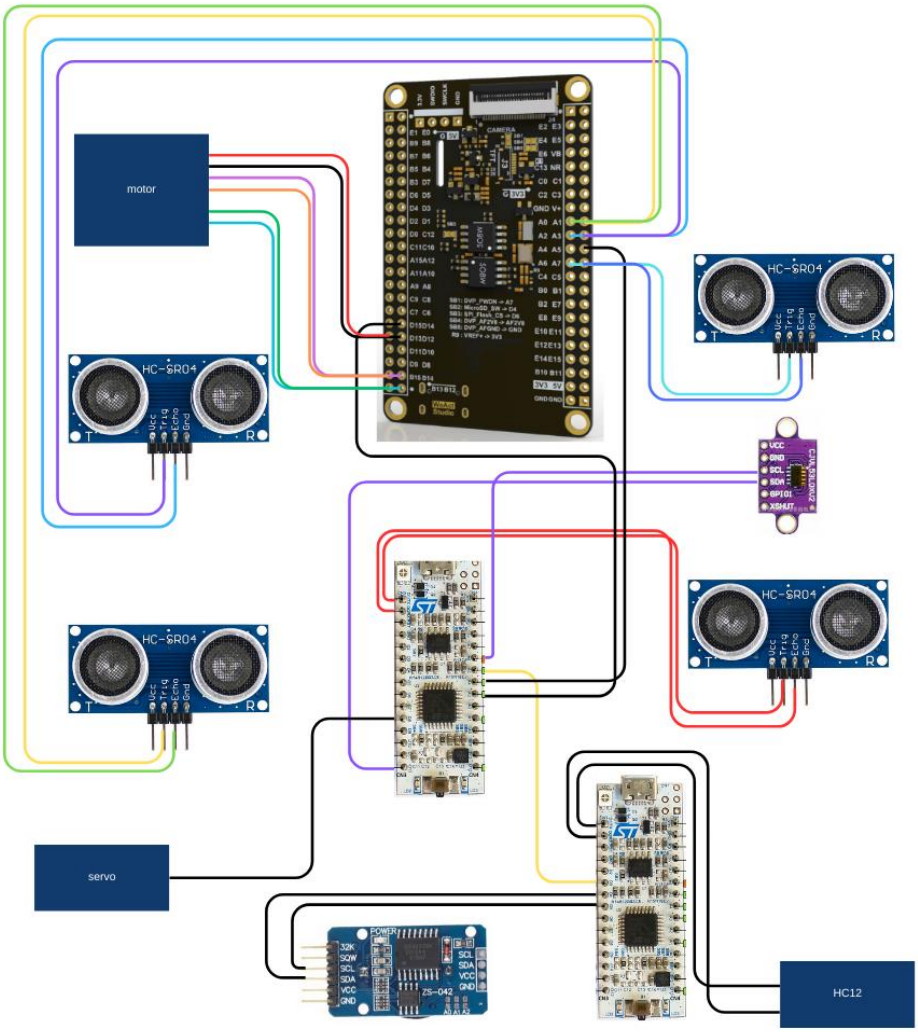



Figure 69. Connections among All Modules

Note: All components are grounded and powered by 5V or 3.3V converted from 18650 lithium batteries.

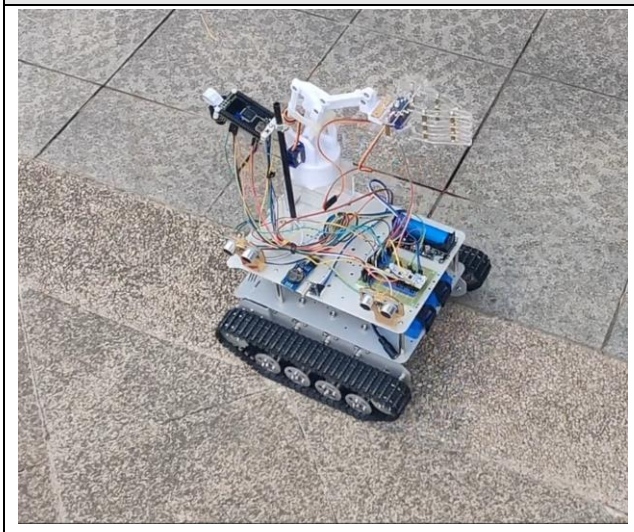
The connection between MBED and OpenMV is established by transmitting pulses through a single port. OpenMV serves as the primary controller for communication. Whenever MBED needs to execute a task, OpenMV sends a valid pulse to MBED. Based on the number of pulses received, MBED carries out the corresponding operations, sends back the relevant return value or status to OpenMV, and then awaits the next valid pulse to determine the subsequent operation.

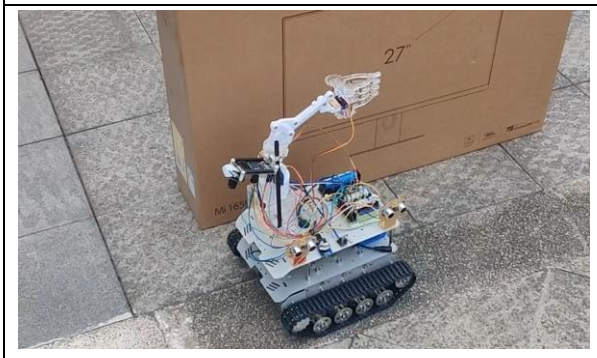
OpenMV acts as the primary controller, controls two mbeds to fulfill distinct functions. The communication between the mbeds is facilitated by OpenMV, which indirectly controls the

second mbed by sending instructions to the first mbed, which in turn controls the second mbed. This cascaded control approach allows for seamless parallelism among different devices without imposing an excessive burden on the control chip.

## 5.3 Results

### 5.3.1 Patio 1

| Results | Discussion of Results |
|---|---|
|  | **Start:** The rover was placed in the middle of the<br>lane and started to travel after three seconds when the battery on. |
|  | **Task 1 Straight Lane:** The rover travelled<br>straight with slight adjustments. |

| | |
|---|---|
|  | **Task 1 Curve:** The rover made a rapid turn at the angle when the OpenMV detected the different edges of the lane.<br><br>When the rover detects the curve, it should turn sharply to prevent travelling off-track. Note that the large pitch angle was a trade-off, as a large angle significantly improves the gravel recognition accuracy. |
|  | **Task 2 90° Turn 1:**<br><br>When the laser detects the beacon ahead, the rover performed an in place 90° turn to the right and aligned with the bridge's central axis. |
|  | **Task 2 Crossing the Bridge:** The rover travelled on the bridge fast. The rover's tracks stuck into the metal bars on the bridge, helping the rover align its heading. |
|  | **Task 2 90° Turn 2:** After detecting the beacon placed in front of its estimated path, the rover made a precise 90° turn to the left and then travelled forward, with approximately 10° error in heading. |

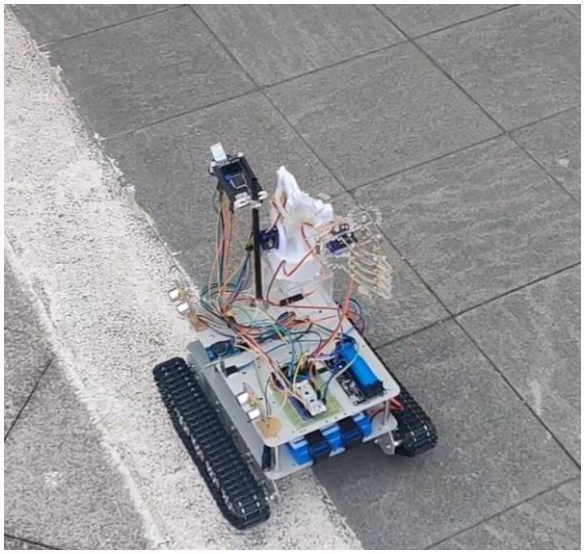
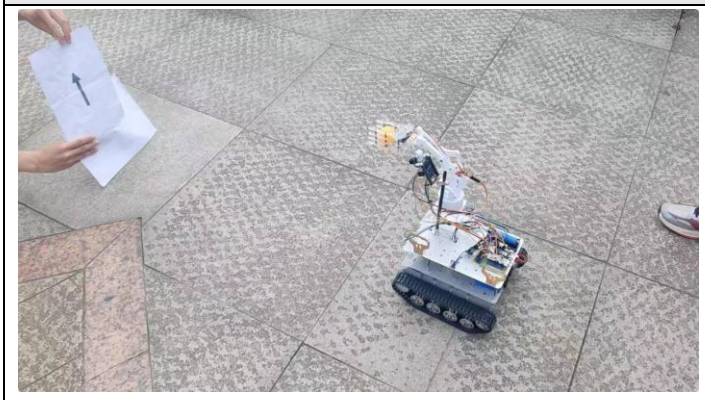
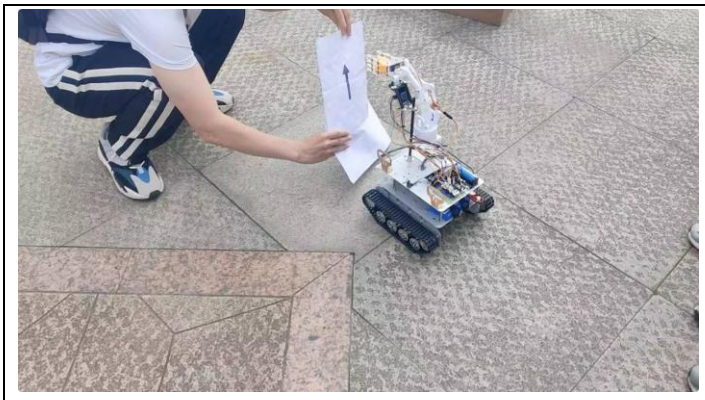
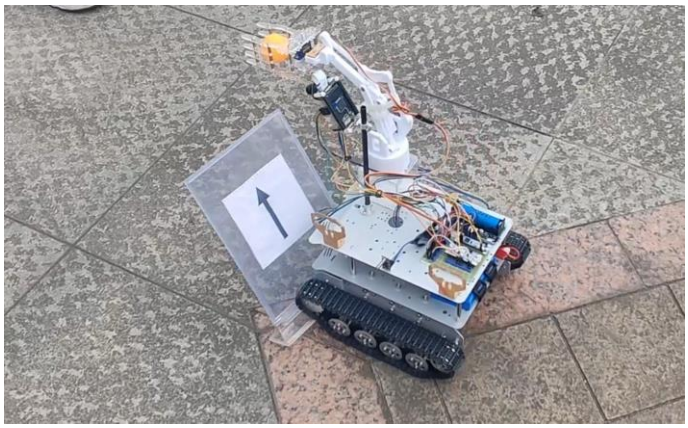
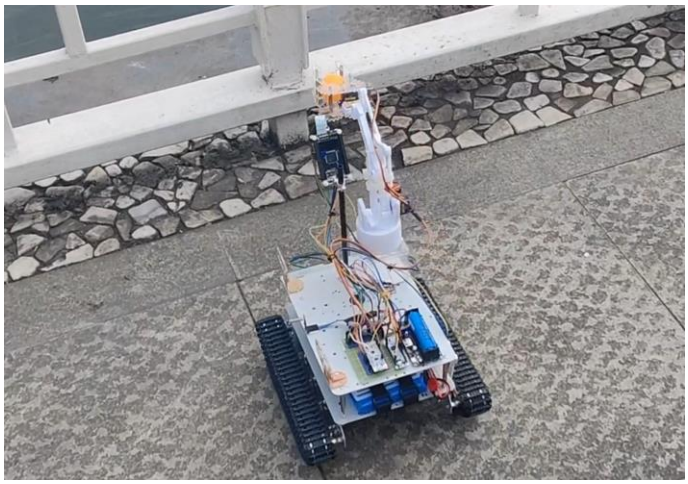
| | |
|---|---|
|  | **Task 3:** After entering the white gravel section, the rover rapidly corrected its heading and started tracking the gravel. The rover patrolled the line and travelled forward. |
|  | **Finish:** The rover stopped short after passing through the bridge, finishing all three tasks in Patio 1 with the time of 1:57. |

Table 13. Results of Patio 1

### 5.3.2 Patio 2

| Results | Discussion of Results |
|---|---|
|  | **Start**: The rover was placed in the middle of the field, where the starting line was located. |

| | |
|---|---|
|  | **Task 1 Shape Recognition:** The rover would recognize the shape on the standing board. |
|  | **Task 1 Target Tracking:** The rover achieved successful tracking in a straight line. The motor control was first designed as an open-loop system, leading to frequent yaw of the rover. To make the rover travel straight, a negative feedback system was added. It was verified to<br>be an effective way to eliminate offsets. |
|  | **Task 1 Strike of Target Board:**<br>The rover successfully strokes the target standing board. The rover<br>was designed to determine its position relative to the target board based on computer vision, in order to make the system more stable and more flexible for direction correction. |

| | |
|---|---|
|  | **The transition from Task 1 to Task 2:**<br><br>By orienting using ultrasound, the rover completes the transition phase.<br><br>The placement of the ultrasonic was crucial due of the uneven height of the fences. The rover consistently shown a reliable performance when moving around the fences. This was due to the fact that our ultrasonic was ultimately installed at a height of 12 cm, which was higher than the valid range of the ultrasound, which was measured in advance at a height of 11.5–12.5 cm. |
|  | **Task 2 Basket Tracking:**<br><br>A mismatch between the rover and the basket could result in the ball falling out of the basket, hence the curved design on the front of the rover was employed to prevent this. In the field tests, it was shown that the curve was helpful and that the basket could be precisely set in front of the rover. The pitching system was hence quite stable. |

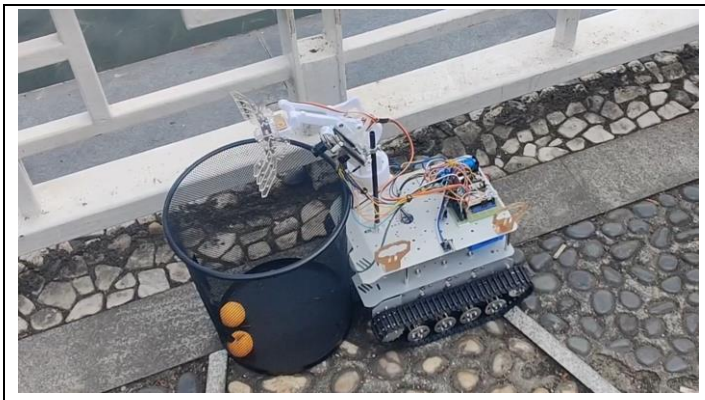
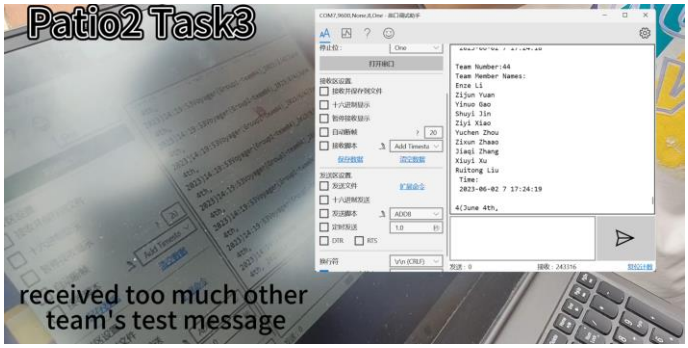
| | |
|---|---|
|  | **Task 2 Ball Releasing:** Controlled by the servo, the ball was successfully pitched in the basket. |
|   | **Task 3 Bluetooth:** The rover would transmit a Bluetooth signal once it had successfully reached the target Region. The rover was made to stop in the job area, giving the Bluetooth adequate time to keep sending signals until it successfully gets the return signal ACK from the computer, in order to prevent signal loss. The rover would next proceed from this area to the finish line. |

Table 14. Results of Patio 2

# 6. Conclusion

In the course Team Design Project and Skills (TDPS), a car which can finish total six tasks automatically in two patios were designed, built, tested and improved from scratch. To achieve each easy actions and functions, different modules were designed or brought as separate components of the system. Then, with the purpose of solving six complicated tasks, these separate components were connected through data transmission lines as well as power supply lines and controlled by three programmed microcontrollers to form a complete system.

The system was constructed by seven parts, including control units, ultrasonic modules, laser-ranging sensor, mechanical arm, motor driven module, wireless communication part

and power supply part.

For the centre of the system, the control units, there are three MCUs. The OpenMV received the input from ultrasonic modules to adjust the directions and achieve fence tracing with PID control. Additionally, OpenMV output the control signal to motor driven module to drive the car. Moreover, it contained line tracking algorithm and arrow recognition algorithm. For one STM32L432KC, it received forward distance signal from laser-ranging senser and controlled the action of mechanical arm. It also sent signal to OpenMV to help it control the system. The other STM32L432KC was only used to control the time signal from RTC module and do the wireless communication. It was worth noting that 12V power supply drove the motors while other components were powered by 5V power supply.

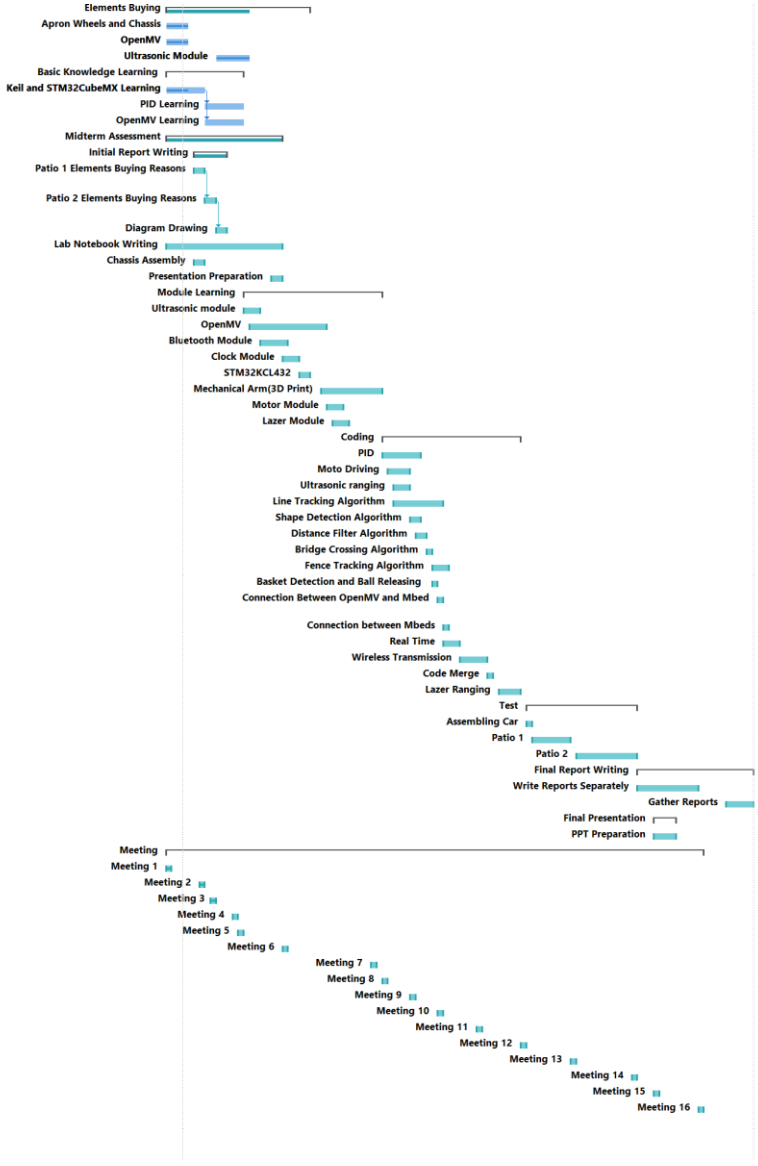
Throughout the process of designing and building the system, the 10 group members actively communicated and all worked hard. Not only did we learn how to build a complex system step by step like an engineer, but we also learnt some important lessons about working in a team. As a result, we finally constructed a perfect project car and demonstrated successfully.

# Appendix A: Financial Record

| Classification | Component | Quantity | Price(each/RMB) | Cost |
|---|---|---|---|---|
| **Control** | OpenMV | 1 | 409.0 | 409.0 |
| **Dynamic** | Ultrasonic Sensor | 4 | 15.0 | 60.0 |
| | Ultrasonic Sensor Holder | 4 | 10.0 | 40.0 |
| | Laser Sensor | 1 | 32.5 | 32.5 |
| | Laser Sensor Holder | 1 | 20.0 | 20.0 |
| | Apron | 1 | 23.2 | 23.2 |
| | Rover Bracket | 1 | 27.2 | 27.2 |
| | Tracks and Frame | 1 | 289.0 | 289.0 |
| | Motor | 2 | 20.0 | 40.0 |
| **Communication** | Bluetooth Module | 1 | 5.0 | 5.0 |
| | Clock Module | 1 | 5.0 | 5.0 |
| | Bluetooth USB Module | 1 | 20.0 | 20.0 |
| **PCB (with components)** | | | | 12.0 |
| **Other Components** | | | | 13.0 |
| | | | Total | 995.9 |

# Appendix B: Gantt Chart

| | | | | |
|---|---|---|---|---|
| **Elements Buying** | **20 个工作日** | **2023年3月6日** | **2023年3月31日** | |
| Apron Wheels and Chassis | 4 个工作日 | 2023年3月6日 | 2023年3月9日 | |
| OpenMV | 4 个工作日 | 2023年3月6日 | 2023年3月9日 | |
| Ultrasonic Module | 4 个工作日 | 2023年3月15日 | 2023年3月20日 | |
| **Basic Knowledge Learning** | **10 个工作日** | **2023年3月6日** | **2023年3月19日** | |
| Keil and STM32CubeMX Learni | 5 个工作日 | 2023年3月6日 | 2023年3月12日 | |
| PID Learning | 5 个工作日 | 2023年3月13日 | 2023年3月19日 | 6 |
| OpenMV Learning | 5 个工作日 | 2023年3月13日 | 2023年3月19日 | 6 |
| **Midterm Assessment** | **16 个工作日?** | **2023年3月6日** | **2023年3月26日** | |
| **Initial Report Writing** | **5 个工作日** | **2023年3月11日** | **2023年3月16日** | |
| Patio 1 Elements Buying Reasons | 2 个工作日 | 2023年3月11日 | 2023年3月12日 | |
| Patio 2 Elements Buying Reasons | 2 个工作日 | 2023年3月13日 | 2023年3月14日 | 11 |
| Diagram Drawing | 2 个工作日 | 2023年3月15日 | 2023年3月16日 | 12 |
| Lab Notebook Writing | 16 个工作日 | 2023年3月6日 | 2023年3月26日 | |
| Chassis Assembly | 2 个工作日 | 2023年3月11日 | 2023年3月12日 | |
| Presentation Preparation | 2 个工作日 | 2023年3月25日 | 2023年3月26日 | |
| **Module Learning** | **19 个工作日** | **2023年3月20日** | **2023年4月13日** | |
| Ultrasonic module | 3 个工作日 | 2023年3月20日 | 2023年3月22日 | |
| OpenMV | 10 个工作日 | 2023年3月21日 | 2023年4月3日 | |
| Bluetooth Module | 3 个工作日 | 2023年3月23日 | 2023年3月27日 | |
| Clock Module | 3 个工作日 | 2023年3月27日 | 2023年3月29日 | |
| STM32KCL432 | 2 个工作日 | 2023年3月30日 | 2023年3月31日 | |
| Mechanical Arm(3D Print) | 9 个工作日 | 2023年4月3日 | 2023年4月13日 | |
| Motor Module | 3 个工作日 | 2023年4月4日 | 2023年4月6日 | |
| Lazer Module | 3 个工作日 | 2023年4月5日 | 2023年4月7日 | |
| **Coding** | **17 个工作日** | **2023年4月14日** | **2023年5月8日** | |
| PID | 5 个工作日 | 2023年4月14日 | 2023年4月20日 | |
| Moto Driving | 3 个工作日 | 2023年4月15日 | 2023年4月18日 | |
| Ultrasonic ranging | 3 个工作日 | 2023年4月16日 | 2023年4月18日 | |
| Line Tracking Algorithm | 7 个工作日 | 2023年4月16日 | 2023年4月24日 | |
| Shape Detection Algorithm | 2 个工作日 | 2023年4月19日 | 2023年4月20日 | |
| Distance Filter Algorithm | 2 个工作日 | 2023年4月20日 | 2023年4月21日 | |
| Bridge Crossing Algorithm | 1 个工作日 | 2023年4月22日 | 2023年4月22日 | |
| Fence Tracking Algorithm | 3 个工作日 | 2023年4月23日 | 2023年4月25日 | |
| Basket Detection and Ball Releasin | 1 个工作日 | 2023年4月23日 | 2023年4月23日 | |
| Connection Between OpenMV and Mbed | 1 个工作日 | 2023年4月24日 | 2023年4月24日 | |
| Connection between Mbeds | 1 个工作日 | 2023年4月25日 | 2023年4月25日 | |
| Real Time | 3 个工作日 | 2023年4月25日 | 2023年4月27日 | |
| Wireless Transmission | 3 个工作日 | 2023年4月28日 | 2023年5月2日 | |
| Code Merge | 1 个工作日 | 2023年5月3日 | 2023年5月3日 | |
| Lazer Ranging | 2 个工作日 | 2023年5月5日 | 2023年5月8日 | |
| **Test** | **14 个工作日** | **2023年5月10日** | **2023年5月29日** | |
| Assembling Car | 1 个工作日 | 2023年5月10日 | 2023年5月10日 | |
| Patio 1 | 5 个工作日 | 2023年5月11日 | 2023年5月17日 | |
| Patio 2 | 7 个工作日 | 2023年5月19日 | 2023年5月29日 | |
| **Final Report Writing** | **15 个工作日** | **2023年5月30日** | **2023年6月19日** | |
| Write Reports Separately | 9 个工作日 | 2023年5月30日 | 2023年6月9日 | |
| Gather Reports | 3 个工作日 | 2023年6月15日 | 2023年6月19日 | |
| **Final Presentation** | **2 个工作日** | **2023年6月2日** | **2023年6月5日** | |
| PPT Preparation | 2 个工作日 | 2023年6月2日 | 2023年6月5日 | |
| **Meeting** | **70 个工作日** | **2023年3月6日** | **2023年6月10日** | |
| Meeting 1 | 1 个工作日 | 2023年3月6日 | 2023年3月6日 | |
| Meeting 2 | 1 个工作日 | 2023年3月12日 | 2023年3月12日 | |
| Meeting 3 | 1 个工作日 | 2023年3月14日 | 2023年3月14日 | |
| Meeting 4 | 1 个工作日 | 2023年3月18日 | 2023年3月18日 | |
| Meeting 5 | 1 个工作日 | 2023年3月19日 | 2023年3月19日 | |
| Meeting 6 | 1 个工作日 | 2023年3月27日 | 2023年3月27日 | |
| Meeting 7 | 1 个工作日 | 2023年4月12日 | 2023年4月12日 | |
| Meeting 8 | 1 个工作日 | 2023年4月14日 | 2023年4月14日 | |
| Meeting 9 | 1 个工作日 | 2023年4月19日 | 2023年4月19日 | |
| Meeting 10 | 1 个工作日 | 2023年4月24日 | 2023年4月24日 | |
| Meeting 11 | 1 个工作日 | 2023年5月1日 | 2023年5月1日 | |
| Meeting 12 | 1 个工作日 | 2023年5月9日 | 2023年5月9日 | |
| Meeting 13 | 1 个工作日 | 2023年5月18日 | 2023年5月18日 | |
| Meeting 14 | 1 个工作日 | 2023年5月29日 | 2023年5月29日 | |
| Meeting 15 | 1 个工作日 | 2023年6月2日 | 2023年6月2日 | |
| Meeting 16 | 1 个工作日 | 2023年6月10日 | 2023年6月10日 | |



# REFERENCE

[1] STMicroelectronics, "Dual Full-Bridge Driver L298n Datasheet," 2000.

[2] T. Williams, *The Circuit Designer's Companion*. Elsevier Science, 2004.

[3] STMicroelectronics, "Precision 500 mA regulators L78M Datasheet," 2020.

[4] B. Schäfer, M. Keuper, and H. Stuckenschmidt, "Arrow R-CNN for handwritten diagram recognition," *International Journal on Document Analysis and Recognition (IJDAR),* vol. 24, no. 1, pp. 3-17, 2021/06/01 2021.

[5] H. Schweitzer, R. Deng, and R. F. Anderson, "A Dual-Bound Algorithm for Very Fast and Exact Template Matching," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 33, no. 3, pp. 459-470, 2011.

[6] J. N. Sarvaiya, S. Patnaik, and S. Bombaywala, "Image Registration by Template Matching Using Normalized Cross-Correlation," in *2009 International Conference on Advances in Computing, Control, and Telecommunication Technologies*, 2009, pp. 819-822.

[7] L. D. Stefano, S. Mattoccia, and M. Mola, "An efficient algorithm for exhaustive template matching based on normalized cross correlation," in *12th International Conference on Image Analysis and Processing, 2003.Proceedings.*, 2003, pp. 322-327.

[8] X. Technology, "OpenMV Embedded Image Processing," 2022, [Online]. Available: https://book.openmv.cc/.

[9] C. Technologies, "HC-SR04 Ultrasonic Sensor User Guide," 2012.

[10] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431-3440.

11] T. He, Z. Zhang, H. Zhang, Z. Zhang, J. Xie, and M. Li, "Bag of tricks for image classification with convolutional neural networks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 558-567.

[12] B. Wu, F. Iandola, P. H. Jin, and K. Keutzer, "Squeezedet: Unified, small, low power fully convolutional neural networks for real-time object detection for autonomous driving," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2017, pp. 129-137.

[13] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted*

*Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, 2015: Springer, pp. 234-241.

[14] c. Wikipedia, *Cross entropy — Wikipedia, The Free Encyclopedia*.

[15] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing,* vol. 13, no. 4, pp. 600-612, 2004.

[16] F. van Beers, A. Lindström, E. Okafor, and M. A. Wiering, "Deep Neural Networks with Intersection over Union Loss for Binary Image Segmentation," in *ICPRAM*, 2019, pp. 438-445.

[17] H. Huang *et al.*, "Unet 3+: A full-scale connected unet for medical image segmentation," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020: IEEE, pp. 1055-1059.