INTEK Jurnal Penelitian
Vol. 10, No. 1, pp. 48-54, April 2023

Real-time Ball Detection and Tracking using Raspberry PI

Dharma Aryani[1,*,a], Kartika Dewi[2,b], Fery Ta'by[3,c], Evita Putri Sanggaria[4,d]

[1,2,3,4]Electrical Engineering Department, Politeknik Negeri Ujung Pandang, Makassar, 90245, Indonesia
[*,a] dharma.aryani@poliupg.ac.id, [b] kartikadewi@poliupg.ac.id, [c] ferytabi26@gmail.com, [d] evitaputrisan16@gmail.com

Abstract*— This paper presents a real-time system for ball detection and tracking system which is reliable in any conditions. Images from the webcam are processed by openCV library running on a Raspberry Pi to move the camera pan and tilt servo and two DC motors to drive the robot body using the Arduino Nano microcontroller. The webcam is integrated in a robot prototype to represent the wheel football robot type. The results show that a ball tracking webcam system is obtained with the capability to detect a ball with a diameter of 17cm within a maximum distance of 200 cm, a stable ball reading when the light intensity is at 32 lux and above. Furthermore, the experimental results demonstrated the system's robustness in detecting and tracking ball in different distance and ligthing conditions.

Keywords— ***Raspberry Pi; Open CV; Webcam; Ball Tracking.***

I. Introduction

Real-time ball tracking robots have emerged as an exciting field of research at the intersection of robotics, computer vision, and artificial intelligence. These robots are made to autonomously locate, track, and follow the motion of a ball in real-time, enabling a variety of applications like interactive sports coaching, automated ball games, and cutting-edge sports analytics. Due to the dynamic nature of ball motion, occlusions, and the requirement for quick responses, the construction of such robots is fraught with difficulty.

Basic computer vision techniques including color-based segmentation and motion estimation were used in early studies on ball tracking robots. These techniques provide a basic grasp of ball tracking, but their inability to deal with intricate settings and moving balls prompted the development of more advanced techniques. Real-time tracking techniques are essential to precisely trace the ball's trajectory after it is discovered. To anticipate and track the ball's position over time, conventional tracking techniques like Kalman filtering and particle filters have been used. In order to account for errors in detection, these algorithms use motion models and observations to anticipate the ball's future placements.

Real-time performance must be attained by ball tracking robots in order to guarantee rapid replies in interactive applications. To accelerate ball tracking algorithms and decrease latency, research has concentrated on parallel processing, hardware acceleration, and optimisation strategies such as in [1] which proposed a set of algorithms for multiple motion tracking from a mobile robot equipped with a monocular camera and a laser rangefinder.

By categorising objects into two groups, [2] proposed an object detection technique. These subcategories include grid-based segmentation and colour blob segmentation. They coupled colour blob segmentation with depth estimation using trigonometric characteristics to quickly identify the red ball. The method was used with the grid-based segmentation pattern and recognition to find the goal post.

Another study by [3] presents the color-coded based on color-based pixel-wise subtraction adaptive estimation for real-time object localisation. They use adaptive estimate for real-time object localization to determine the ball's and goalkeeper's positions. This technique also uses a threshold to divide the hue, saturation, and value (HSV) of the colour when identifying the ball and goalkeeper on the field. In order to enhance the vision to detect ball and goal, the You Only Look Once (YOLO) methods is used as deep-learning object detection method by [4] to present the real-time object detection integrated to humanoid robot soccer. A study in [5[used Haar-like features

DOI : http://dx.doi.org/10.31963/intek.v10i1.4301

trained by an adaboost algorithm to get a colorless representation of the ball. This method detecting and tracking the ball without the need for color information.

Indonesia is a country that has participated in developing the field of robotics, an example of an effort being made is holding the Indonesian Robot Contest (KRI). There are various divisions that are contested in this robot contest, one of which is the Wheeled Football Robot Contest (KRSBI-Wheel). This robot contest is a competition for robots who play ball like humans, but the robots used are robots that move or maneuver using wheels and play games automatically without human intervention by utilizing the camera sensor as a ball visualization and imaging tool.

Based on this background, this paper explores the Ball Tracking Webcam System Design Based on Raspberry Pi which focuses on detecting and tracking object by performing image processing to detect and track the position of the ball. Ball tracking is executed using openCV python image processing, with the color tracking method. Data from ball tracking is applied to a two-wheeled mobile robot with an Arduino microcontroller as a controller for wheel rotation and servo movement. The end result of this research is the formation of a two-wheeled mobile robot that tracks and follows the movement of the ball.

The rest of this paper is organized as follows: Section II will describe about proposed method which been used as object detection approach. In section III, the results of experiments are presented followed by section IV as the conclusion and future work.

II. Research Methodology

Raspberry based webcam ball tracking system is a ball-tracking robot system that is configured using Raspberry Pi as the robot control center, then the webcam is used to retrieve visual data and managed using the OpenCV library to obtain ball position data which is then managed by Arduino Nano

A. Software Architecture

OpenCV (Open Source Computer Vision Library) is an open-source computer vision and machine learning software library. It provides a wide range of functions and algorithms for image processing, computer vision tasks, and machine learning applications. Key Features and functionalities of OpenCV, such as image and video processing where openCV allows users to read, write, and process images and videos from various file formats. It provides a wide range of image processing functions, such as filtering, resizing, morphological operations, and color space conversions. OpenCV supports various object detection algorithms and real-time computer vision, making it suitable for real-time computer vision applications, including real-time object detection, tracking, and augmented reality [6]. OpenCV is primarily written in C++, but it also provides an easy-to-use Python interface. This makes it accessible to developers who prefer working in Python for rapid prototyping and development.

The software system is divided into 2 main systems, namely the vision system and localization. A motion control system to obtain information about ball position, ball color, ball distance, presence or absence of balls, then the data is used for a motion processing system to control the movement of DC motors and servo motors to form a complex system that works in parallel, resulting in a robot that can detect the ball, track the ball and follow or approach the ball object (Figure 1). Furthermore, the flowchart on the vision and localization system as well as motion control is presented in Figure 2.

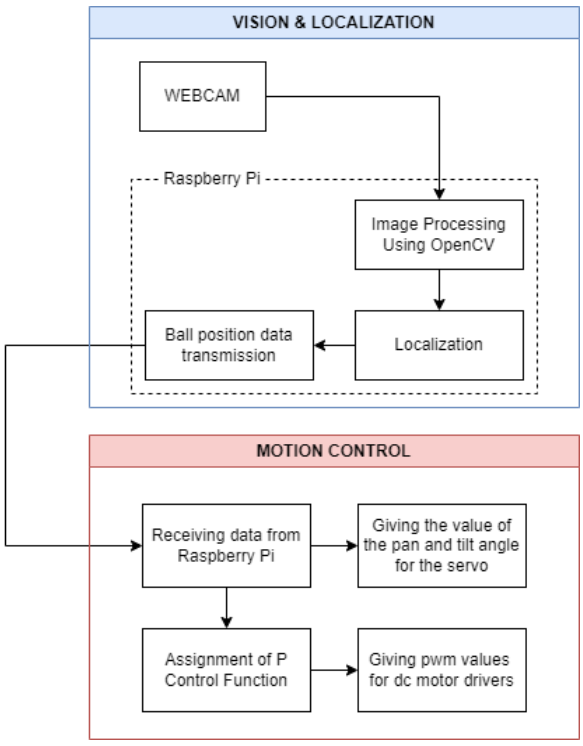


Figure 1. System architecture

DOI : http://dx.doi.org/10.31963/intek.v10i1.4301

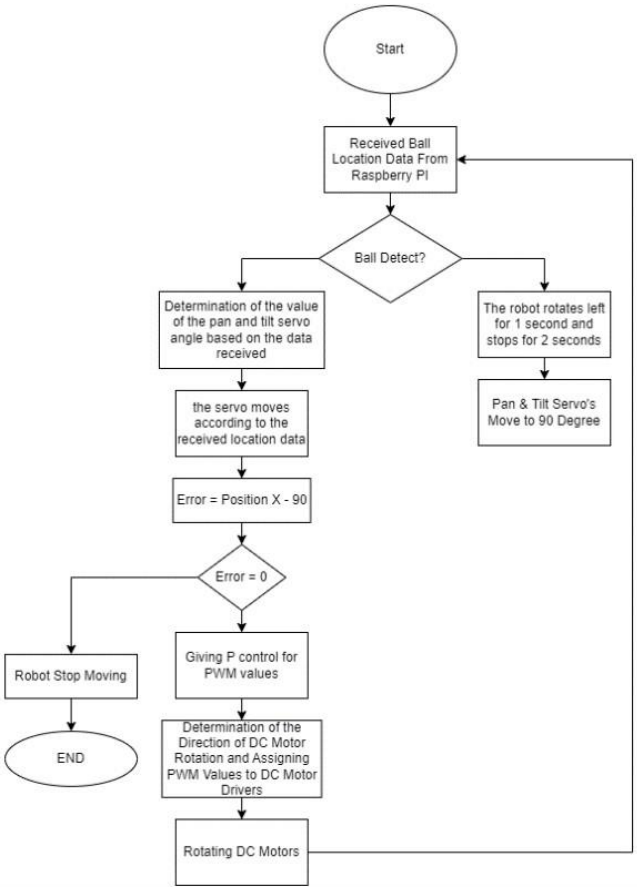


Figure 2. Flowchart Vision and Motion Control

The procedures for software setting are:

1. Receiving Image From Camera

```
if not args.get("video", False):
    camera = cv2.VideoCapture(0)
else:
    camera = cv2.VideoCapture(args["video"])


while True:
    (grabbed, frame) = camera.read()

    if args.get("video") and not grabbed:
        break
```

Through the program script above, raspberry will receive image data from the internal camera and then stored in the camera variable, but if there is no data from the camera, it will do a break.

2. HSV Color Setting

Setting the HSV color space is done by providing a limit value for the HSV color to be detected. The script is as follows:

```
greenLower = (34, 120, 116)
greenUpper = (56, 255, 255)

hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)
```

To determine the hsv value of the ball color, the HSV color picker application is used. The following is an example of picking up colors with the color picker application:

Figure 3. Color Picker App

3. Thresholding

Thresholding is the process of converting an hsv color image into a binary image where if the detected color image meets a predetermined range value, it will turn white, whereas if it does not, it will turn black. The following program script used in this study is as follows:

```
mask = cv2.inRange(hsv, orangeLower, orangeUpper)
```

The range value used is using the hsv color space that has been set previously. The thresholding process is also known as the masking process

4. Morphological Transformation

Is a method that utilizes the erosion and dilation functions to make the masking more objectified. The erosion function will remove small color spots and leave a strong color cast. This is then combined with the dilation function to enlarge the set of colors that have been pre-sorted by the erosion function. So that its application to ball detection, detection will be more focused towards the ball. The following program script used in this study is as follows:

```
mask = cv2.erode(mask, None, iterations=2)
mask = cv2.dilate(mask, None, iterations=2)
```

5. Draw Contour

Contours are defined as lines joining all points along the image boundary that have the same intensity. Contours are useful in shape analysis, finding the size of objects of interest, and object detection. Contours can be generated when all the previous image processing steps have been worked out. The draw contours program script used in this study is as follows:

```
cnts = cv2.findContours(mask.copy(), cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)[-2]
```

6. Draw Circle

The next stage is to make a circle using the parameters of the contour results that have been found previously. To do this, the function used is cv2.circle and to find the position of the center point of the circle and the radius of the circle, the function used is cv2.minEnclosingCircle.

DOI : http://dx.doi.org/10.31963/intek.v10i1.4301

The following program script used in this study is as follows:

```
cv2.circle(frame, (int(x), int(y)), int(radius), (0, 255, 255), 2)
```

7. Ball Localization

Ball localization is useful for determining the position of the ball in the image frame by creating a grid in the frame, making it easier to send data to arduino.

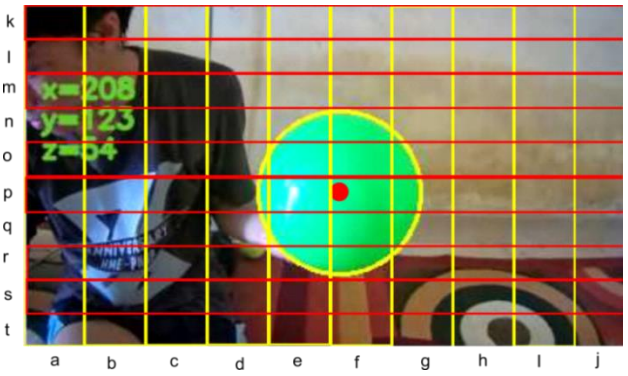
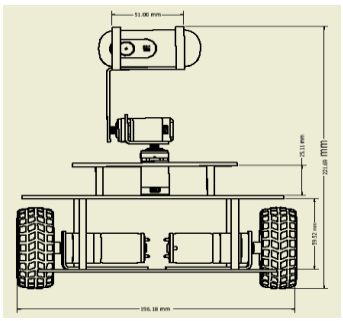


Figure 4. Ball Localization

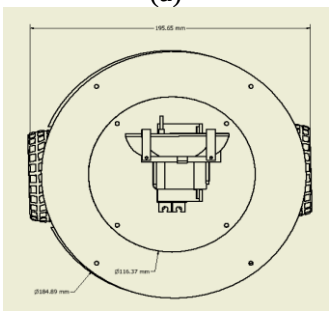
B. Hardware

1. Robot Design

The robot mechanics designed in this study aim to enable the robot to detect, track and move towards the ball properly. The robot design is designed in Autodesk Inventor 2016 application. The design of the robot prototype is shown in Fig 5.



(a)



(b)

Figure 5. Robot Prototype a) Front view b)Upper view

2. Electronics System

The electronic design used is divided into 2 main systems, namely vision and localization and motion control. The block diagram is as follows:

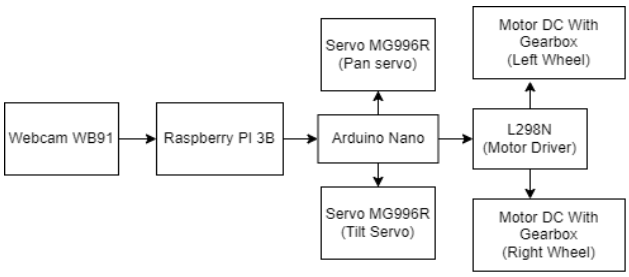


Figure 6. Electronics System

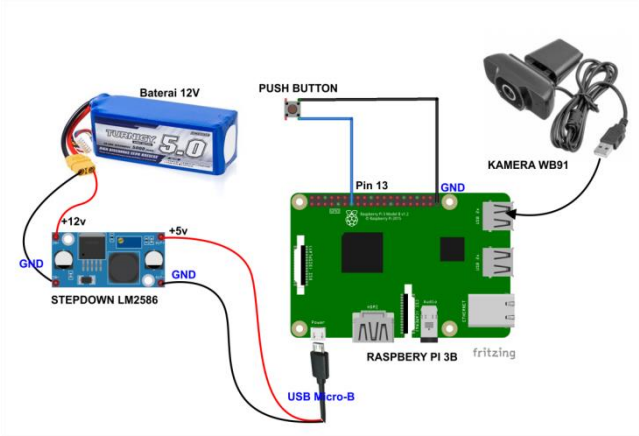


Figure 7. Vision & Localization Electronics Design

The electronic design of vision and localization consists of a webcam WB91 and Raspberry pi 3B and a push button, in which the webcam acts as a capturer of the surrounding image, raspberries acts as a vision and localization processor then send ball position information to electronic motion control and a push button as a shutdown button.

The Raspberry Pi is a single-board computers which has high adaptability and open-source nature, these credit-card-sized boards have become quite popular across a wide range of industries. The Raspberry Pi boards feature connectivity choices like USB, HDMI, Ethernet, Wi-Fi, and Bluetooth, support a variety of operating systems, and have GPIO pins for interacting with external hardware. They are employed in a variety of applications, including robots, media centres, home automation, Internet of Things, and more [8].

In Electronic Motion Control there is an arduino nano as the main controller, the L298N module as a DC motor driver, 2 servos to support pan and tilt movements,

DOI : http://dx.doi.org/10.31963/intek.v10i1.4301

stepdown to reduce 12v to 5v, there is also a pin header for serial communication ports between raspberries and arduino. For battery ports, lights, and switches use block pins for easy wiring.

III. Results and Discussion

The specifications of the robot is measures 23.5cm high, robot weight 0.971 kg, the robot assembly material uses 5052 aluminum with a thickness of 1.5mm and acrylic with a thickness of 3mm, WB91 camera as image input. raspberry Pi 3 Model B as an image processor, arduino nano as a servo controller and dc motor, servo MG996R, as an actuator on the camera. 12v DC motor with a speed of 125rpm and a torque of 7.5kg, as an actuator on the wheel (Figure 8)

Figure 8. Hardware Design Result

A. *Image Processing Testing*

After going through the stages of taking pictures from the webcam, converting RGG to HSV, thresholding, erosion, dilatation, and draw circles, the results are as shown in the following figure

Figure 9. Image Processing Result

At this stage, the object's midpoint has been calculated using the find contours method, and a circle has been drawn on the 0bject, as well as displaying the object's position value in the upper left corner of the frame.

B. *Object capture distance testing*

Testing the catching distance of objects in this study used a dark green ball with a ball diameter of 17cm provided that the color calibration has been carried out according to the color of the ball caught. The results are as shown in the following figure.

Figure 10. Maximum Object Capture Distance

Based on the data obtained, the maximum distance of 17cm ball detection is at 200m.

C. *Testing based on different object colors*

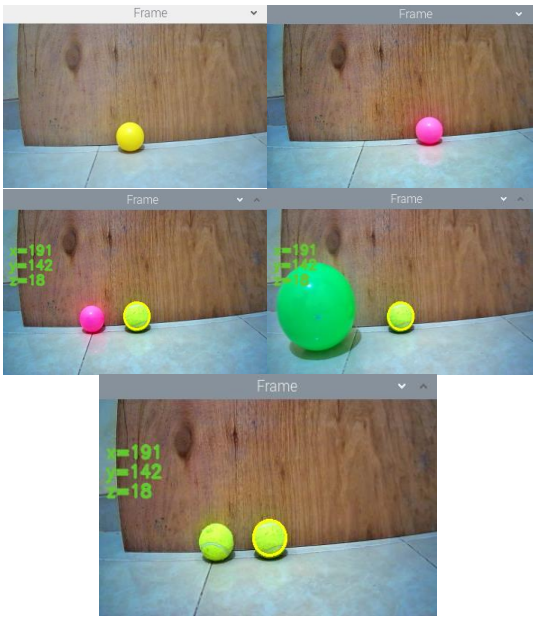


Figure 11. Testing different object color

From the results of data collection above, it can be concluded that the ball detected by the robot is the ball

DOI : http://dx.doi.org/10.31963/intek.v10i1.4301

that has the closest color value to the HSV value setting in the script.

D. *Testing based on ball diameter*

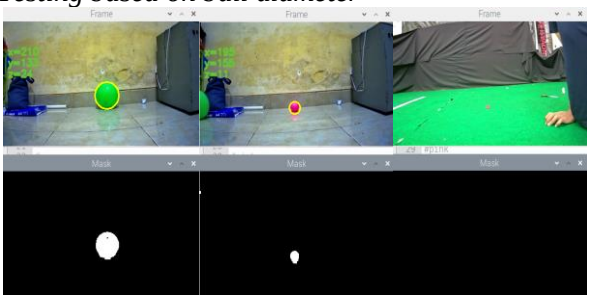


Figure 12. Testing based on ball diameter

From the experimental data that has been obtained in the image above, at a distance of 1 m, the maximum ball diameter that can be detected is 6cm.

E. Testing the Effect of Ball Detection on Light Intensity

Testing the effect of ball readings on light intensity is carried out by measuring the lux of room light using the lux light meter application on an Android device.

Table 1. Ball detection based on Light Intensity

No	Light Intensity (Lux)	Condition	Ball Detection Response
1	0	Ball not detect	No detection
2	4	Ball Detect	Unstable detection
3	12	Ball Detect	Unstable detection
4	16	Ball Detect	Unstable detection
5	32	Ball Detect	Stable detection
6	45	Ball Detect	Stable detection
7	99	Ball Detect	Stable detection
8	200	Ball Detect	Stable detection

The stable ball readings are when the light intensity is at 32 lux and above. From the data above, it can be concluded that light intensity affects ball detection, the greater the value of the light intensity, the better the ball detection, conversely the smaller the value of the light intensity, the worse the ball detection. The effect of webcam quality is also one of the ball detection factors.

F. Robot Speed Test Chasing the ball

Testing how fast the robot is in chasing the ball is carried out when the ball is 1 meter in front of the robot, on a tiled floor, and to measure time using a stopwatch.

Table 1. Ball detection speed

No	Trial	Time (second)
1	1	4.10
2	2	3.80
3	3	15
4	4	4
5	5	4.5

In testing a 1 meter distance, the robot can track the ball until it approaches the ball with an average travel time of 4 seconds. In addition to the affomentioned testings, the response of the robot head in following the ball shows that when the ball is not detected, the robot will do a motion looking for the ball, that is, the robot rotates to the left for 1 second and a delay of 2 seconds, then spins again so on until the ball is detected.

IV. Conclusion

The robot can detect and track spherical objects properly based on the images from the webcam. The images from the webcam are processed using image processing with the openCV Python library running on Raspberry Pi. The robot can move closer to the spherical object automatically by processing data from raspberries using arduino. The data received from raspberry is ball position data which is used as a parameter to determine the motion of the servo and dc motor which functions as a robot actuator. For a ball with a diameter of 17cm it can be detected up to a maximum distance of 200 cm. Objects that are the same color as the detected ball will still be read by the robot even with a different shape, this is because object reading is based on color. If there are two ball objects with he same color, the ball that is detected by the robot is the ball that has the most appropriate HSV color value. At a distance of 1m, the minimum ball diameter that can be detected is 6cm and the ball reading is stable when the light intensity is at 32 lux and above.

References

[1] Jung, B., Sukhatme, G.S. Real-time Motion Tracking from a Mobile Robot. Int J of Soc Robotics 2, 63–78 (2010)

[2] Peter Chondro, Shanq-Jang Ruan, “An adaptive background estimation for real-time obejct localization on a color-coded environment,” IEEEInternational Conference on Advanced Computer Science and Information (ICACSIS), pp. 464-469, Malang-Indonesia, October 2016.

[3] Phua Seong Hock, S. Parasuraman, “Motion synchronization with predefined rhythms for humanoid robot,” IEEE-Recent Advances in Intelegent Computational Systems (RAICS), pp. 294-299, TrivandrumIndia, December 2015.

DOI : http://dx.doi.org/10.31963/intek.v10i1.4301

[4] Susanto, Eko Rudiawan, Riska Analia, Sutopo Daniel, Hendawan Soebakti, "The deep learning development for real-time ball and goal detection of barelang-FC." ,*IEEE 2017 International Electronics Symposium on Engineering Technology and Applications (IES-ETA) - Surabaya 2017*

[5] André Treptow, Andreas Zell, "Real-time object tracking for soccer-robots without color information", Robotics and Autonomous Systems,Volume 48, Issue 1, 2004

[6] OpenCV Documentation, OpenCV - Open Computer Vision Library.

[7] Muharom, S., Automatics detect and Shooter Robot based on object detection using camera. Przegląd Elektrotechniczny, 2022.

[8] Raspberry Pi. "Raspberry Pi 3 Model B", (https://www.raspberrypi.org/products/raspberry-pi-3-model-b/?resellerType=home, diakses 23 November 2020).

DOI : http://dx.doi.org/10.31963/intek.v10i1.4301