MINISTRY OF EDUCATION & TRAINING

UNIVERSITY OF TECHNOLOGY AND EDUCATION

FACULTY OF MECHANICAL ENGINEERING

GRADUATION PROJECT

Topic: **DESIGN AND MANUFACTURE PICKLEBALL MACHINE**

Instructor:	**TS. ĐỖ VĂN HIẾN**	
Performer:	**NGUYỄN QUANG GIAO**	**MSSV: 19146121**
	BÙI TRƯƠNG VĨNH PHÚC	**MSSV: 20146518**
Class:	**19146CLA1 ; 201461A**	
Course:	**2019 – 2023 ; 2020-2024**	

Ho Chi Minh, January 2025

UNIVERSITY OF TECHNOLOGY AND EDUCATION

FACULTY OF MECHANICAL ENGINEERING

DEPARTMENT OF MECHATRONICS

GRADUATION PROJECT

Topic: DESIGN AND MANUFACTURE PICKLEBALL MACHINE

Instructor:	**TS. ĐỖ VĂN HIẾN**	
Performer:	**NGUYỄN QUANG GIAO**	**MSSV: 19146121**
	BÙI TRƯƠNG VĨNH PHÚC	**MSSV: 20146518**
Class:	**19165CLA1 ; 201461A**	
Course:	**2019- 2023 ; 2020-2024**	

Ho Chi Minh, January 2025

TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HCM	CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM
KHOA CƠ KHÍ CHẾ TẠO MÁY	***Độc lập - Tự do – Hạnh phúc***
BỘ MÔN CƠ ĐIỆN TỬ	

NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP

Học kỳ 1/ năm học 2024-2025

Giảng viên hướng dẫn: Ths. Đỗ Văn Hiến Mã số GV: 0214

Sinh viên thực hiện:

Nguyễn Quang Giao	MSSV: 19146121	Điện Thoại: 0345031248
Bùi Trương Vĩnh Phúc	MSSV: 20146518	Điện Thoại: 0845024312

1. Mã số đề tài: CTM-86

Tên đề tài: Thiết kế và chế tạo máy nhặt bóng pickle ball

2. Các số liệu, tài liệu ban đầu:

- Dựa trên máy nhặt bóng có sẵn để thiết kế lại thành Robot nhặt bóng.
- Dựa trên tài liệu có sẵn từ các nguồn tham khảo khác.
- Diện tích làm việc giới hạn: 40m^2.
- Thời gian hoạt động tối đa của robot: 45 phút.

3. Nội dung chính của đồ án:

- Thiết kế và gia công cơ khí Robot.
- Thiết kế và làm mạch điều khiển.
- Lập trình và điều khiển Robot thông qua thiết bị điều khiển.

4. Các sản phẩm dự kiến

- Robot nhặt bóng chạy thông qua thiết bị điều khiển trên ứng dụng.
- Robot nhặt bóng chạy tự động.

5. Ngày giao đồ án: 19/08/2024

6. Ngày nộp đồ án:

7. Ngôn ngữ trình bày: ***Bản báo cáo:*** ***Tiếng Anh*** ☐ ***Tiếng Việt*** ☐

Trình bày bảo vệ: ***Tiếng Anh*** ☐ ***Tiếng Việt*** ☐

TRƯỞNG KHOA	**TRƯỞNG BỘ MÔN**	**GIẢNG VIÊN HƯỚNG DẪN**
(Ký, ghi rõ họ tên)	*(Ký, ghi rõ họ tên)*	*(Ký, ghi rõ họ tên)*

☐ Được phép bảo vệ ………………………………………………………….

(GVHD ký, ghi rõ họ tên)

LỜI CAM KẾT

- Tên đề tài: Thiết kế và chế tạo máy nhặt bóng pickle ball.

- GVHD: TS. Đỗ Văn Hiến.

- Họ tên sinh viên 1: Nguyễn Quang Giao

- MSSV: 19146121 Lớp: 19146CLA1

- Địa chỉ sinh viên 1: Lâm Đồng

- Số điện thoại liên lạc: 0345031248

- Email: nguyenquanggiaoo@gmail.com

- Họ tên sinh viên 2: Bùi Trương Vĩnh Phúc

- MSSV: 20146518 Lớp: 201461A

- Địa chỉ sinh viên 2: Tiền Giang

- Số điện thoại liên lạc: 0845024312

- Email: phucbt1203@gmail.com

- Ngày nộp khóa luận tốt nghiệp:

- Lời cam kết: "*Tôi xin cam đoan Đồ Án Tốt Nghiệp (ĐATN) này là công trình do chính tôi nghiên cứu và thực hiện. Tôi không sao chép từ bất kỳ một bài viết nào đã được công bố mà không trích dẫn Source gốc. Nếu có bất kỳ một sự vi phạm nào, tôi xin chịu hoàn toàn trách nhiệm*".

Tp. Hồ Chí Minh, ngày … tháng … năm 2025

Ký tên

ACKNOWLEDGMENTS

First of all, we don't know what to say other than to express our deep gratitude to the teachers. Throughout the journey of studying and making graduation projects, we have always received the dedicated guidance and help of teachers.

We would like to express our sincere thanks to Mr. Do Van Hien - Lecturer at Ho Chi Minh City University of Technology and Education, who has wholeheartedly guided and guided us during the process of making the Project.

We would also like to sincerely thank the teachers in Ho Chi Minh City University of Technology and Education in general and in the Department of Mechatronics in particular for teaching us knowledge about general subjects as well as specialized subjects, helping us to have a solid theoretical basis and creating conditions to help us throughout the learning process.

Finally, we would like to sincerely thank our family and friends, who have always facilitated, cared, helped and encouraged us throughout the process of studying and completing our graduation Project.

With the limited time and experience of a student, this project cannot avoid shortcomings. We are very much looking forward to receiving guidance and comments from teachers so that we have the conditions to supplement and improve our awareness, better serve the practical work in the future.

Our group would like to sincerely thank you. Respect!

Ho Chi Minh, on ... month... in 2025

Team

Nguyễn Quang Giao

Bùi Trương Vĩnh Phúc

PROJECT SUMMARY

DESIGN AND MANUFACTURE OF PICKLE BALL PICKER

This topic focuses on the implementation and construction of the basic structure and control of the pickleball picker. Offering a mechanical structure that meets the needs of load and movement, the base circuit comes with a simple and easy-to-use control program. After consulting quite suitable options, mechanically, my team made the decision to use a plate robot body with lightweight meca material, in terms of control, we will choose to use the ROS operating system to create an environment for robot operation and control algorithms.

After the design, it will be the implementation and launch of the actual model product, the process of testing the operation of the robot. As a result, the robot runs stably on the terrain of the pickleball field. However, the limitation encountered is that the operating time is quite limited when the supply is a Cell Li-on battery. In terms of control, the ROS environment is successfully integrated into the controller to run the Coverage Path Planner algorithm, which gives a suitable moving trajectory for the robot to follow automatically. But when running in practice, it is still misleading and inaccurate due to hardware conditions, terrain, and interference,…

In the future, the solution will focus on improving the operating time. Along with that, we will further improve the control algorithm and the accuracy of the robot's automatic movement.

Performer

Nguyễn Quang Giao

Bùi Trương Vĩnh Phúc

TABLE OF CONTENTS

LIST OF IMAGES

Picture **Page**

LIST OF TABLES

LIST OF ABBREVIATIONS

UART:	Universal Asynchronous Receiver-Transmitter
PWM:	Pulse Width Modulation

CHAPTER 1: INTRODUCTION

1.1 Introduction to the topic

The research will focus on the design and manufacture of an automatic ball picker based on the working principle and general structure of some ball pickers currently on the market.

1.2 Reason for choosing the topic

Pickleball is a rapidly growing sport globally, leading to an increase in demand for supporting equipment, including pickleball pickers. However, specific information on the market share and market size of pickleball pickers is currently limited.

According to a report by Market Research Future, the global pickleball-related products market is estimated to reach $71.47 billion by 2024 and is expected to grow to $151 billion by 2032, at a CAGR of 9.8%. While the report does not break down individual product types, it can be inferred that assistive devices such as ball retrievers will contribute to this total value.

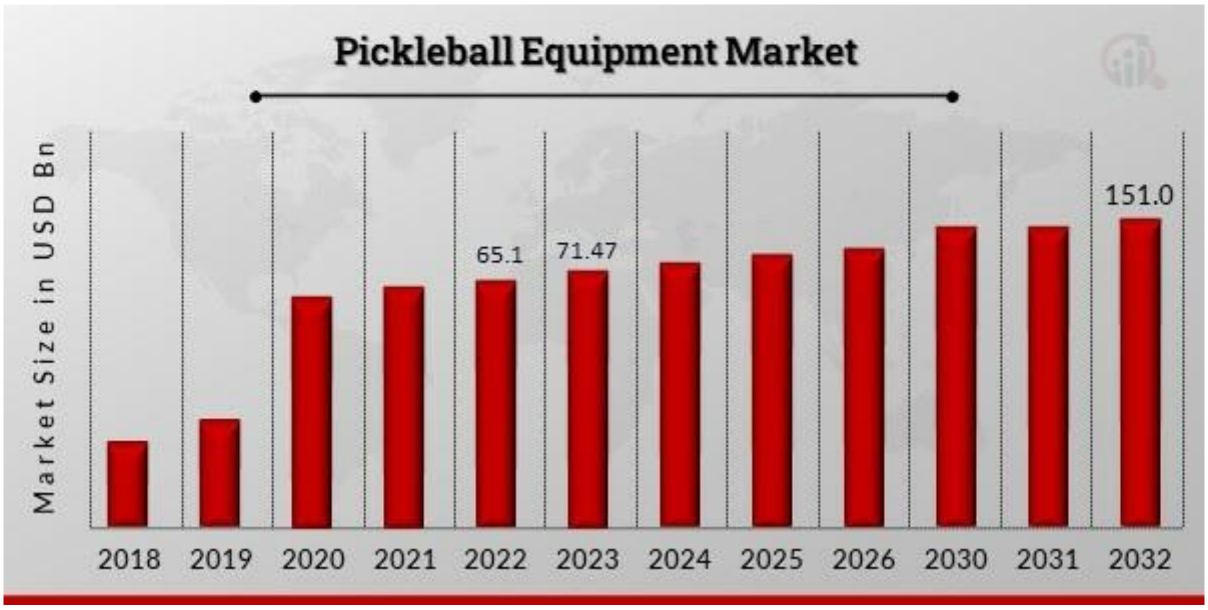


Figure 1.1: Pickleball Equipment Market

(Source: https://www.marketresearchfuture.com/reports/pickleball-equipment-market-12490)

In Vietnam, pickleball began to appear in 2018 and has quickly become popular. Many playgrounds and pickleball clubs have been established in Hanoi, Ho Chi Minh City and other provinces. The increase in the number of players has led to a high demand for supporting equipment, including ball pickers. However, there is currently no official data on the market share and market size of pickleball ball pickers in Vietnam.

The above is a result of the increasing technological advancements in artificial intelligence and robotics in the field of automation, which has led to the creation of more efficient devices and reduced human intervention. With the increase in the income levels of people living around the world, there is an increase in the sales of these devices which enhance the user experience.

Europe is expected to have a significant share in the global market and will grow at a rapid pace due to the region’s high demand for professional equipment and automated tools. Furthermore, the shift of customers away from manual ball picking and towards newer, efficient, intelligent and battery-powered automatic ball pickers is expanding Europe’s growth prospects.

In Vietnam, this is a fairly new sport and has many shortcomings compared to the West. Picking up the ball after hitting will take up about 35% of each player's practice time, pickleball players will become bored and tired after each time picking up the ball, leading to reduced efficiency. Therefore, having a robot in charge of picking up the ball on the field will help players feel more comfortable and practice more effectively, and at the same time have time to rest while the ball-picking robot works.

In fact, the current situation of using ball-picking robots to support training still faces many difficulties such as:

- High cost when importing robots from abroad.
- User interface and instructions are difficult to access for Vietnamese people.
- Warranty and repair issues are complicated and time-consuming.
- Difficulty with power sources between countries.
-…..

Because this is a new and very hot sport in Vietnam and to in line with the trend with the desire to create a pickleball picker machine with reasonable cost and user-friendly in the country. We decided to design and manufacture a “Pickleball Picker Robot”.

1.3 Objective of the topic

Benefits of Pickleball picker Robot:

- No time wasted picking up balls.

- Has specific control software and easy-to-understand viewing interface.
- Can collect balls quickly and smoothly.
- The robot operates independently without a human operator.
- The robot is easy to control.

1.4 Topic Limitation

- Based on the existing ball picker to redesign into a Ball Picker Robot.
- Based on material available from other reference sources.
- Limited working area: 40m^2.
- Operating conditions: Outdoors, no rain.
- Operates on flat terrain.
- Maximum robot operating time: 45 minutes.

1.5 Objectives and scope of research

1.5.1 Research objectives

- Development of intelligent control system using Robot Operating System (ROS) and microcontroller to control automatic ball picker.
- Optimal and efficient ball pick-up mechanism design.
- Building a test model and evaluating the performance of an automatic ball picker.

1.5.2 Scope of research

- The research scope includes control system design, sensor integration and software development to control the automatic ball picker.
- Research focused on the pickleball court area.
- Evaluate machine performance and reliability under various real-world conditions.

1.6 Research methods

1.6.1 Data collection method

- Literature Review: Study scientific literature, books, newspapers and other information sources related to automatic control systems, ROS, and automatic ball picker design methods.
- Field survey: Collect information about the types of automatic ball pickers currently sold on the market in practice, survey the structure and working methods of models from different manufacturers.

1.6.2 Design and development methods

- Control system design: Using Robot Operating System (ROS) to build a central control system for an automatic ball picker.
- Programming microcontroller and raspberry pi to process data from sensors and control motors.
- Mechanical design: Develop the ball picker mechanism and machine frame to optimize ball picker performance and ensure machine durability.

1.6.3 Testing and evaluation methods

- Model Testing: Build and test a model of an automated ball picker under different conditions to check the system's performance and accuracy.
- Performance Evaluation: Evaluate the performance of the automatic ball picker based on criteria such as ball picking efficiency, operating time and energy consumption.

1.6.4 Optimization method

- Control Algorithm Optimization: Research and apply optimization algorithms to improve control performance and save energy.
- Mechanical Design Optimization: Use computational and simulation methods to optimize the structure and design of the ball picker to ensure the highest performance and durability.

1.7 Graduation project structure

The project content includes the following:

Chapter 1: Overview.

The main content introduces the topic, working principle, general structure and some advantages and disadvantages of ball picking machines currently sold on the market. Then, the research method, objectives and scope of the research are presented.

Chapter 2: Theoretical basis.

Mention the basic theories that will be used in this topic, from mechanics, control systems, the boards that will be used, actuators, sensors and system control software.

Chapter 3: Design and construction.

Provide basic requirements for the topic of engineering, model design, and electrical circuits. Provide block diagrams and functions of each block in it. Propose solutions to the requirements, then choose a solution that is suitable for the accompanying conditions of the topic.

Chapter 4: Algorithms and Control.

Describe the process of mechanical processing, electronic circuit assembly and control software development.

Chapter 5: Experiment.

This is the part that will cover testing of each item built above from mechanical, electromagnetic circuit, control software. Finally, the experiment when integrating the whole system into a system to operate.

Chapter 6: Conclusion.

Summarize what the topic has achieved and what limitations it has encountered. Propose appropriate development directions for the topic.

CHAPTER 2: THEORETICAL BASIS

2.1 Common ways to pick up the ball

2.1.1 Pick up the ball by hand

Pickleballs are picked up by hand by the simplest and most common method. This is often seen in major matches, when players serve a foul or end a point, the picker will run to the ball and pick it up, then return it to the player to start the next shot. Pickleballs in a practice environment are also time-consuming and labor-intensive, but require less physical strength than picking up balls in matches.

2.1.2 Pick up the ball with a tool

Collectaball is a tool designed to optimize time and effort when picking up pickleballs. This tool can collect many balls in one roll without having to bend down like the usual way of picking up pickleballs.

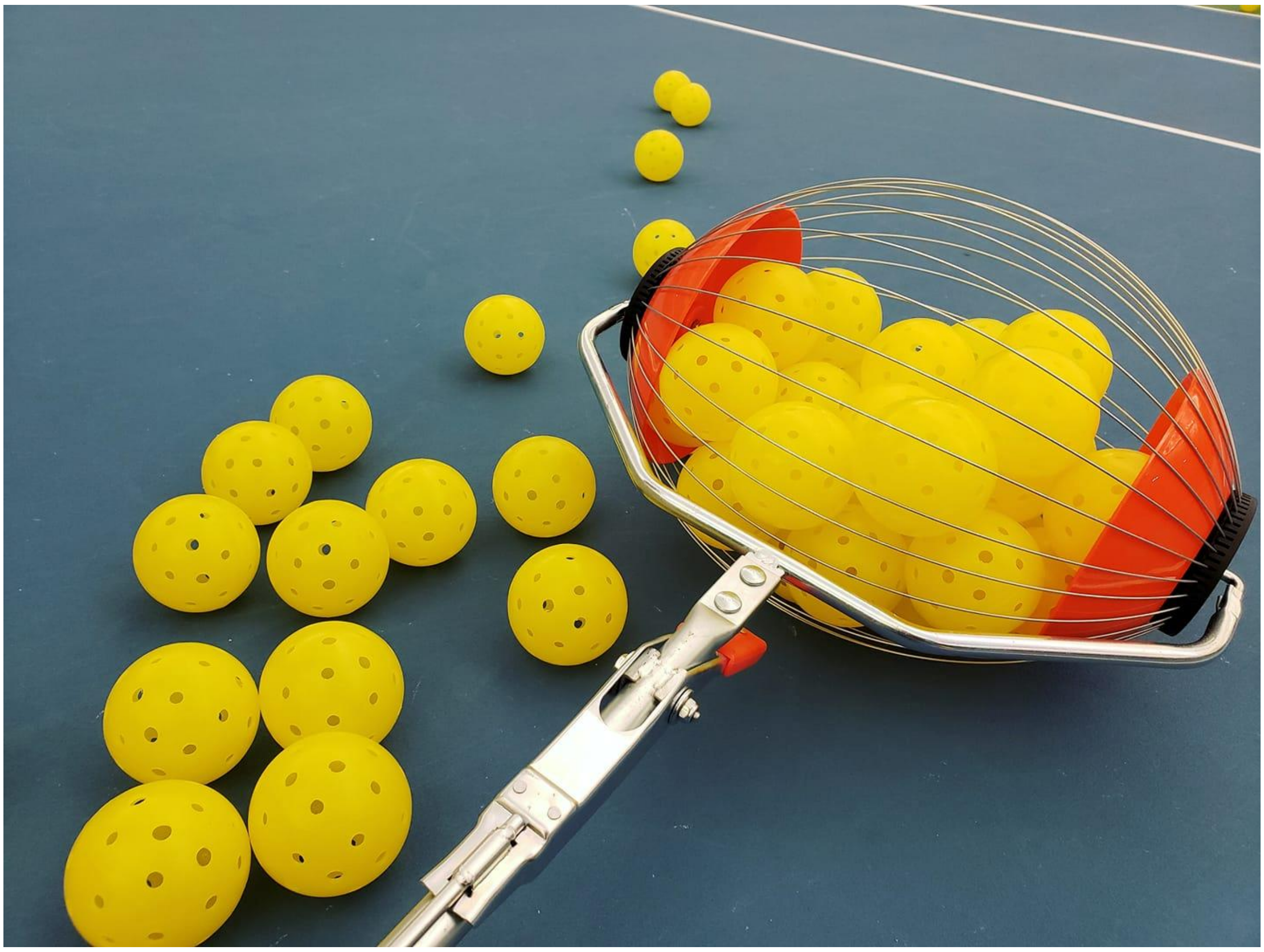


Figure 2.1: Pickleball Equipment Market

2.2 Ball Pickup Methods

2.2.1 Paddle methods

The paddle mechanism is a clever design that is used in many types of machines and devices, especially in automation systems. This mechanism works on the principle of using one or more paddles to gather spherical or elliptical objects into a concentrated position.

Operating principle:

Paddle blades: The paddle blades are specially designed with suitable shapes and angles to create suction and push forces on objects.

Motion: When the paddle blades rotate, they create a force that helps to pull objects inside.

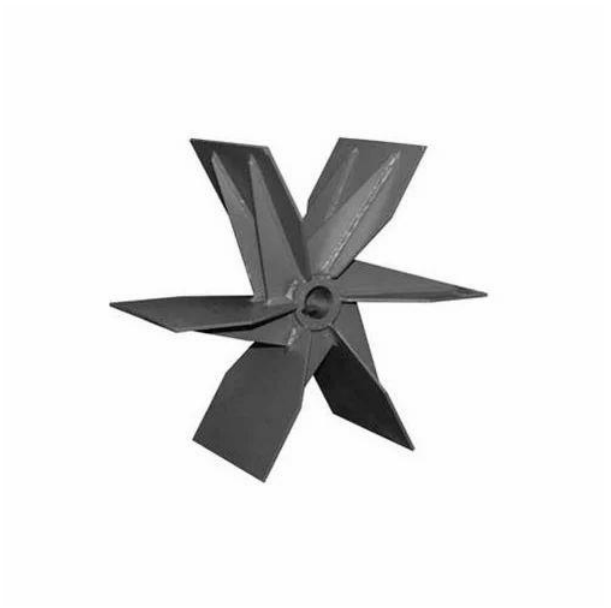


Figure 2.2 Paddle mechanism

Advantages

- High efficiency: The paddle wheel can collect a large number of balls in a short time, especially when the balls are distributed relatively evenly on the surface.
- Simple: The structure of the paddle wheel is relatively simple, easy to manufacture and maintain.
- Flexible: The paddle wheel can be designed in many different sizes and shapes to suit different types of balls and working spaces.
- Less damage: The paddle wheel causes less damage to the surface of the ball.

Disadvantages

- Difficulty with different sized balls: The paddle wheel is usually designed to collect a certain size of ball. Collecting different sized balls at the same time can be difficult.
- Ineffective with balls in hard-to-reach places: The paddle wheel often has difficulty reaching balls in corners, crevices or too high.
- Space-consuming: The paddle wheel requires a certain amount of space to operate, which may not be suitable for very small spaces.

2.2.2 Tomohopper

Tomohoppers typically use one or more large propellers to create a powerful stream of air, sucking the ball from a long distance into a large tube.

The Tomohopper picks up 90 balls quickly, smoothly, quietly and easily. When the balls are collected, the basket hangs on the handle for easy access and is ready to go home, and it all folds together into a compact unit that fits in the back seat of a car. If you are a professional athlete or simply someone who wants to get better and spends a lot of time practicing, the Tomohopper is a smart and easy-to-use design:

- Collects up to 90 balls quickly in just minutes.
- Non-marking wheels that run on all court surfaces.
- 42-inch wide collection arm for efficient collection at nets and fences. The arm locks and detaches easily for quick use and easy storage.

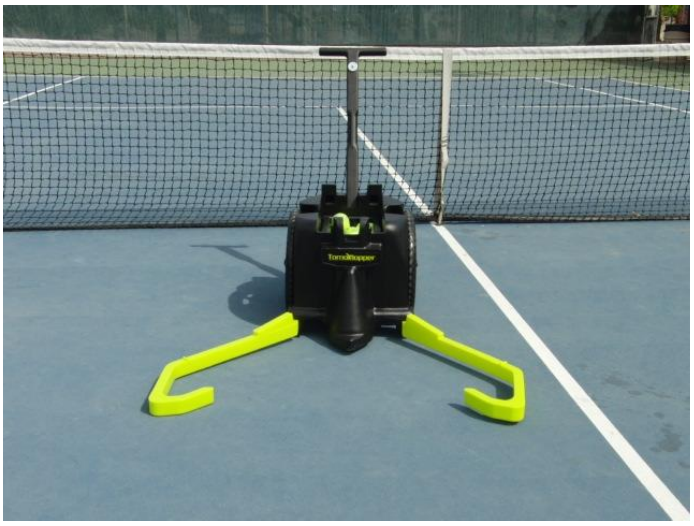


Figure 2.3 Tomohopper robot

2.2.3 Conveyor or small fans

Use a conveyor system or small fans to collect the balls. As the robot moves through the area with the balls, the fans will spin and suck the balls into a holding tank.

Advantages:

- Stable: Conveyors usually create a steady and steady motion, helping to transport balls smoothly and limit the possibility of them falling.
- Versatile handling: Conveyors can be designed to transport multiple balls at once, and can adjust the speed to suit the needs of use.
- The small fan can create strong suction to collect balls from many different directions, including balls in high or far positions.

Disadvantages:

- Less flexible: Conveyors often have fixed paths, making it difficult to adjust to accommodate complex terrain or unusually positioned balls.
- High power consumption: Small fans need to use a certain amount of electricity to create suction, which can affect the robot's operating time.
- Less stability: In some cases, small fans can cause balls to fly around or get stuck in the fan blades.
- Noise: Small fans often make a certain amount of noise when operating, which can be annoying to users.

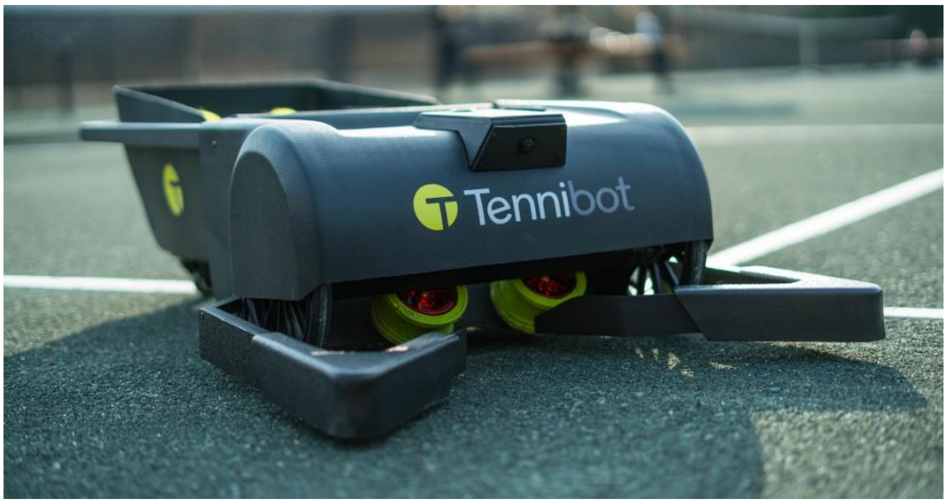


Figure 2.4 Tennibot use conveyor and small fans to picking ball

2.2.4 Select methods

Description of pickleball

Material: Pickleballs are made from polypropylene or polyethylene, which are lighter in weight and have lower elasticity than other types of balls such as tennis balls.

Design: Pickleballs have many small holes on the surface, which helps reduce air resistance and create softer hits.

Because the design of the pickleball contains small holes that help reduce air resistance then the method of using a fan to suck the ball is not reasonable.

After considering the methods, we decided to choose "**Paddle Methods"** to collect the ball.

-With strong rotation force, when the pickleball to touch the fan blade to easily collect the pickleball inside.

- The structure of the paddle wheel is relatively simple, easy to manufacture, maintain and can change the size of the fan blade depending on the size of the ball easily.

2.3 Introduction to image processing

2.3.1 What is image processing?

Image processing is the process of converting images into digital form and performing operations to extract useful information from that image. Image processing systems typically treat any image as a 2D signal by applying predefined methods.

Description of pickleball

Material: Pickleballs are made from polypropylene or polyethylene, which are lighter in weight and have lower elasticity than other types of balls such as tennis balls.

Design: Pickleballs have many small holes on the surface, which helps reduce air resistance and create softer hits.

Because the design of the pickleball contains small holes that help reduce air resistance, the method of using a fan to suck the ball is not reasonable

After considering the methods, we decided to choose method X to collect the ball

2.3.2 Basic steps of image processing of the system

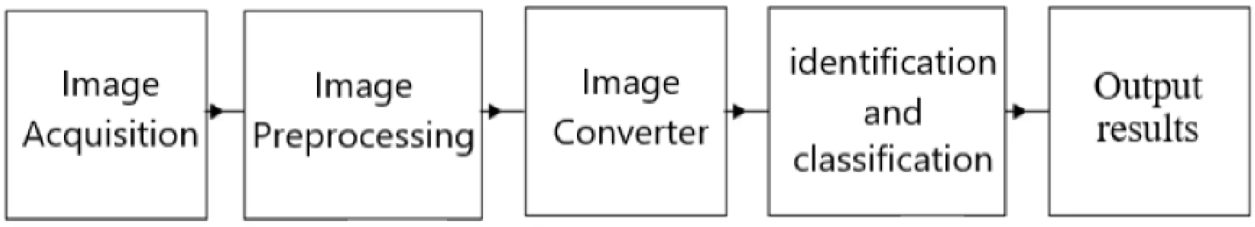


Figure 2.5: Basic steps of image processing

-Image acquisition : Get images extracted from the camera or memory to process the image.

-Image preprocessing This process aims to filter noise, increase brightness to make the image better quality for easy identification.

-Image converter In this process, we will convert the RGB color space to HSV screen space to suit the image processing process.

-Identification and classification: Each color and shape will have its own characteristics. In this process, we will identify the blue color and round shape of the pickleball, then identify which is the pickleball.

-Output results: After identifying the color and shape, the camera will output the image of the pickleball.

2.3.3 What is digital photography? Types of digital photography

Digital images are the numerical representation of images in a computer, usually expressed in binary form. Digital images can be divided into two main types:

-Raster images: Raster images are a type of digital image composed of a finite set of pixels (picture elements). Each pixel is a small square containing a numerical value representing properties such as color and brightness of a small element in the image. Raster images are mainly obtained from projectors, cameras... and are the subject of image processing.

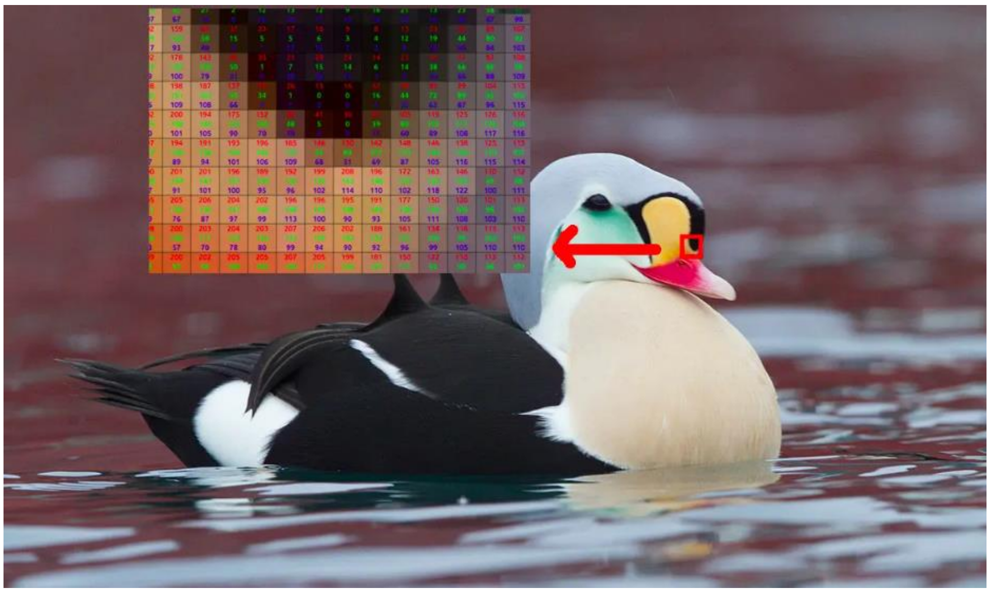


Figure 2.6: Raster image

-Vector images: Vector images are a type of digital image created using lines, points and geometric shapes instead of pixels. The main feature of vector images is that they are defined by mathematical formulas, they can be stretched, scaled, allowing resizing without losing image quality. Vector image data is small so it saves more storage space than raster images. Usually, vector images are used in logo design, banners... Almost no mention of image processing.

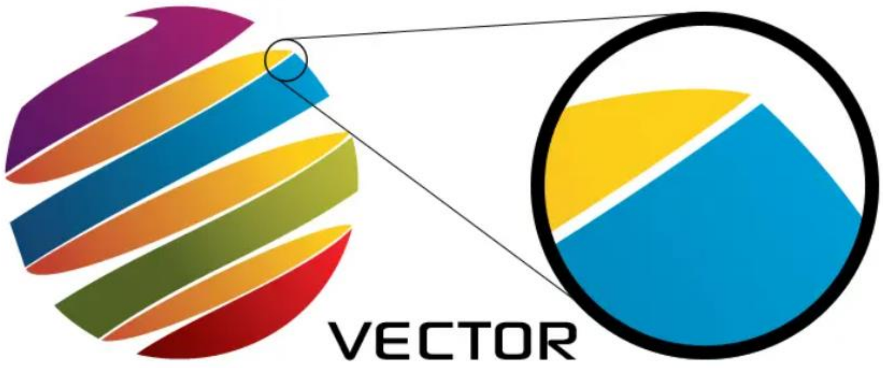


Figure 2.7: Vector image

2.3.4 Image Processing Techniques

Image preprocessing is an important step in image processing. In this step, digital images are transformed to achieve certain purposes. The relevant pixels in the image are retained and other irrelevant pixels are filtered out.

Image Filters

Gaussian Filter: is an important tool in image processing, widely used for image smoothing, noise reduction and edge blurring. It works on the mathematical principle of Gaussian distribution, a mathematical function that describes a bell curve.

Mean filter: is one of the simplest methods in image processing for image smoothing and noise reduction. It works by replacing the value of each pixel with the average value of neighboring pixels in a given neighborhood.

Disadvantage of Mean Filter

-Blurring edges: This is the biggest drawback of the mean filter. By averaging the values of neighboring pixels, the filter will blur the sharp edges of objects in the image. This reduces the sharpness and detail of the image.

-Loss of details: Besides blurring edges, the mean filter also loses fine details in the image. These details can be very important in some applications such as object recognition or medical image analysis.

-The Gaussian filter uses a weighted kernel, where pixels closer to the center have a larger weight, which helps preserve edges better.

Therefore, we choose the "Gaussian filter" for the model.

2.3.5 Edge detection

Edge detection is a technique in image processing that identifies points in an image that have sudden changes in light intensity. In other words, it helps to find the boundaries, the contours of objects in the image.

Canny:

Advantages:

-Effective noise reduction: Canny uses a Gaussian filter to smooth the image before calculating the gradient, which reduces noise and improves the accuracy of the results.

Weak edge removal: Canny uses two thresholds to remove weak edges and only retain strong edges, resulting in clearer results.

-Optimization: Canny is designed to find the most accurate edges, by combining noise reduction, gradient calculation, non-maximum suppression, and double thresholding steps.

Disadvantages:

More complex: Canny is more complex than Sobel, requiring more computational steps.

Sobel:

Advantages: Simple, easy to implement and fast to compute.

Disadvantages:

-Sensitive to noise: Sobel is very sensitive to noise in the image, easily creating fake edges.

No weak edge removal mechanism: Sobel does not have a mechanism to remove weak or unimportant edges, leading to noisy edge detection results.

-No optimization: Sobel is not optimized to find the most accurate edges, but simply calculates the gradient of the image.

Canny edge detection is superior to other edge detection techniques in identifying spurious edges. There is a risk that if the system detects spurious edges, it may fire the robot to the wrong location where the desired objects are not present.

Canny edge detection has good noise immunity and detects real edge points with minimal error.

So that we choose “Canny Edge” detection.

2.3.6 Object Recognition

Hough Transform: Finding Simple Shapes

The Hough transform is a mathematical technique used to find simple shapes such as lines and circles in an image. Instead of directly searching for points belonging to a shape, the Hough transform converts the image space into a parametric space, where each point represents a specific shape.

Advantages:

Ability to detect simple shapes even when they are noisy or partially occluded.

Less dependent on the geometric features of the object.

Disadvantages:

Intensive in computing resources, especially with high-resolution images.

Difficult to apply to complex shapes.

Color Segmentation: Separating Color Regions

Color segmentation is the process of dividing an image into regions of uniform color. Each of these regions represents an object or part of an object in the image.

Advantages:

Simple, easy to implement.

Effective in separating objects with distinct color differences.

Disadvantages: Sensitive to light, shadow, and other lighting effects.

Difficult to segment objects with similar colors or color gradients.

Because while the car is moving, the camera will go through many different light areas which can increase the noise of an image.

We choose “Color Segmentation” so that the camera can recognize the pickleball ball in the best way.

2.3.7 Locate the object

The location of the object will be determined by the centroids. The centroid is the center point of a group of white pixels. The values are in x and y coordinates. There will be a point marked in the image representing the center of the ball.

Once the centroids are found, the location of the pickleballs can be determined by performing mathematical calculations. The x and y values represent the locations corresponding to the regions where the pickleballs exist.

2.4 Raspberry Pi 4 Model B 8 Gb

Raspberry Pi is a small-sized computer (SBC - Single Board Computer) with the size of a credit card, but has all the functions of a desktop computer. Developed by the Raspberry Pi Foundation, Raspberry Pi is designed for educational purposes, making programming and electronics accessible to everyone. The main purpose of Raspberry Pi is to provide a versatile and powerful platform for developing applications from simple to complex. Raspberry Pi is capable of running operating systems such as Linux, allowing users to install and use many different types of software, from web browsers, word processors, to programming and software development tools. In addition, Raspberry Pi also supports network connections, HDMI, USB and many other types of connection ports, allowing it to connect to many different peripherals and systems. Raspberry Pi is widely used in many fields. In education, it is a useful tool for teaching programming and electronic engineering. In DIY (Do It Yourself) projects, Raspberry Pi is used to build smart home systems, security monitoring systems, and IoT (Internet of Things) devices. In industry, Raspberry Pi can be used to control and monitor automated manufacturing processes. In addition, Raspberry Pi is also used in scientific research, as a personal server, and even in digital art projects. Thanks to its versatility and low cost, Raspberry Pi has become a popular tool for both beginners and experts in many fields of technology.

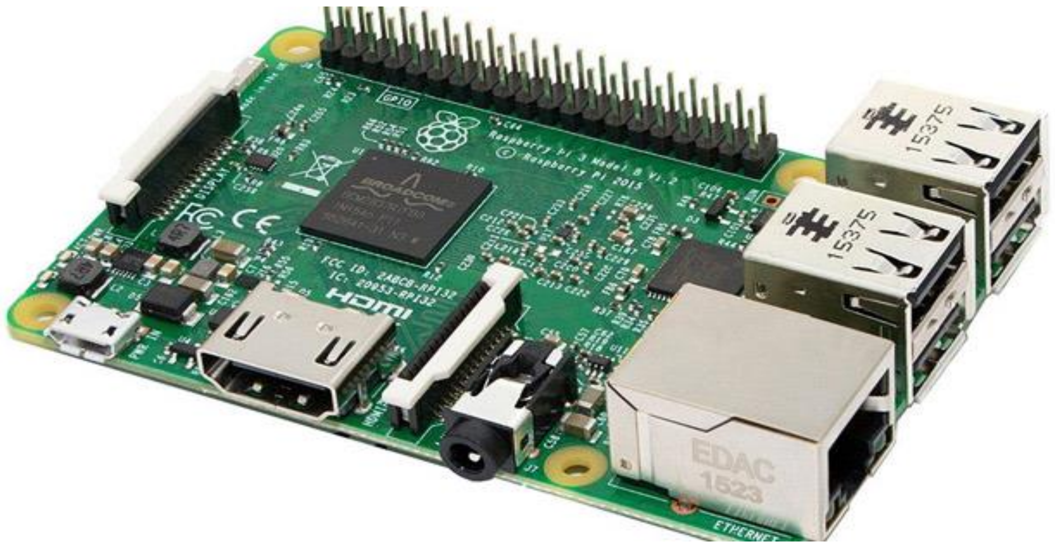
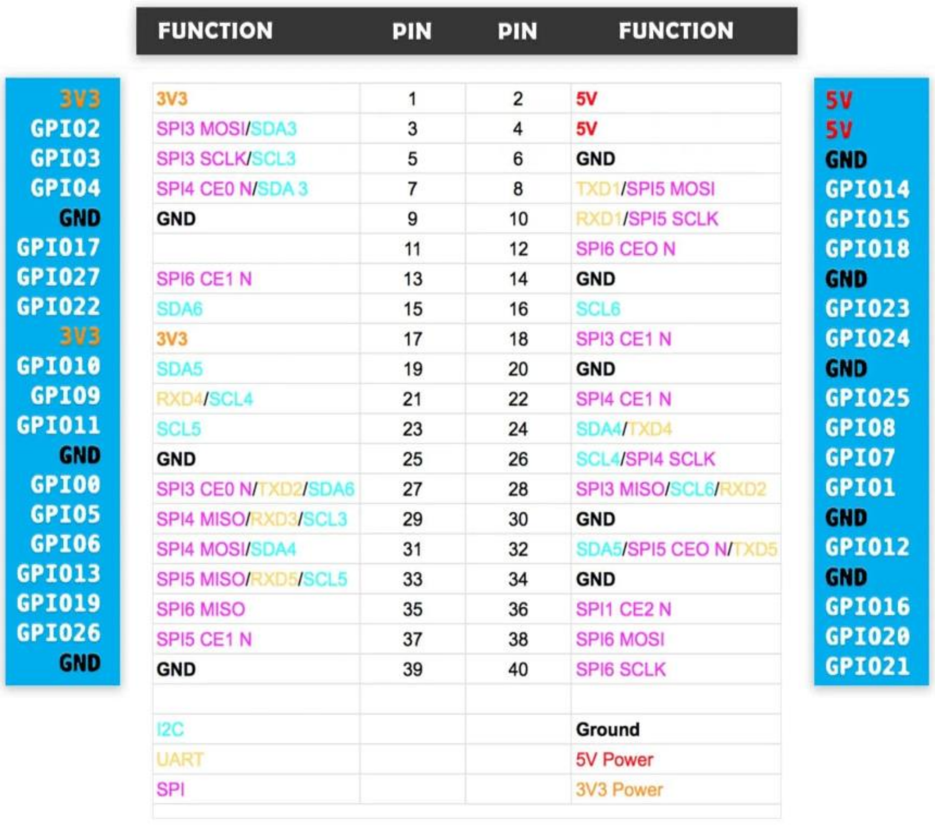


Figure 2.8: Raspberry Pi 4B+

(Source: Internet)

Raspberry Pi pin configuration



	FUNCTION	PIN	PIN	FUNCTION	
3V3	3V3	1	2	5V	5V
GPIO2	SPI3 MOSI/SDA3	3	4	5V	5V
GPIO3	SPI3 SCLK/SCL3	5	6	GND	GND
GPIO4	SPI4 CE0 N/SDA 3	7	8	TXD1/SPI5 MOSI	GPIO14
GND	GND	9	10	RXD1/SPI5 SCLK	GPIO15
GPIO17		11	12	SPI6 CEO N	GPIO18
GPIO27	SPI6 CE1 N	13	14	GND	GND
GPIO22	SDA6	15	16	SCL6	GPIO23
3V3	3V3	17	18	SPI3 CE1 N	GPIO24
GPIO10	SDA5	19	20	GND	GND
GPIO9	RXD4/SCL4	21	22	SPI4 CE1 N	GPIO25
GPIO11	SCL5	23	24	SDA4/TXD4	GPIO8
GND	GND	25	26	SCL4/SPI4 SCLK	GPIO7
GPIO0	SPI3 CE0 N/TXD2/SDA6	27	28	SPI3 MISO/SCL6/RXD2	GPIO1
GPIO5	SPI4 MISO/RXD3/SCL3	29	30	GND	GND
GPIO6	SPI4 MOSI/SDA4	31	32	SDA5/SPI5 CEO N/TXD5	GPIO12
GPIO13	SPI5 MISO/RXD5/SCL5	33	34	GND	GND
GPIO19	SPI6 MISO	35	36	SPI1 CE2 N	GPIO16
GPIO26	SPI5 CE1 N	37	38	SPI6 MOSI	GPIO20
GND	GND	39	40	SPI6 SCLK	GPIO21
	I2C			Ground	
	UART			5V Power	
	SPI			3V3 Power	

Figure 2.9: Raspberry Pi pin configuration

Raspberry Pi4 can be used in external embedded systems for signal communication. Raspberry Pi4 has a total of 40 pins, of which 28 are GPIO pins and the remaining pins are power pins. GPIO pins not only perform simple I/O functions but also support protocols such as UART, SPI and I2C….

Applications of Raspberry Pi

-Used in home automation systems.

-Used in study, research, student projects.

-In the fields of robotics and embedded systems.

2.5 Arduino Uno R3

Arduino Uno R3 is one of the most popular boards in the Arduino product line, widely used in electronics projects and embedded programming. You can use Arduino Uno R3 widely in projects such as: Autonomous robots, remote control systems, measuring and monitoring devices, home automation systems….

The Arduino board uses Atmel's 8-bit AVR microprocessor line, in which the two most popular chips are ATmega328 and ATmega2560. These microprocessors allow programming of complex control applications thanks to their powerful configuration with ROM, RAM and Flash memory, digital I/O input/output ports (many of which are capable of outputting PWM signals), analog signal reading ports, and support for diverse communication standards such as UART, SPI and TWI (I2C).

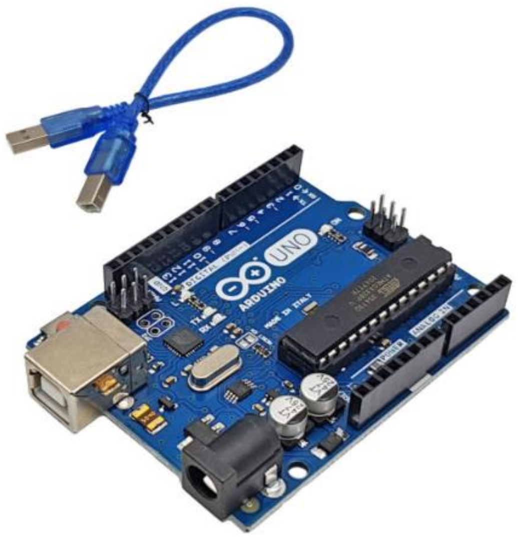


Figure 2.10: Arduino Uno R3

Datasheet:

Table 2.1: Datasheet of Arduino Uno R3

Microcontroller	ATmega328 8bit
Operating voltage	5~12V DC (recommended)
Operating frequency	16 MHz
Consumption stream	About 30mA
Input voltage limit	19V DC
Number of Digital I/O pins	14 (6 chân PWM)
Number of Analog Pins	6 (10bit resolution)
Maximum current per I/O pin	30 mA
Maximum output current (5V)	500 mA
Maximum output current (3.3V)	50 mA
Flash memory	32 KB (ATmega328) with 0.5KB used by bootloader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Mass	25gram

2.6 Motor JGA25 – 370 130RPM

The JGA25 motor is a 12V DC geared motor, made of metal with high durability and stability.

-Robot: JGA25 motor is widely used in projects such as autonomous robots, remote control robots, free line tracing robots with the ability to provide high torque and precise speed control.

-Smart home devices: JGA25 motor can be used in smart home devices such as automatic curtains, electronic door locks, and some other control devices…

-Automation equipment: In automation line systems, the JGA25 motor can be used to control actuators, conveyors or other moving equipment…

With such a wide range of applications, the JGA25 is a popular choice in engineering projects and technology applications.

Figure 2.11: Motor JGA25 130RPM

Datasheet:

-Supply voltage: 6-18VDC

-Speed after gearbox: 130rpm

-Torque: 5 kg.cm = 0.49 N.m

-No-load current: 50mA

2.7 Motor control circuit L298N

This is a device used to control motors, such as DC (Direct Current) motors and stepper motors. This module controls the rotation direction, speed and position of the motor according to the requirements of the specific application.

The L298N DC motor control module can control two DC motors, the maximum current is 2A per motor, this module includes a protection diode and a 7805 power IC to supply 5VDC to other modules (use this 5V only if the power supply is <12VDC). This circuit is simple and easy to use, has a low price and is the best choice in the price range.

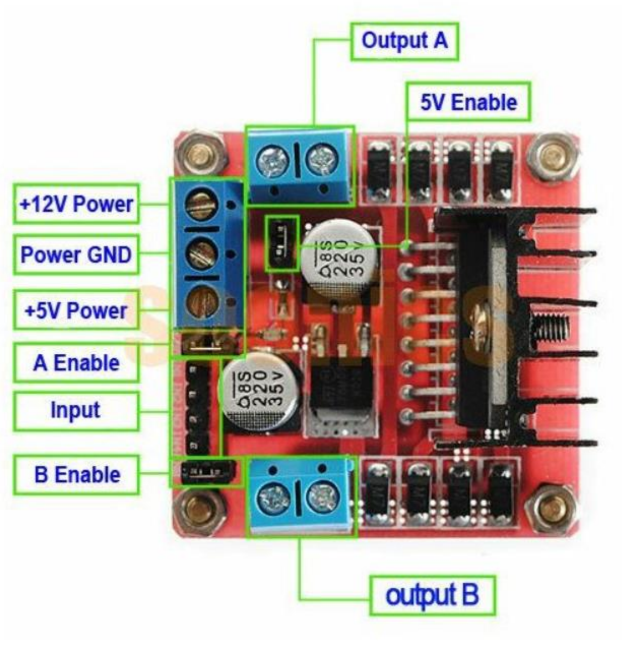


Figure 2.12: Motor control circuit L298N

Datasheet

- Main IC is L298 – Dual Full Bridge Driver.
- Input voltage from 5 to 30VDC.
- Maximum power is 25W for each bridge (note that power is calculated by current multiplied by voltage, so when the input voltage is higher, the current is smaller, the power remains the same at 25W).
- Maximum current for each H-bridge is 2A.
- Logic voltage levels: Low from -0.3V to 1.5V, High from 2.3V to Vss (ground).
- The size of the circuit is 43x43x27mm.

2.8 UART protocol (Universal Asynchronous Receiver-Transmitter)

Basic features:

UART, or Universal Asynchronous Receiver Transmitter, is a simple method of digital communication between devices. You can find UARTs as integrated circuits (ICs) or as individual components. UART communication takes place between two separate devices via a pair of wires and a common ground.

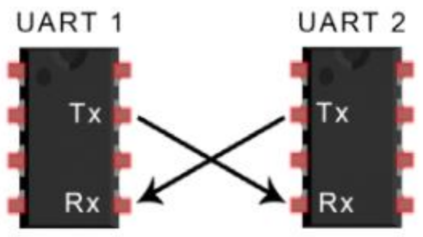


Figure 2.13: UART protocol

UART is an asynchronous protocol, so it does not have a clock line to regulate the data transfer rate. Therefore, the user must configure both devices to communicate at the same speed, called the baud rate, usually measured in bits per second (bps). This baud rate can vary from 9600 baud to 115200 , and even higher. The difference between the transmit and receive rates should only fluctuate around 10% to avoid excessive bit timing skew.

Advantage:

-Simple, easy to deploy

-Many research papers are widely used

-Only 2 wires are used

-No clock signal is needed

-Compatible with many devices

-Low power consumption

Disadvantages

-The size of the data frame is only 9bit maximum

-Multiple slave or master systems are not supported

-The baud rate of each UART is limited to within 10% of each other

-The baud rate is slower than other protocols

-UART is susceptible to noise

Popular Applications:

-Communication between microcontrollers and peripherals.

-Remote control and smart home appliances: UARTs are used in remote control systems, such as controlling lights, fans, and other smart home appliances.

-Medical devices: UARTs are used in medical devices such as blood pressure monitors, heart rate monitors to transmit medical data from the device to a storage or display system.

-Embedded systems and IoT: In Internet of Things (IoT) applications, UARTs are used to connect sensors, communicators, and control devices together.

2.9 Python language and OpenCV library

Python is a general-purpose programming language, widely used in fields such as web development, software development, data science, and machine learning (ML). Developers love Python because of its efficiency, ease of learning, and ability to run on many different platforms. Python is open source software, freely downloadable, and integrates well with different systems, helping to maximize product development speed.

Figure 2.14: Programming Language Python

Python is one of the popular languages used in image processing, thanks to powerful libraries such as OpenCV (Open Source Computer Vision Library). OpenCV is an open source library that provides tools and algorithms for efficient image processing and computer vision.

Through Python and OpenCV, programmers can perform Missions such as:

-Image Reading and Display: Python easily reads image files and displays them on the screen.

-Basic Processing: Smooth images, adjust brightness, contrast, or perform simple transformations.

-Segmentation and Object Recognition: Use image segmentation or object recognition algorithms to detect and identify objects in images.

-Real-Time Processing: Apply image processing and computer vision to real-time applications such as face recognition, object tracking, or motion detection.

-Image Data Analysis: Use Python to analyze data from images, such as calculating geometric features or statistics from image datasets.

Figure 2.15: Opencv Library

2.10 Visual Studio Code Software

Visual Studio Code (VS Code) is a free, open-source, cross-platform source code editor developed by Microsoft. It is designed to assist programmers in writing, debugging, and managing the source code of software applications. Here are some of the key features of Visual Studio Code:

-Cross-Platform: Visual Studio Code can run on Windows, macOS, and Linux.

-Supports features like tabs, split views, and terminal integration.

-Multi-Language Support: Supports multiple programming languages like JavaScript, Python, Java, C++, C#, PHP, Go, and many more through extension).

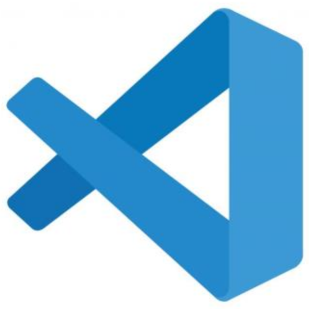


Figure 2.16: Visual Studio Code Software

2.11 Arduino IDE Software

Arduino IDE (Integrated Development Environment) is the official software for programming Arduino boards. It is the essential tool for writing, compiling, and uploading source code to the Arduino board. It is available for operating systems such as macOS, Windows, and Linux, and runs on the Java platform. It provides important built-in functions and commands to support debugging, editing, and compiling code in the development environment.

The IDE environment mainly consists of two basic parts: Editor and Compiler. The Editor is used to write source code, while the Compiler is used to compile and upload the code to the Arduino module. The IDE supports programming in C and C++ languages.

Figure 2.17: Arduino IDE Software

2.12 Color spaces representing images

RGB color space

The RGB color space is based on the principle of color addition, in which three primary colors (red, green, blue) are combined in different proportions to create many different colors. RGB is commonly used in digital devices such as cameras, computers, etc. Usually in the 24bit model, each color channel will use 8bit, so the value of R, G, B will be in the range of 0-255. With the 24bit color model, the maximum possible color is 16581375.

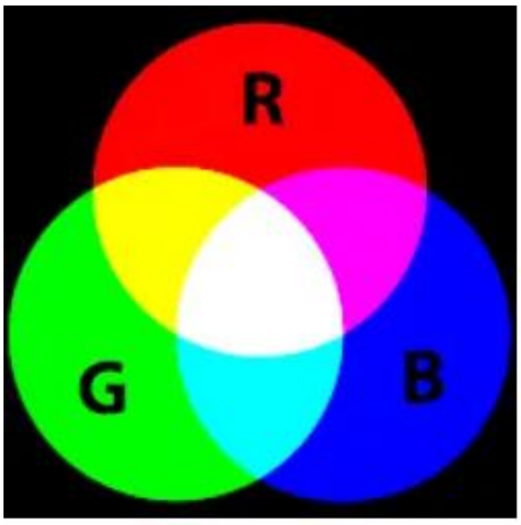


Figure 2.18: RGB color space

CMYK color space

The CMYK color space is a color model used primarily in the printing industry. CMYK stands for four primary colors: Cyan, Magenta, Yellow, and Key.
The CMYK color space is widely used in the printing industry, including printing books, newspapers, magazines, and advertising materials. Printers use CMYK inks to produce high-quality prints.

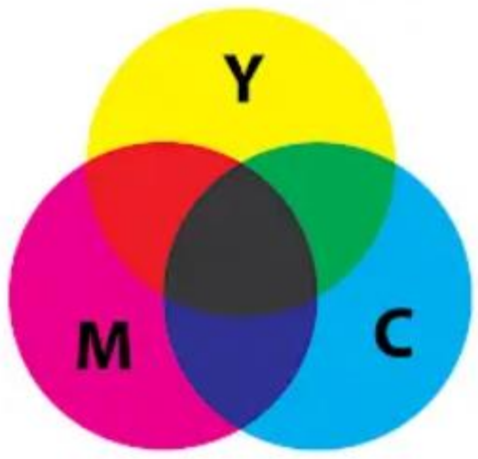


Figure 2.19: CMYK color space

HSV color space

The HSV (Hue, Saturation, Value) color space is a colorimetric system used to describe colors in a way that closely approximates the way humans typically perceive and distinguish colors. HSV is one of the most popular color spaces used in image processing and computer graphics. Here is a detailed explanation of the components of the HSV color space:

-Hue (H):

- Description: Hue represents the color range of a pixel.
- Value Range: In the HSV color space, Hue is typically represented from 0 to 179.

-Saturation (S)

- Description: Saturation measures the purity of a color. It reflects the saturation of the color, with high values representing pure colors and low values representing pale or gray colors.
- Value Range: Saturation typically ranges from 0-255.

-Value (V)

- Description: Value measures the brightness of a color. It indicates the brightness of the color, with high values representing bright colors and low values representing dark colors.
- Value Range: Value typically ranges from 0-255.

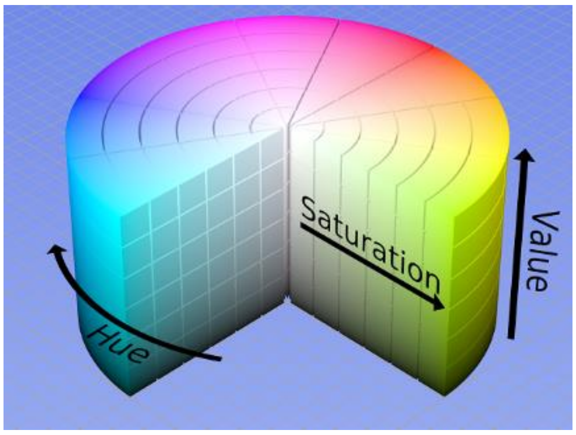


Figure 2.20: HSV color space

HSV color filters are often preferred over other color filters in many image processing applications, especially in tasks related to color segmentation, object recognition and image editing with advantages such as:

Less affected by light

Object tracking: Use HSV color filters to separate objects from the background and track their movements.

So we choose HSV color filters

2.13 Blynk

Blynk is a platform with mobile applications that makes it easy to connect and control microcontrollers such as Arduino, Esp8266, Esp32 or Raspberry over the Internet.

The Blynk App is a digital dashboard that helps you create a graphical interface for your project by dragging and dropping pre-designed widgets from the vendor.

Blynk is not limited to a specific board or shield. Instead, it is compatible with any hardware you choose. Whether your Arduino or Raspberry Pi connects to the Internet via Wi-Fi, Ethernet, or the ESP8266 chip, Blynk will help you connect and prepare for IoT projects.

Figure 2.21: Blynk

2.14 Introduction to IMU (Inertial Measurement Unit) sensor

Inertial Measurement Unit (IMU) is an electronic device used to measure and report specific forces acting on a body, angular velocity, and sometimes orientation of the body, using a combination of accelerometers, accelerometers, and sometimes magnetometers. When a magnetometer is present, the IMU is called an IMMU.

IMUs are commonly used to control modern vehicles such as motorcycles, missiles, aircraft (positioning and navigation systems), including unmanned aerial vehicles (UAVs), and space equipment such as satellites and landing gear. Recent advances have allowed the production of GPS devices that incorporate an IMU. An IMU allows a GPS receiver to operate when GPS signals are not available, such as in tunnels, inside buildings, or in the presence of electromagnetic interference. IMUs are important tools in automatic control systems, motion monitoring, navigation assistance, and other high-tech applications, providing flexibility and high accuracy in tracking and controlling vehicles and mobile devices in various environments.

Figure 2.22: Unit of inertia in the Apollo spacecraft *(source: wikipedia)*

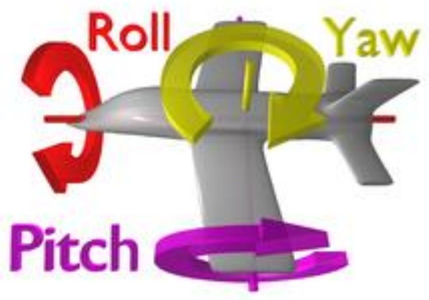


Figure 2.23: IMU activity tracks changes in roll, yaw, pitch *(source: wikipedia)*

In smaller scope of use, we will often see modules and circuits with integrated IMU circuits such as MPU6050, MPU9250, BNO055, ADIS16470, LSM9DS1. And among them will belong to one of the following 3 types of IMU:

IMU integrates 2 sensors:

This type of IMU consists of an accelerometer and a gyroscope. Typically, each sensor has two to three degrees of freedom defined for the x, y, and z axes, with a total of four to six degrees of freedom when combined. The acceleration values collected from the accelerometer and the angular velocity from the gyroscope are kept separate. The rotation angle can be measured from both sensors, so both data can be calibrated as shown in Figure 1 to obtain more accurate output data. These modules are widely used in applications such as positioning and navigation systems, robot and autonomous vehicle control, motion monitoring, and IoT (Internet of Things) applications. They are often connected to microcontrollers to process the data and control the systems based on the information from the IMU.

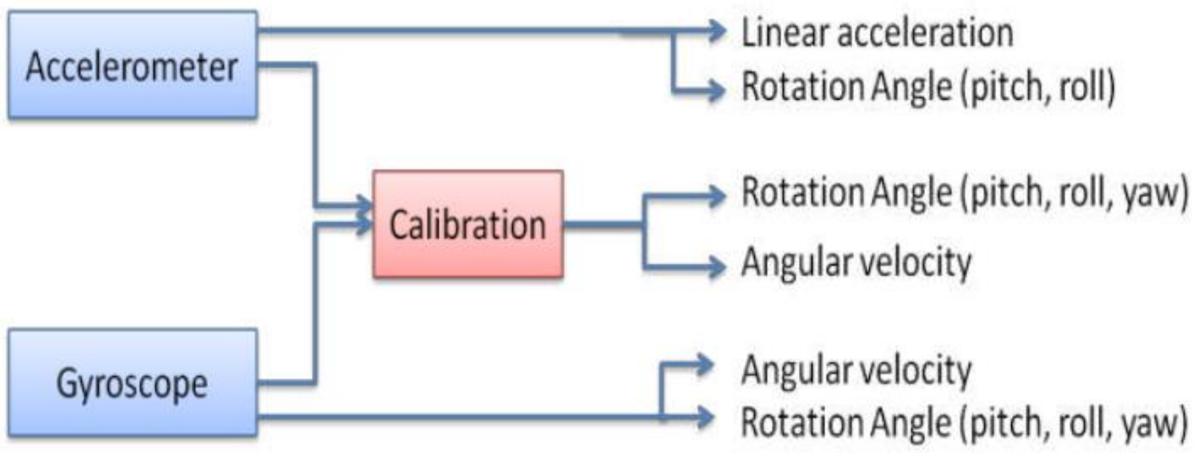

Accelerometer
Linear acceleration
Rotation Angle (pitch, roll)
Calibration
Rotation Angle (pitch, roll, yaw)
Angular velocity
Gyroscope
Angular velocity
Rotation Angle (pitch, roll, yaw)

Figure 2.24: IMU type based on 2 sensors

(Source: Reviews on Various Inertial Measurement Unit (IMU) Sensor Applications (by Norhafizan Ahmad, Raja Ariffin Raja Ghazilla, and Nazirah M. Khairi), 2013.)

IMU integrates 3 sensors:

This type of IMU includes an accelerometer, gyroscope, and magnetic sensor - usually all three axes to measure in three different axes, for a total of 9 degrees of freedom. The magnetic sensor is used to measure the yaw angle, so it can be calibrated with data from the gyroscope to improve the problem of large drift. This type of sensor is suitable for calculating both short- and long-term direction of motion where there is little drift error. However, there is a drawback to having a magnetic sensor in the package. If the IMU is used in an environment surrounded by ferromagnetic metals, the measurements may be affected by interference from the magnetic field.

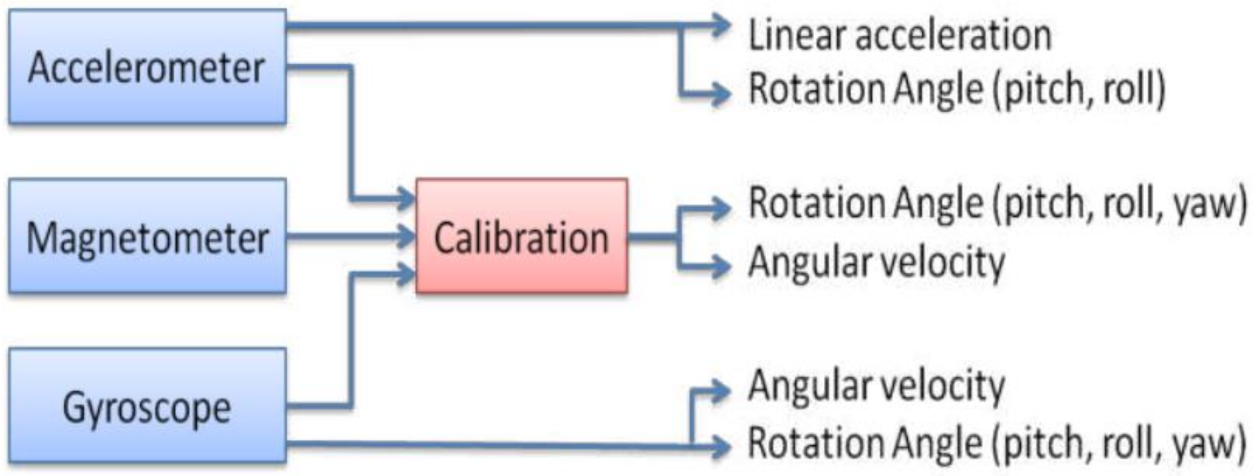


Figure 2.25: IMU type based on 3 sensors

(Source: Reviews on Various Inertial Measurement Unit (IMU) Sensor Applications (by Norhafizan Ahmad, Raja Ariffin Raja Ghazilla, and Nazirah M. Khairi), 2013.)

When choosing which type of sensor is suitable for the job being used, rely on the following factors to decide and consider.

Size: Many consumer products need to be designed with compact and lightweight sensors to fit the product and ensure good portability. For example, a smartphone used in consumer products as well as a smart baseball used to measure the trajectory of the ball both prefer to use small-sized IMU sensors that can be embedded in small spaces. Compared to aircraft applications, the size of the sensor may not be a major issue. The most widely used type of IMU in the consumer industry is the MEMS (Micro Electro Mechanical Systems) sensor.

Accuracy: Gyroscopes are known to have drift errors over long periods of use, while accelerometers are sensitive to acceleration in any direction when the object is accelerating rapidly or rotating. Improvements have been made such as the use of Kalman filtering to calibrate data from accelerometers and gyroscopes. Some applications only require a specific measurement range and can accept accuracy within a certain threshold. Therefore, fine-grained filtering is not a major concern here.

Response speed: A good sensor should have a fast response rate. For example, a sensor with a sampling rate of 50 Hz should be able to perform 50 measurements per second. Most devices that involve motion applications require a higher response rate and need an initial validation before proceeding with the application. Typical current sampling rates range from 1 Hz to 50 Hz. Some advanced applications such as vehicle guidance systems may require higher response rates of up to 200 Hz to ensure that data is collected quickly and accurately in a timely manner.

Degrees of freedom: The number of degrees of freedom (DOF) determines the number of independent parameters in a system. There are many IMU (Inertial Measurement Unit) sensors available on the market, with the number of degrees of freedom ranging from 2 to 9 DOF. The difference in the number of DOFs depends on the type of sensor included in the IMU and the number of axes the sensor will measure. Depending on the intended use and the features required by the application, the total number of DOFs can vary. In position tracking applications, most applications use a dual-sensor IMU with 6 DOF or a triple-sensor IMU with 9 DOF, with the number of DOFs representing the measurements in the x, y, and z axes for each sensor. Typically, the higher the number of DOFs, the more accurately we can sample the data.

2.15 Introduction to ROS (Robot Operating System)

ROS (Robot Operating System) is a special operating system developed to support and optimize robot control. It is not only a pure operating system, but also a powerful framework for robot software development. ROS is built on the open source community, creating a flexible platform for researchers, developers and students of robotics.

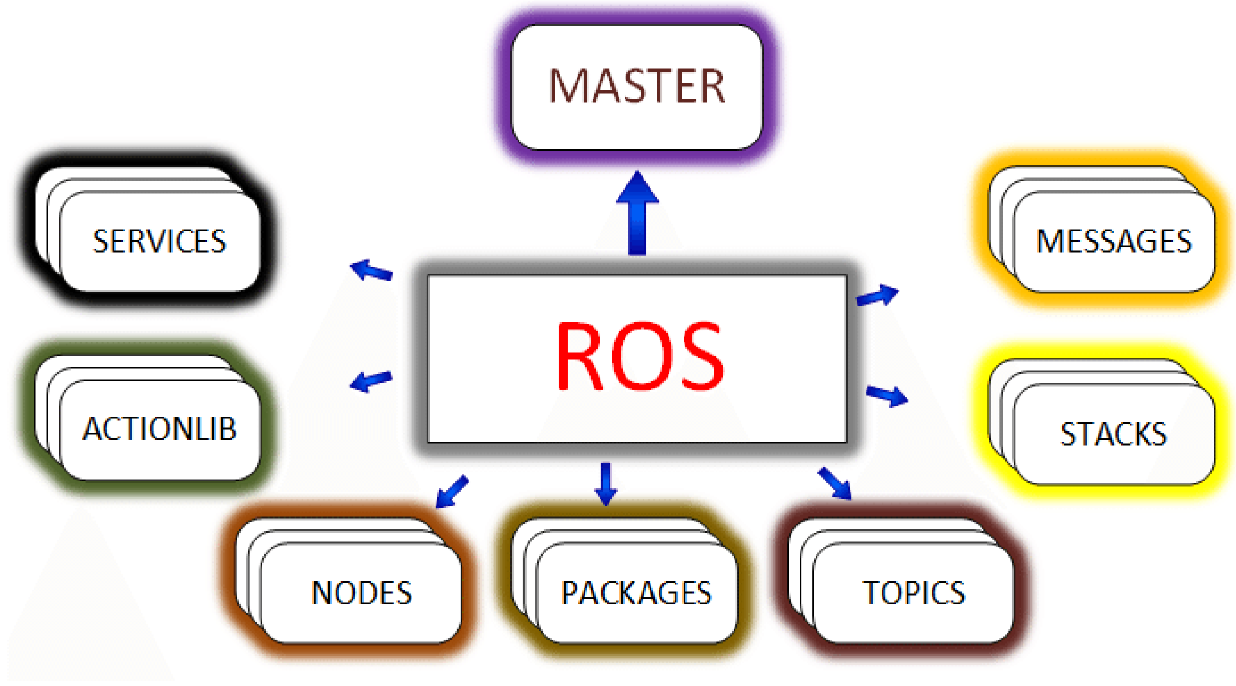


Figure 2.26: ROS framework

(Source: ROS-Based Approach for robot as a service in cloud computing (by Labib Sadek Terrissa, Radhia Bouziane, Soheyb Ayad, and Jean-Francois Brethe), 2016.)

ROS (Robot Operating System) is a special operating system developed to support and optimize robot control. It is not only a pure operating system, but also a powerful framework for robot software development. ROS is built on the open source community, creating a flexible platform for researchers, developers, and students of robotics.

ROS provides a wide range of functions and services, including device management, communication between software and hardware, sensor data management, and many other features that make robot application development convenient and efficient. With its powerful infrastructure, ROS allows users to focus on designing and deploying complex robot systems easily, enhancing the power and flexibility of robot projects in many different fields.

Using ROS (Robot Operating System) brings many benefits to the development and management of robot systems. Below are the benefits it brings:

1. Easy Integration: ROS provides a powerful framework for robot software development, making it easy to integrate different robot components and functions.
2. Open Source Community: ROS is built on a large open source community. This means that users can leverage the power and diversity of the community, as well as share and reuse open source code.

3. Communication Standardization: ROS provides communication standards between robot software and hardware, allowing flexibility in connecting and interacting between different components.
4. Device Management: ROS supports the management of robot devices and sensors, making it easy to add new ones and configure them flexibly.
5. Rapid Development: For robot developers, ROS provides powerful tools and libraries to reduce the time and effort required for development. Multi-Platform Support: ROS can run on a variety of platforms, including Linux, macOS, and Windows, providing flexibility in deployment across different systems.
6. Runs on a Variety of Robots: ROS is not limited to the type of robot, from mobile robots to industrial and service robots. This makes ROS a popular choice in the robotics and research community.

ROS operates on a distributed architecture, in which software components are called nodes. Each node is an independent process responsible for performing a specific function, such as controlling a motor, collecting data from a sensor, or processing images. These nodes communicate with each other through a publisher-subscriber system. In this system, a node can publish data to a topic, and other nodes can subscribe to that topic to receive data. ROS also supports services, which allow nodes to make synchronous requests and wait for responses.

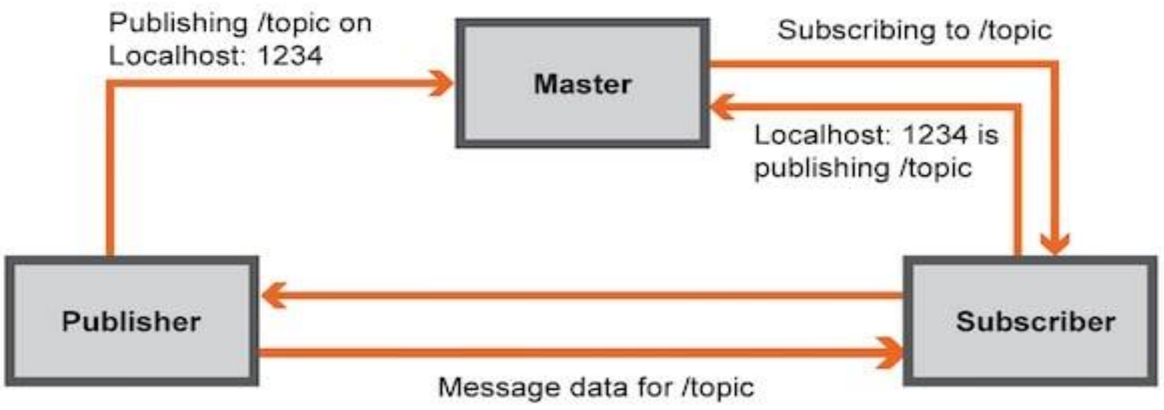

Publishing /topic on Localhost: 1234
Master
Subscribing to /topic
Localhost: 1234 is publishing /topic
Publisher
Subscriber
Message data for /topic

Figure 2.27: ROS Activity

(Source: An Introduction to Robot Operating System (ROS) (by Yahya Tawil), 2017.)

In addition, ROS also provides actions, which allow for more complex Missions to be performed with the ability to track progress and abort if necessary. Another important component of ROS is the ROS master, which is responsible for managing information about nodes and helping them find and connect to each other. The ROS master helps ensure that nodes can discover and communicate with each other effectively.

ROS also integrates many development support tools such as rviz for data visualization and visualization, and Gazebo for simulating robots in 3D environments. ROS libraries provide pre-developed software packages, which reduce the developer's workload and speed up the application development process. Thanks to its flexible architecture and rich tooling system, ROS simplifies the development of complex robotic systems, from basic control to programming intelligent behaviors, creating a powerful platform for modern robotic projects.

In ROS, there are common and commonly used support packages in the field of robotics and automation as follows:

Move Base: Move Base is one of the important software packages in the Robot Operating System (ROS) ecosystem and plays a central role in controlling the movement of mobile robots. Move Base provides the necessary tools and interfaces to navigate the robot from a current position to a target position safely and efficiently. Below is a detailed description of Move Base, its effects, structure and components.

Move Base is a node in ROS that is responsible for navigating the robot in the environment. It performs the following main functions:

Path planning: Using global and local planner algorithms, Move Base creates a safe and efficient path to achieve the goal. Obstacle avoidance: Move Base integrates the ability to recognize and avoid obstacles based on the robot's sensor information such as lidar, camera, or sensors. Move Base is an important node in ROS (Robot Operating System) with the main role of navigating the mobile robot. It provides the tools and interfaces needed to help the robot move from its current position to a target position safely and efficiently.

Functions Move Base has many important functions in the robot navigation system: Receiving and processing navigation goals: Move Base receives goal coordinates from the user or from higher-level systems through ROS messages. Path planning: Move Base uses global and local planner algorithms to create a safe and efficient path to achieve the navigation goal. Obstacle avoidance: Move Base integrates the ability to recognize and avoid obstacles based on the robot's sensor information such as lidar, camera, or ultrasonic sensors.

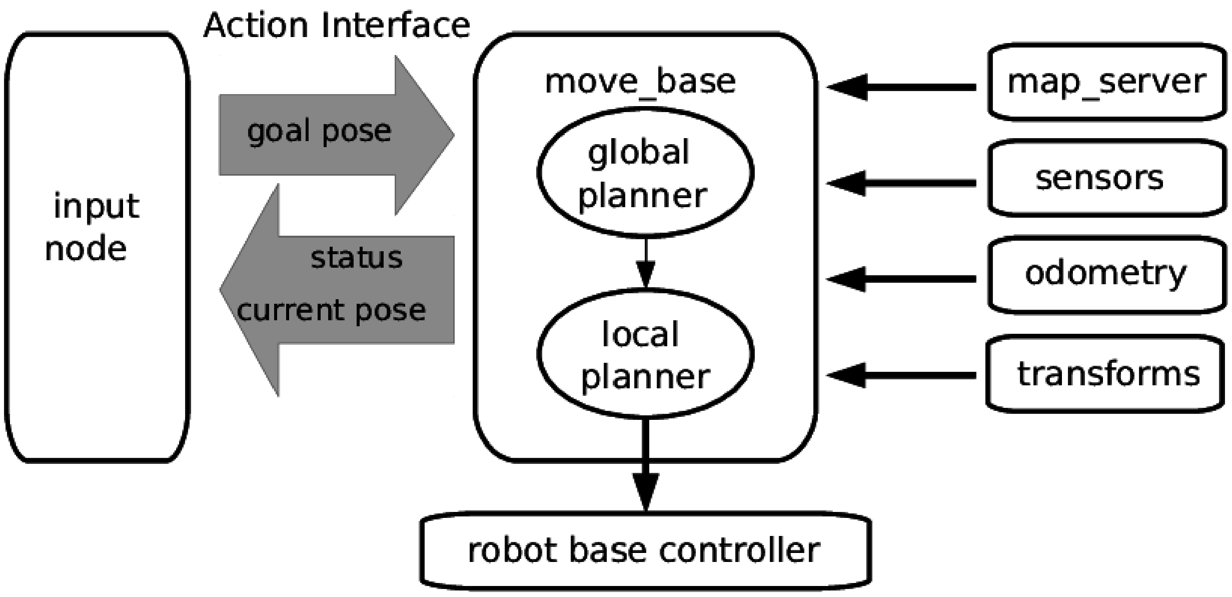


Figure 2.28: Move_Base Diagram

(Source: ROSoClingo: A ROS package for ASP-based robot control (by Benjamin Andres, Philipp Obermeier, Orkunt Sabuncu, Torsten Schaub, and David Rajaratnam), 2013.)

The Move Base has a complex structure with many components, each playing a specific role in robot navigation. Below are the main components of the Move Base:

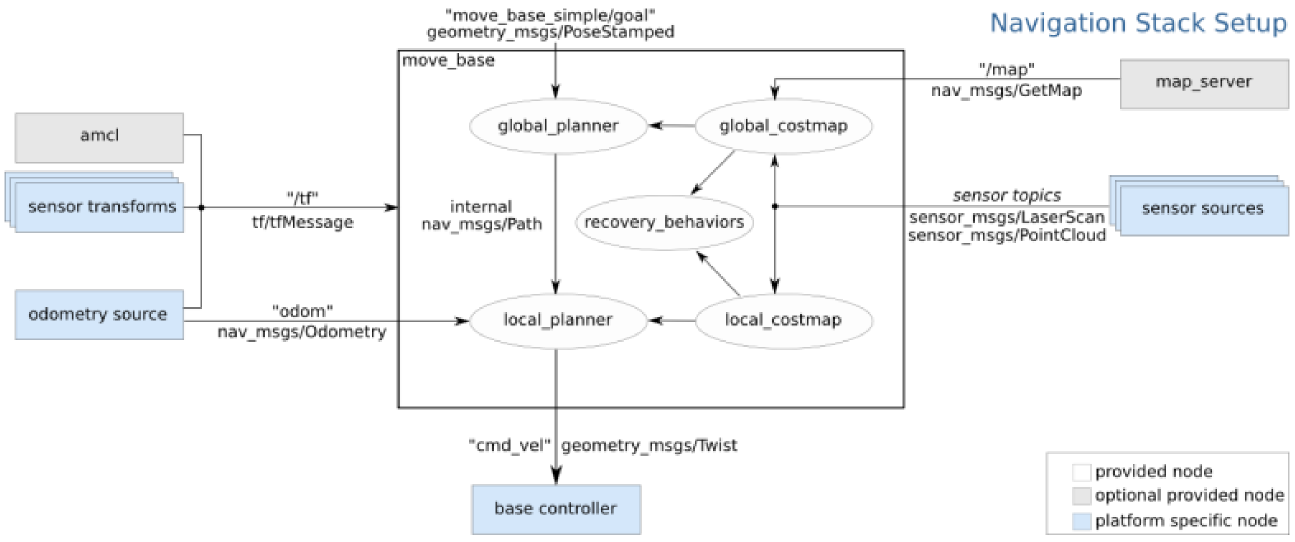


Figure 2.29: The figure shows the interaction between the move base and other components

(Source: https://wiki.ros.org/move_base)

Global Planner (Global planning):

- Mission: Generate a general path from the current position to the target position on a static map of the environment.
- Algorithm: Commonly used algorithms include Dijkstra, A*, v.v.

Local Planner (Local planning):

- Mission: Create small paths and quickly adjust the robot's direction of movement to avoid obstacles in a dynamic environment.
- Algorithm: Popular algorithms include DWA (Dynamic Window Approach), TEB (Timed Elastic Band).

Costmap (Cost Map):

- Mission: Generate global and local cost maps, representing the robot's surroundings and the costs associated with moving through different areas.
- Structure: The costmap is continuously updated based on data from sensors and consists of layers such as static_layer, obstacle_layer, inflation_layer.

Recovery Behaviors (Restorative behavior):

- Mission: Provide recovery strategies when the robot encounters difficult situations or gets stuck, such as turning back and finding another path. Example: Clear Costmap, Rotate Recovery.

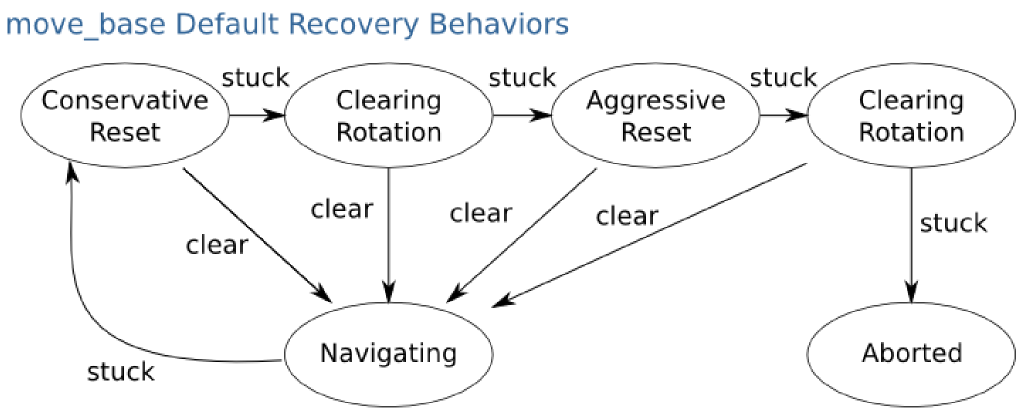


Figure 2.30: Operation diagram of Recovery Behaviors function

(Source: https://wiki.ros.org/move_base)

Robot Localization: Robot Localization is a software package in the ROS (Robot Operating System) system that is used to determine the position of the robot in the environment. This is an important component to ensure that the robot can know its current position with high accuracy, thereby performing navigation Missions and interacting with the environment effectively.

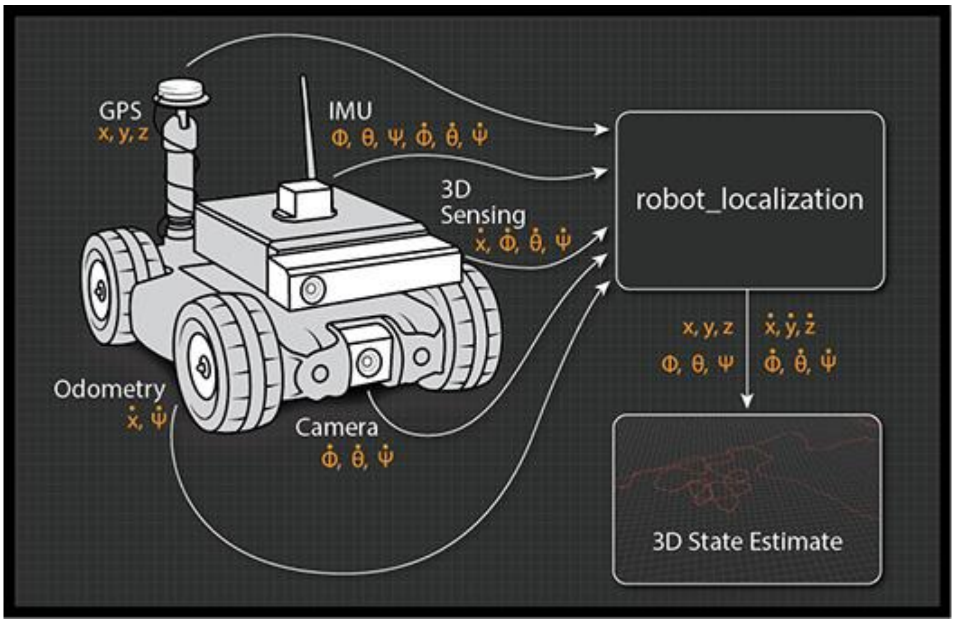

GPS
x, y, z
IMU
$\Phi, \theta, \Psi, \dot{\Phi}, \dot{\theta}, \dot{\Psi}$
3D Sensing
$\dot{x}, \dot{\Phi}, \dot{\theta}, \dot{\Psi}$
Odometry
$\dot{x}, \dot{\Psi}$
Camera
$\dot{\Phi}, \dot{\theta}, \dot{\Psi}$
robot_localization
x, y, z
Φ, θ, Ψ
$\dot{x}, \dot{y}, \dot{z}$
$\dot{\Phi}, \dot{\theta}, \dot{\Psi}$
3D State Estimate

Figure 2.31: The figure shows the schematic diagram of robot localization operation

(Source: https://cra.com/charles-river-analytics-presents-research-in-unmanned-systems-autonomous-robotics-and-network-optimization-at-mors-symposium/.)

The robot localization package has the following important effects:

- Accurately locate the robot using advanced filtering techniques to estimate the robot's position and direction of movement in the environment.
- Improved navigation: Providing accurate position information helps navigation packages such as Move Base work more efficiently.
- Sensor data integration: Combine data from different types of sensors such as IMU (Inertial Measurement Unit), encoder, and lidar to improve the accuracy of position estimation.

Robot Localization typically uses advanced filtering algorithms to integrate and process data from sensors. Here are the main components and architecture:

Extended Kalman Filter (EKF):

Mission: Use a dynamic state model and sensor measurements to estimate the robot's position and orientation.

Operation: EKF uses two main steps: prediction and update. In the prediction step, the robot's state is predicted based on the kinematic model and control inputs. In the update step, the predicted state is adjusted based on current sensor measurements.

Unscented Kalman Filter (UKF):

Mission: Similar to EKF but uses sample points (sigma points) to estimate the state probability distribution, which improves accuracy in highly nonlinear systems.

How it works: UKF generates a set of sample points from the current state distribution, predicts the states of these points, and then updates the state distribution based on new measurements.

Particle Filter (PF):

Mission: Use a set of particles to represent the probability distribution of the robot state. This is a common method in complex and non-linear environments.

How it works: Particle Filter uses the same prediction and update steps as EKF and UKF, but instead of using Gaussian state and measurement models, it uses a set of particles to estimate the state distribution.

Sensor Fusion:

Mission: Combine data from different types of sensors such as IMU, encoder, lidar, and GPS to improve the accuracy and reliability of position estimation. Operation: Sensor Fusion uses advanced filtering algorithms such as EKF, UKF, and PF to combine sensor data in an optimal way, minimizing measurement errors and noise.

Tf và tf2: TF (Transform) is an important software package in the ROS (Robot Operating System) system, which is used to track and manage the coordinate systems of the robot and its surrounding environment. It allows the different components of the robot system to know the position and orientation of parts and objects in three-dimensional space in real time. TF has the following functions:

Managing coordinate systems: Helps robots track and manage the coordinate systems of different parts such as the robot body, sensors, joints, and other objects in the environment.

Coordinate system conversion: Provides tools to convert between different coordinate systems, helping system components understand the position and orientation of objects accurately.

Time synchronization: Ensures that coordinate system transformations are synchronized in real time, helping to maintain accuracy and consistency during navigation and interaction with the environment.

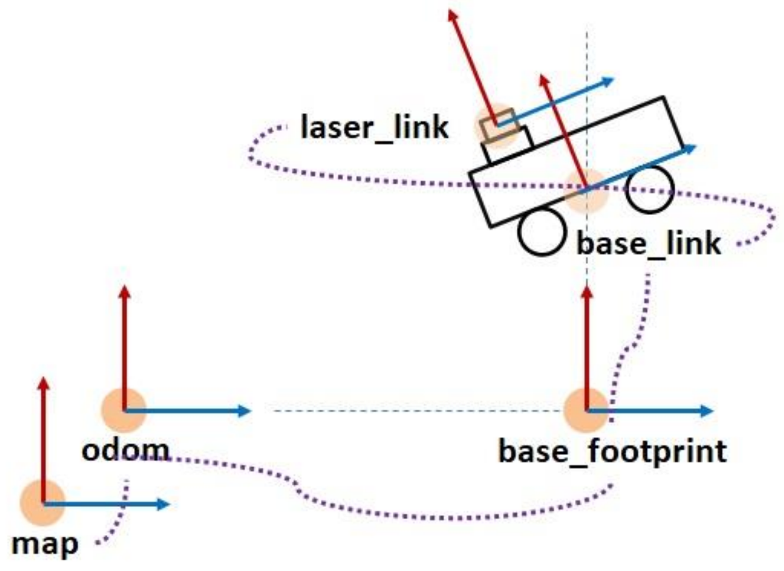


Figure 2.32: Illustration of transformation between coordinate axes in ROS

(Source: https://memo.soarcloud.com/ros%E3%81%AE%E5%BA%A7%E6%A8%99%E5%A4%89%E6%8F%9Btf%E3%81%AB%E3%81%A4%E3%81%84%E3%81%A6/.)

The TF package in ROS has a complex structure with many different components, each of which takes on a specific Mission to ensure that coordinate management and transformation are performed correctly and efficiently. Below are the main components of TF.

tf::TransformListener:

Mission: Listen and store transformations between coordinate systems over time.

Operation: TransformListener receives information about transformations from other nodes in the system and stores them in a buffer for later retrieval.

tf::TransformBroadcaster:

Mission: Broadcast transformations between coordinate systems to the ROS system.

Operation: TransformBroadcaster sends information about transformations to ROS topics, from which other components can listen and use this information.

tf::Transform:

Mission: Represents a transformation between two coordinate systems, including position and orientation information.

Structure: A tf::Transform consists of a translation and a rotation, usually represented by a quaternion or rotation matrix.

tf::StampedTransform:

Mission: A timestamped transformation.

What it does: tf::StampedTransform extends tf::Transform by adding time information and names of the origin and destination coordinate systems, which helps ensure time accuracy during transformations.

About TF2 is an enhanced version of the original TF package, providing features and improvements in memory, processing performance, and integration with other systems in ROS. It also supports different data types and provides better tools for handling complex transformations.

Teb Local Planner: TEB (Timed Elastic Band) Local Planner is a software package in the Robot Operating System (ROS) ecosystem used for local path planning for mobile robots. This package focuses on generating optimal short-term paths for robots, especially in dynamic and complex environments, helping robots move safely and efficiently from their current position to their target position.

Coverage Path Planning and Boustrophedon Cellular Decomposition Algorithms: To be able to pick up the ball efficiently, an algorithm that can plan a path that covers the entire area is needed. Here we have the coverage path planning algorithm along with the Boustrophedon Cellular Decomposition to give the basic directions of straight up and straight down.

The algorithm will adapt to any type of closed boundary input, thereby providing a more suitable trajectory for movement.

Boustrophedon Cellular Decomposition (BCD) is an advanced technique in the field of coverage path planning of mobile robots. This method is especially effective in ensuring that the robot can cover the entire working area without missing any part, especially in complex environments with many obstacles. BCD is widely used in many fields such as floor cleaning, ball picking "Boustrophedon" comes from Greek, meaning "go and return", similar to the way farmers plow the fields. This method divides the working area into cells and then the robot will move back and forth between each cell in a zig-zag pattern, ensuring the entire area is covered. This technique focuses not only on spatial path optimization but also on feasibility and time efficiency.

The algorithm will work in the following order:

Building Environmental Charts (Environment Map): First, a detailed map of the work area needs to be created, including obstacles and boundaries of the area to be covered. This map is often built using sensors such as LIDAR or cameras.

Cell Division (Cell Decomposition): Uses a sweep-line algorithm to divide the area into non-overlapping cells. Each cell is defined based on the location and size of obstacles, creating simpler zones to manage and navigate.

Plan Paths for Each Cell (Path Planning for Each Cell): Within each cell, the robot plans an optimal path in a Boustrophedon-like fashion. This process involves moving along parallel paths, then turning around and moving back in the opposite direction, ensuring that the entire surface of the cell is covered.

Combine Paths: After planning each cell, the paths of each cell are combined into one continuous path. This allows the robot to move smoothly from cell to cell, without repeating paths or missing any areas.

2.16 Wheel Hex Joint 68mm– Type 3

Wheel Hex Joint 68mm– Type 3 has very good quality. The wheel uses a common hexagon joint (12mm) which is easy to assemble and apply to the design.

-Good quality with a soft rubber tire surface for very good friction, much better than type 1, type 2.

-Optimal tire tread design for the highest friction and grip

-Has a very good elastic thick foam layer inside to prevent the wheel from collapsing when loaded and has the best friction with the road.

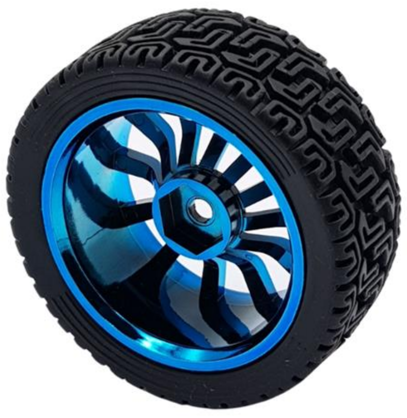


Figure 2.33: Wheel Hex Joint 68mm– Type 3

Datasheet:

-Material: Hard plastic, foam padding, good rubber.

-Diameter: 68mm.

-Wheel width: 27mm

2.17 7inch HDMI Picture waveshare monitor for Raspberry Pi

7inch HDMI LCD touch screen allows Raspberry Pi to be connected via HDMI port, 800×480 resolution capacitive touch, supports Raspberry Pi and can also be used as a computer monitor. In this project, Waveshare screen will be connected to Raspberry Pi to display the image of the environment in front of the camera when detecting for the purpose of processing Pickleball image.

Application:

-Display Systems: Make displays for embedded systems or IoT devices.

-Learning and Teaching: Great learning and teaching tool for programming and electronics courses.

-Projects with Raspberry Pi: Use in DIY projects, such as custom tablets, smart home control systems, and many other creative applications.

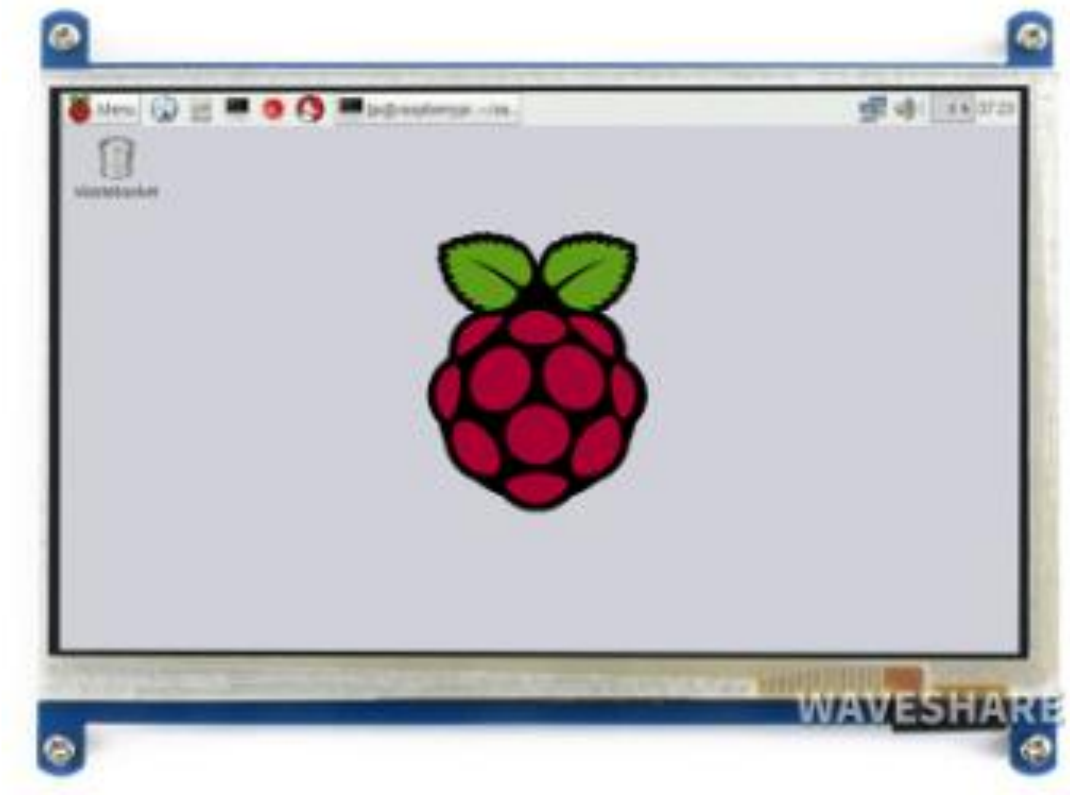


Figure 2.34: Waveshare 7inch

2.18 Li-ion cell battery NCR 18650GA 3400mAh

-The NCR18650GA Li-ion cell is a 3400mAh cylindrical lithium-ion battery produced by Panasonic, which is a high-capacity and good-performance battery mainly used to replace laptop batteries, high-end flashlights, solar lamp battery sources, backup power sources, and can also be used for various rechargeable battery packs. In addition, this 18650 battery can be recharged many times and operates in high-temperature environments.

-In this project, the NCR 18650GA Li-ion cell will be powered to control the L298N modules and the motors to operate.

Datasheet:

-Brand: Panasonic

-Model: NCR18650GA

-Voltage: 3.7V, full charge: 4.2V

-Lifespan: 1000 times

-Capacity: ~3400 mAh

-Diameter: 18mm

-Length: 65mm

Figure 2.35: Li-ion cell battery NCR 18650GA 3400mAh

2.19 120W Power Bank

Similar to previous products, 120W 20000mAh power bank is equipped with Li-Po (Lithium Polymer) battery core from leading manufacturers such as LG/Panasonic. Therefore, you can rest assured about the quality and longevity of the product.

In this project, we can use the power bank to power the Raspberry Pi to operate with an output of 5V.

Datasheet:

- Slim and lightweight design
- Battery capacity: 20.000 mAh
- Fast charging speed with a capacity of 18 W
- Includes 2 Micro USB and Type C input ports
- Equipped with 2 convenient USB output ports that can charge 2 devices at the same time.
- The backup charger uses a Polymer battery core with safety protection against overcurrent and overload for the devices in use

Figure 2.36: 120W Power Bank

2.20 Webcam

Webcam is a compact digital video recording device, usually mounted on a computer or integrated into a laptop, to capture and transmit video directly over the internet or store it on a computer. Webcam can be used for many different purposes, including: Video calling, security monitoring... in this topic, the webcam is intended to connect to a raspberry to process images and identify pickleballs.

Design

-Compact size, easy to carry anytime, anywhere.

-Has a convenient clip for use with computers and TVs.

Uses

-Record live video with clear, detailed HD image quality.

-Talk with the built-in microphone in the HD Webcam for computers.

Datasheet

-Dimensions: 79mm x 31mm x 38.5mm.

-Video resolution: 1080p.

-Photo resolution: 10MP.

-Frame rate: 60FPS.

-Color: Black - Resolution: HD 1080

-Connection: USB 2.0

-Cable length: 1.5m

Figure 2.37: Webcam

CHAPTER 3: DESIGN AND CONSTRUCTION

3.1 Block diagram

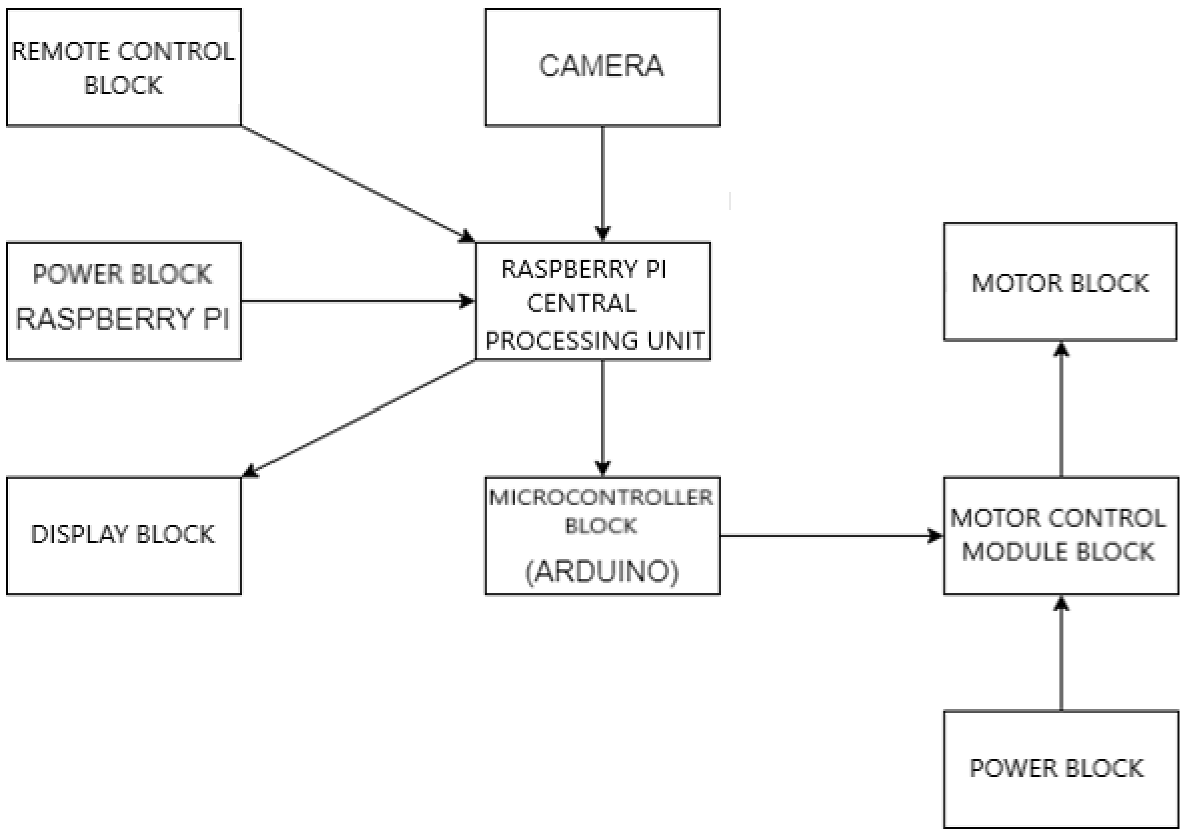

REMOTE CONTROL BLOCK
CAMERA
POWER BLOCK RASPBERRY PI
RASPBERRY PI CENTRAL PROCESSING UNIT
MOTOR BLOCK
DISPLAY BLOCK
MICROCONTROLLER BLOCK (ARDUINO)
MOTOR CONTROL MODULE BLOCK
POWER BLOCK

Figure 3.1: Block diagram of the system

The block diagram of this system includes the following blocks:

-Raspberry Pi power block: The power supply for Raspberry Pi is 5V and the output is USB Type C. With the need for flexibility to operate within the scope of this topic, I used the power of the backup charger to supply the Raspberry Pi to operate.

-Module power block: I used 3 3400mah batteries to supply the motor control module, with these 3 batteries, it will help control the motor for a period of time enough for the scope of the student project.

-Raspberry Pi central processing block: In this project, due to the requirement for flexibility and compactness, we will use a Raspberry Pi 4 computer instead of a regular computer, although much more powerful but large, heavy and bulky, not suitable for the needs of the project. Although Raspberry Pi 4 is not as powerful as a regular computer, its processing capacity is enough for the needs of the topic, besides there are other reasons for us to use it for this topic, especially its compactness.

-Camera block: This block is used to record and collect images from the environment. After the camera records the image, it will be sent to the Raspberry Pi central processor. Using Python language on Geany programming software with the results we programmed before, the information will be transmitted to Arduino via UART communication and Arduino will proceed to process the next steps.

-Control block: After the camera collects images, it is sent to the central processor, the Raspberry Pi, to process the image using the Python language that we programmed before. Then, using UART communication, Raspberry Pi will send information to Arduino and then Arduino will control the motor through the motor control module. If the object is in frame 1,2, Raspberry Pi will transmit a signal to Arduino to control the motor to rotate to the left, if it is in frame 6,7, Raspberry Pi will transmit a signal to Arduino to control the motor to rotate to the right. The remaining frames are frame 3,4,5, Arduino will control the motor to make the car run straight forward. With this way of dividing the cells to control the motor, it will bring quite high accuracy.

-Motor control module block: L298N is a module that integrates a protection diode and a 7805 power IC, it is capable of controlling 2 DC motors. Thanks to this module, the DC motors attached to the wheels will operate effectively and with high accuracy.

-Motor block: This block includes 5 motors, 4 motors attached to the wheels to move the robot smoothly, 1 motor attached to the ball collecting mechanism will help collect the balls quickly and effectively.

-Display block: We use a 7-inch screen to connect to Raspberry Pi to display the main screen of Raspberry Pi, execute commands for Raspberry Pi to start the program and at the same time can observe the camera angle to identify the Pickleball ball.

-emote control block: We use the Blynk app to connect to control the motor in 2 modes: auto and manual.

3.2 System wiring diagram

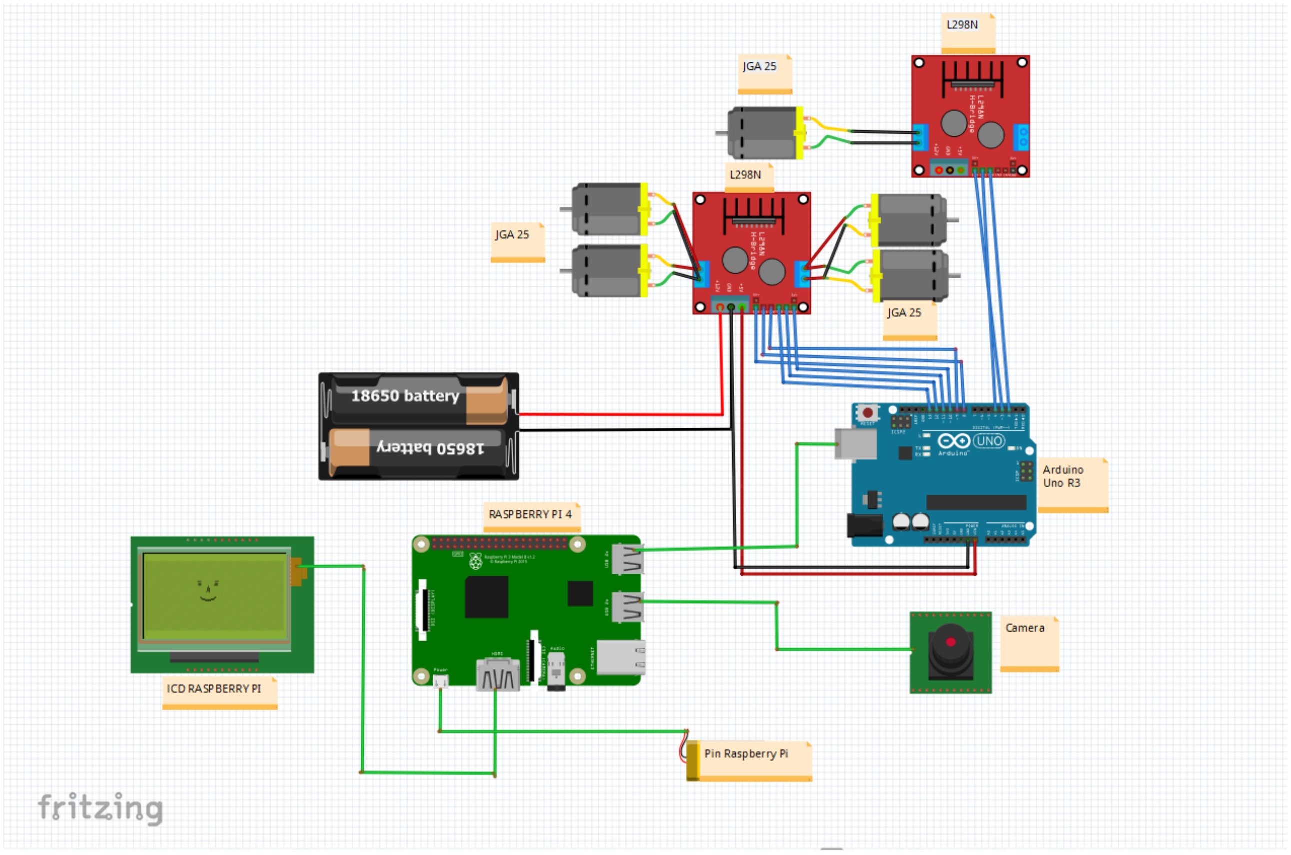


Figure 3.2: Wiring diagram

The system wiring diagram includes the following components:

-Raspberry Pi 4: Acts as the central processor of the system

-2 L298N motor control modules

-Arduino Uno R3: Arduino controls the motor through the L298N motor control circuit

-5 JGA25 DC motors: These motors are for the robot to move, and are also a Pickleball collecting mechanism

-Power source: here we use 2 sources: the power source for the Raspberry Pi is a backup battery and the power source for the motor control module is 3 18650 3200mAh batteries

3.3 Operational description

When the camera is started and the camera captures the image of the Pickleball, the information will be transmitted to the Raspberry Pi 4 computer, where the image of the ball will be analyzed based on the previously programmed Python code. If it is determined that it is a Pickleball, the Raspberry Pi will transmit the information to the Arduino via the UART protocol to control the DC motors to operate, helping the robot move closer to the ball and at the same time the ball collection mechanism will also collect the ball and push the ball into the container.

3.4 Robot drawings

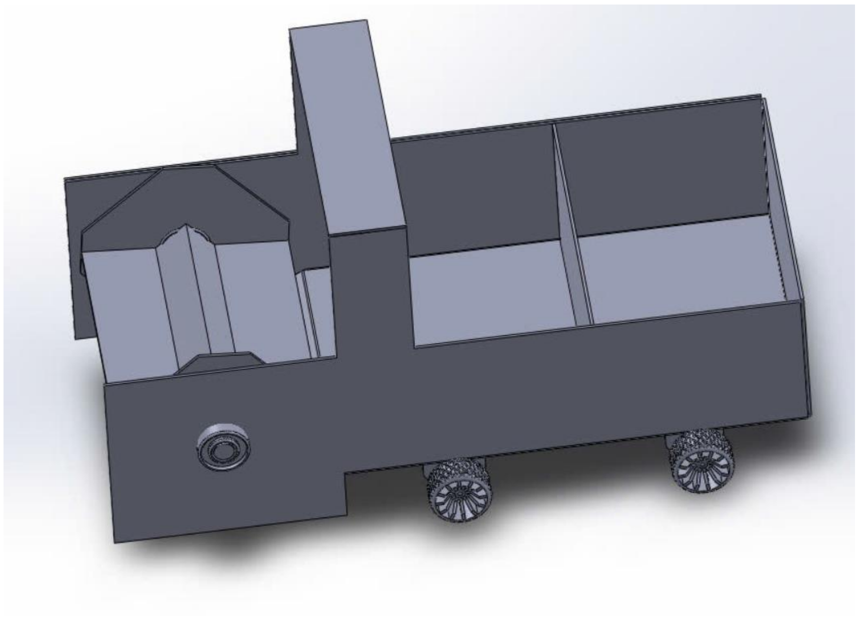


Figure 3.3: Overall Robot Drawing

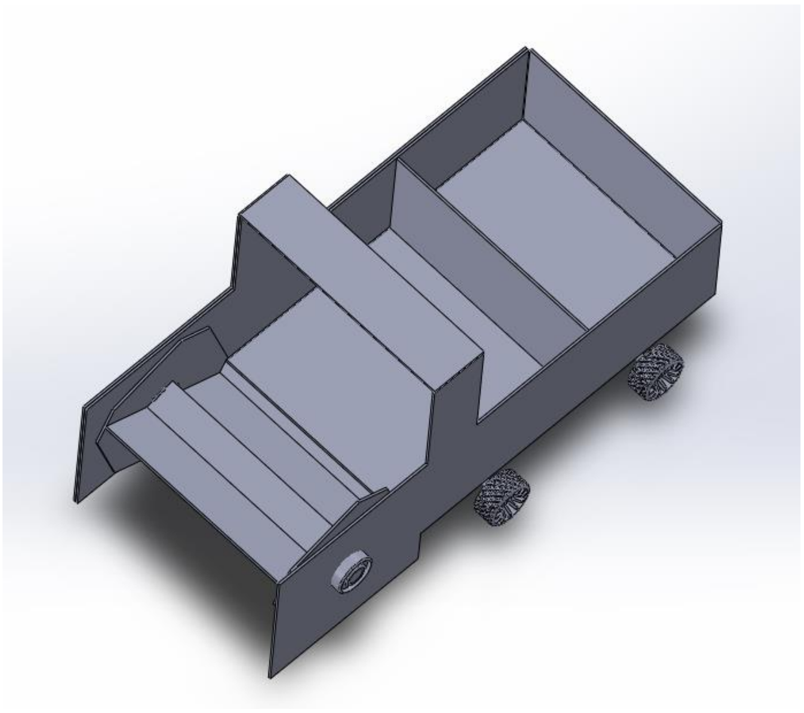


Figure 3.4: Robot drawing top view

3.4.1 Transmission system

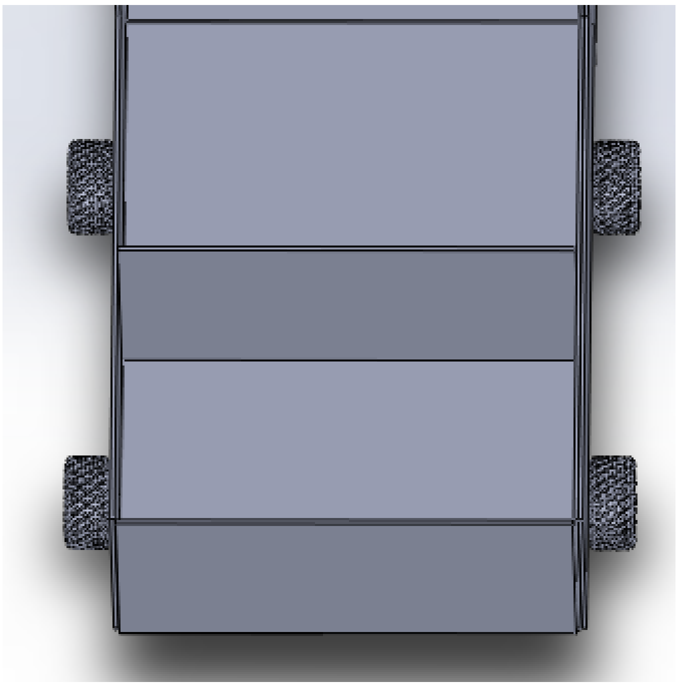


Figure 3.5: Transmission system

The robot moves using a JGA-25 4-motor drive attached to 4 wheels. The robot will move when the camera detects the position of the Pickleball.

3.4.2 Ball collection mechanism

The ball collecting trough is cut and glued and attached to the middle of the shaft connecting to the horizontal bearing bearing. The principle of this ball collecting mechanism is that the motor will rotate continuously, when detecting the ball, the robot will approach and the ball will be pushed into the basket.

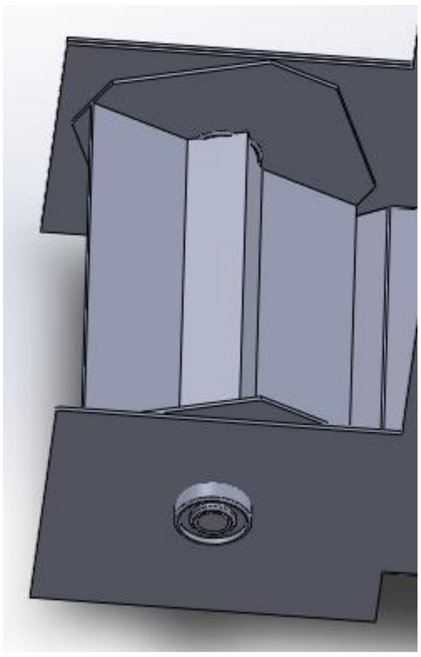


Figure 3.6: Ball collection mechanism

3.4.3 Ball container and component compartment

Behind the ball collection part will include a box to hold the balls and a separate compartment to store components, with a spacious design so it can hold many balls at the same time.

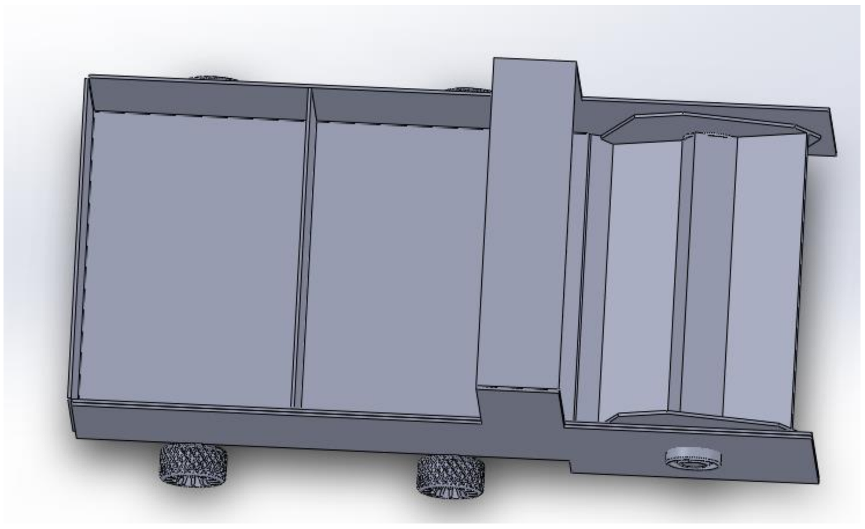


Figure 3.7: Ball and accessory storage bin

3.5 Construction

Step 1: Choose materials for the robot in the topic we will use mica because it is easy to buy, not too heavy.

Step 2: Choose the size of the robot to proceed to cut mica and assemble according to the desired size: Robot frame has a size of 30x45cm.

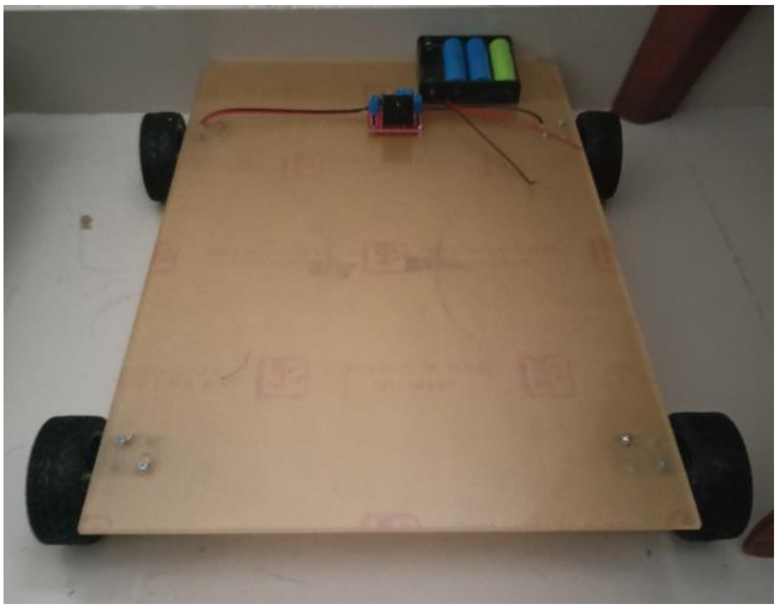


Figure 3.8: Robot frame

Step 3: Install the wheels, mount the motor to the 4 motors to make 4 wheels for the robot to move.

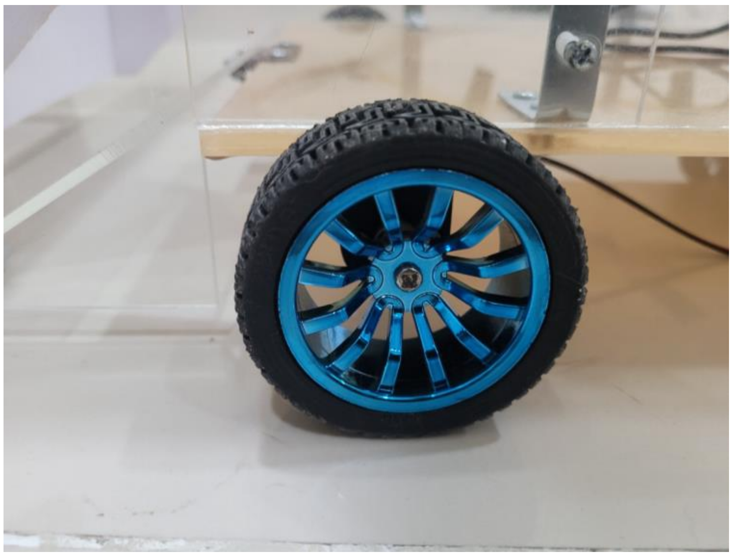


Figure 3.9: Robot wheel

Step 4: Install the component compartment for the robot, this will hold all the components of the system

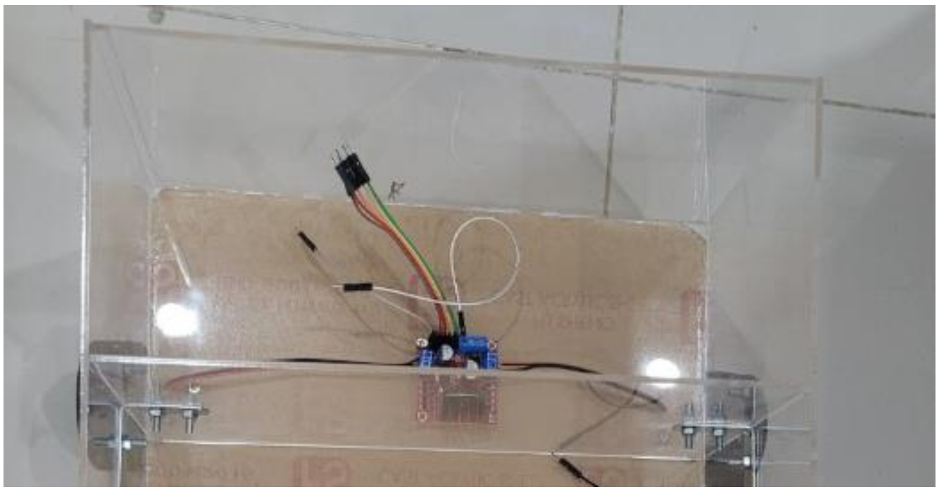


Figure 3.10: Component compartment

Step 5: Install a ramp to help roll the ball in for retrieval.

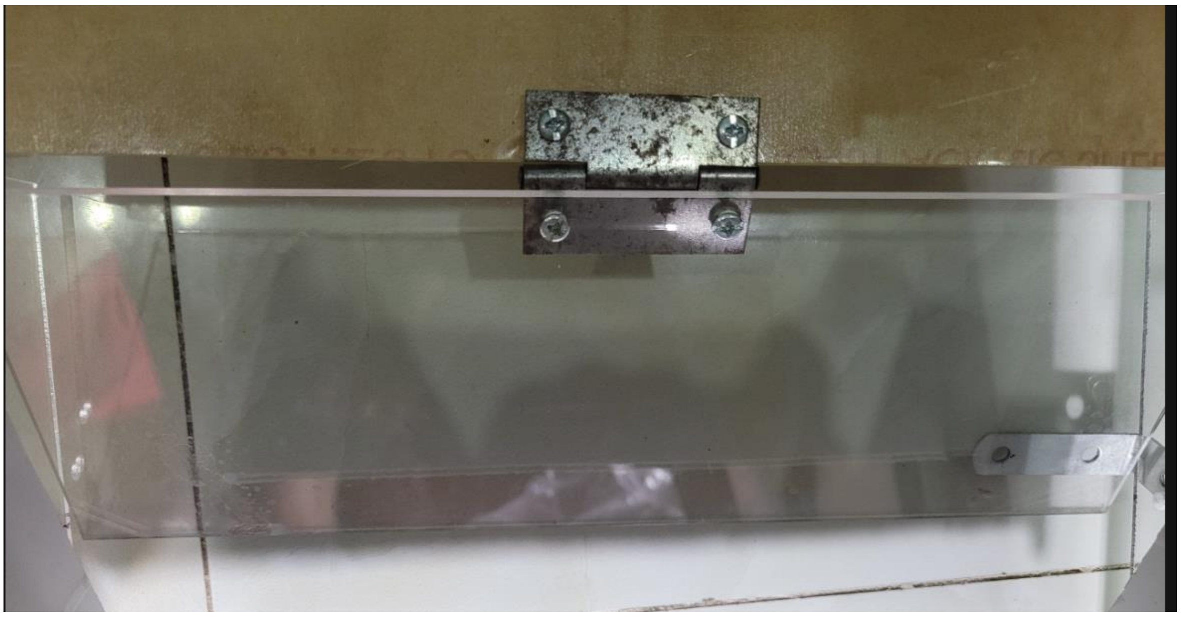


Figure 3.11: Ramp

Step 6: Install the camera mount for the robot

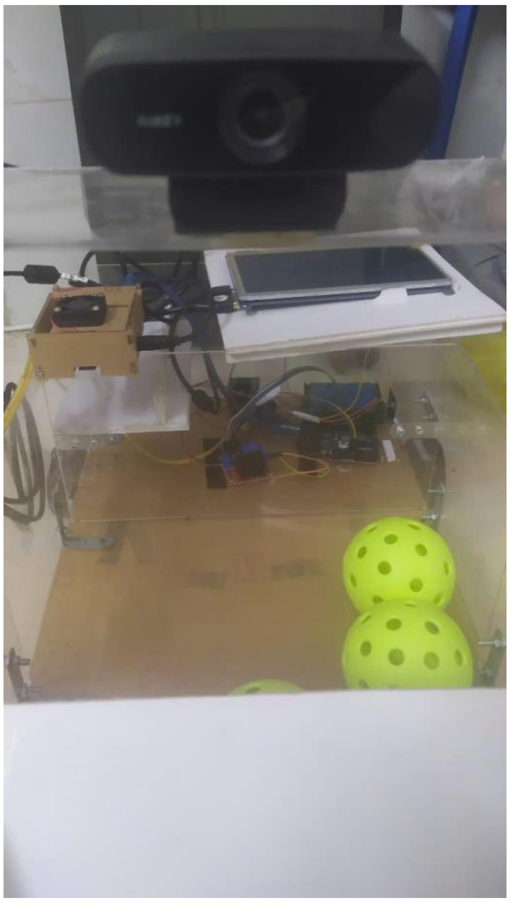


Figure 3.12: Camera mount

Step 7: Cut and paste the pickleball collecting mechanism for the robot

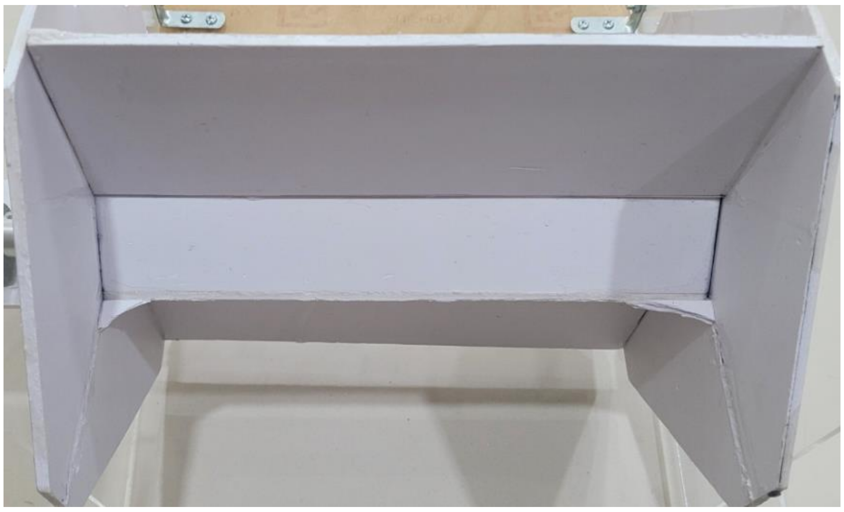


Figure 3.13: Pickleball chute

Step 8: Attach the horizontal shaft to the ball collecting motor

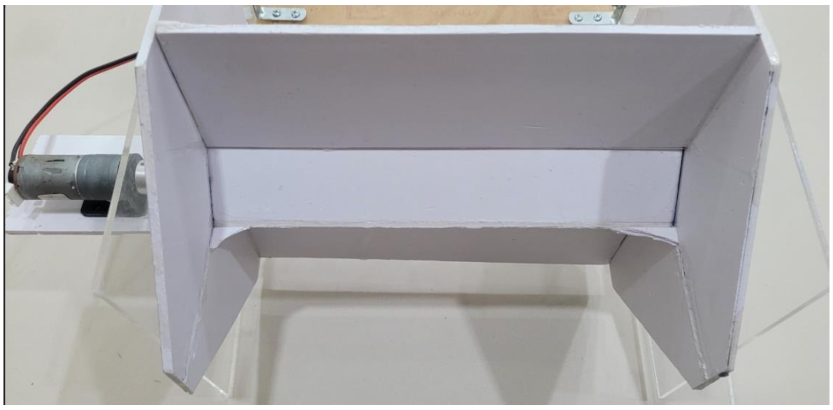


Figure 3.14: Horizontal shaft mounting of ball collecting motor

Step 9: Connect the wires as designed

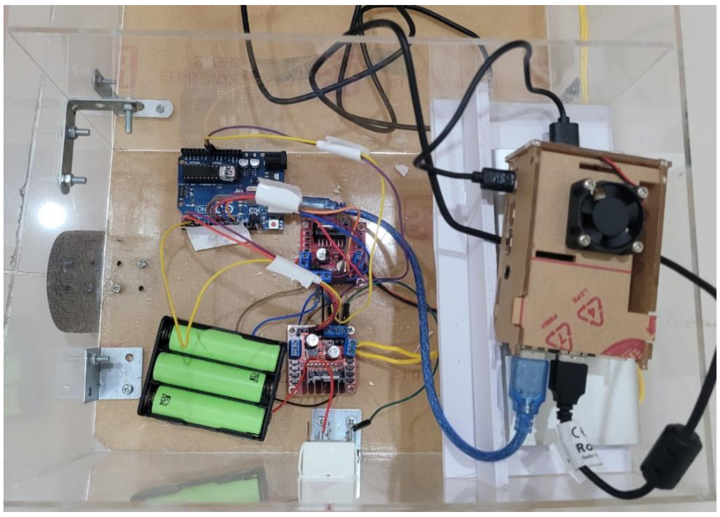


Figure 3.15: Connect the wires according to the diagram

Step 10: Complete the overall robot

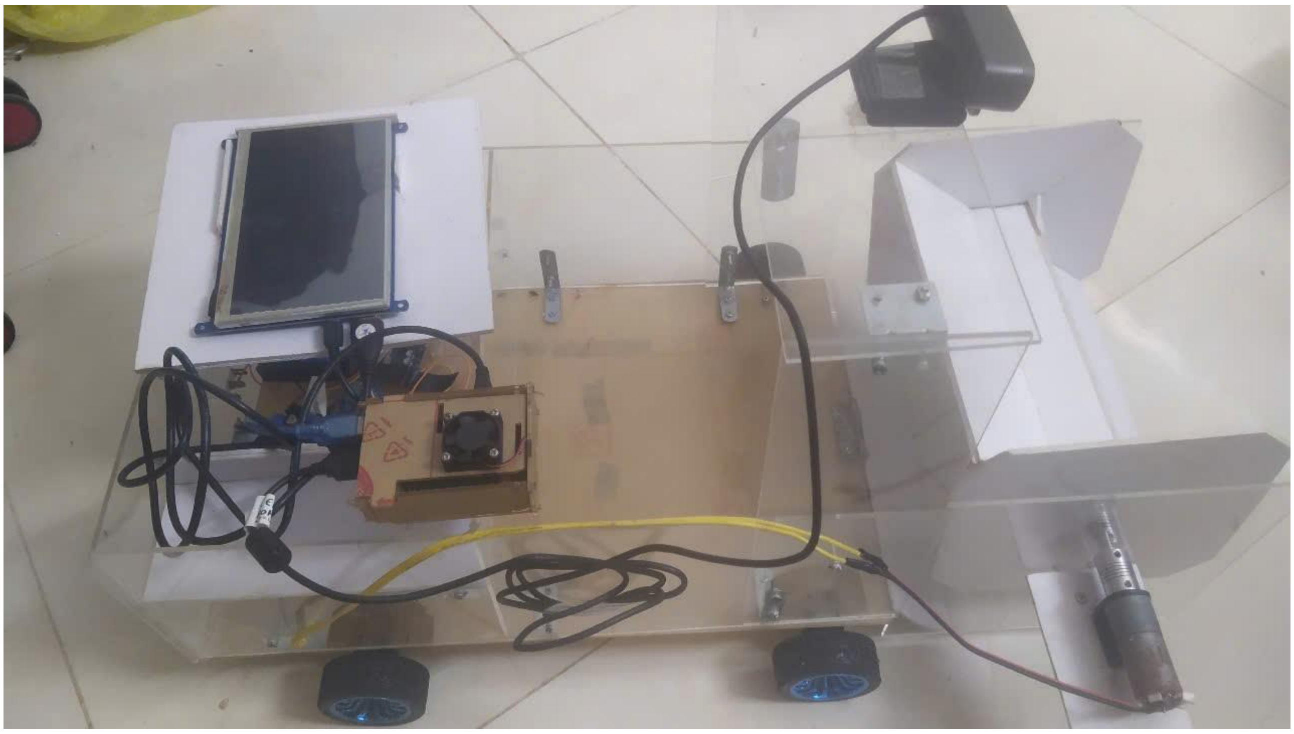


Figure 3.16: Robot Overview

Datasheet

Table 3.1: Robot datasheet

Datasheet	**Value**
Total weight	4,2kg
Total length	65cm
Total width	30cm
Frame height	15cm
Frame length	45cm
Frame Width	30cm
Total height including camera stand	46cm

3.6 Install Raspberry Pi operating system

First we need a micro sd card to store the operating system, then use Raspberry Pi Imager to install the operating system following these steps: Raspberry.

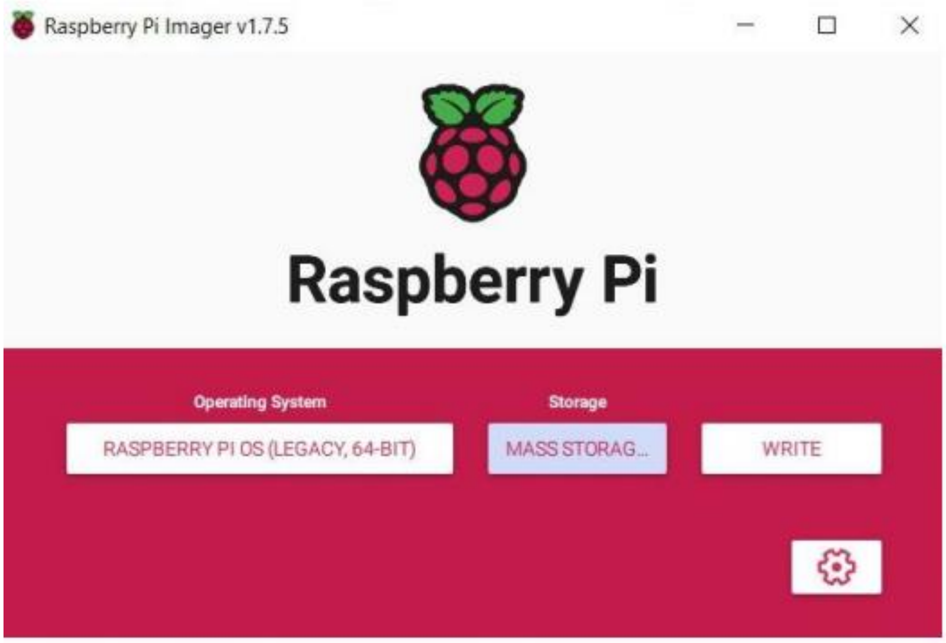


Figure 3.17: Install Raspberry Pi OS

Select Raspberry Pi OS (LEGACY, 64-BIT)

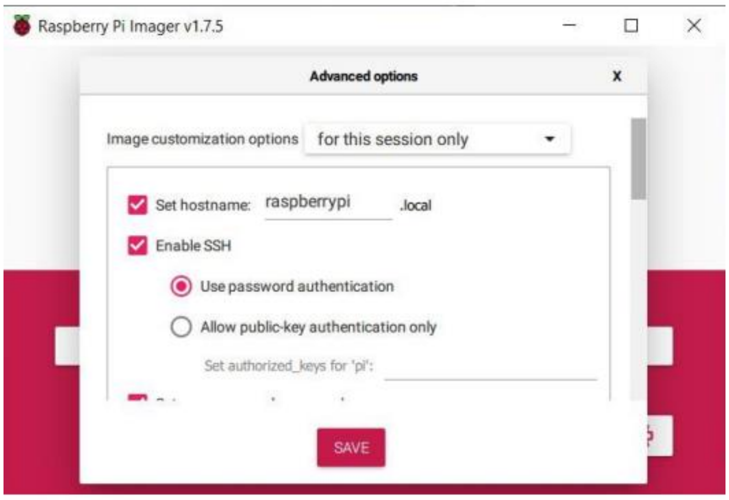


Image customization options: for this session only

Set hostname: raspberrypi .local

Enable SSH

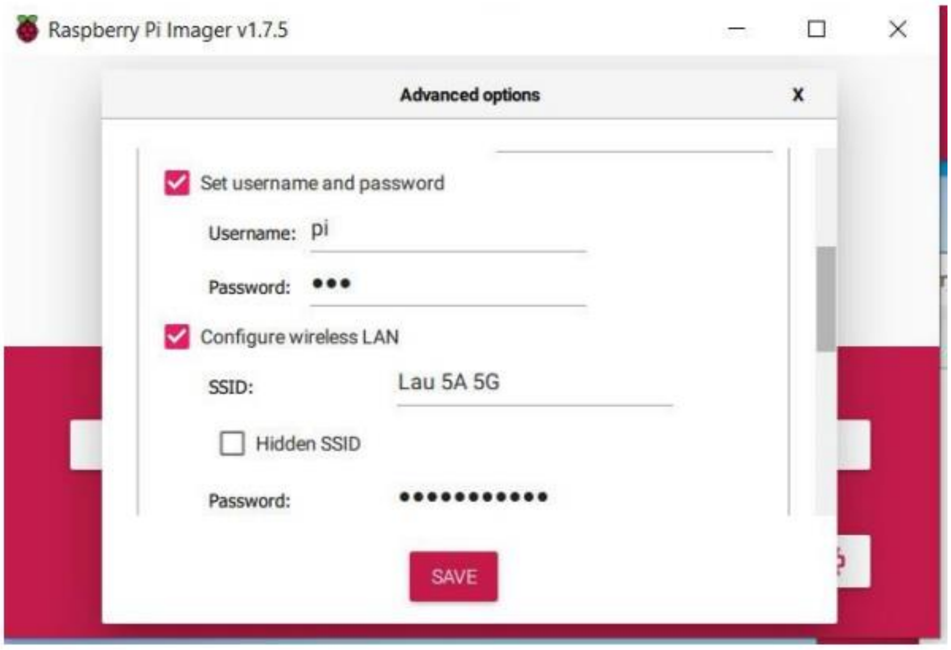
Use password authentication

Allow public-key authentication only

Set authorized_keys for 'pi':

Figure 3.18: Set HostName and Enable SSH for Raspberry Pi

In the Setting section, we select the hostname as raspberrypi. Next, select Enable SSH so that Raspberry Pi 4 can connect to Wifi and be controlled remotely via Wifi with VNC.



Raspberry Pi Imager v1.7.5

Advanced options

☑ Set username and password

Username: pi

Password: •••

☑ Configure wireless LAN

SSID: Lau 5A 5G

☐ Hidden SSID

Password: ••••••••••••

Figure 3.19: Setting up an account for Raspberry Pi

3.7 Raspberry Pi Picture screen interface

This is the main screen of Raspberry Pi, we can operate, use this screen to run programs, observe the camera angle on the screen.

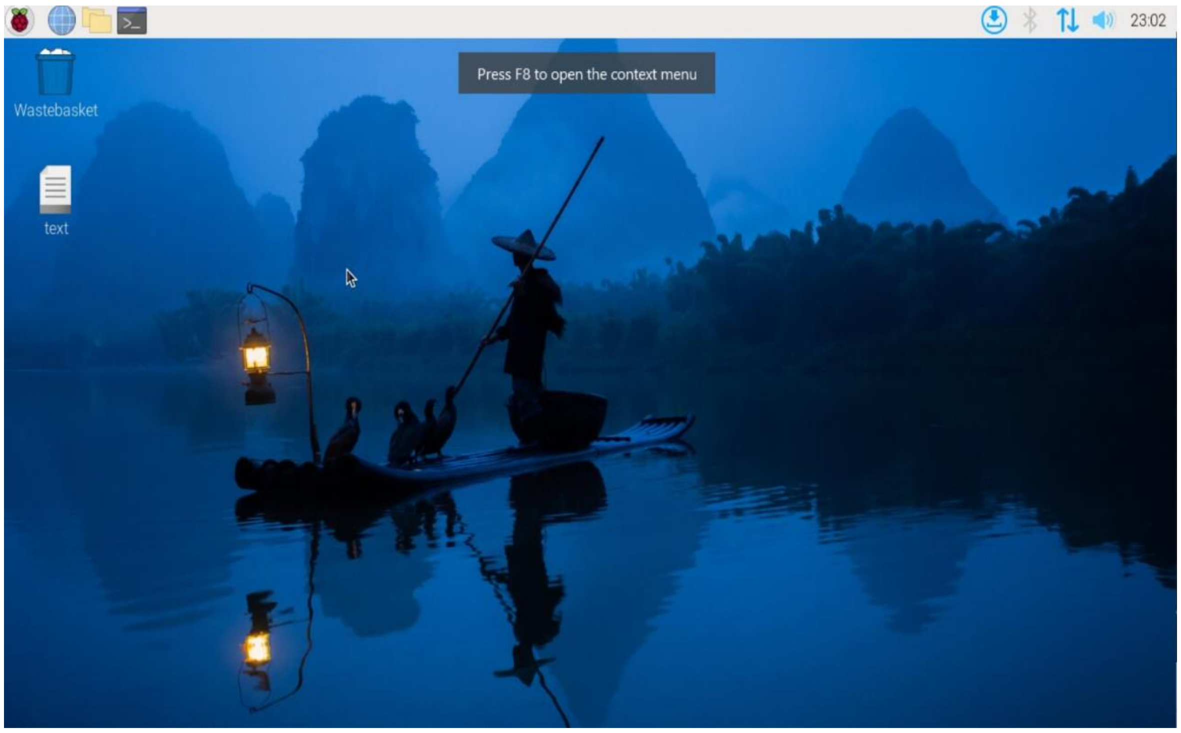


Figure 3.20: Raspberry Pi screen interface

CHAPTER 4: ALGORITHM AND CONTROL

4.1 Algorithm flowchart

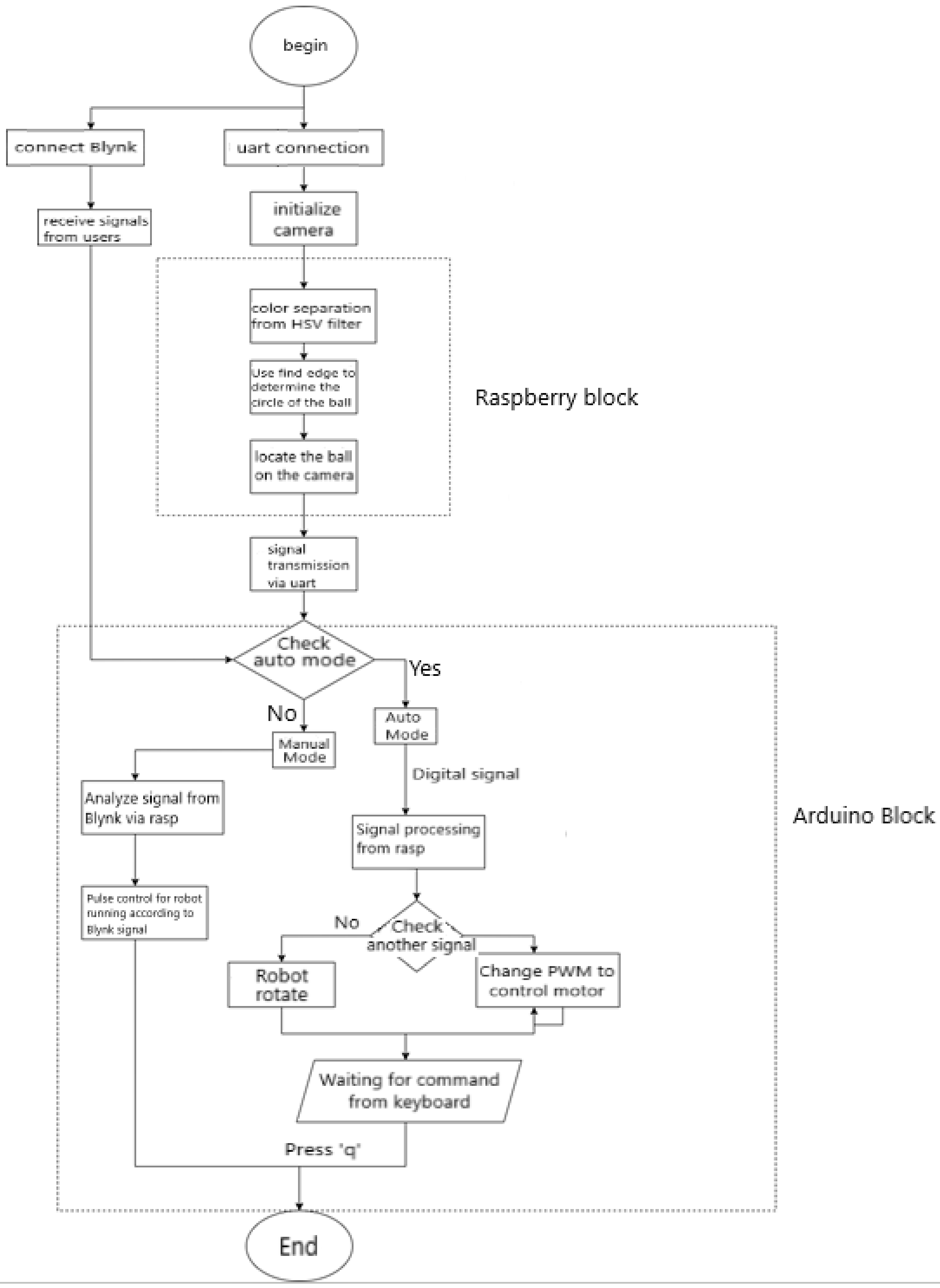

begin
connect Blynk
uart connection
receive signals from users
initialize camera
color separation from HSV filter
Use find edge to determine the circle of the ball
locate the ball on the camera
Raspberry block
signal transmission via uart
Check auto mode
Yes
No
Auto Mode
Manual Mode
Digital signal
Analyze signal from Blynk via rasp
Signal processing from rasp
Arduino Block
Pulse control for robot running according to Blynk signal
No
Check another signal
Robot rotate
Change PWM to control motor
Waiting for command from keyboard
Press 'q'
End

Figure 4.1: Algorithm flowchart

4.2 System operating principle

First, when we turn on the power, the system will operate, the components used will start up. When the components are finished starting up, the Arduino will be connected to the

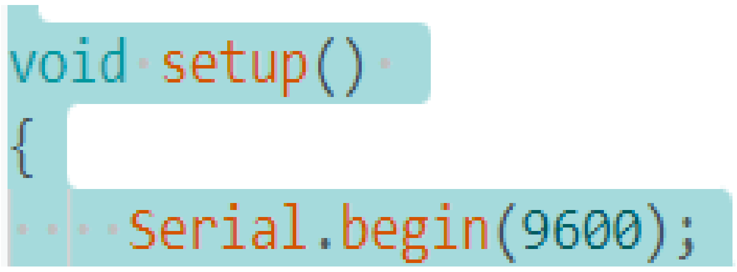
Raspberry Pi via the UART protocol. After the camera collects images to the Raspberry Pi, the Raspberry Pi will process and analyze the signal from the previously programmed Python language. From here, the Arduino will control the motor module to control the motor as we have programmed. If at signal 1,2, the Robot will turn left, and at signal 3,4,5 the Robot will go straight, and 6,7 is the remaining signal the Robot will turn right. After picking up the ball, the Raspberry will re-analyze the data from the surrounding environment and repeat the operations as before. If there is no ball in the camera's operating area, the robot will rotate to find the object, we can end the program by pressing "q" or turning off the system's power.

4.3 Receive data with Serial communication (UART) on Arduino

The UART serial communication standard on Arduino, also known as Serial, is a very popular protocol in embedded system applications.

UART declaration:

There are two ways to declare the use of UART on Arduino, but the most common is to use the command Serial.begin(9600), where 9600 is the baud rate and uses the default 8-N-1 transmission frame (8 data bits, no parity bit, 1 stop bit). This declaration also converts digital pins 0 and 1 into data transmission functions: digital pin 0 is connected to the microcontroller's internal data receiver and digital pin 1 is connected to the microcontroller's internal data transmitter.



```
void setup()
{
    Serial.begin(9600);
```

Figure 4.2: Commands for UART communication on Arduino

How to connect:

-If you use the Arduino Board to communicate with the computer (using the Serial monitor window on ARDUINO IDE), you don't need to connect anything else because the Arduino board has already done that, just plug in the USB cable.

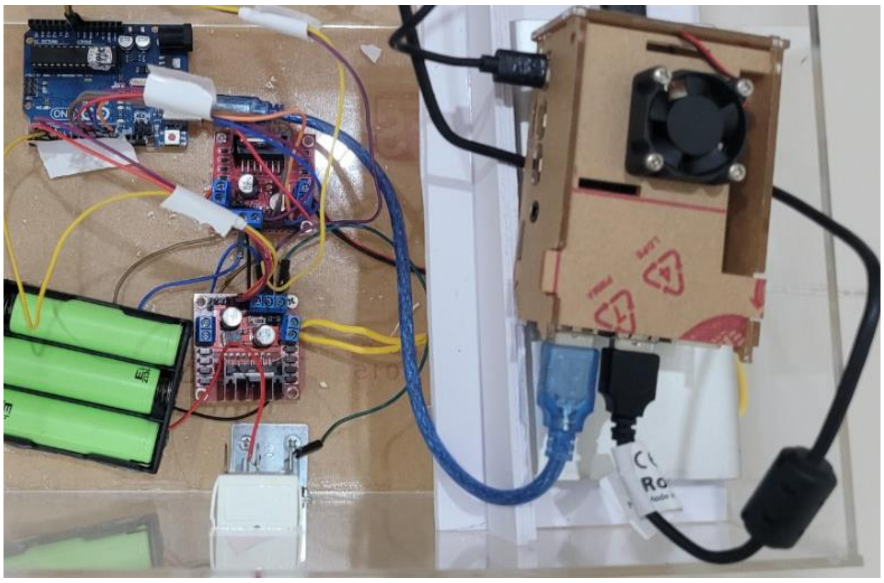


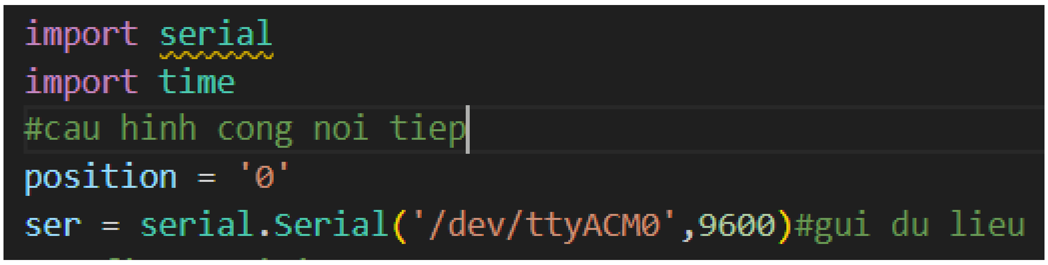
Figure 4.3: Arduino connected to Raspberry Pi via USB port

-When using Arduino to communicate with another device or module, you need to cross-connect the pins: TX – RX and RX – TX. Don't forget to check if both devices have the same GND, because if not, they will not understand each other's logic level and therefore will not be able to communicate.

-In other more complicated cases, when 2 devices have different logic levels, for example, a 5v device needs to be connected to a 3.3v device, we need to add a circuit to convert the voltage accordingly.

4.4 Passing data from Python to Arduino

In this project we need to communicate both Arduino and Python. How to transfer data from Python to Arduino is as follows:



```
import serial
import time
#cau hinh cong noi tiep
position = '0'
ser = serial.Serial('/dev/ttyACM0',9600)#gui du lieu
```

Figure 4.4: Passing data from Python to Arduino

Step 1: Configure the serial port using the "import serial" and "import time" commands

Step 2: ser = serial.Serial('/dev/ttyACM0', 9600) in Python using the pyserial library to establish a serial connection with a device in this case Arduino

-ser: This is an object of the Serial class from the pyserial library. This object will be used to perform reading, writing and managing the serial connection.

-serial.Serial(): This is the constructor of the Serial class in the pyserial library. This function establishes and opens the serial connection with the provided parameters.

-'/dev/ttyACM0': This is the name of the serial port that your device (e.g. Arduino) is connecting to. This port name may vary depending on your operating system and device

-On Linux, the serial port is usually named /dev/ttyACM0, /dev/ttyUSB0, etc. On Windows, the serial port is usually named COM1, COM2, COM3, etc. On macOS, the serial port is usually named /dev/tty.usbmodem1411, /dev/tty.usbserial-1420, v.v.

-9600: This is the baud rate, i.e. the speed at which data is transmitted over the serial connection. This rate must match the baud rate set on your device (e.g. Arduino). In this case, the baud rate is 9600 bits per second.

4.5 Communicating L298N DC Motor Control Module with Arduino

PWM (Pulse Width Modulation) - Adjusts the speed of the motor

PWM (Pulse Width Modulation) is a method of controlling the speed of a DC motor. In PWM, the signal is generated by varying the width of pulses at a specified frequency. The working principle of PWM is to generate a series of pulses with different widths in each cycle. The width of the pulse (the duration of time the pulse is on) is adjusted to control the power level or the corresponding analog signal. This power level is adjusted by changing the ratio between the time the pulse is on and the time the pulse is off in a cycle.

When using Arduino, we can use the PWM function on some digital pins to control the pulse width and generate an analog signal. The Arduino analogWrite() function is designed to generate PWM signals on PWM-enabled pins.

Pinout of L298N DC Motor Control Module

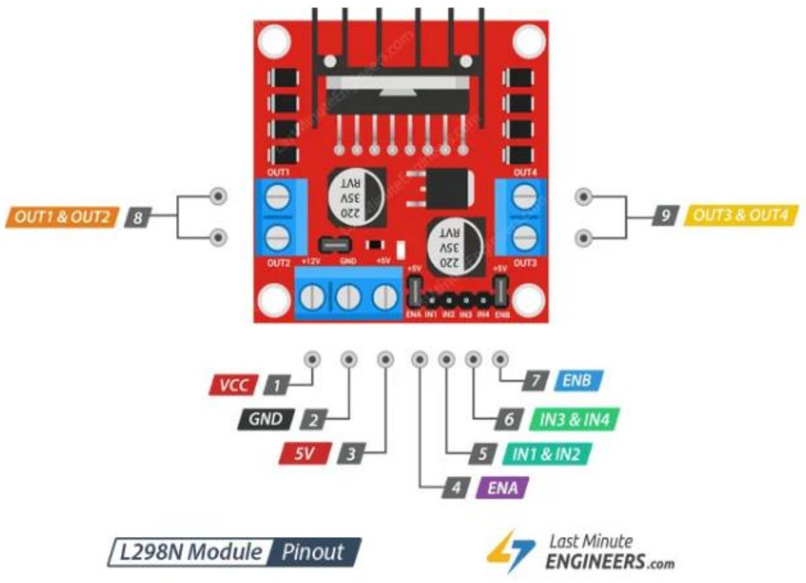


Figure 4.5: Pinout of L298N DC Motor Control Module

Motor power supply pins (Power Pins)

L298N DC motor driver module with Arduino has 2 input power pins VS and VSS

-SV: This pin is used to supply power to the H-bridge circuit inside the IC. The input voltage ranges from 5V to 12V.

-VCC: This pin is used to supply power to the logic circuit inside the L298N IC, with a voltage ranging from 5V to 7V.

-GND: This pin is used to ground.

Output pins

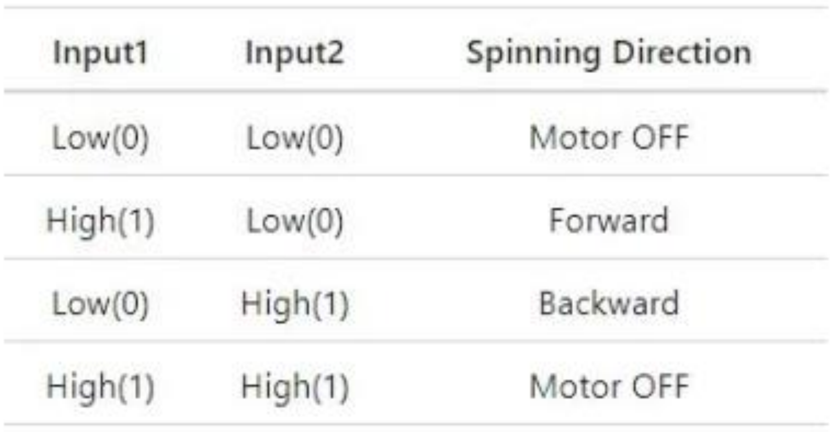
The output channels of the L298N motor driver module when connected to Arduino are divided as follows: OUT1 and OUT2 are for motor A, while OUT3 and OUT4 are used for motor B. This module supports controlling two DC motors with operating voltage from 5V to 12V.

Direction Control Pins

The pins on the L298N motor driver module allow you to adjust the direction of the motor's movement, such as forward or reverse. The principle of operation of this function is based on controlling the switches in the H-bridge circuit of the L298N IC, which helps change the rotation direction of the motor.

The L298N motor control module when connected to Arduino has two pairs of pins used to adjust the direction of movement of the motor. Pins IN1 and IN2 are used to control the direction of movement of motor A, while pins IN3 and IN4 adjust the direction of movement of motor B.

The direction of motor movement can be adjusted by applying a logic HIGH (5V) or logic LOW (GND) signal. The table below shows the different directions of movement based on the input pairs combined together.



Input1	Input2	Spinning Direction
Low(0)	Low(0)	Motor OFF
High(1)	Low(0)	Forward
Low(0)	High(1)	Backward
High(1)	High(1)	Motor OFF

Figure 4.6: Motor control logic board

Speed Control Pins

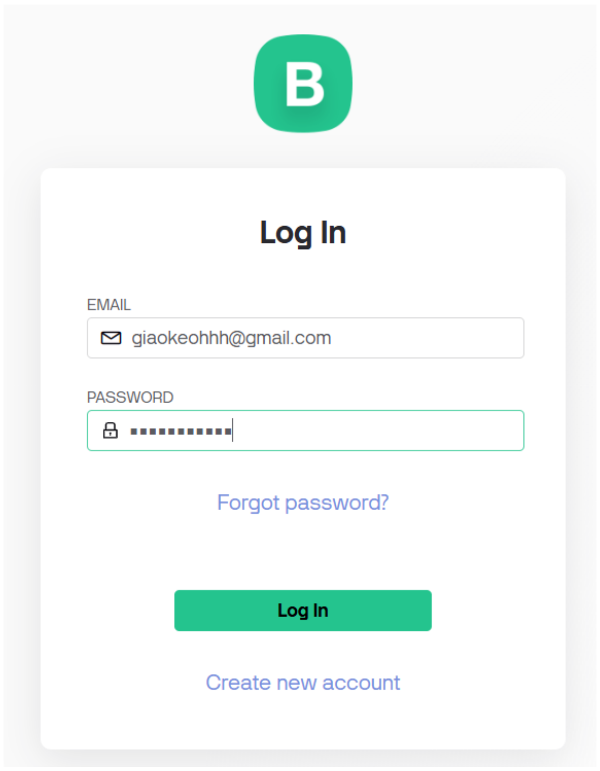
The ENA and ENB speed control pins are used to turn the motors on/off and control their rotation speed.

When the ENA and ENB pins are set to HIGH, the motors will rotate, while LOW will stop the motors. Additionally, you can adjust the rotation speed of the motors using PWM (Pulse Width Modulation).

There is a Jumper included on the L298N Arduino module. When this Jumper is plugged in, the motors will rotate at their maximum speed. If you want to control the speed of the motors through the program, you need to remove the Jumper and connect the Arduino's PWM-enabled pins to the module.

4.6 Create Blynk to control the system

Step 1: Create a Blynk account and log in

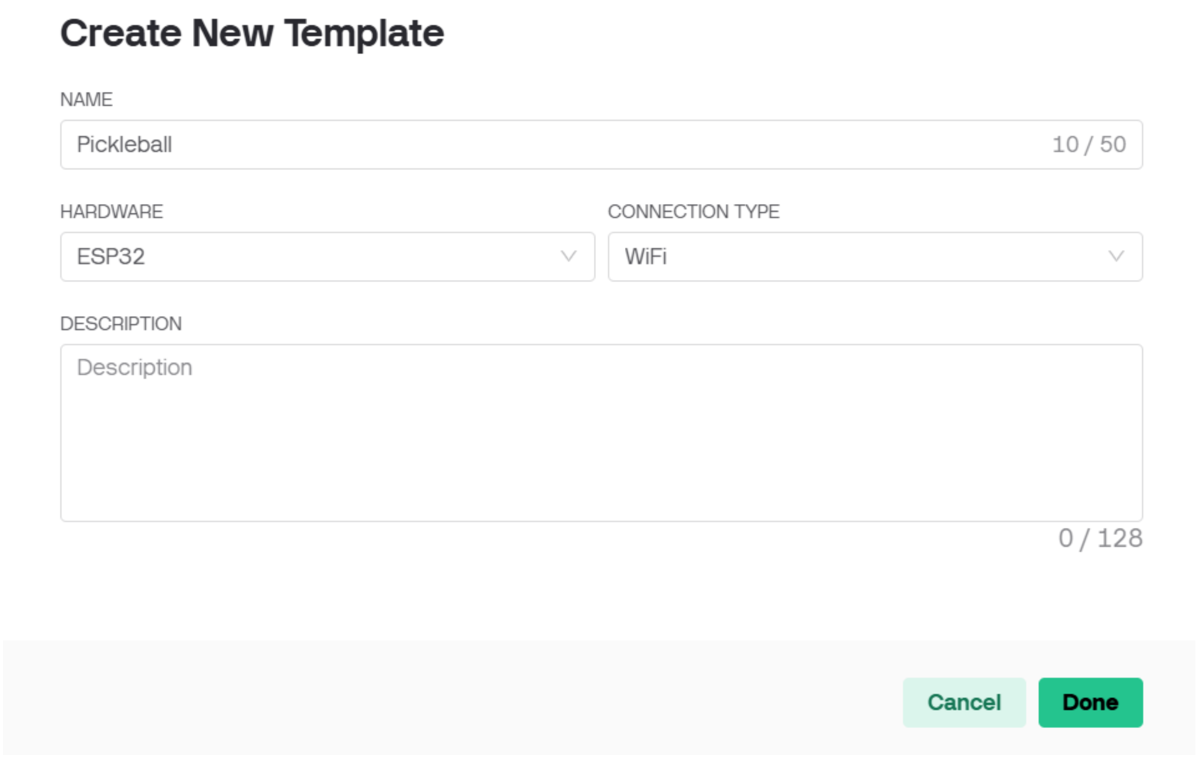

Log In

EMAIL: giaokeohhh@gmail.com

PASSWORD: ••••••••••••

Figure 4.7: Create a Blynk account

Step 2: Create new template


Create New Template

NAME: Pickleball 10 / 50

HARDWARE: ESP32

CONNECTION TYPE: WiFi

DESCRIPTION: Description 0 / 128

Figure 4.8: Create new template

Step 3: Create New Datastream => Virtual Pin

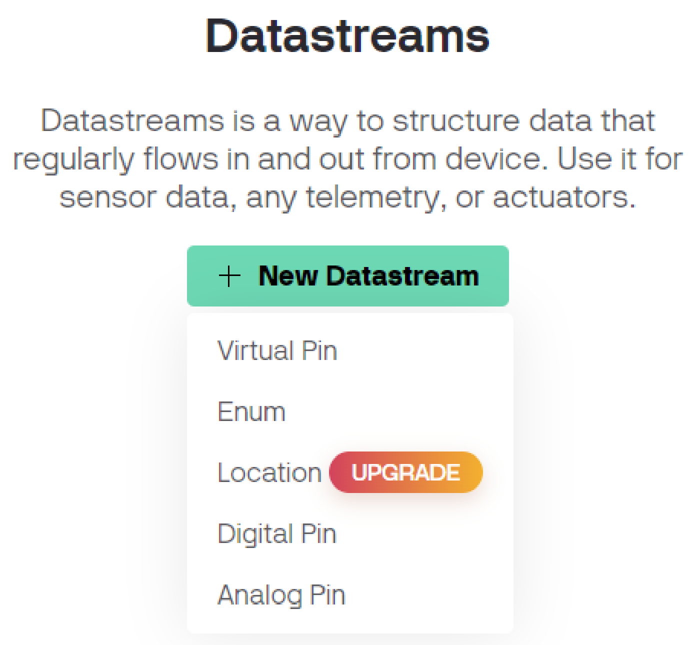
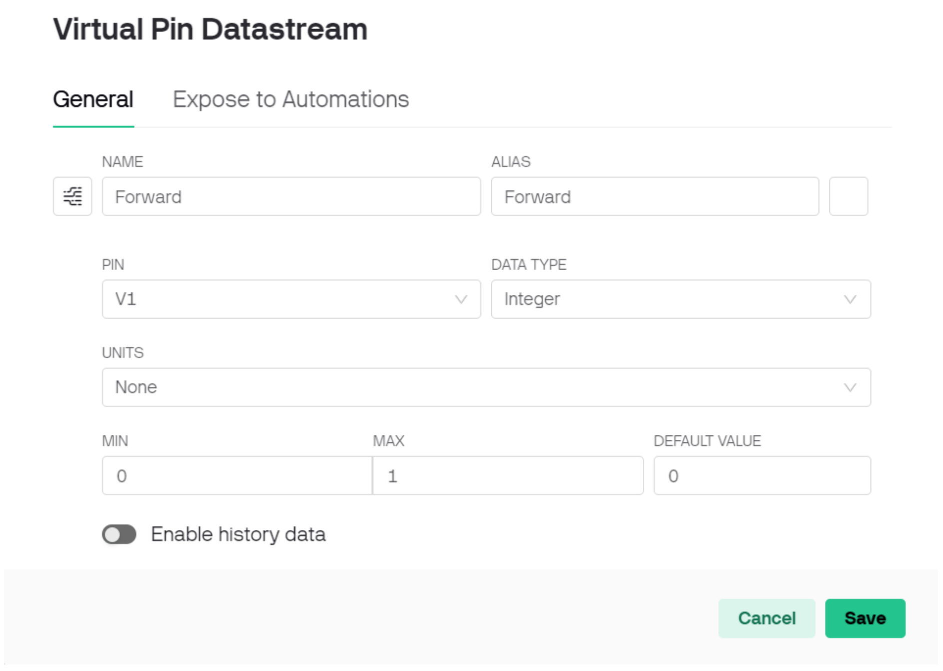

Datastreams
Datastreams is a way to structure data that regularly flows in and out from device. Use it for sensor data, any telemetry, or actuators.
+ New Datastream
Virtual Pin
Enum
Location
UPGRADE
Digital Pin
Analog Pin

Figure 4.9: Create New Datastream

Step 4: Create Virtual Pin for Forward



Virtual Pin Datastream

General | Expose to Automations

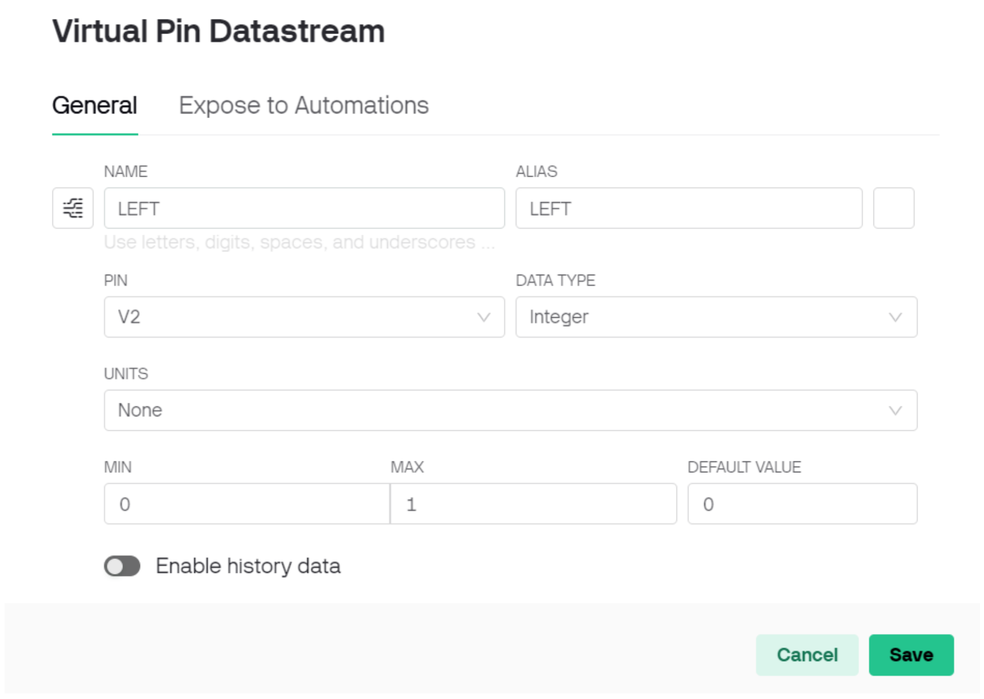
NAME	Forward
ALIAS	Forward
PIN	V1
DATA TYPE	Integer
UNITS	None
MIN	0
MAX	1
DEFAULT VALUE	0

Enable history data

Cancel | Save

Figure 4.10: Create Virtual Pin for Forward

Step 5: Create Virtual Pin for Left

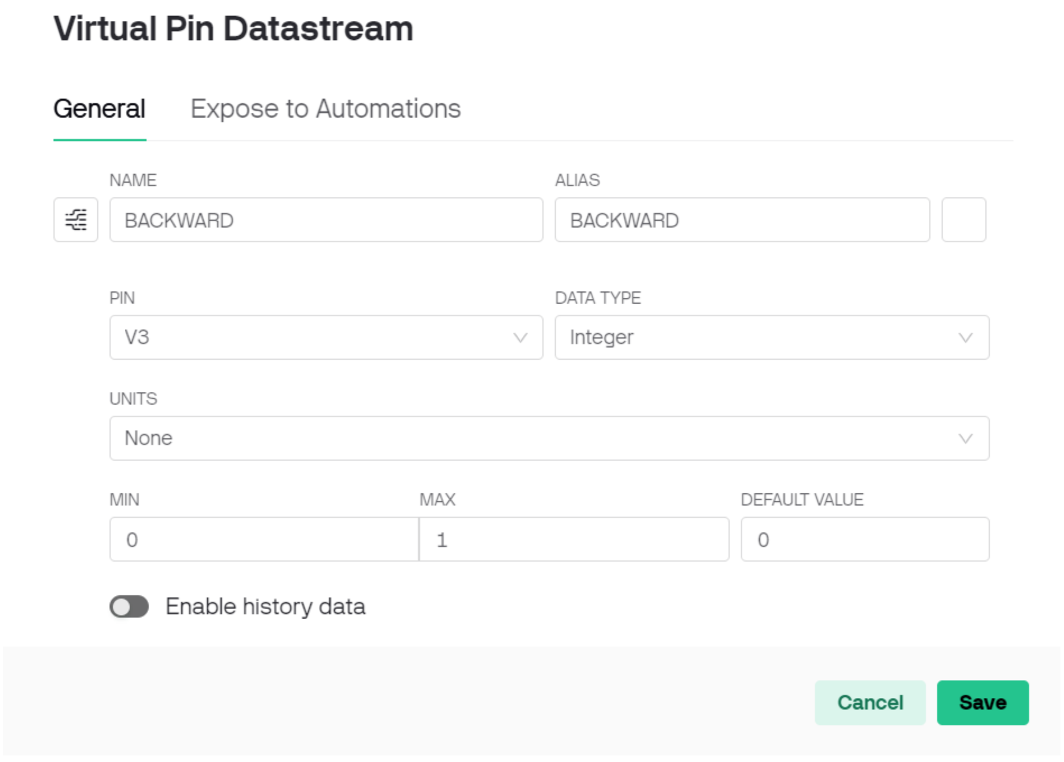

Virtual Pin Datastream

General | Expose to Automations

NAME: LEFT
ALIAS: LEFT
Use letters, digits, spaces, and underscores ...
PIN: V2
DATA TYPE: Integer
UNITS: None
MIN: 0
MAX: 1
DEFAULT VALUE: 0
Enable history data

Figure 4.11: Create Virtual Pin for Left

Step 6: Create Virtual Pin for Backward

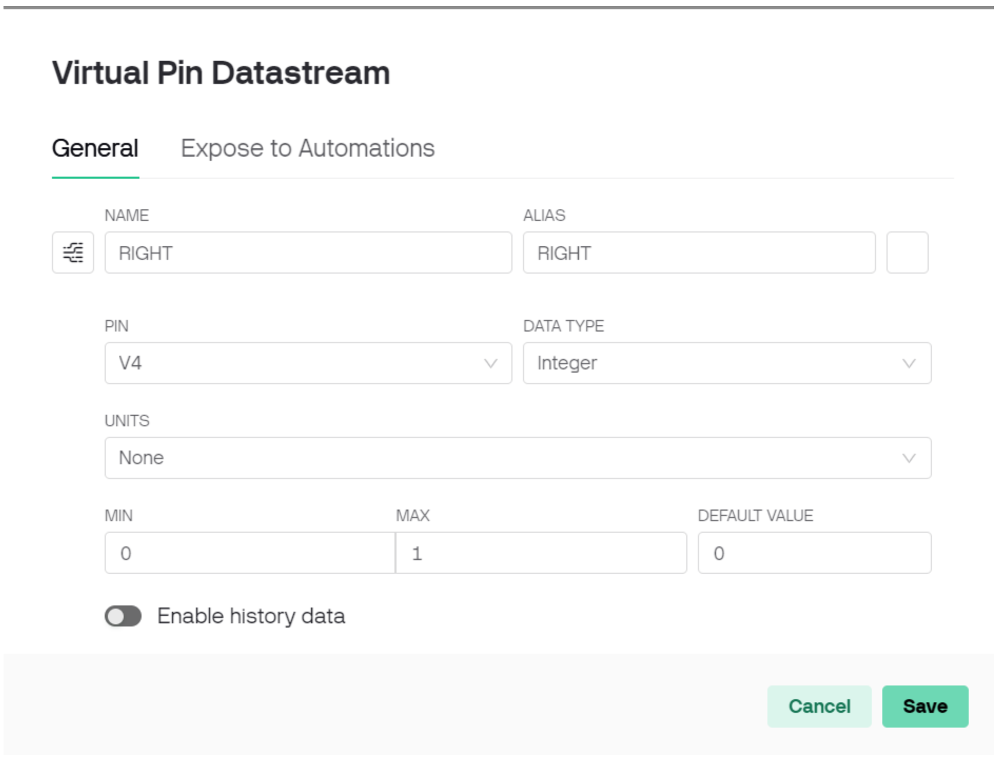

Virtual Pin Datastream

General | Expose to Automations

NAME: BACKWARD
ALIAS: BACKWARD
PIN: V3
DATA TYPE: Integer
UNITS: None
MIN: 0
MAX: 1
DEFAULT VALUE: 0
Enable history data

Figure 4.12: Create Virtual Pin for Backward

Step 7: Create Virtual Pin for Right

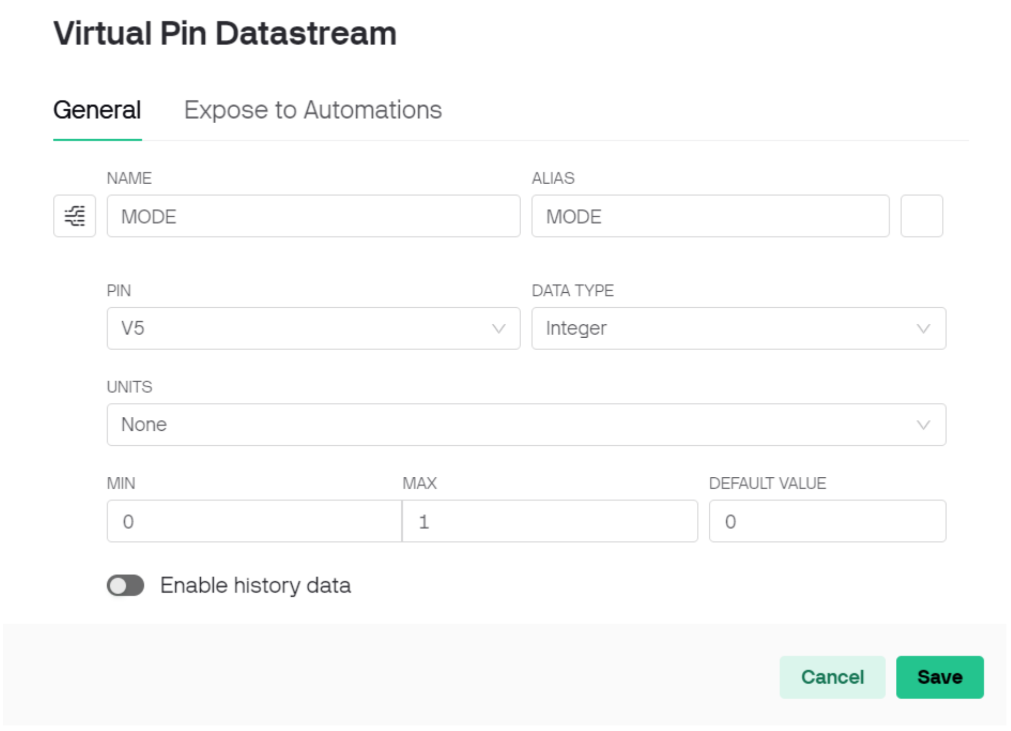

Virtual Pin Datastream

NAME	RIGHT
ALIAS	RIGHT
PIN	V4
DATA TYPE	Integer
UNITS	None
MIN	0
MAX	1
DEFAULT VALUE	0

Enable history data

Figure 4.13: Create Virtual Pin for Right

Step 8: Create virtual Pin for Mode


Virtual Pin Datastream

NAME	MODE
ALIAS	MODE
PIN	V5
DATA TYPE	Integer
UNITS	None
MIN	0
MAX	1
DEFAULT VALUE	0

Enable history data

Figure 4.14: Create virtual Pin for Mode

Step 9: Create Web Dashboard

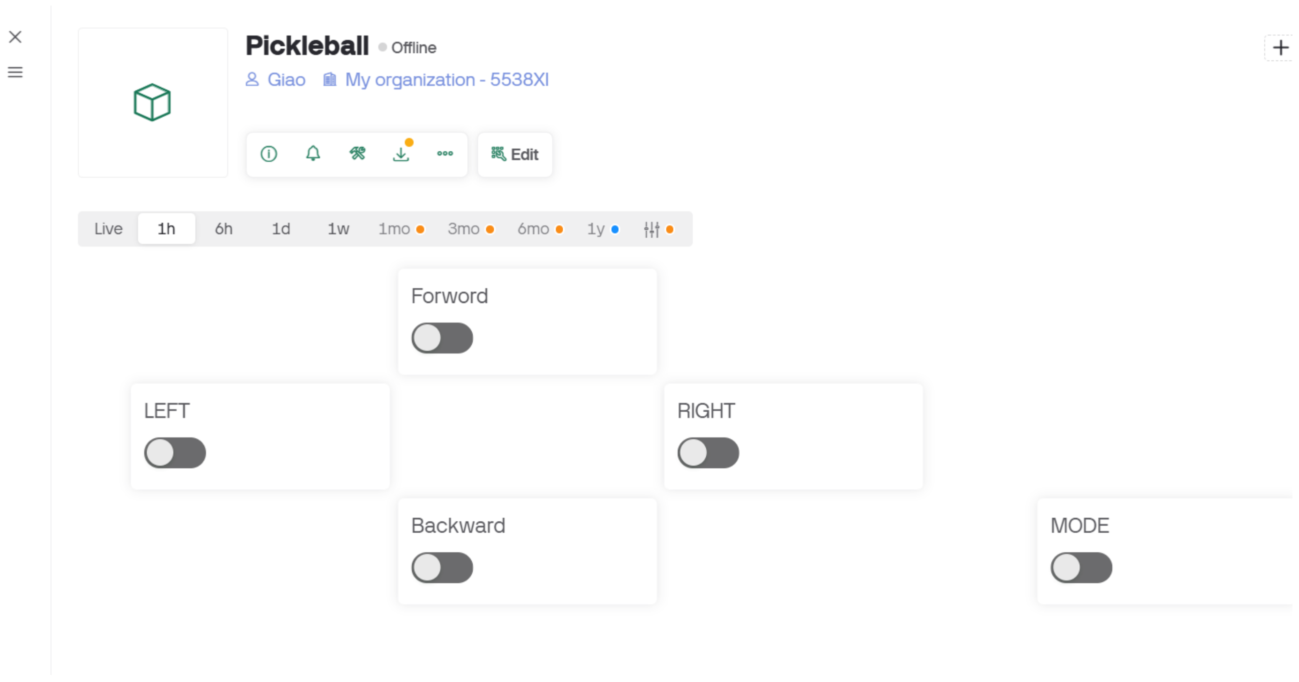
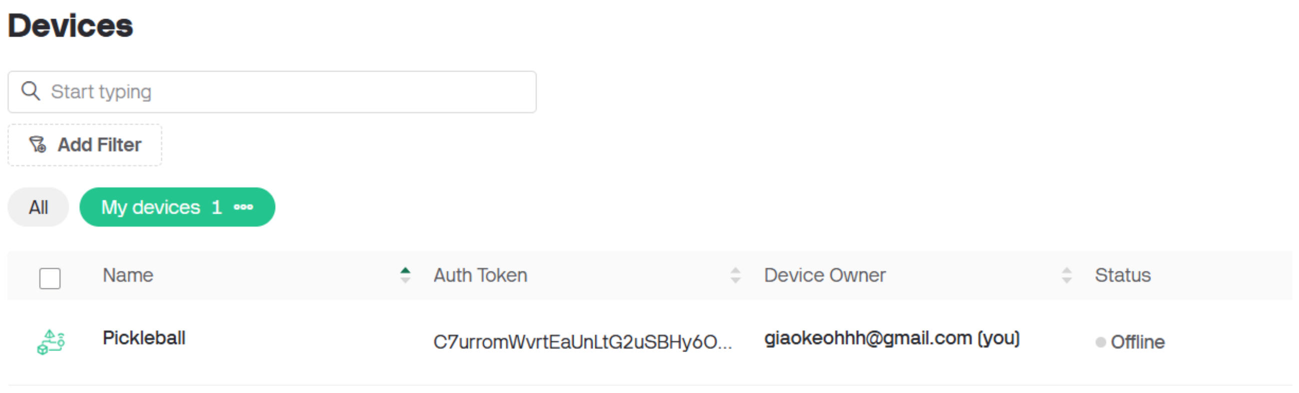


Figure 4.15: Create Web Dashboard

Step 10: Create New Device and Finish



	Name	Auth Token	Device Owner	Status
	Pickleball	C7urromWvrtEaUnLtG2uSBHy6O...	giaokeohhh@gmail.com (you)	Offline

Figure 4.16: Create New Device and Finish

Step 11: Get the Token code to assign to the code to communicate with Blynk in Raspberry Pi

CHAPTER 5: EXPERIMENT

5.1 Identifying the mid-position pickleball

When the pickleball is correctly identified at position 3,4,5, the motor will go straight and collect the ball

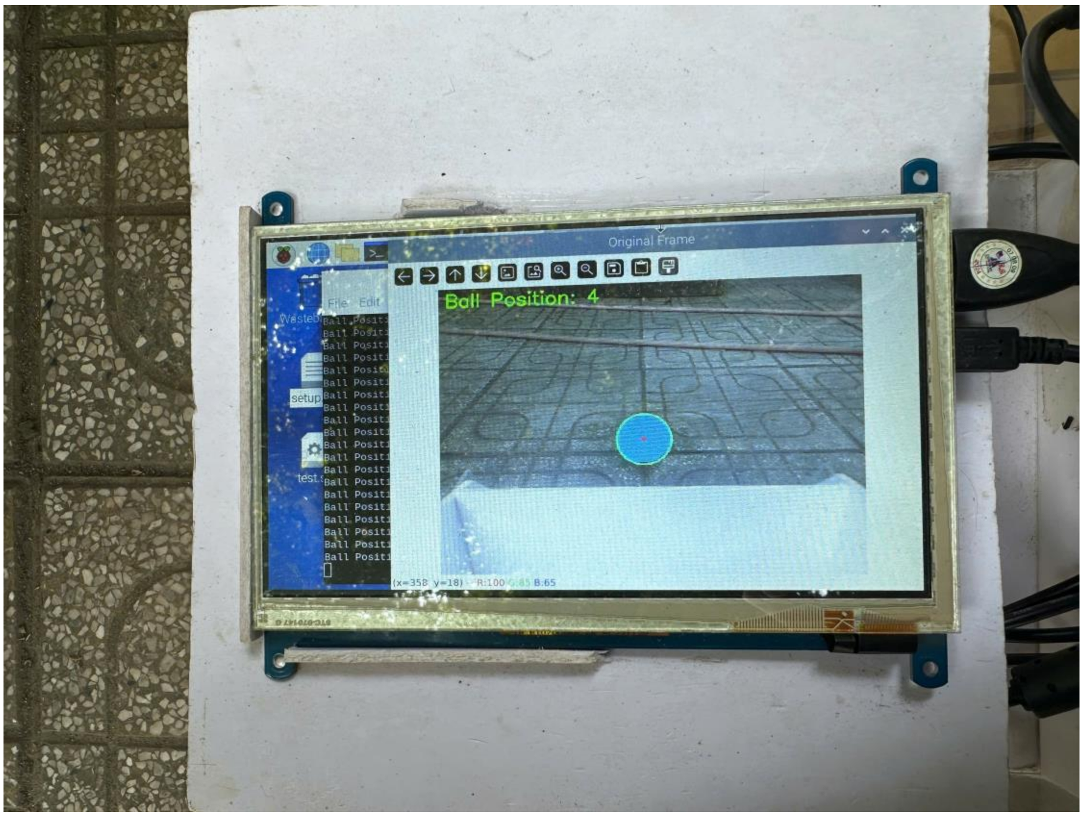


Figure 5.1: Pickleball recognition in the middle position

5.2 Identify the pickleball in the left position

When the pickleball is at position 1,2, it will be processed as being in the left position. After processing the position image from Raspberry Pi, it will transmit a signal to Arduino to control the motor to rotate to the left

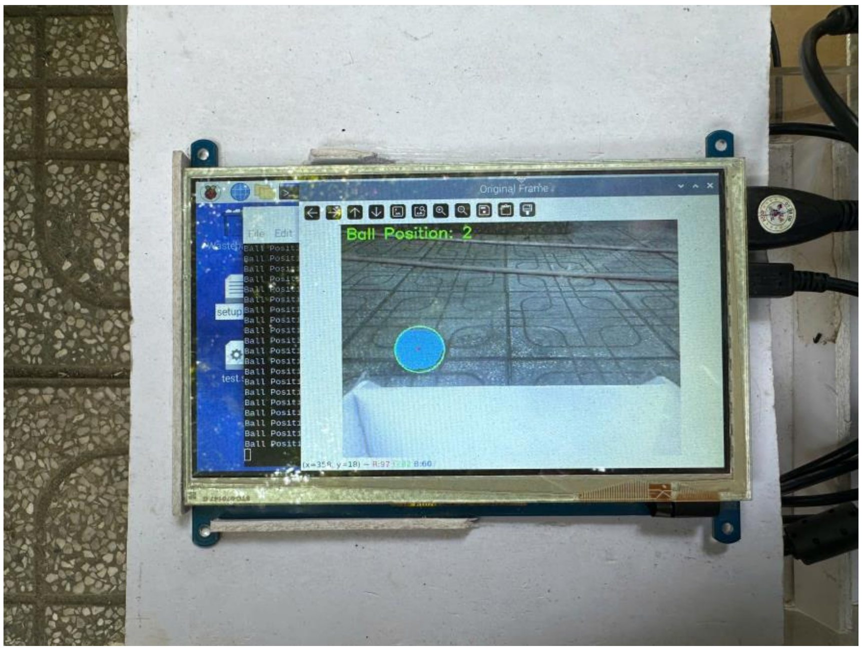


Figure 5.2: Identify the pickleball in the left position

5.3 Identifying the right-positioned pickleball

When the pickleball is at position 6,7, it will be processed as being in the right position. After processing the position image from Raspberry Pi, it will transmit a signal to Arduino to control the motor to rotate to the right

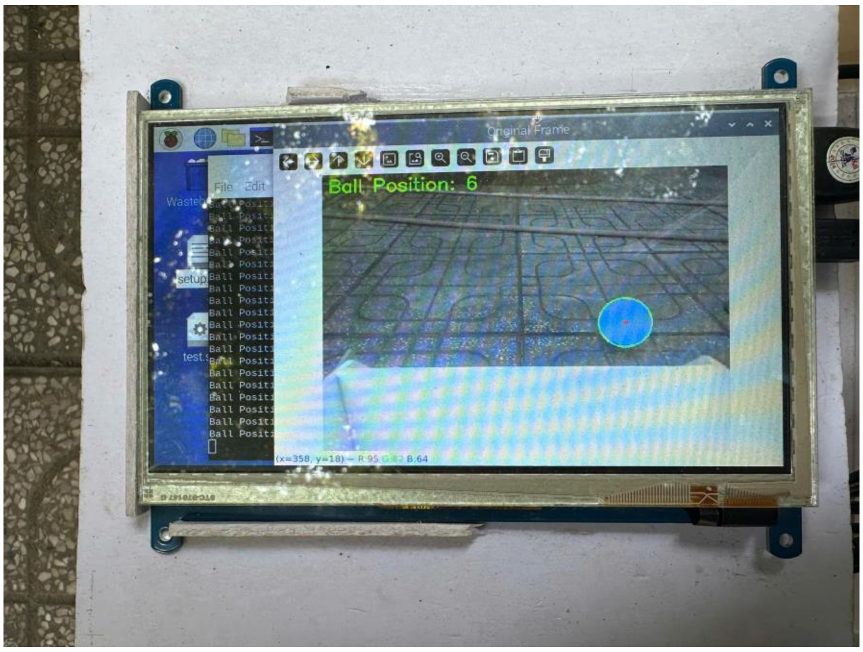


Figure 5.3: Identifying the right-positioned pickleball

5.4 Ball Recovery

The robot processes the pickleball image correctly and the motor runs exactly in the correct position to retrieve the ball as originally calculated

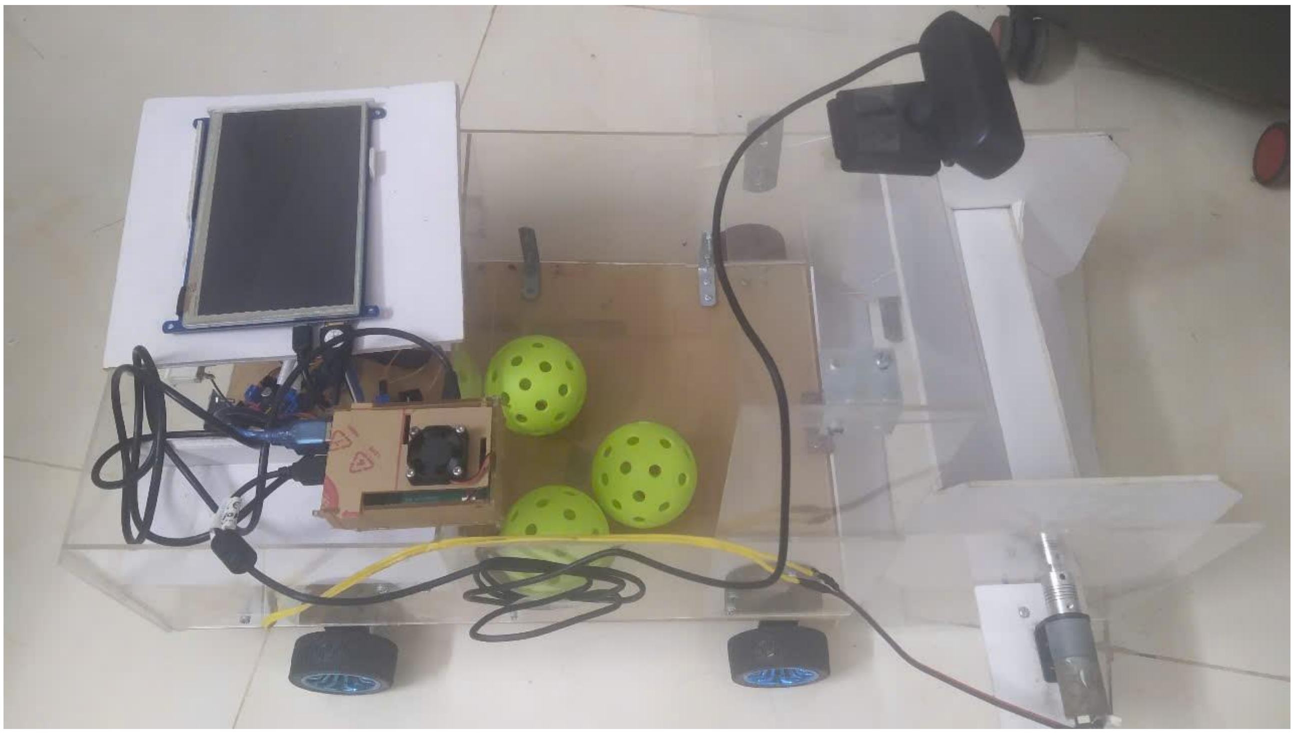


Figure 5.4: Ball Recovery

5.5 Experimental results

After testing in left, straight, and right positions, the robot can work quite well. With 10 tests, the robot has the following results:

Table 5.1: Experimental results

Time	1	2	3	4	5	6	7	8	9	10
Results	2/3	3/3	3/3	3/3	2/3	3/3	2/3	2/3	3/3	3/3

Conclusion: Testing with the pickleball in all 3 directions, the robot recovered the ball with a fairly high rate of 87%.

CHAPTER 6: CONCLUSION

6.1 Conclusion

-The model runs exactly as the topic requires

-Image processing correctly identifies the pickleball according to the problem requirements

-Pickleball retrieval is quite accurate with relative speed

-Precise manual mode control

6.2 Advantages

-The robot has a relatively moderate size to be able to hold many balls.

-The hardware is made of mica, lightweight cardboard and is cheap

-The processing speed is quite fast and relatively accurate

-Can be controlled via app from phone

6.3 Disadvantages

-Lack of calculation experience makes the robot unstable

-Operating time is not long-lasting

-There are also limitations in the camera's vision so sometimes the ball can be next to the side of the car without the car detecting it.

-The robot can collect balls in areas close to the wall, but it still cannot reach the corner areas to collect balls.

6.4 Difficulties in implementing the project

- Difficulty in choosing suitable materials and testing the ball collecting mechanism, when assembling the paddle wheel is not firm, leading to deformation or popping out of the paddle wheel while collecting the ball. The paddle wheel speed must be reasonable, if the speed is too high, the ball will not have time to go inside but will be continuously pushed out by the paddle wheel, if the speed is too low, there will not be enough force to push the ball in.
- The ball guide mechanism has problems that can cause the ball to get stuck when the paddle wheel is rotating.
- Some devices are purchased old, so they cannot be used well and it takes time to buy new devices.
- Difficulty in finding reference documents and available project information sources.

- When experimenting with moving, there are problems such as not being able to locate the ball position, the vehicle does not move in the desired direction in the code, there are many errors in programming and battery capacity problems, so the engine cannot operate strongly enough and long enough.

6.5 Development direction

-Find ways to optimize accuracy, ball detection and ball picking speed.

-Create a new ball collecting mechanism that can collect balls lying in the corner of walls or corners of strange objects.

-Can withstand impact when hit by a ball.

-Make a phone application that connects to the robot to monitor parameters such as battery, number of balls in the basket, and alert when there is a problem.

-Design to increase the size of the basket to hold more balls.

-Add ambient sensors to continuously avoid obstacles or when approaching walls.

-Compact design can be easily moved by just one person

REFERENCES

[1] Lê Cảnh Trung, P. Q. (2016). Lập trình điều khiển với ARDUINO. NXB Khoa học và kỹ thuật.

[2] Phạm Quang Huy, V. M. (2017). Lập trình điều khiển với Raspberry. NXB Thanh Niên.

[3] https://arduinokit.vn/giao-tiep-module-dieu-khien-dong-co-dc-l298n-voi-arduino

[4] Phạm Quang Huy, V. M. (2017). Xử lý ảnh với Arduino & Raspberry Pi NXB Thanh Niên.

[5] https://dientutuonglai.com/giao-tiep-uart-la-gi.html

[6] Vehbi Umur Çabuk, Ahmet Kubilay Şavkan, Ramazan Kahraman, Ferdi

Karaduman, Okan Kırıl , Volkan Sezer "Design and Control of a Tennis Ball

Collector Robot", 2018.

[7] Fromaget, P. (2020). Master your Raspberry Pi in 30 days.

[8] https://thuvienso.hcmute.edu.vn/tag/thiet-ke-va-che-tao-robot-nhat-bong-tennis-ung-dung-xu-ly-anh.html

[9] Asala Tibi,Hala Barqawi, An-Najah National University, Tennis Ball Collector, 2023.

APPENDIX

Code của Arduino

```
//-----L298N-----
#define enA 11
#define in1 8
#define in2 9
#define in3 12
#define in4 13
#define enB 10
#define en1A 3
#define in1_1 2
#define in1_2 4
bool flag = false;

unsigned long forwardDurationStartTime = 0;
bool lastForward = false;
char lastCommand = '0';

void DongCo() {
  analogWrite(en1A, 50);
  digitalWrite(in1_1, HIGH);
  digitalWrite(in1_2, LOW);
}

void Forward() {
  analogWrite(enA, 90);
  analogWrite(enB, 90);
  digitalWrite(in1, LOW);
  digitalWrite(in2, HIGH);
  digitalWrite(in3, HIGH);
  digitalWrite(in4, LOW);
}
```

```
void Back() {
  analogWrite(enA, 90);
  analogWrite(enB, 90);
  digitalWrite(in1, HIGH);
  digitalWrite(in2, LOW);
  digitalWrite(in3, LOW);
  digitalWrite(in4, HIGH);
}

void Left() {
  analogWrite(enA, 240);
  analogWrite(enB, 240);
  digitalWrite(in1, HIGH);
  digitalWrite(in2, LOW);
  digitalWrite(in3, HIGH);
  digitalWrite(in4, LOW);
}

void Right() {
  analogWrite(enA, 240);
  analogWrite(enB, 240);
  digitalWrite(in1, LOW);
  digitalWrite(in2, HIGH);
  digitalWrite(in3, LOW);
  digitalWrite(in4, HIGH);
}
void Stop() {
  digitalWrite(in1, LOW);
  digitalWrite(in2, LOW);
  digitalWrite(in3, LOW);
  digitalWrite(in4, LOW);
}
```

```
void setup() {
  Serial.begin(9600);
  pinMode(in1, OUTPUT);
  pinMode(in2, OUTPUT);
  pinMode(in3, OUTPUT);
  pinMode(in4, OUTPUT);
  pinMode(enA, OUTPUT);
  pinMode(enB, OUTPUT);
  pinMode(en1A, OUTPUT);
  pinMode(in1_1, OUTPUT);
  pinMode(in1_2, OUTPUT);
  DongCo();
  countTime = millis();
}

void loop()
{
  unsigned long currentMillis = millis();
  if (Serial.available() > 0)  {
    char receivedChar = Serial.read();
    Serial.println(receivedChar);
    if(receivedChar == 'A') //
    {
      flag = false;
    }
    else if(receivedChar == 'M')
    {
      flag = true;
    }
    if(!flag)
    {
```

```
  if(receivedChar == '1' || receivedChar == '2' )
  {
   if(lastForward)
   {
    if (currentMillis - forwardDurationStartTime >= 800 || forwardDurationStartTime ==
0)
    {
     forwardDurationStartTime = millis();
     Forward();
     lastForward = false;
    }
   }
   else //       {
    forwardDurationStartTime = millis();
    Left();
   }
  }
  else if (receivedChar == '4' || receivedChar == '3' || receivedChar == '5')
  {
   Forward();
   lastForward = true;
  }
  else if (receivedChar == '6' || receivedChar == '7'|| receivedChar == '0')
  {
   if (lastForward)
   {
    if (currentMillis - forwardDurationStartTime >= 800 || forwardDurationStartTime ==
0)
    {
     forwardDurationStartTime = millis();
     Forward();
     lastForward = false;
```

```
      }
    }
    else
    {
      forwardDurationStartTime = millis();
      Right();
    }
  }
}
else if(flag)
{
  if(receivedChar == 'F')
  {
    Forward();
  }
  else if(receivedChar == 'L')
  {
    Left();
  }
  else if(receivedChar == 'B')
  {
    Back();
  }
  else if(receivedChar == 'R')
  {
    Right();
  }
  else if(receivedChar == 'S')
  {
    Stop();
  }
}
```

```
  }
}
```

Code xử lý ảnh

```
import cv2
import numpy as np
import threading
from blynk_function import *
import serial
import time
try:
    cap = cv2.VideoCapture(0)
except:
    cap = cv2.VideoCapture(1)
greenlow = (30, 70, 90)
greenup = (45, 140, 255)
stop_event = threading.Event()
position = '0'


# Define positions
positions = {
    "1": (0, 1),#LEFT3
    "2": (1, 2),#Left2
    "3": (2, 3),#Left1
    "4": (3, 4),#Forward
    "5": (4, 5),#RIGHT1
    "6": (5, 6),#RIGHT2
    "7": (6, 7)#RIGHT3
}

def extract_green(frame, greenlow, greenup):
    blur = cv2.GaussianBlur(frame, (21, 21), 0)
    hsv = cv2.cvtColor(blur, cv2.COLOR_BGR2HSV)
    mask = cv2.inRange(hsv, greenlow, greenup)
    mask = cv2.dilate(mask, None, iterations=4)
    return mask

def contour_ext(mask, img):
    conts, _ = cv2.findContours(mask.copy(), cv2.RETR_EXTERNAL, 
cv2.CHAIN_APPROX_SIMPLE)
    center = None
    position = None
    if len(conts) > 0:
        c = max(conts, key=cv2.contourArea)
        perimeter = cv2.arcLength(c, True)
        approx = cv2.approxPolyDP(c, .03 * cv2.arcLength(c, True), True)
        area = cv2.contourArea(c)
        cv2.imshow('Disp Frame', mask)
        #print(len(approx))
        if len(approx)>1 and area / (perimeter * perimeter) > 0.05:
            #cv2.drawContours(img, [c], 0, (220, 152, 91), -1)
            ((x, y), radius) = cv2.minEnclosingCircle(c)
            M = cv2.moments(c)
            center = (int(M["m10"] / M["m00"]), int(M["m01"] / M["m00"]))
            if radius > 20:
                cv2.circle(img, (int(x), int(y)), int(radius), (126, 255, 60), 2)
                cv2.circle(img, center, 2, (75, 54, 255), 2)
                position = get_position(center, img.shape[1])
    return img, position

def get_position(center, img_width):
    x_center = center[0]
    segment_width = img_width / 7
    segment_id = int(x_center // segment_width) + 1
    position = '0'
    for key, value in positions.items():
        if segment_id in value:
            position = key
            break
    return position



def thread_function_1():
    global position, controlMode
    while True:
            _, frame = cap.read()
        frame=cv2.resize(frame,(640,480))
        frame_height, frame_width, _ = frame.shape

        for i in range(1, 7):
            x_coordinate = frame_width // 7 * i
            cv2.line(frame, (x_coordinate, 0), (x_coordinate, frame_height), (255, 255, 255), 1,
lineType=cv2.LINE_AA)
        disp = extract_green(frame, greenlow, greenup)
        frame, position = contour_ext(disp, frame)
        cv2.putText(frame, f'Ball Position: {position}', (10, 30),
cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 255, 0), 2, cv2.LINE_AA)
        cv2.imshow('Original Frame', frame)
        if cv2.waitKey(20) & 0xFF == ord('q'):
            break
    cap.release()
    cv2.destroyAllWindows()
def send_data(position1):
    global controlMode
    if position1 == None:
        position1 = '0'
    if not controlMode:
        ser.write(position1.encode())
        time.sleep(0.5)

def thread_function_2():
    global controlMode
    while True: #not stop_event.is_set() and
        send_data(position)


        print(f"Ball Position: {position}")

if __name__ == "__main__":
    thread1 = threading.Thread(target=thread_function_1)
    thread2 = threading.Thread(target=thread_function_2)
    thread3 = threading.Thread(target=blynk_activate)
    thread1.start()
    thread2.start()
    thread3.start()
```

Code xử lý ảnh chọn vùng màu của banh

```
import cv2
import numpy as np

def get_hsv_value(event, x, y, flags, param):
    if event == cv2.EVENT_LBUTTONDOWN:  # Check if the left mouse button was clicked
        hsv_value = hsv_frame[y, x]
        print(f"HSV Value at ({x}, {y}): {hsv_value}")
# Initialize webcam capture
cap = cv2.VideoCapture(0)

cv2.namedWindow("Frame")
cv2.setMouseCallback("Frame", get_hsv_value)

while True:
    _, frame = cap.read()
    hsv_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)

    # Display the frame
    cv2.imshow("Frame", frame)

    key = cv2.waitKey(1)
    if key == 27:  # Esc key to break
        break

# Release the webcam and close windows
cap.release()
cv2.destroyAllWindows()
```

Code giao tiếp Blynk với Raspberry Pi

```
import BlynkLib
import serial
import time
```

```
BLYNK_AUTH_TOKEN = 'fv1QDFbmcSUs9KNC2Jm9npL75On4p04k'
blynk = BlynkLib.Blynk(BLYNK_AUTH_TOKEN)
controlMode = 0 is_enable = 0
my_variable = 0
forward = 0
left = 0
right = 0
backward = 0
#from TennisDetection import thread_function_1, thread_function_2
try:
   ser = serial.Serial('/dev/ttyACM0',9600)
   #ser = serial.Serial('/dev/ttyUSB0',9600)
except:
   ser = serial.Serial('/dev/ttyACM1',9600)
   #ser = serial.Serial('/dev/ttyUSB1',9600)
#FORWARD
@blynk.on("V1")
def v1_write_handler(value):
     global is_enable, forward
     if is_enable:
        if(value[0] == '1'):
                blynk.virtual_write(2, 0)
                blynk.virtual_write(3, 0)
                blynk.virtual_write(4, 0)
                forward = 'F'
        else:
                forwar = 'S'
        ser.write(forward.encode())
        print(f'FORWARD:{forward}')
#TURN LEFT
@blynk.on("V2")
def v2_write_handler(value):
        global is_enable, left
        if is_enable:
            if(value[0] == '1'):
                blynk.virtual_write(1, 0)
                blynk.virtual_write(3, 0)
                blynk.virtual_write(4, 0)
                left ='L'
            else:
                left = 'S'
            ser.write(left.encode())
            print(f'LEFT:{left}')
#BACKWARD
@blynk.on("V3")
def v3_write_handler(value):
        global is_enable, backward
        if is_enable:
            if(value[0] == '1'):
                blynk.virtual_write(1, 0)
                blynk.virtual_write(2, 0)
                blynk.virtual_write(4, 0)
                backward = 'B'
            else:
                backward = 'S'
            ser.write(backward.encode())
            print(f'BACKWARD:{backward}')
#TURN RIGHT
@blynk.on("V4")
def v4_write_handler(value):
        global is_enable, right
        if is_enable:
            if(value[0] == '1'):
                blynk.virtual_write(1, 0)
                blynk.virtual_write(3, 0)
                blynk.virtual_write(2, 0)
                right = 'R'
        else:
                right = 'S'
        ser.write(right.encode())
        print(f'RIGHT:{right}')
#MODE

@blynk.on("V5")
def v5_write_handler(value):
     global controlMode, is_enable
     controlMode = value[0]
     print(f'controlMode: {controlMode}')
     if controlMode != '0' and controlMode != 0:
        blynk.virtual_write(1, 0)
        blynk.virtual_write(2, 0)
        blynk.virtual_write(3, 0)
        blynk.virtual_write(4, 0)
        ser.write('M'.encode())
        is_enable = 1 # Bat mode manual
        print('Mode: Manual')
     else:
        blynk.virtual_write(1, 0)
        blynk.virtual_write(2, 0)
        blynk.virtual_write(3, 0)
        blynk.virtual_write(4, 0)
        is_enable = 0# Bat mode auto
        ser.write('A'.encode())
        print(controlMode)
#CHECK_CONNECTION
@blynk.on("connected")
```

```
def blynk_connected():
    print("Raspberry Pi Connected to New Blynk")
def blynk_activate():
    while True:
        blynk.run()
        time.sleep(0.5)
```