Jurnal Elektro
p-ISSN: 1979-9780
e-ISSN: 2746-4288

Jurnal Elektro Vol.18 No.1 April 2025
journal homepage: https://ejournal.atmajaya.ac.id/index.php/JTE

Design and Implementation of a Vision-based Wheeled Mobile Robot Using HSV Color Segmentation and P-D Control

Wira Aditama[1], David Herianto[1], Nico Fernando[1], Carolus Henry[1], Nova Eka Budiyanta[1]

[1]Department of Electrical Engineering, School of Bioscience, Technology, and Innovation, Atma Jaya Catholic University of Indonesia, Jakarta 12930, Indonesia

Article Info

Article history:

Received
26-05-2025

Accepted
04-06-2025

Keywords:
Mobile Robot
Object Detection
Raspberry Pi
OpenCV
PD Control

Abstract

This study presents the design and implementation of a wheeled mobile robot capable of detecting and tracking a ping-pong ball using vision-based processing. The system integrates a Raspberry Pi 3 Model B+ as the main controller, a Raspberry Pi Camera Rev 1.3 for visual input, and DC motors driven by an L298N motor driver for actuation. Object detection is achieved through color segmentation in the HSV color space using the OpenCV library, followed by morphological filtering and contour analysis. A proportional-derivative (PD) control algorithm is employed to adjust motor speeds dynamically based on the ball's horizontal position in the frame. The experimental results demonstrate that the robot can successfully detect and follow a ping-pong ball, although it exhibits limitations in processing speed and motion stability. The average frame rate during operation was 5 FPS, which is sufficient for basic tracking tasks but suboptimal for high-speed applications. This project highlights the feasibility of vision based robotic systems for simple object tracking tasks.

1. INTRODUCTION

The rapid advancement of technology in the modern era has significantly transformed various aspects of human life. These technological developments have made it possible to perform tasks more efficiently and effectively. Fields such as control systems, instrumentation, and electronic devices are central to this transformation. Among the various branches of technology, robotics has emerged as one of the most fascinating and widely adopted fields. This is evident in recent works where robotics integrates visual perception, embedded systems, and autonomous control [1]. Recent developments in mobile robotics, such as the implementation of real-time navigation and remote-control using ROS and Raspberry Pi, illustrate the increasing autonomy and reliability of robotic systems in structured environments [2].

Robots come in many forms, including mobile robots, manipulators, humanoid robots, animaloid robots, underwater robots, and more. This study focuses on the mobile robot category, which is designed to move from one location to another. Mobile robots are generally divided into two types: wheeled and legged robots. The system developed in this research is based on a wheeled mobile robot, selected for its relative mechanical simplicity and stability in structured environments. The proposed robot in this project is designed to detect specific objects, in this case, ping-pong balls, while navigating their surroundings. To achieve this functionality, the robot is equipped with a camera module that acts as a vision-based sensor for object detection. Furthermore, a Raspberry Pi 3 serves as the processing unit, managing image processing and controlling the robot's movements through programmed instructions.

Based on the aforementioned context, this study seeks to address the problem of how a wheeled mobile robot can be designed and implemented to autonomously detect and respond to the presence of ping-pong balls using vision-based sensing and embedded control systems. To achieve this, the research aims to design and develop a wheeled mobile robot equipped with a camera and a Raspberry Pi 3 as its processing unit, enabling the robot to autonomously detect ping-pong balls and perform basic navigation behaviors based on visual input.

*Corresponding author. Nova Eka Budiyanta
Email address: nova.eka@atmajaya.ac.id

2. LITERATURE REVIEW

2.1 Mobile Robots

Mobile robotics is a key area within the field of robotics that focuses on the design and implementation of autonomous systems capable of navigating their environment. These robots often integrate mechanical structures, motor control systems, sensor-based perception, and real-time decision-making algorithms to perform dynamic tasks. An example of such an application can be found in the work of Kao and Ho, who developed an omnidirectional mobile robot for ball-catching tasks using stereo vision and a PID control strategy [3]. Their system demonstrates how sensor fusion and precise actuation enable robots to perform real-time object tracking and motion control in dynamic scenarios. This highlights the increasing sophistication and autonomy of mobile robotic platforms in structured environments.

2.2 Sensors and Camera Module

A sensor is a component that measures physical quantities and converts them into signals readable by a system. In mobile robots, sensors are critical for acquiring parameters such as speed, distance, and position. This project employs a camera sensor for object detection. The Raspberry Pi Camera Module Rev 1.3 is used, which is capable of capturing high-resolution images and transmitting them for processing. It is based on the Omnivision 5647 CMOS sensor and connects via a 15-pin MIPI Camera Serial Interface (CSI-2).

Table 1. Specifications of Raspberry Pi Camera Rev 1.3 [4]

Specification	Description
Image Sensor	Omnivision 5647 CMOS, fixed focus, with IR filter
Resolution	5 Megapixels
Max Picture Size	2592 × 1944
Frame Rate	1080p@30fps, 720p@60fps
Interface	15-pin ribbon to MIPI CSI-2
Image Controls	Auto exposure, white balance, band filter, luminance detection, etc.
Operating Temp.	−30°C to 0°C (operational), stable at 0°C to 50°C
Lens Size	1/4 inch
Dimensions & Weight	20 × 25 × 10 mm, 3 g

2.3 Computer Vision: OpenCV

OpenCV (Open Source Computer Vision Library) is a widely adopted open-source framework designed for real-time computer vision applications. It supports multiple programming languages, including C++, Python, and Java, and can be deployed across various operating systems such as Windows, Linux, macOS, Android, and iOS. Its modular architecture provides extensive functionalities for tasks including image processing, object detection, motion analysis, and camera calibration [5]. Recent studies have demonstrated the practical use of OpenCV in intelligent traffic systems. For instance, OpenCV has been integrated with deep learning frameworks such as YOLO to build real-time vehicle detection and traffic monitoring applications, highlighting its effectiveness in dynamic, real-world scenarios [6], [7].

Table 2. Key Features of OpenCV

Feature	Description
Image Data Manipulation	Allocation, release, conversion, and configuration
Image/Video I/O	Input via files or camera; output via files or display
Matrix & Linear Algebra	Matrix operations, eigenvalues, SVD, etc.
Dynamic Data Structures	Includes lists, queues, graphs
Image Processing	Filters, edge detection, color conversion, morphology
Structural Analysis	Contours, Hough transforms, polygon approximation
Camera Calibration	Pattern tracking, matrix estimation, stereo correspondence
Motion Analysis	Optical flow, motion segmentation, object tracking
Object Recognition	Eigenfaces, HMM-based methods
GUI Functions	Image labeling, drawing, UI elements

2.4 Embedded Processing: Raspberry Pi 3 Model B+

The Raspberry Pi is a single-board computer (SBC) developed by the Raspberry Pi Foundation in the UK. It integrates a microprocessor, memory, and I/O interfaces into a compact form factor similar to a credit card. For this project, a Raspberry Pi 3 Model B+ is utilized. This model features a 64-bit quad-core ARM Cortex-A53 processor clocked at 1.4 GHz, dual-band WiFi, Bluetooth 4.2, and Gigabit Ethernet over USB 2.0.

Table 3. Specifications of Raspberry Pi 3 Model B+ [8]

Specification	Description
Processor	Broadcom BCM2837B0, Cortex-A53, 64-bit, 1.4 GHz
Memory	1 GB LPDDR2 SDRAM
Wireless Connectivity	2.4GHz & 5GHz 802.11 b/g/n/ac, Bluetooth 4.2, BLE
Ethernet	Gigabit Ethernet (via USB 2.0, max ~300 Mbps)
Ports	4× USB 2.0, HDMI, MIPI DSI & CSI ports, GPIO header
Multimedia Support	H.264 & MPEG-4 decoding/encoding, OpenGL ES 1.1/2.0
Power Supply	5V/2.5A via micro-USB or GPIO; PoE enabled via HAT
Operating Temp.	0–50°C

2.5 Actuation: DC Motors

A DC motor is an electromechanical device that converts direct current electrical energy into mechanical rotation. It consists of a stator (stationary field) and a rotor (rotating armature), with commutators and brushes to manage current flow. The core principle underlying DC motor operation is the Lorentz force, generated when current-carrying conductors interact with a magnetic field, producing torque. The Lorentz force is given by Eq. (1).

$$F = BIL \sin\theta \quad (1)$$

where:
$F =$ Force (N)
$B =$ Magnetic flux density (Wb/m^2)
$I =$ Current (A)
$L =$ Length of conductor (m)
$\theta =$ Angle between current direction and magnetic field

DC motors are categorized into:
- Brushed motors: Simple and low-cost, using mechanical commutation.
- Brushless motors: Rely on electronic commutation, more efficient and durable.
- Permanent magnet motors: Use magnets in the stator to produce a constant field, reducing weight and size.

DC motors, particularly brushed types, are commonly used in control systems research due to their simplicity and predictability. Several recent studies have focused on enhancing their performance through advanced PD [9] and PID tuning methods [10][11].

3. METHODOLOGY

3.1 System Design Overview

The proposed system is designed as a wheeled mobile robot capable of detecting and responding to the presence of a ping-pong ball using a vision-based processing approach. The overall architecture of the system is illustrated in the block diagram as Figure 3.1.

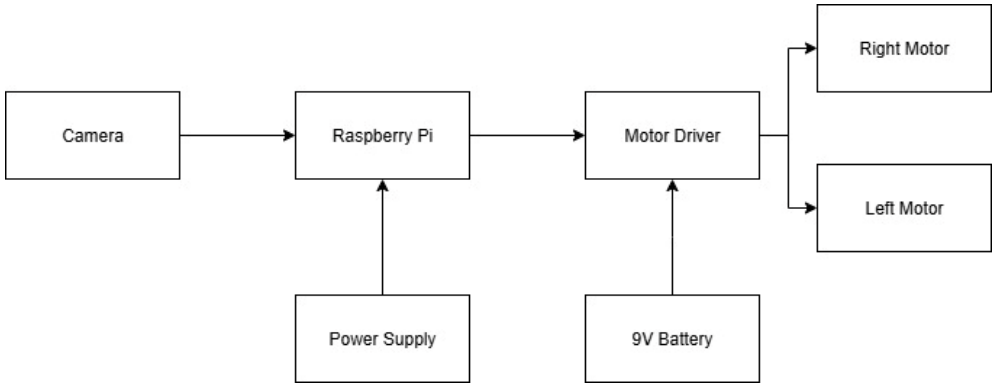

Figure 3.1. Block diagram of the ping-pong ball detection robot

The robot consists of three main subsystems: the input system (camera module), the processing unit (Raspberry Pi 3), and the actuation system (DC motors and driver module). The camera captures visual input, which is processed by the Raspberry Pi to determine the location of the ping-pong ball. Based on this information, appropriate movement commands are sent to the motors to navigate toward or stop near the ball.

3.2 List of Components

The components used in this project are listed in the table below.

Table 3.1. Components used in the robot system

No	Component	Quantity
1	Raspberry Pi Camera Rev 1.3	1
2	Raspberry Pi 3 Model B+	1
3	Motor Driver L298N	1
4	DC Motor	2
5	Acrylic Chassis	1
6	Wheel	2
7	9V Battery	1
8	Male-to-Male Jumper Wires	7

3.3 Hardware Design

The robot's physical structure is designed in 3D and visualized from multiple perspectives to ensure optimal component placement and stability. The following figures illustrate the robot's structure from various viewpoints.

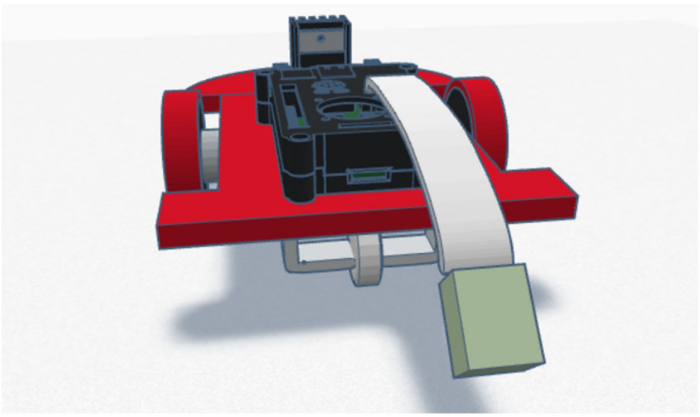

Figure 3.3a. 3D view of the robot design

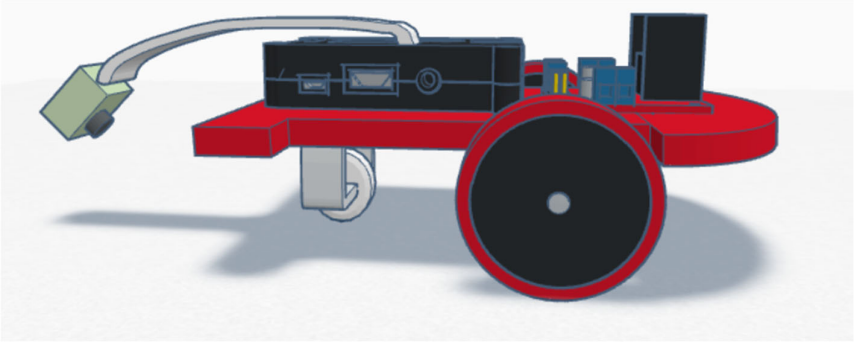

Figure 3.3b. Right side view of the robot

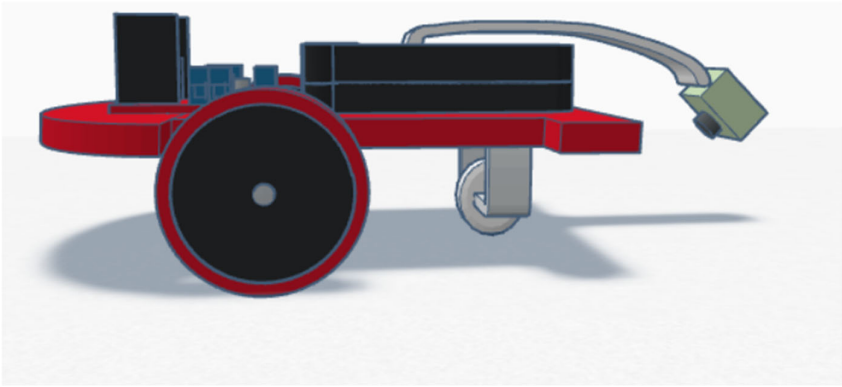

Figure 3.3c. Left side view of the robot

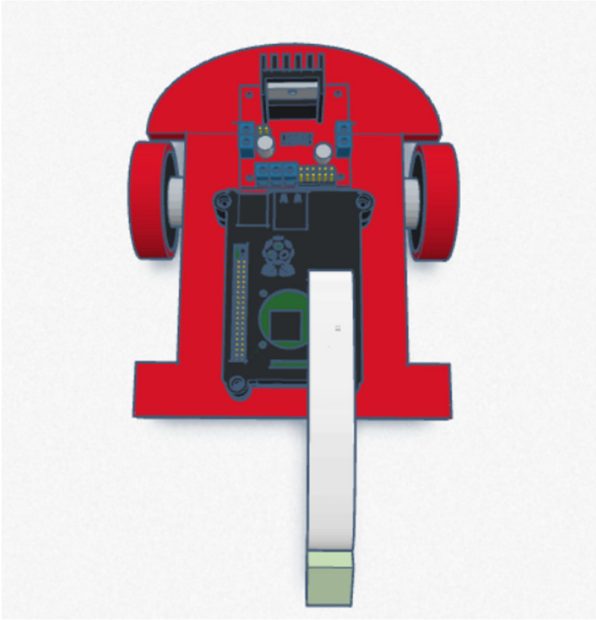

Figure 3.3d. Top view of the robot

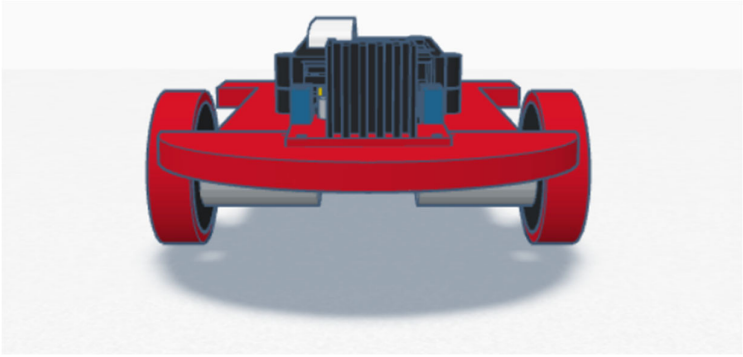

Figure 3.3e. Rear view of the robot

3.4 Prosess Flowchart

The logical flow of the robot's operation is represented in the flowchart below.

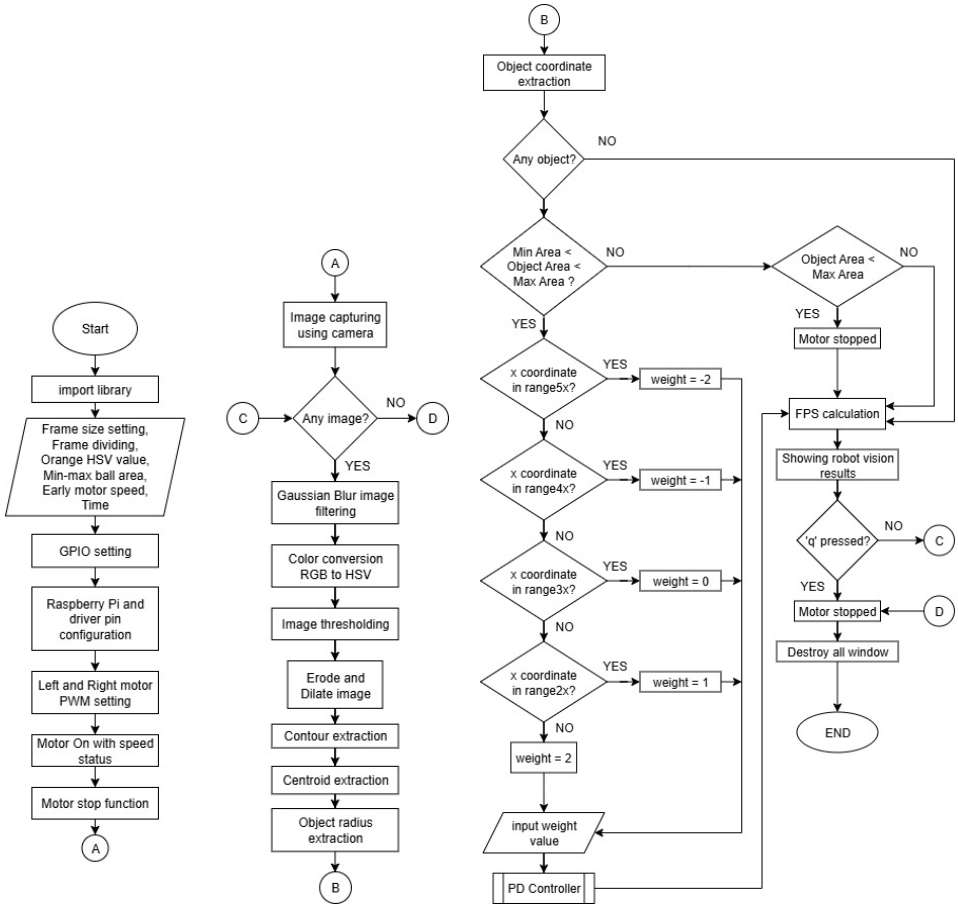

Figure 3.4. Robot's Flowchart

3.5 Operational Workflow

The robot utilizes a camera as its primary input sensor and a pair of DC motors as the output actuators. The system begins by initializing the video stream using the VideoCapture() function. Within a continuous loop, individual frames from the video feed are captured and undergo a series of preprocessing steps, including resizing, Gaussian blurring, and color space conversion from RGB to HSV. If the system fails to retrieve a frame, the loop is terminated to avoid further processing errors. Once a frame is acquired

and converted to HSV, the image undergoes color filtering to isolate regions that match the characteristic orange hue of a ping-pong ball. To suppress noise and enhance object boundaries, morphological operations are applied, specifically, erosion with four iterations followed by dilation with two iterations. The cleaned image is then analyzed for contours using the imutils.grab_contours() function.

If no contours are found, the loop continues to the next frame. However, when contours are detected, the system identifies the largest one and calculates its center coordinates, radius, and area. If the radius exceeds a threshold of 10 pixels, a circle is drawn around the object, and the center position is marked. The area of the contour is also evaluated to determine whether it falls within a predefined valid range. The image frame is logically divided into five horizontal segments, each assigned a specific weight value based on the ball's location.

These weight values are input into a Proportional-Derivative (PD) controller that adjusts the speed and direction of the robot's wheels in real time. If the ball's detected area exceeds the upper threshold, indicating close proximity, the robot halts to prevent collision. Throughout the entire operation, the system calculates the frame processing rate (frames per second or FPS) and displays it on the top-left corner of the video feed. The user can terminate the program at any time by pressing the 'Q' key, which safely stops the motors and ends the process.

4. RESULT AND DISCUSSION

4.1 System Schematic and Implementation

The schematic diagram of the proposed robot system is shown in Figure 4.1, which illustrates the integration of all components involved in the design and implementation.

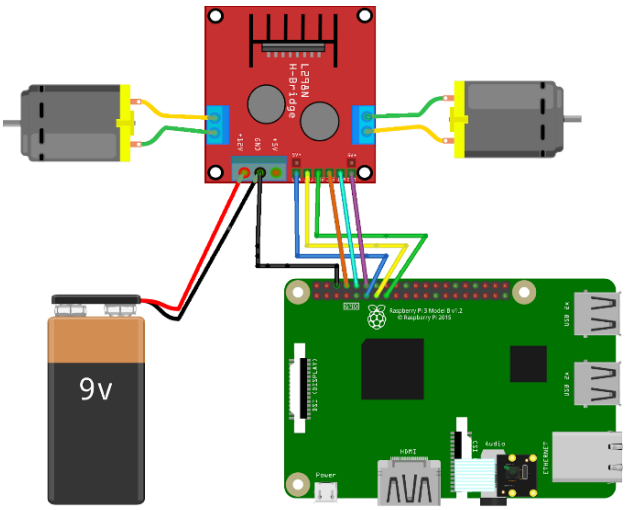

Figure 4.1. Schematic diagram of the ping-pong ball detection robot

4.2 Program Implementation

The robot control system was developed using the Python programming language, leveraging the OpenCV and GPIO libraries for image processing and motor control, respectively. The source code initializes the camera, processes each video frame to detect the ping-pong ball based on its orange color in the HSV space, and applies morphological filtering to reduce noise. Contours are extracted from the binary mask, and if a contour is detected, its center coordinates and area are computed. Based on the object's position, the image frame is segmented into five horizontal regions, each assigned a directional weight. These weights are passed into a custom Proportional-Derivative (PD) controller, which dynamically adjusts the motor speeds to orient the robot toward the ball. Motor speed adjustments are made via PWM signals sent through the L298N motor driver. The system continuously monitors the frame rate (FPS), displays it on the video stream, and allows termination by pressing the 'Q' key, upon which all motors are safely stopped.

4.3 Camera Testing and Object Detection

Initial tests focused on evaluating the robot's object detection capabilities using the Raspberry Pi Camera Rev 1.3. The robot was able to accurately detect the ping-pong ball under standard indoor lighting conditions. The captured detection frames are shown in Figure 4.3a and Figure 4.3b.

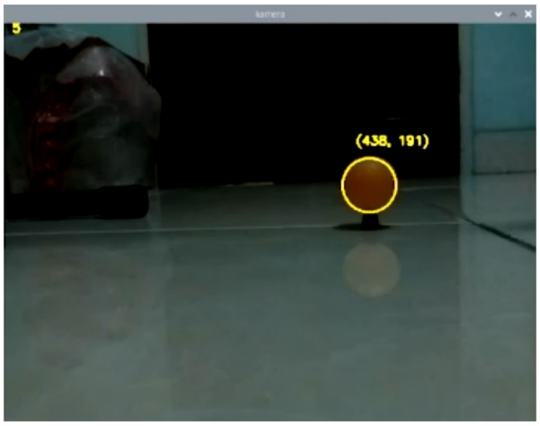

Figure 4.3a. Camera detection process (Part I)

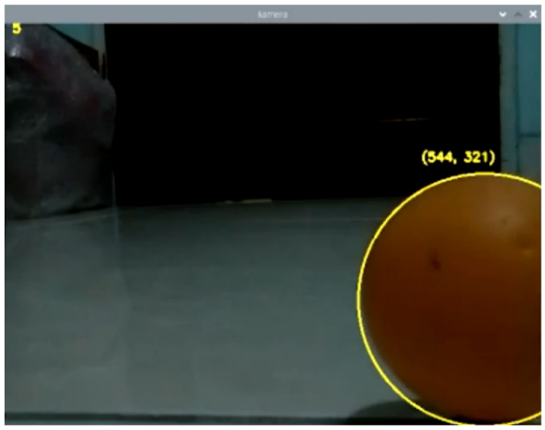

Figure 4.3b. Camera detection process (Part II)

The HSV color filtering was effective in isolating the orange object, and the morphological preprocessing steps minimized background noise. The detection performance was consistent when the ball remained within a moderate distance and central field of view.

4.4 System Integration Test

A complete system-level test was conducted to evaluate the performance of all integrated components, as summarized in Table 4.1.

Table 4.1. Functional verification of system components

Component	Status	Description
Raspberry Pi Camera Rev 1.3	Functional	Successfully integrated with the robot. Operated reliably in OpenCV and during testing. Enabled consistent detection of the ping-pong ball.
Motor Driver L298N	Functional	Accurately controlled motor direction and speed for both left and right motors.
Right DC Motor	Functional	Successfully drove the right wheel when the robot was in motion.
Left DC Motor	Functional	Successfully drove the left wheel during robot navigation.

A PD controller was implemented with gain values of $K_p = 2$ and $K_d = 7$, which provided realtime corrections based on the ball's horizontal position. The results of the system operation are illustrated in Figure 4.3c and Figure 4.3d.

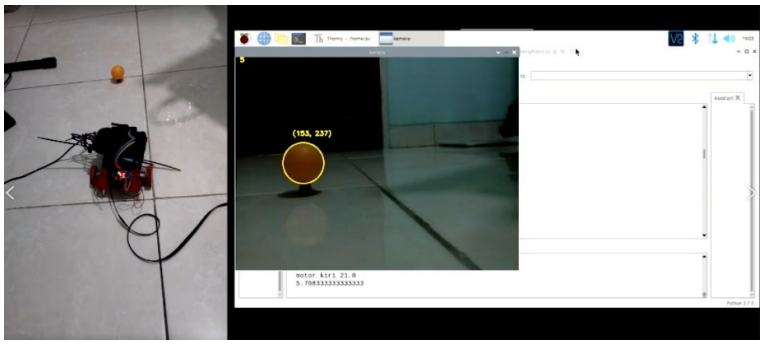

Figure 4.3c. Full system test – robot tracking (Part I)

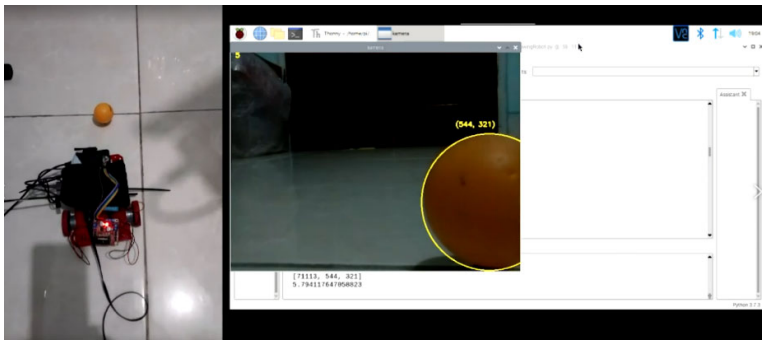

Figure 4.3d. Full system test – robot tracking (Part II)

The robot successfully detected the ping-pong ball and responded accordingly. However, there were some limitations observed during the tracking process. The average FPS (frames per second) achieved during testing was approximately 5 FPS, which indicates limited real-time performance. Moreover, the robot exhibited a slight lateral deviation during motion, failing to maintain a perfectly straight trajectory. This may be attributed to asymmetrical motor output or imperfect weight adjustment by the PD controller. Additionally, the ball was not always centered in the camera view during motion, which affected precision.

5. CONCLUSION

This study successfully demonstrated the design and implementation of a low-cost, vision-based wheeled mobile robot for detecting and tracking a ping-pong ball. The system utilized a Raspberry Pi 3 Model B+ and a Raspberry Pi Camera Rev 1.3 for visual input, combined with DC motors and an L298N motor driver for movement. By applying HSV color filtering and morphological operations using OpenCV, the robot was able to isolate and identify the ping-pong ball in real time. A Proportional-Derivative (PD) control algorithm was implemented to adjust motor speeds dynamically based on the horizontal position of the ball within the camera frame. The experimental results confirmed that the robot could detect and follow the target object effectively, although it faced certain limitations in terms of stability and frame processing speed. The average frame rate of 5 FPS was sufficient for basic tracking tasks but may not be adequate for faster or more complex movements. Additionally, the robot occasionally struggled to maintain a straight trajectory, indicating room for improvement in motor calibration or control logic. Future work may focus on improving the system's response time and tracking accuracy by optimizing image processing algorithms or upgrading to a more powerful processing unit. Enhancing the PID tuning mechanism and incorporating predictive tracking methods may also improve motion stability and responsiveness. Overall, this project validates the feasibility of using affordable embedded platforms for simple vision-guided robotic applications.

REFERENCE

[1] D. Dang and H. Bui, "Real-Time Obstacle Avoidance for Mobile Robots Using Semantic Segmentation and Monocular Camera," Electronics, vol. 12, no. 8, pp. 1932–1945, 2023. [Online]. Available: https://www.mdpi.com/2079-9292/12/8/1932

[2] W. C. Kao and S. T. Ho, "An Omnidirectional Mobile Robot for Ball-Catching Tasks Using Stereo Vision and PID Control," Sensors, vol. 21, no. 9, Art. no. 3208, May 2021. [Online]. Available: https://www.mdpi.com/1424-8220/21/9/3208

[3] Belaidi, H., Demim, F., Mezhary, A., Rouigueb, A., Yahoui, S., Nemra, A. (2024). Enhanced Navigation Based Obstacle Avoidance and Target Tracking with Raspberry-Pi for Mobile Robots. In: Djamaa, B., Boudane, A., Mazari Abdessameud, O., Hosni, A.I.E. (eds) Advances in Computing Systems and Applications. CSA 2024. Lecture Notes in Networks and Systems, vol 1145. Springer, Cham. [Online]. Available: https://doi.org/10.1007/978-3-031-71848-9_22

[4] "Raspberry Pi Camera Module Rev 1.3," RS Components, pdf data sheet (A700000007835051), 2013. [Online]. Available: https://docs.rs-online.com/2888/0900766b8127db0a.pdf

[5] OpenCV.org, "OpenCV Documentation," [Online]. Available: https://docs.opencv.org/4.x/

[6] S. Hamza and R. Rathod, "Smart Traffic Monitoring Using YOLO and OpenCV," Journal of Information Systems Engineering and Management, vol. 10, no. 1, 2025. [Online]. Available: https://jisem-journal.com/index.php/journal/article/view/5765

[7] M. Bagwe, S. Prakash, and A. Deshmukh, "Implementation of Smart Traffic System Using OpenCV and Python," Electronics, vol. 11, no. 4, Art. no. 67, 2022. [Online]. Available: https://www.mdpi.com/2673-7590/4/4/67

[8] Raspberry Pi Foundation, "Raspberry Pi 3 Model B+ Product Brief," datasheets.raspberrypi.com, May 2018. [Online]. Available: https://datasheets.raspberrypi.com/rpi3/raspberry-pi-3-b-plus-product-brief.pdf

[9] N. E. Budiyanta, C. O. Sereati, and L. Lukas, "P-D controller computer vision and robotics integration based for student's programming comprehension improvement," TELKOMNIKA (Telecommunication, Computing, Electronics and Control), vol. 18, no. 2, pp. 899-906, 2020, doi:10.12928/TELKOMNIKA.v18i2.14881.

[10] A. Saravanan, K. Ravi, and S. Shanmugam, "Performance enhancement of PID-controlled DC motor using Kookaburra and Red Panda optimization algorithms," Scientific Reports, vol. 15, no. 1, pp. 1–13, 2025. [Online]. Available: https://www.nature.com/articles/s41598-025-87607-2

[11] I. C. Yuniar, R. Nurhadi, and B. Setiawan, "Optimization of PID controller parameters using particle swarm optimization on DC motor speed control," Int. J. of Robotics and Control Systems, vol. 2, no. 2, pp. 134–142, Jun. 2022. [Online]. Available: https://arxiv.org/abs/2209.09170