**Project report on**

**Empowering Robot Vision with a Raspberry Pi-based system for Real -Time Ball Tracking**

Submitted by

**Aryan Mishra (2241013129)**

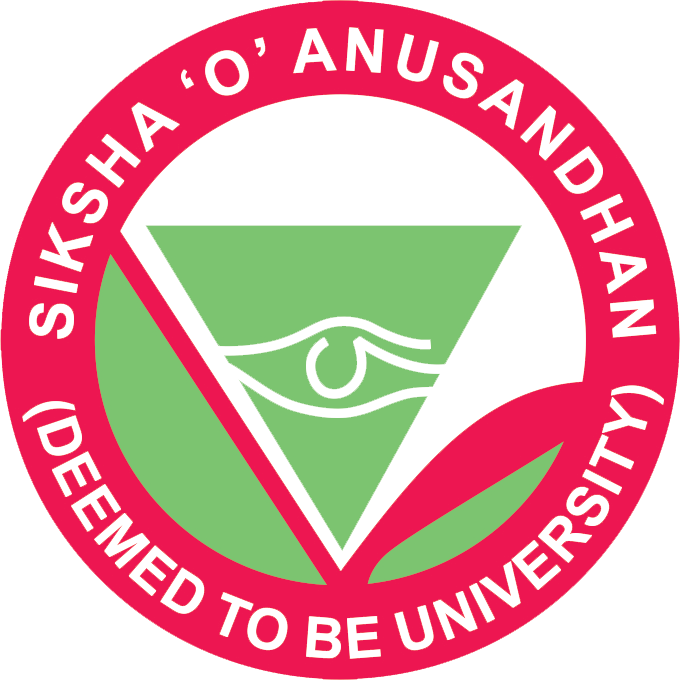
**Gourav Kumar Padhy (2241016518)**

**Pritik Kumar Sha (2241019385)**

**Bade Naveen Kumar (2241011121)**

**Rohan Prakash Nayak (2241014117)**

**B. Tech. (CSE(IOT) 4th Semester (Section–2241009)**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

Institute of Technical Education and Research

**SIKSHA ‘O’ ANUSANDHAN**

**DEEMED TO BE UNIVERSITY**

Bhubaneswar, Odisha, India.

(May, 2024)

# Abstract

This project endeavors to pioneer the development of an innovative ball-tracking robot leveraging Raspberry Pi technology. Integrating Robotics and Intelligent Color Tracking, our objective is to engineer a sophisticated system adept at dynamically tracking and pursuing a moving ball. Inspired by the nuanced maneuvers exhibited by football-playing robots, our endeavor aims to achieve a high degree of precision and adaptability in ball tracking. Through meticulous design and implementation, this endeavor seeks to push the boundaries of robotics, offering potential applications in sports training, entertainment, and beyond.

# Contents

[Abstract i](#_Toc166580400)

[Contents ii](#_Toc166580401)

[Chapter 01: Introduction 1](#_Toc166580402)

[1.1. Introduction 1](#_Toc166580403)

[1.2. Background 1](#_Toc166580404)

[1.3. Project Objectives 1](#_Toc166580405)

[1.4. Scope 1](#_Toc166580406)

[ Development of a single ball-tracking robot prototype. 1](#_Toc166580407)

[ Hardware integration of Raspberry Pi, camera module, motor driver, and motors. 1](#_Toc166580408)

[ Software development using the Processing IDE for ARM architecture. 1](#_Toc166580409)

[ Implementation of image processing algorithms for ball detection and tracking. 1](#_Toc166580410)

[ Testing the robot's functionality and performance under varying conditions. 1](#_Toc166580411)

[1.5. Project Management 1](#_Toc166580412)

[1.6. Overview and Benefits 2](#_Toc166580413)

[1.7. Organization of the Report 2](#_Toc166580414)

[Chapter 02: Background Review & Survey 3](#_Toc166580415)

[2.1. Related Works 3](#_Toc166580418)

[Chapter 03: Theoretical Aspects 5](#_Toc166580419)

[3.1. Ball Tracking Robot 5](#_Toc166580422)

[3.2. Features of Ball-Tracking Robot 5](#_Toc166580423)

[3.3. Advantages of Ball-Tracking Technology 5](#_Toc166580424)

[3.4. Disadvantages 5](#_Toc166580425)

[3.5. Application areas of IoT 5](#_Toc166580426)

[3.6. Project Layout 6](#_Toc166580427)

[3.6.1. Brief Description 6](#_Toc166580428)

[Chapter 04: Hardware Requirements 8](#_Toc166580429)

[4.1. Raspberry Pi Zero 2W 8](#_Toc166580432)

[4.1.1. Features 8](#_Toc166580433)

[4.1.2. Pin Configuration 8](#_Toc166580434)

[4.2. USB Camera 8](#_Toc166580435)

[4.3. Robot Chassis 8](#_Toc166580436)

[4.3.1. Acrylic Body 8](#_Toc166580437)

[4.3.2. Motors with Gears 8](#_Toc166580438)

[4.4. HW 095 Motor Driver 8](#_Toc166580439)

[4.5. Block diagram of the proposed system 8](#_Toc166580440)

[4.5.1. Working of the system 8](#_Toc166580441)

[4.5.2. Circuit Diagram 9](#_Toc166580442)

[4.5.3. Components Required 10](#_Toc166580443)

[Chapter 05: Software Requirements 11](#_Toc166580444)

[5.1. Processing IDE 11](#_Toc166580447)

[5.2. Logic and Flowchart 11](#_Toc166580448)

[Chapter 06: Project development & Testing Aspects 12](#_Toc166580449)

[6.1. Development Process 12](#_Toc166580452)

[Chapter 07: Conclusion & Future Scope 14](#_Toc166580453)

[7.1. Result 14](#_Toc166580456)

[7.2. Conclusion 14](#_Toc166580457)

[7.3. Limitations 14](#_Toc166580458)

[7.4. Further Enhancement and Future Scope 14](#_Toc166580459)

[References 15](#_Toc166580460)

[Appendix 01 16](#_Toc166580461)

[A01.1. Code Listing 16](#_Toc166580462)

[A01.2. Main Code 20](#_Toc166580463)

[A01.3. Libraries 20](#_Toc166580464)

[Appendix 02 21](#_Toc166580465)

[A02.1. Project Proposal Form 21](#_Toc166580466)

[A02.2. Project Management 21](#_Toc166580467)

[ 21](#_Toc166580468)

[A02.3. Bill of Material 21](#_Toc166580469)

[Appendix 03 22](#_Toc166580470)

[A03.1. Data Sheets 22](#_Toc166580471)

# Chapter 01: Introduction

## Introduction

The project is centered around the creation of an advanced ball-tracking robot using Raspberry Pi technology. By harnessing the synergies of Robotics, Intelligent Color Tracking, the objective is to develop a sophisticated system capable of dynamically tracking and following a moving ball, mirroring the intricate behavior observed in robots designed for playing football.

## Background

Robotics, Artificial Intelligence, and Machine Learning are rapidly evolving fields with the potential to transform human lifestyles. Robots interact with the real world through sensors and machine learning, with image recognition being a popular method for understanding objects. This project leverages Raspberry Pi technology to build a robot capable of tracking and following a ball, akin to football-playing robots.

## Project Objectives

* Develop a ball-tracking robot using Raspberry Pi technology.
* Utilize image processing techniques for ball detection and tracking.
* Integrate hardware components including Raspberry Pi, camera module, motor driver, and motors.
* Program the robot to dynamically adjust its movements to follow the ball's trajectory.
* Enable the robot to interact with its environment in real-time.

## Scope

## Development of a single ball-tracking robot prototype.

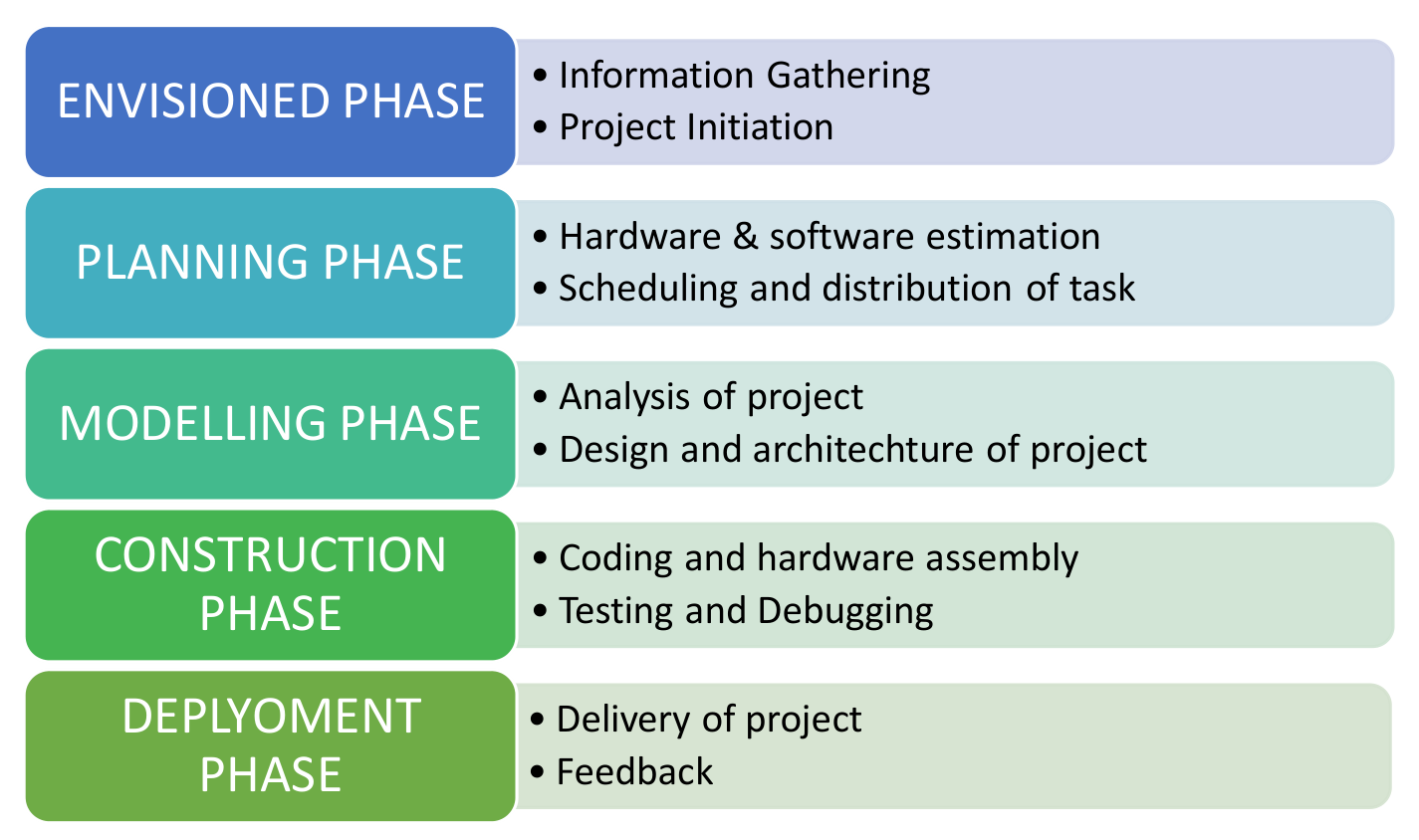
## Hardware integration of Raspberry Pi, camera module, motor driver, and motors.

## Software development using the Processing IDE for ARM architecture.

## Implementation of image processing algorithms for ball detection and tracking.

## Testing the robot's functionality and performance under varying conditions.

## Project Management



**Figure 1. Model of phases in project management.**

## Overview and Benefits

* The project aims to revolutionize human-robot interaction through intelligent automation.
* Benefits include enhanced precision, adaptability, versatility, innovation, and educational value.
* Potential applications range from sports training to interactive displays and beyond.

## Organization of the Report

The report is organised into the following chapters. Each chapter is unique on its own and is described with necessary theory to comprehend it.

Chapter 2 deals with background survey and review, Chapter 3 has the description of the theoretical aspects that has been acquired to commence the project work.

# Chapter 02: Background Review & Survey



## Related Works

Several projects and research initiatives have explored the intersection of robotics, image processing, and ball tracking. Some notable works include:

* OpenCV-Based Ball Tracking Systems:

Many projects have utilized OpenCV, an open-source computer vision library, for ball tracking. These systems often employ webcam or camera modules for real-time image capture and processing. Techniques such as color segmentation and object tracking algorithms are commonly employed to detect and follow the ball's movement.

* ROS-Based Robotics Projects:

The Robot Operating System (ROS) is widely used in robotics research and development. Numerous projects within the ROS community have focused on ball tracking and manipulation tasks. These projects often integrate ROS packages for camera control, image processing, and robot control, enabling complex behaviors such as ball tracking and interception.

* Academic Research in Sports Robotics:

Universities and research institutions have conducted studies on robotic systems designed for sports applications, particularly in soccer-playing robots. These projects explore advanced control algorithms, sensor fusion techniques, and strategy optimization for autonomous gameplay. Research findings from these endeavors contribute valuable insights to the broader field of robotics and intelligent systems.

* Commercial Ball Tracking Solutions:

In addition to academic and DIY projects, commercial solutions exist for ball tracking in sports and entertainment. These systems are often used in training facilities, broadcasting, and interactive installations. They incorporate high-speed cameras, machine learning algorithms, and specialized hardware for accurate and reliable ball tracking in diverse environments.

By reviewing and building upon these related works, this project aims to contribute to the advancement of ball-tracking robotics technology, particularly focusing on the integration of Raspberry Pi for accessible and versatile implementations.

# Chapter 03: Theoretical Aspects



## Ball Tracking Robot

The theoretical aspects of our project encompass various concepts and principles that underpin its design and functionality. In particular, our project leverages key theoretical foundations in the fields of robotics, image processing, and Internet of Things (IoT) technology. Here are some insights into how these theoretical aspects contribute to the development and operation of our ball-tracking robot:

## Features of Ball-Tracking Robot

1. Intelligent Color Tracking: Our robot utilizes sophisticated image processing algorithms to detect and track the color of the ball in real-time, enabling dynamic movement and precise control.
2. Dynamic Motor Control: By integrating Raspberry Pi technology and motor driver modules, the robot can adjust its movements based on the position of the tracked ball, demonstrating agility and responsiveness.

## Advantages of Ball-Tracking Technology

1. Enhanced Efficiency: The automated tracking and following capabilities of our robot streamline tasks such as object monitoring and surveillance, increasing operational efficiency and productivity.
2. Real-Time Interaction: With its ability to track moving objects in real-time, our robot opens up opportunities for interactive applications in entertainment, sports training, and educational settings

## Disadvantages

1. Privacy and Security Concerns: The use of camera-based tracking systems raises privacy considerations, requiring measures to ensure data protection and user privacy.
2. Environmental Limitations: Factors such as varying lighting conditions and background clutter can pose challenges to accurate color detection and tracking, necessitating robust algorithms and calibration procedures.

## Application areas of IoT

* Sports Training: Our ball-tracking robot has applications in sports training and coaching, where it can simulate gameplay scenarios and assist athletes in improving their skills.
* Surveillance and Monitoring: The robot's ability to track objects of interest makes it suitable for surveillance tasks in security and monitoring applications, such as tracking intruders or monitoring crowd movements.
* Education and Research: Our project serves as an educational tool for teaching concepts in robotics, image processing, and IoT technology, providing hands-on learning experiences for students and researchers.

## Project Layout

**Figure 2. Layout of project module**

### Brief Description

The project aims to develop a Raspberry Pi-based ball-tracking robot using Processing IDE, focusing on simplicity and accessibility. It utilizes Processing IDE for ARM architecture instead of Python, simplifying the development process. The robot comprises a Raspberry Pi, a camera module with a ribbon cable, a robot chassis, gear motors with wheels, an L293D motor driver, and a power bank or other portable power source.

The setup process involves connecting the Raspberry Pi to a monitor, keyboard, and mouse, ensuring an active internet connection for downloading necessary components. Processing ARM software is downloaded and extracted, and libraries for Raspberry Pi, GL Video, and Hardware I/O are installed within the Processing environment.

The circuit diagram involves connecting the Pi camera, motor driver module, and motors to the Raspberry Pi, powered by a mobile power bank. Pins GPIO14, 4, 17, and 18 control the motors' movements.

Programming in Processing involves initializing output pins for motor control, capturing video from the Pi camera, and processing each frame to detect the ball's color. A color is selected by clicking on the ball's image, and the program tracks this color by comparing it with each pixel's color in the video frame. Based on the color match, the robot adjusts its movements to keep the ball's position centered on the screen.

After setting up and testing the robot's functionality, it can be deployed for real-world applications. The project emphasizes simplicity, accessibility, and practicality in building a ball-tracking robot using Raspberry Pi and Processing IDE.

# Chapter 04: Hardware Requirements



## Raspberry Pi Zero 2W

### Features

* Improved Performance: Raspberry Pi Zero 2W features a BCM2710A1 die Cortex-A53 CPU and 512MB LPDDR2 DRAM, offering enhanced processing power for a variety of tasks.
* It also has a 802.11b/g/n wireless LAN and Bluetooth 4.2 for enhanced connectivity.

### Pin Configuration

GPIO pins 27, 14, 17, and 18 are used for motor control, and they are set as OUTPUT pins using the GPIO.pinMode() function. These pins would then be used to control the direction and speed of the motors connected to the HW 095 motor driver module.

## USB Camera

## Robot Chassis

### Acrylic Body

### Motors with Gears

## HW 095 Motor Driver

## Block diagram of the proposed system

### Working of the system

The system works by utilizing a Raspberry Pi-based setup along with a USB Camera and motor control to track and follow a moving ball in real-time. Here's an overview of how the system works:

Initialization:

The Raspberry Pi is connected to a monitor, keyboard, and mouse, and the Processing IDE environment is set up.

Necessary libraries for Raspberry Pi, Processing Video, and Hardware I/O are installed within the Processing environment.

The circuit is set up, connecting the Raspberry Pi with the USB Camera and motor driver module, powered by a portable power source.

Color Tracking:

The system captures video feed from the USB Camera in real-time using the Processing IDE.

The user selects the color of the ball by clicking on it, which is stored as the track color.

Each frame of the video feed is processed, and pixels are analyzed to detect the track color using color matching algorithms.

The position of the ball in the frame is determined based on the detected color, allowing the system to track its movement.

Motor Control:

The system adjusts the movement of the robot chassis based on the position of the ball.

If the ball moves to the left side of the frame, the motors are activated to turn the robot chassis to the left.

If the ball moves to the right side of the frame, the motors turn the robot chassis to the right.

The system continuously adjusts the robot's movement to keep the ball's position centered on the screen.

Deployment and Testing:

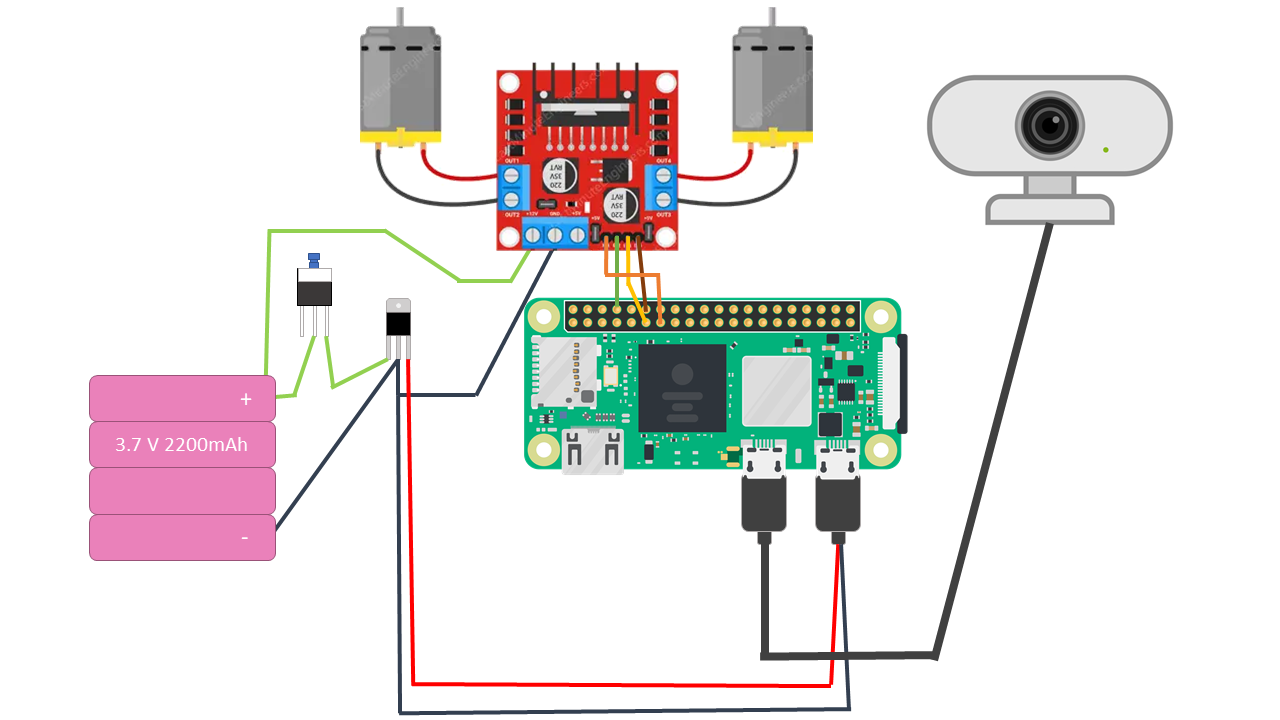
Once the system is set up and configured, it can be deployed for real-world applications.

The user can test the functionality of the ball-tracking robot by releasing it on the ground and observing its movement.

The robot adjusts its movement based on the ball's position, effectively tracking and following its movement in real-time.

Overall, the system leverages Raspberry Pi technology and image processing algorithms to create a ball-tracking robot capable of autonomously following a moving ball, demonstrating the integration of hardware and software components for practical applications.

### Circuit Diagram



### Components Required

**Table 1. Component listing.**

|  |  |  |
| --- | --- | --- |
| **Sl. No.** | **Component and Specification** | **Quantity** |
|  | Raspberry Pi Zero 2W | 1 |
|  | HW 095 Motor Driver | 1 |
|  | Batteries | 4 |
|  | 7805 IC | 1 |
|  | Robot Chassis | 1 |
|  | Motor with Gears | 2 |
|  | Jumper Wires | As per required |

# Chapter 05: Software Requirements



## Processing IDE

We used Processing 3.3 for our project. We installed Processing Video Library, and Hardware I/O Library for connecting to USB Camera and Raspberry Pi GPIO pins respectively

## D:\Projects\Ball_Tracking_Bot_Rasp_Pi\mermaid-diagram-2024-05-11-024903.pngLogic and Flowchart

# Chapter 06: Project development & Testing Aspects



## Development Process

The development process for the ball-tracking robot project involves several stages, including hardware setup, software development, integration, and testing. Here's an overview:

1. Hardware Setup:
   * Assemble the robot chassis, motors, motor driver module, Raspberry Pi, and camera module according to the provided specifications.
   * Connect the components and peripherals, ensuring proper wiring and power supply.
2. Software Installation:
   * Install the necessary software components, including the Processing IDE, GPIO library for Processing, and any additional libraries required for camera access and image processing.
   * Configure the Raspberry Pi and install the operating system (e.g., Raspbian) to support the software environment.
3. Code Development:
   * Develop the code in the Processing IDE to capture video from the camera module, process each frame to detect the ball's color, track its position, and control the motors accordingly.
   * Implement algorithms for color detection, object tracking, and motor control, ensuring efficient and accurate performance.
4. Integration:
   * Integrate the software code with the hardware components, ensuring proper communication and functionality between the Raspberry Pi, camera module, motor driver, and motors.
   * Test the integrated system to verify correct sensor readings, motor responses, and overall system behavior.
5. Calibration and Optimization:
   * Fine-tune the parameters and settings of the software algorithms, such as color thresholds, motor control logic, and camera settings, to optimize the performance of the ball-tracking system.
   * Conduct calibration tests to ensure accurate tracking and smooth motor movements under different lighting conditions and environments.
6. Testing:
   * Perform comprehensive testing of the ball-tracking robot in various scenarios, including different ball colors, speeds, and trajectories.
   * Evaluate the system's performance, accuracy, and responsiveness through rigorous testing and analysis.
   * Identify and debug any issues or errors encountered during testing, making necessary adjustments to the hardware or software as needed.
7. Documentation:
   * Document the development process, including hardware setup instructions, software code documentation, calibration procedures, and testing results.
   * Provide clear guidelines and troubleshooting tips for users to set up, operate, and maintain the ball-tracking robot effectively.
   * By following a systematic development process and conducting thorough testing, the project ensures the successful implementation of the ball-tracking robot with reliable performance and functionality.

# Chapter 07: Conclusion & Future Scope



## Result

The result of the ball-tracking robot project demonstrates the successful implementation of a Raspberry Pi-based system capable of dynamically tracking and following a moving ball. Through the integration of hardware components, software algorithms, and real-time image processing techniques, the robot effectively detects the ball's color, tracks its position, and adjusts motor movements to keep the ball centered on the screen.

## Conclusion

In conclusion, the development of the ball-tracking robot project highlights the potential of Raspberry Pi technology and Processing IDE for creating intelligent robotic systems. By harnessing the synergies of robotics, image processing, and embedded computing, the project achieves its objective of designing a sophisticated ball-tracking system inspired by robots used in sports applications.

## Limitations

Despite its success, the project has certain limitations:

* The system's performance may vary under different lighting conditions and environmental factors, affecting color detection and tracking accuracy.
* The hardware setup and software implementation may require technical expertise, limiting accessibility for inexperienced users.
* The robot's speed and agility may be limited by the capabilities of the motors and motor driver module used in the project.

## Further Enhancement and Future Scope

To address these limitations and enhance the project's capabilities, several avenues for further enhancement and future scope can be explored:

* Implementing advanced image processing algorithms, such as machine learning-based object detection and tracking, to improve the system's robustness and adaptability to diverse scenarios.
* Integrating additional sensors, such as ultrasonic or infrared sensors, to enhance the robot's obstacle avoidance capabilities and navigation in complex environments.
* Upgrading the hardware components, such as using higher-performance motors and motor driver modules, to increase the robot's speed, agility, and payload capacity.
* Developing a user-friendly graphical interface or mobile application for controlling and monitoring the robot remotely, enabling seamless interaction and operation by users of all skill levels.

By pursuing these enhancements and future developments, the ball-tracking robot project can evolve into a more versatile and capable robotic platform with broader applications in areas such as education, research, entertainment, and industrial automation.

# References

* https://circuitdigest.com/microcontroller-projects/raspberry-pi-ball-tracking-robot-using-processing
* https://chatgpt.com/
* https://www.google.co.in/
* https://www.raspberrypi.com/products/raspberry-pi-zero-2-w/

# Appendix 01

## A01.1. Code Listing

import processing.video.\*;

import processing.io.\*;

Capture video;

color trackColor;

void setup() {

size(320, 240, P2D);

String[] cameras = Capture.list();

if (cameras.length == 0) {

println("No cameras available");

exit();

} else {

println("Available cameras:");

for (int i = 0; i < cameras.length; i++) {

println(cameras[i]);

}

}

video = new Capture(this, cameras[0]); // Change index to select appropriate camera

video.start();

trackColor = color(255, 0, 0); // Default track color is red

GPIO.pinMode(27, GPIO.OUTPUT);

GPIO.pinMode(14, GPIO.OUTPUT);

GPIO.pinMode(17, GPIO.OUTPUT);

GPIO.pinMode(18, GPIO.OUTPUT);

}

void draw() {

background(0);

if (video.available()) {

video.read();

}

video.loadPixels();

image(video, 0, 0);

float worldRecord = 500;

int closestX = 0;

int closestY = 0;

// Begin loop to walk through every pixel

for (int x = 0; x < video.width; x++) {

for (int y = 0; y < video.height; y++) {

int loc = x + y \* video.width;

// What is current color

color currentColor = video.pixels[loc];

float r1 = red(currentColor);

float g1 = green(currentColor);

float b1 = blue(currentColor);

float r2 = red(trackColor);

float g2 = green(trackColor);

float b2 = blue(trackColor);

// Using euclidean distance to compare colors

float d = dist(r1, g1, b1, r2, g2, b2);

// If current color is more similar to tracked color than

// closest color, save current location and current difference

if (d < worldRecord) {

worldRecord = d;

closestX = x;

closestY = y;

}

}

}

if (worldRecord < 10) {

// Draw a circle at the tracked pixel

fill(trackColor);

strokeWeight(4.0);

stroke(0);

ellipse(closestX, closestY, 16, 16);

println(closestX, closestY);

if (closestX < 140) {

GPIO.digitalWrite(27, GPIO.HIGH);

GPIO.digitalWrite(14, GPIO.HIGH);

GPIO.digitalWrite(17, GPIO.HIGH);

GPIO.digitalWrite(18, GPIO.LOW);

delay(10);

GPIO.digitalWrite(27, GPIO.HIGH);

GPIO.digitalWrite(14, GPIO.HIGH);

GPIO.digitalWrite(17, GPIO.HIGH);

GPIO.digitalWrite(18, GPIO.HIGH);

println("Turn Right");

} else if (closestX > 200) {

GPIO.digitalWrite(27, GPIO.HIGH);

GPIO.digitalWrite(14, GPIO.LOW);

GPIO.digitalWrite(17, GPIO.HIGH);

GPIO.digitalWrite(18, GPIO.HIGH);

delay(10);

GPIO.digitalWrite(27, GPIO.HIGH);

GPIO.digitalWrite(14, GPIO.HIGH);

GPIO.digitalWrite(17, GPIO.HIGH);

GPIO.digitalWrite(18, GPIO.HIGH);

println("Turn Left");

} else if (closestY < 170) {

GPIO.digitalWrite(27, GPIO.HIGH);

GPIO.digitalWrite(14, GPIO.LOW);

GPIO.digitalWrite(17, GPIO.HIGH);

GPIO.digitalWrite(18, GPIO.LOW);

delay(10);

GPIO.digitalWrite(27, GPIO.HIGH);

GPIO.digitalWrite(14, GPIO.HIGH);

GPIO.digitalWrite(17, GPIO.HIGH);

GPIO.digitalWrite(18, GPIO.HIGH);

println("Go Forward");

} else {

GPIO.digitalWrite(27, GPIO.HIGH);

GPIO.digitalWrite(14, GPIO.HIGH);

GPIO.digitalWrite(17, GPIO.HIGH);

GPIO.digitalWrite(18, GPIO.HIGH);

}

} else {

GPIO.digitalWrite(27, GPIO.HIGH);

GPIO.digitalWrite(14, GPIO.HIGH);

GPIO.digitalWrite(17, GPIO.HIGH);

GPIO.digitalWrite(18, GPIO.HIGH);

}

}

void mousePressed() {

// Save color where the mouse is clicked in trackColor variable

int loc = mouseX + mouseY \* video.width;

trackColor = video.pixels[loc];

}

## A01.2. Main Code

The main code consists of the Processing sketch responsible for capturing video, processing frames, detecting and tracking the ball's position, and controlling the motors accordingly.

## A01.3. Libraries

The project utilizes the Processing video library for capturing video from the camera module and the GPIO library for Processing for controlling the Raspberry Pi's GPIO pins. These libraries enable seamless integration of hardware and software components for the ball-tracking robot project.

# Appendix 02

## A02.1. Project Proposal Form

The project proposal form was prepared and duly signed from our Faculty-in-Charge Dr. Biswaranjan Swain. The same is attached at the last of this report.

## A02.2. Project Management

## 

## A02.3. Bill of Material

**Table 1. Component listing.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **#** | **Component** | **Unit Cost** | **Quantity** | **Total** |
|  | Raspberry Pi Zero 2W | 1800 | 1 | 1800 |
|  | HW 095 Motor Driver | 240 | 1 | 240 |
|  | Batteries | 100 | 4 | 400 |
|  | USB Camera | 800 | 1 | 800 |
|  | Robot Chassis with Geared Motors | 1200 | 1 | 1200 |
|  | 7805 IC | 150 | 1 | 150 |
|  | USB – C Connector | 60 | 1 | 60 |
|  | Jumper Wires | 200 | As per required | 200 |
| Grand Total | | 4850 | | |

# Appendix 03

## A03.1. Data Sheets

* Raspberry Pi 4:

|  |  |
| --- | --- |
| **Video and Sound** | 2 × micro HDMI ports (up to 4Kp60 supported), 2-Lane MIPI CSI Camera Port, 2-Lane MIPI DSI Display Port, 4-Pole Stereo Audio and Composite Video Port |
| **Clock Speed** | 1.5 GHz |
| **Micro-SD Card Slot** | Yes (FAT32 format), support maximum 32G Micro SD Card Memory Features |
| **Operating Temperature (°C)** | 0 to 50 |
| **Dimensions (LxWxH) mm** | 85 x 56 x 19 |
| **Weight (g):** | 52 |
| **UPC** | 692759968863 |
| **GTIN** | 765756931182 |
| **Shipping Weight** | 0.057 kg |
| **Shipping Dimensions** | 10 × 6 × 2 cm |