IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Journal Européen des Systèmes Automatisés
Vol. 59, No. 7, July, 2026, pp. 2015-2026
Journal homepage: http://iieta.org/journals/jesa

Real-Time Ball Tracking and Prediction for Autonomous Wheeled Soccer Robot

Achmad Fiqhi Ibadillah[1*], Achmad Zain Nur[1], Aeri Rachmad[2], Yuli Panca Asmara[3], Muhammad Syauqi Annafi'[1]

[1] Department of Electrical Engineering, Faculty of Engineering, University of Trunodjoyo Madura, Bangkalan 69162, Indonesia
[2] Department of Information Systems, Faculty of Engineering, University of Trunodjoyo Madura, Bangkalan 69162, Indonesia
[3] Department of Mechanical Engineering, Faculty of Engineering and Quantity Surveying, INTI International University, Nilai 71800, Malaysia

Corresponding Author Email: fiqhi.ibadillah@trunojoyo.ac.id

https://doi.org/10.18280/jesa.590717

Received: 22 April 2026
Revised: 17 June 2026
Accepted: 2 July 2026
Available online: 31 July 2026

Keywords:
autonomous wheeled soccer robot, ball tracking and prediction, Kalman filter, omnidirectional camera, real-time system, third-order polynomial model, product innovation

ABSTRACT

A reliable robotic system with high and affordable performance is urgently needed to support product innovation. This study supports Sustainable Development Goal (SDG) 9 by providing an innovative product solution through a real-time process for ball tracking and prediction in autonomous wheeled soccer robots. The study combines Hue, Saturation, and Value (HSV)-based color segmentation, morphological closing operations to reduce noise, Circular Hough Transform (CHT) for verifying shapes, and a Kalman filter to predict ball trajectories. We also propose a third-order polynomial for calibrating ball distances, enabling reliable detection between 20 cm and 350 cm. Deployed on an Intel NUC Mini PC paired with an omnidirectional camera, the proposed vision processing pipeline delivers 79.04 frames per second, outperforming YOLO and Optical Flow. The Kalman filter matches predicted to actual positions with 82.81% accuracy, while exhibiting a Mean Absolute Error (MAE) of 53.01 px, an MSE of 3,392.91 px^2, and a Root Mean Square Error (RMSE) of 56.20 px for all scenarios. Our third-order polynomial model limits maximum errors to under 1 cm over the full range. All experiments were conducted in a controlled indoor field.

1. INTRODUCTION

The Sustainable Development Goals (SDGs) continue to attract worldwide attention. The ninth SDG target focuses on industry, innovation, and infrastructure; therefore, this issue needs to be addressed through the development of innovative products, cost-effective performance, and reliable robotic systems [1]. Research in this field has mainly focused on several applications, including autonomous navigation, real-time object tracking, and predictive control. Wheeled soccer robots can serve as challenging test platforms for evaluating tracking algorithms in image processing, which can also be applied to industrial and service robots [2]. Technological sustainability can be achieved through the integration of robotics and computer vision using advanced tracking systems. Surveillance systems, automated warehouse systems, and logistics applications require fundamental functions such as object tracking and trajectory prediction. The main objective of this research is to improve the control capability of wheeled soccer robots for real-time ball tracking and movement prediction, thereby supporting the development of sustainable and high-performance automated technologies.

Autonomous mobile robots are increasingly employed in industrial, service, recreational, and competitive environments. Their effective operation requires the integration of reliable mobile platforms, environmental sensing, autonomous navigation, and coordinated control capabilities [3]. The RoboCup competition, which began in the late 1990s, provides a standardized yet dynamic environment in which robots must perceive their surroundings, track moving objects, coordinate with teammates, and execute complex actions [4, 5]. Unlike conventional industrial automation systems operating under relatively controlled conditions, soccer robots must cope with moving objects, temporary occlusions, illumination variations, and interactions among multiple robots. These characteristics make robotic soccer a demanding and useful platform for evaluating vision-based navigation, object-tracking, and predictive-control algorithms. A fundamental capability required for autonomous soccer robots is the reliable detection and tracking of the ball under real-time conditions [6, 7]. In this research, the ball is considered the primary object of interest, and the robot's ability to locate, approach, and manipulate the ball directly affects its overall performance. Vision-based ball tracking systems must address several challenges, including accurate detection under varying lighting conditions, robust tracking during ball occlusion, real-time processing for rapid control responses, and trajectory prediction to anticipate future movements [8, 9]. Traditional approaches commonly use color-based segmentation through Hue, Saturation, and Value

(HSV) thresholding, which provides efficient performance under different lighting conditions [10, 11]. However, relying solely on color-based detection may generate false positives from similarly colored objects on the field. Therefore, additional shape validation processes, such as the Circular Hough Transform (CHT), are required to verify candidate ball regions [12]. The combination of color segmentation and shape validation has been applied in various soccer robot applications due to its computational efficiency without requiring the high computational resources associated with deep learning methods.

Although the aforementioned detection algorithms can identify the ball in a single image frame, autonomous robots require sequential position information to intercept and control moving objects effectively [13, 14]. The latency introduced between vision processing and control output can reduce performance in dynamic interception tasks, particularly when the ball moves at high speed. Predictive filtering methods, such as the Kalman filter, can address these challenges by estimating the ball's state, including position and velocity, and predicting its future trajectory. The Kalman filter employs a recursive framework that enables optimal state estimation under uncertainty by combining noisy measurements with motion models to obtain more accurate estimates. For wheeled soccer robots operating in environments with relatively predictable ball movements, the Kalman filter provides high computational efficiency and accurate trajectory prediction performance [15].

2. RELATED WORK

Research on vision-based ball tracking for soccer robots has developed through several main approaches, including color-based and shape-based detection, motion prediction using Kalman filtering, and object detection based on deep learning. Each approach has different strengths and limitations. Color-based methods, such as HSV segmentation, are advantageous in terms of computational efficiency because they do not require model training and can be implemented in real-time robotic systems. However, this approach is sensitive to lighting variation and may produce false detections when other objects with similar colors appear in the field of view. Therefore, color-based detection needs to be combined with shape validation methods, such as contour analysis or CHT, so that the detected object is not only similar in color but also consistent with the geometric characteristics of a ball.

Kao and Ho [16] developed a ball-catching system for an omnidirectional wheeled mobile robot by integrating image processing, Kalman filtering, and PID control. Their study shows that ball interception requires the integration of object detection, motion estimation, and robot control. Rahmat et al. implemented a Kalman filter on the Barelang63 wheeled soccer robot by utilizing information from an omnidirectional camera and a stereo camera [17]. Their results showed that the robot was able to intercept the ball at a speed of approximately 0.8 to 1 m/s. These studies demonstrate that Kalman filtering can support the estimation of ball position and velocity in interception scenarios. However, interception success is not only determined by the prediction algorithm, but also by ball speed, actuator response, processing latency, and the real-time performance of the vision system. Therefore, ball prediction needs to be evaluated together with prediction error, trajectory angle, frame rate, and the characteristics of the robotic platform used.

Deep learning-based approaches have also been widely applied to improve object detection accuracy in soccer robots. Farhan and Candra developed a goal detection system using YOLOv8 and a ball detection system using CNN for KRSBI-Beroda robots equipped with omnidirectional cameras [18]. Habibi et al. [19] proposed a YOLO-based ball detection system in ROS and evaluated its detection efficiency at distances ranging from 1 to 12 m. Bordado et al. [20] applied YOLO for ball detection and tracking in the RoboCup Soccer Humanoid League Kidsize category, while Jati et al. [21] compared YOLO-NAS with YOLOv8 and YOLOv7 for ball tracking, goal boundary detection, and obstacle avoidance in humanoid soccer robots. These studies show that YOLO- and CNN-based methods provide strong object recognition capability, particularly in visually complex environments. However, deep learning methods generally require datasets, training processes, model optimization, and higher computational resources, so their implementation in wheeled soccer robots with fast response requirements must carefully consider latency and frame rate. The comparative summary of the proposed method and other methods are shown in Table 1 below.

Table 1. Comparative summary of vision-based ball tracking methods

Ref.	Vision Method	Embedded Devices	Frame Rate	Accuracy	Real Prototype
[19]	YOLOv8	Laptop (Ryzen 7 5800H + NVIDIA RTX 3060, 16GB RAM)	30 fps	51%	No
[20]	YOLO + Optical Flow	Intel NUC (Core i7, 16 GB RAM)	15 fps	N/A	Yes
[21]	YOLO-NAS	Jetson Nano 4 GB RAM	21 fps	48%	Yes
This work	HSV + CHT + Kalman Filter	Intel NUC Mini PC (Core i7, 16 GB RAM)	79.04 fps	82.81%	Yes

Computational limitation remains an important issue in the implementation of vision systems for soccer robots. Platforms such as Raspberry Pi, NVIDIA Jetson, and Intel NUC have different capabilities in running detection and prediction algorithms. Frame rate is closely related to robot response because a higher processing rate allows the system to update the ball position more frequently and reduce prediction uncertainty [22]. However, deep learning-based approaches do not always provide sufficient speed on embedded devices. A lightweight CNN-based study for ball manipulation achieved only 3.2 frames per second on a Raspberry Pi platform [23]. This result indicates that detection accuracy alone is not sufficient; soccer robot systems also require algorithms that can maintain real-time processing speed according to the control requirements of the robot. Based on this comparison, previous studies still leave a research gap in balancing detection accuracy, prediction capability, and computational efficiency for wheeled soccer robot vision systems. YOLO and CNN-based approaches provide strong detection capability, but they generally require training datasets and higher

computational resources. In contrast, HSV segmentation, morphological operations, and CHT are computationally lightweight, but they require additional calibration and prediction mechanisms to improve tracking performance during ball motion. The Kalman filter is relevant for this purpose because it can estimate the ball position based on previous measurements, but its performance needs to be validated through trajectory-angle testing, prediction error analysis, and frame-rate evaluation. This study addresses this gap by integrating HSV segmentation, morphological filtering, CHT, centroid detection, third-order polynomial distance conversion, and Kalman filter-based trajectory prediction. Unlike deep learning-based studies that emphasize detection accuracy with higher computational requirements, this study focuses on a lightweight vision system that can operate in real-time on a wheeled soccer robot, as seen in Figure 1. The main distinction of this study lies in the integration of classical vision methods with nonlinear distance calibration and real-time ball motion prediction on an Intel NUC-based robotic platform. This integration is expected to maintain a balance between computational efficiency, distance estimation accuracy, and ball trajectory prediction capability.

Figure 1. Omni-wheeled soccer robot

3. METHOD

The whole process of the proposed method is shown in Figure 2. The proposed system integrates an omnidirectional camera, a wheeled soccer robot, and a control system to achieve autonomous ball interception with a real-time image processing stage. The proposed method is divided into six major stages: image acquisition and preprocessing, color detection and segmentation, shape validation, centroid detection and estimated ball position calculation using a third-order polynomial equation, Kalman filter prediction, and finally action execution. The proposed method runs on hardware and software specifications as shown in Table 2 below.

Table 2. System hardware and software specifications

Component	Specification
Programming Language	C++
Vision Library	OpenCV (Version 4.10)
Computing Platform	Asus NUC Mini PC
Processor (CPU)	Intel Core i7 (7th Generation)
Graphics Card (GPU)	Intel Iris 650
Memory (RAM)	8 GB
Processing Mode	CPU-based (No GPU acceleration)
Camera Resolution	640 × 480 pixels

3.1 Image acquisition and pre-processing

The system begins with the initialization and activation of an omnidirectional camera with a convex reflector placed on the top of the robot. The camera used is an Omnivision ELP Full HD webcam wide-angle camera with an OmniVision OV2710 CMOS sensor, Full HD 1080p resolution, and a 3.6 mm M12 lens. The camera is directed towards the convex mirror to produce a 360° view of the robot's surroundings in a single image frame, as shown in Figures 3 and 4.

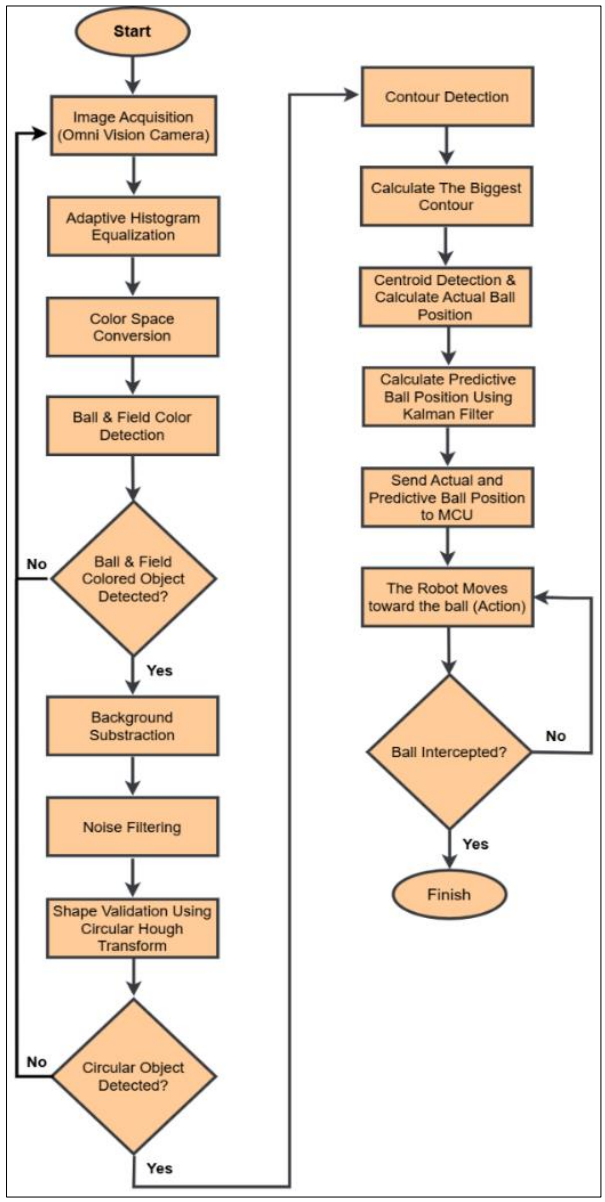

Figure 2. The overall methodology of the proposed system

Figure 3. Omnidirectional camera

Figure 4. 360° panoramic view of omnidirectional camera

There are several advantages that omnidirectional camera can offer such as: (1) full field coverage that enables robot to observe the ball, teammates, opponents, goalposts, and field boundaries at the same time without any movement mechanical parts [24]; reduced latency because there are no moving parts so it will eliminate delay; (3) temporal consistency because all its environmental objects/features are captured in the same time so it will support for accurate state prediction.

Following image capture using an omnidirectional camera, the system applies Adaptive Histogram Equalization (AHE) to enhance image contrast with the parameter values 2.0 for clip limit and 8 × 8 for grid size. This preprocessing technique adaptively redistributes pixel intensity values to achieve a uniform histogram, thereby improving the visibility of objects in both underexposed and overexposed image regions. This transformation maps the original intensity values to new values that approximate a uniform distribution, effectively spreading the most frequent intensity values and compressing less frequent ones. The result is enhanced local contrast, which is particularly beneficial for detecting the ball under varying illumination conditions commonly encountered on soccer fields, such as shadows, reflections, and changing sunlight angles [25].

After the AHE process, the system converts the color space from RGB to HSV. This conversion is important because RGB is more sensitive to lighting variations, while HSV separates color or hue information from intensity or value. Thus, the ball color detection process can run more stable. In the implementation in OpenCV, the HSV calibration range is shown in Table 3.

Table 3. Hue, Saturation, Value (HSV) calibration ranges

Object	Color Channel	Lower Threshold	Upper Threshold
Orange Ball	Hue (H)	10	25
	Saturation (S)	55	89
	Value (V)	88	255
Green Field	Hue (H)	47	99
	Saturation (S)	39	99
	Value (V)	58	255

3.2 Ball and field color detection with background subtraction

After conversion of color space, the system conducts ball and field color detection using a thresholding method. This dual detection method results several purposes: detecting the ball as main target object and setting the green field to determine the boundary of play areas.

For each pixel in the image, the system examines whether the HSV color values are inside the ranges. Pixels that fulfill the conditions are set as potential ball or field pixels that will results binary masks for each object. If the color detection process fails to detect both ball and field colors so the system activates background subtraction process. This method removes non-relevant objects from the image by subtracting current frame with a background model [26]. Specifically, the frame differencing method computes the absolute difference between the current frame $I_t(x, y)$ and the reference background $B_t(x, y)$, expressed as:

$$D_t(x,y) = |I_t(x,y) - B_t(x,y)| \quad (1)$$

The difference image obtained from the above formula is then thresholded to generate a binary mask, where pixels with values higher than the threshold are classified as foreground (moving objects), while the remaining pixels are considered background. This method detects moving regions in the scene and removes static regions, making it suitable for real-time object detection in dynamic environments. After the background subtraction phase, the system returns to the color detection phase, forming an iterative loop that continues until the ball is detected. This closed-loop approach ensures robust performance even in crowded environments. The results of the background subtraction phase, including the remaining noise, are shown in Figure 5.

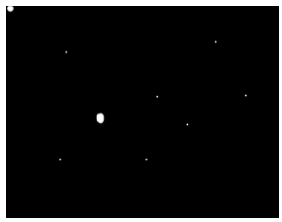

Figure 5. Background subtraction result with its noise

3.3 Noise filtering and shape validation using Circular Hough Transform

After the color detection and background subtraction phase, the system implements noise filtering that can remove small dots, artifacts, and noise caused by camera sensor. The noise filtering phase uses opening and closing as part of morphological operations. Opening performs erosion process followed by a dilation process in order to smooth object contours, while closing performs dilation process followed by an erosion process in order to fill in small holes and gaps. In the system implementation, opening and closing use a 5 × 5 rectangular Kernel. The HSV mask of the orange ball is also cleaned using 3 × 3 erosion and 5 × 5 dilation. Based on the test, the closing operation produces a more intact ball contour than opening. The result of noise filtering is shown below in Figure 6.

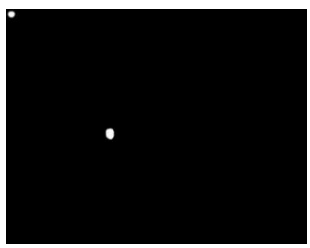

Figure 6. Noise filtering result

After noise filtering, the system implements the CHT process, a feature extraction method that designed to detect circular or ellipse pattern in images. The CHT is suitable for ball detection because ball appears as circle or ellipse in the image plane [27].

The CHT results several parameters of the detected circles: coordinates (x_c, y_c) and radius r. If CHT detects one or more than one circles so the system conducts contour detection process. If CHT cannot detect circular objects so the system returns to Stage 1 (image acquisition), it creates feedback loop until circular objects are detected. CHT runs on several parameters as shown in Table 4 below.

Table 4. Circular Hough Transform (CHT) parameter configuration

Parameter	Argument	Configured Value
Detection Method	method	HOUGH_GRADIENT (Advanced)
Accumulator Resolution	dp	1
Minimum Distance Between Centers	minDist	85 pixels
Canny Edge Detector Higher Threshold	param1	100
Accumulator Threshold for Candidate Centers	param2	30
Minimum Circle Radius	minRadius	10 pixels
Maximum Circle Radius	maxRadius	50 pixels

3.4 Centroid detection and estimated ball position calculation using third-order polynomial equation

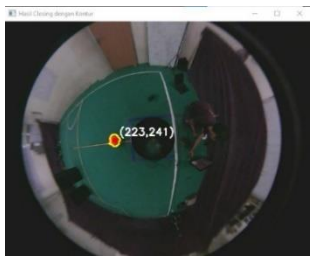

Figure 7. Centroid detection result

After the shape validation phase is successful, the system conducts contour detection to detect the boundaries of the detected ball precisely. Contour detection algorithms track the external boundaries of interconnected regions inside a binary image. The result of contour detection is a list of contours, where every contour is defined as vector of (x, y) points. Then the system chooses contours based on its properties such as: Area, Perimeter, Circularity, Convexity, and solidity [28]. The contour retrieval component and simple contour approximation method are utilized for contour parameters setting. Then the system finds the largest contour by comparing the area of each detected contour. This method is based on the reasonable assumption that in the robot's field of view, the ball is the largest object that the robot can see. The robot targets the largest detected ball and tracks it. The result of centroid detection of the ball is shown in Figure 7.

After the largest contour is detected, the system calculates the centroid of the largest contour to determine its center coordinate. This can be calculated by using image moments, where the spatial moments M_{00}, M_{10}, and M_{01} are calculated from the region of the largest contour. The centroid coordinates (x_c, y_c) are then obtained using:

$$x_c = \frac{M_{10}}{M_{00}}, y_c = \frac{M_{01}}{M_{00}} \quad (2)$$

where, M_{00} shows the area of the largest contour, while M_{10} and M_{01} show to the first-order moments along the x and y axes, respectively. This centroid of the largest contour shows the center of detected ball in the field and is used as reference coordinate point for tracking the ball.

After centroid detection of the ball, the system converts the ball position in the image into real-world coordinates (cm). This conversion process is important because the use of convex mirror as reflector produces high nonlinear distortion that causing the Euclidean distance in pixels (d_p) to increase in nonlinear way with actual distance (d_{cm}). It differs with conventional pinhole cameras; the convex mirror cannot be approximated by linear or inverse model. The resulting third-order polynomial equation to convert Euclidean distance in pixels to actual distance in cm is:

$$d_{cm} = -0.000108 \cdot d_p^3 + 0.0429 \cdot d_p^2 - 4.13 \cdot d_p + 135.8 \quad (3)$$

3.5 Ball position prediction using Kalman filter

The Kalman filter is a recursive algorithm that calculates the state of moving system from measurements that contain noise and incomplete data. In soccer robot applications, the Kalman filter can predict future locations of both the ball position and velocity [29]. The Kalman filter works in two main stages: prediction and update. At the prediction step, this filter utilizes the previous state estimation result to predict the current state of the ball position and velocity. At the update step, the result of prediction state is compared with the actual result that obtained from centroid of detected ball. Kalman filter runs on several parameters as shown in Table 5.

Table 5. Kalman filter parameter and matrix configuration

Parameter	Notation	Argument	Configured Value
State Vector Dimension	x	dynamParams	4 Elements: $[x, y, v_x, v_y]^T$
Measurement Vector Dimension	z	measureParams	2 Elements: $[x_{meas}, y_{meas}]^T$
Control Input Vector	u	N/A	0 (None utilized)
State Transition Matrix	A	transitionMatrix	[1 0 1 0; 0 1 0 1; 0 0 1 0; 0 0 0 1] (Constant Velocity)
Measurement Matrix	H	measurementMatrix	[1 0 0 0; 0 1 0 0]
Process Noise Covariance	Q	processNoiseCov	$10^{-4}\times$ I₄ (Diagonal Matrix)
Measurement Noise Covariance	R	measurementNoiseCov	$10^{-1}\times$ I₂ (Diagonal Matrix)

Comparing the predicted position with the actual detected position continuously, the Kalman filter refines its estimations over time so it results the ball tracking in smoother and more accurate way. This iterative process makes the robot can

anticipate the ball trajectory, allows more responsive, and precise movement during play. The result of Kalman filter prediction with blue colored circle and the actual ball position with orange colored circle is shown in Figure 8.

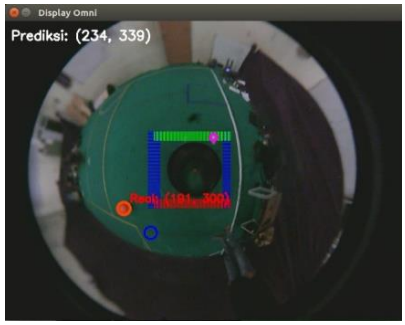

Figure 8. Kalman filter prediction result

3.6 Data transmission and robot action

After calculating the actual and predicted positions of the ball, the system transmits the obtained data to the microcontroller unit (MCU) STM32 F407 VGT6 series. The data transfer between the computer vision processing unit that occurs in the Intel NUC and the MCU for motor control is important for real-time performance, demanding a high-speed serial communication interface configured at 115200 bps to ensure low latency and reliable data transfer between vision and motor control unit.

The MCU processes the received ball position data and triggers commands for the robot's wheels. The robot moves forward to the predicted location rather than moving to actual ball location. This predictive control movement enables smoother and more efficient motion. In addition, by using the predicted position from the Kalman filter, the system decreases the effect of noise and quick changes in the detected position [30, 31]. This results in better navigation and decreases unimportant oscillatory movement. The control algorithm updates the robot trajectory every time based on new data that have been received, enabling adaptive behavior in the environment.

4. RESULT AND DISCUSSION

The experimental results are presented based on distance conversion accuracy and Kalman filter prediction performance.

4.1 Ball detection distance conversion accuracy using third-order polynomial model

To assess the accuracy of the distance conversion from pixel based using Euclidean to real world in centimeters, experiments with certain value conducted using a camera that reflected at convex mirror. The experimental setup involved placing the orange ball at distances ranging from 20 cm to 350 cm from the robot, with increments of 10 cm. For each distance, the Euclidean distance in pixels (d_p) was computed from the detected ball's centroid to the robot center point. The actual distance or real-world distance (d_{cm}) was calculated using the proposed third-order polynomial model that derived from calibration data. The setup of robot to ball distance measurement is shown in Figure 9.

Figure 9. Robot-to-ball distance measurement setup

Table 6. Distance conversion accuracy using third-order polynomial model

No.	Actual Distance (cm)	Euclidean Distance (px)	Estimated Distance (cm)	Error (cm)	Error (%)
1	20	48	20.12	0.12	0.60
2	30	52	29.89	0.11	0.37
3	40	61	40.23	0.23	0.58
4	50	72	49.78	0.22	0.44
5	60	83	60.34	0.34	0.57
6	70	89	69.67	0.33	0.47
7	80	98	80.51	0.51	0.64
8	90	105	89.42	0.58	0.64
9	100	111	99.76	0.24	0.24
10	110	117	110.23	0.23	0.21
11	120	122	119.67	0.33	0.28
12	130	124	130.42	0.42	0.32
13	140	128	139.58	0.42	0.30
14	150	135	150.89	0.89	0.59
15	160	138	159.34	0.66	0.41
16	170	141	170.23	0.23	0.14
17	180	145	179.67	0.33	0.18
18	190	147	190.78	0.78	0.41
19	200	150	199.45	0.55	0.28
20	210	152	210.89	0.89	0.42
21	220	154	219.34	0.66	0.30
22	230	156	230.23	0.23	0.10
23	240	158	239.67	0.33	0.14
24	250	159	250.78	0.78	0.31

25	260	161	259.45	0.55	0.21
26	270	163	270.89	0.89	0.33
27	280	164	279.34	0.66	0.24
28	290	166	290.23	0.23	0.08
29	300	167	299.67	0.33	0.11
30	310	168	310.78	0.78	0.25
31	320	169	319.45	0.55	0.17
32	330	170	330.89	0.89	0.27
33	340	172	339.34	0.66	0.19
34	350	173	350.23	0.23	0.07

From all candidate models that are evaluated such as linear regression, second-order polynomial, fourth-order polynomial, power law, and rational functions. The third-order polynomial is the lowest error metrics from the entire measurement range. Table 6 shows the distance conversion from the actual distance to the estimated by using third-order polynomial model.

As concluded from Table 6 the third-order polynomial model yielded excellent results between actual and estimated distances from 20–350 cm range. The maximum absolute error is only 0.89 cm, while the Mean Absolute Error (MAE) is 0.48 cm, and the Root Mean Square Error (RMSE) is 0.54 cm. The results shows that the proposed method model results below 1 cm accuracy even though the convex mirror introduces significant nonlinear distortion.

4.2 Comparative performance of candidate models

A comparative analysis was performed against several alternative models, such as linear regression, second-order polynomial, fourth-order polynomial, power law, and rational function, to prove the selection of the third-order polynomial as the best model. Table 7 shows the error metrics for each candidate model.

Table 7. Comparative error metrics of candidate distance conversion models

Model	RMSE (cm)	MAE (cm)	Max. Error (cm)	R^2
Linear (two-point)	68.42	54.37	80.25	0.8912
Second-Order Polynomial	71.15	58.93	167.61	0.8824
Third-Order Polynomial	0.54	0.48	0.89	0.9999
Fourth-Order Polynomial	1.31	1.05	3.10	0.9997
Power Law	347.82	289.46	1041.40	0.9981
Rational Function	2.15	1.72	4.30	0.9994

Note: RMSE = Root Mean Square Error; MAE = Mean Absolute Error

The comparative analysis shows that the third-order polynomial outperformed the other tested models. Linear and second-order polynomial models were less able to represent the nonlinear distortion produced by the convex mirror, while the fourth-order polynomial showed a slight tendency toward overfitting. Although the power law and rational function models are theoretically applicable to some optical systems, they were less suitable for this omnidirectional camera configuration. The third-order polynomial achieved R^2 = 0.9999, indicating that almost all variance in the calibration data was explained by the fitted model. Its, residual error also remained below 1 cm across the tested range of 20–350 cm, with a maximum error of 0.89 cm. Therefore, this model provides the most balanced performance in terms of accuracy, stability, and model complexity.

4.3 Kalman filter trajectory prediction

Figure 10. The experimental setup for all angle and direction scenarios

In a real soccer robot match, the ball continuously moves and changes direction. Therefore, one of the main contributions of this study is the implementation of the Kalman filter to predict the ball motion state. This capability allows the robot to move toward the predicted ball position rather than only reacting to the current detected position. To evaluate this capability, the ball was moved at three trajectory angles, namely 10°, 20°, and 30°, and two directions for each angle to assess how well the prediction algorithm handled lateral motion as seen in Figure 10.

The dynamic performance of the Kalman filter was evaluated across 30 sequential test samples for each trial scenario and 15 trials for different angle and direction. For each trajectory angle, the Euclidean distance between the actual ball coordinate and the predicted coordinate is calculated for each sample. This sequential sample-based evaluation is used to observe how the prediction accuracy and error metrics such as MAE, MSE and RMSE changed during dynamic ball motion. The frame rate was also recorded for each sample to verify whether the prediction process maintained real-time performance during trajectory estimation.

Tables 8 and 9 show the system performance for ball movements directed to the right and to the left of the robot, respectively, and three angular conditions over 15 trials per scenario.

Table 8. System performance for 10°, 20°, 30° to the right of the robot

Trial/ Degree	Av. Acc (%)	Av. Frame Rate	MAE (px)	MSE (px^2)	RMSE (px)
1/10°	88.73	81.73	32.22	1,411.35	37.57
2/10°	88.22	78.90	32.23	1,411.39	37.57
3/10°	89.04	75.10	30.22	1,222.96	34.97
4/10°	87.56	80.50	35.16	1,520.33	38.99
5/10°	89.47	76.70	31.76	1,389.34	37.27
6/10°	88.22	72.40	33.67	1,476.98	38.43
7/10°	89.50	81.80	30.30	1,229.88	35.07
8/10°	86.86	77.80	37.79	1,745.55	41.78
9/10°	87.20	73.90	37.96	1,756.58	41.91
10/10°	87.90	69.50	32.86	1,404.33	37.47
11/10°	89.60	80.40	30.30	1,229.88	35.07
12/10°	89.70	76.50	30.47	1,231.10	35.09
13/10°	89.00	72.60	30.74	1,329.28	36.46
14/10°	88.70	68.70	32.97	1,408.16	37.53
15/10°	88.50	78.23	32.86	1,404.32	37.47
1/20°	82.47	82.46	32.22	1,411.35	37.57
2/20°	85.63	79.53	40.59	1,884.52	43.41
3/20°	85.79	80.66	39.39	1,791.23	42.32
4/20°	85.50	80.37	40.71	1,885.67	43.42
5/20°	85.80	81.33	40.71	1,885.67	43.42
6/20°	83.21	79.16	45.47	2,216.53	47.08
7/20°	85.34	79.20	44.07	2,101.40	45.84
8/20°	83.12	78.84	46.10	2,259.43	47.53
9/20°	83.68	78.65	46.50	2,365.15	48.63
10/20°	84.14	79.16	43.14	2,019.26	44.94
11/20°	85.01	80.63	41.28	1,872.74	43.28
12/20°	85.73	81.20	39.51	1,794.27	42.36
13/20°	83.92	79.86	44.68	2,135.73	46.21
14/20°	84.97	79.79	41.38	1,952.13	44.18
15/20°	85.63	79.70	39.58	1,795.99	42.37
1/30°	72.76	79.51	70.34	5,417.71	73.60
2/30°	71.98	78.45	71.88	5,794.54	76.12
3/30°	70.12	76.45	76.45	6,386.93	79.91
4/30°	72.04	75.01	72.06	5,836.87	76.39
5/30°	71.04	76.54	74.77	6150.04	78.42
6/30°	70.89	79.93	76.28	6452.63	80.32
7/30°	71.93	83.88	72.42	5885.64	76.71
8/30°	71.38	73.13	75.83	6393.59	79.95
9/30°	71.93	76.67	74.57	6128.52	78.28
10/30°	71.91	76.41	73.01	5946.06	77.11
11/30°	71.91	75.34	72.5	5900.56	76.81
12/30°	70.99	77.2	77.04	6351.49	79.69
13/30°	71.88	76.65	72.53	5923.63	76.96
14/30°	71.71	75.83	74.3	6092.74	78.05
15/30°	71.08	77.88	75.81	6375.09	79.84

Note: RMSE = Root Mean Square Error; MAE = Mean Absolute Error; MSE = Mean Squared Error

Table 9. System performance for 10°, 20°, 30° to the left of robot

Trial/ Degree	Av. Acc (%)	Av. Frame Rate	MAE (px)	MSE (px^2)	RMSE (px)
1/10°	88.43	83.81	40.91	1857.27	43.09
2/10°	87.31	78.58	43.57	1981.04	44.5
3/10°	86.95	81.45	45.19	2113.57	45.97
4/10°	87.15	79.8	44.39	2051.5	45.29
5/10°	85.51	85.34	51.4	2679.57	51.76
6/10°	87.36	80.02	43.58	1982.27	44.52
7/10°	86.91	84.12	45.6	2151.1	46.38
8/10°	86.98	82.78	46.94	2321.51	48.18
9/10°	86.25	83.36	48.62	2419.64	49.18
10/10°	85.58	83.36	51.68	2708.77	52.04
11/10°	84.5	81.7	60.86	4286.96	65.47
12/10°	87.01	81.36	45.42	2137.94	46.23
13/10°	86.93	81.1	45.19	2113.57	45.97
14/10°	84.91	87.07	59.08	3636.46	60.3
15/10°	84.95	81.91	54.34	3110.13	55.7
1/20°	85.13	85.45	51.46	2990.22	54.68
2/20°	89.52	81.23	48.28	2565.12	50.64
3/20°	88.34	82.05	49.92	2739.73	52.34

4/20°	88.94	82.2	49.33	2697.8	51.94
5/20°	88.14	79.03	51.27	2916.11	54
6/20°	89.17	82.64	51.5	2947.55	54.29
7/20°	88.54	79.87	51.97	2976.31	54.55
8/20°	89.19	79.23	49.29	2658.55	51.56
9/20°	88.29	83.3	52.44	3031.96	55.06
10/20°	87.62	79.42	53.73	3167.3	56.27
11/20°	89.65	82.59	49.81	2800.74	52.92
12/20°	88.19	77.11	53.11	3107.1	55.74
13/20°	87.72	79.33	51.53	2919.46	54.03
14/20°	89.67	78.99	47.34	2489.99	49.89
15/20°	88.8	79.83	52.17	3016.46	54.92
1/30°	72.06	80.4	70.92	5644.32	75.12
2/30°	77.29	81.05	69.84	5477.27	74.01
3/30°	77.77	74.95	69.94	5493.42	74.12
4/30°	76.34	75.98	69.38	5413.43	73.58
5/30°	78.61	74.34	69.76	5468.2	73.95
6/30°	77.31	78.43	69.85	5481.56	74.04
7/30°	75.89	77.41	69.83	5479.16	74.02
8/30°	78.76	78.43	70.18	5533.47	74.39
9/30°	76.37	79.13	71.59	5740.16	75.76
10/30°	78.4	75.98	69.41	5418.06	73.61
11/30°	77.47	75.3	70.84	5633.88	75.06
12/30°	77.33	77.02	69.76	5474.31	73.99
13/30°	78.13	78.29	72.27	5862.99	76.57
14/30°	76.21	77.19	70.35	5560.02	74.57
15/30°	80.06	77.9	70.14	5527.01	74.34

Note: RMSE = Root Mean Square Error; MAE = Mean Absolute Error; MSE = Mean Squared Error

To provide a visual comparison of system performance between rightward (R) and leftward (L) movements, Figures 11-15 present the trends in average accuracy, average frame rate, MAE, MSE, and RMSE across all angle. Figure 11 shows the average accuracy for both directions, while Figure 12 illustrates the average frame rate. Figures 13-15 display the MAE, MSE, and RMSE values, respectively, for the same angular conditions.

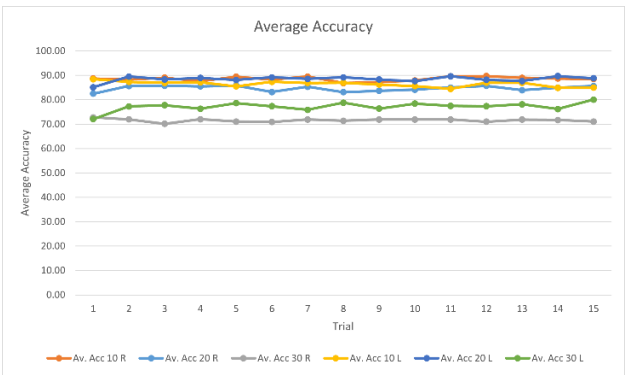

Figure 11. Average accuracy for all scenarios

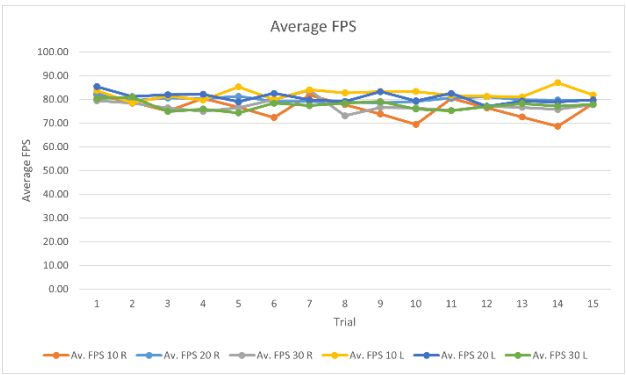

Figure 12. Average frame rate for all scenarios

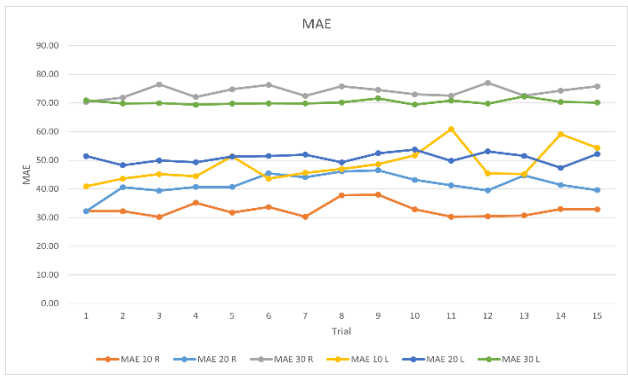

Figure 13. Mean absolute error (MAE) value for all scenarios

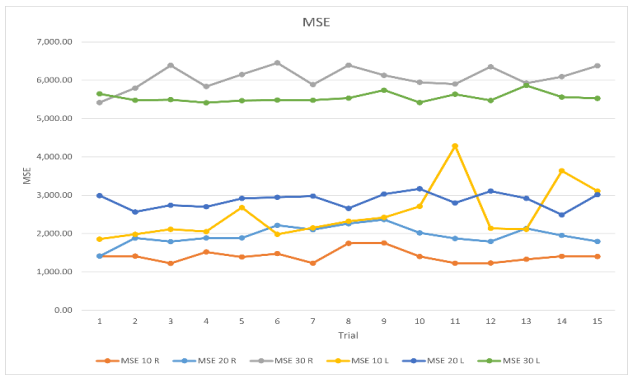

Figure 14. Mean Squared Error (MSE) value for all scenarios

To provide a brief overview of the system's directional performance, Table 10 summarizes the average values of accuracy, frame rate, MAE, MSE, and RMSE for movements to the right and left of the robot at 10°, 20°, and 30°. These values are derived from the 15 trials presented in Tables 8 and 9, showing a clear comparison between rightward (R) and

leftward (L) conditions across increasing angles.

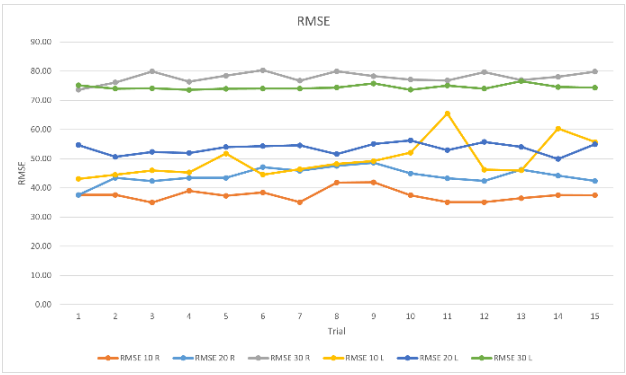

Figure 15. Root Mean Square Error (RMSE) value for all scenarios

Table 10. Average system performance for all scenarios

Dir./ Degree	Acc (%)	Frame Rate	MAE (px)	MSE (px^2)	RMSE (px)
R/10°	88.55	76.32	32.77	1,411.43	37.51
R/20°	84.66	80.04	41.69	1,958.07	44.17
R/30°	71.57	77.26	73.99	6,069.08	77.88
L/10°	86.44	82.38	48.45	2,503.42	49.64
L/20°	88.46	80.82	50.88	2,868.29	53.35
L/30°	76.95	77.45	70.27	5,547.15	74.48

Note: RMSE = Root Mean Square Error; MAE = Mean Absolute Error; MSE = Mean Squared Error

The average system performance shows an inverse correlation between tracking accuracy and spatial error metrics across all scenarios. The proposed system shows peak tracking fidelity at a minor rightward deviation (R/10°), yielding the highest accuracy of 88.55% alongside minimal coordinate drift (MAE = 32.77 px, RMSE = 37.51 px). Conversely, wider angular variation significantly degrades tracking stability; the lowest performance is recorded at R/30°, where accuracy drops to 71.57% due to a sharp change in spatial errors (MAE = 73.99 px, RMSE = 77.88 px). Despite these spatial variances, the computational throughput average remains highly stable between 76.32 FPS and 82.38 FPS, successfully validating the system's robustness for real-time processing.

4.4 Discussion

The benchmarking results between these methods are presented in Table 1. Table 1 compares the performance of four different frameworks—YOLOv8, YOLO with Optical Flow, YOLO-NAS, and the proposed method—in terms of vision methods, embedded hardware devices, frame rate, tracking accuracy, and deployment feasibility on real prototypes.

From the accuracy perspective, the proposed method achieves the highest tracking accuracy of 82.81%, significantly outperforming YOLOv8 [19] (51%) and YOLO-NAS [21] (48%). This substantial accuracy improvement indicates that the combination of traditional color segmentation (HSV) and geometric feature extraction (CHT), refined by temporal state estimation (Kalman filter), can effectively detect the ball without requiring heavy, data-driven deep hierarchical feature learning. The lower accuracy scores observed in deep learning methods [19, 21] suggest that lightweight object detection models struggle to maintain high tracking stability under rapid motion or resource-constrained environments unless extensively trained on massive, domain-specific datasets.

In terms of computational efficiency, the proposed framework exhibits an outstanding performance profile, achieving the highest frame rate of 79.04 FPS on an Intel NUC Mini PC. This processing speed drastically outperforms YOLOv8 [19] (30 FPS), YOLO-NAS [21] (21 FPS), and YOLO + Optical Flow [20] (15 FPS). While reference [19] utilizes a high-end dedicated GPU setup (Ryzen 7 5800H + NVIDIA RTX 3060) to achieve a standard 30 FPS, it fails to deliver acceptable accuracy and lacks a physical prototype implementation. Conversely, though models like YOLO-NAS [21] are deployed on specialized edge-AI platforms like the Jetson Nano, they suffer from low computational throughput (21 FPS) and inadequate accuracy (48%) due to the heavy overhead of deep convolutional layers on edge hardware.

The proposed method demonstrates a highly balanced and superior performance profile. By avoiding resource-intensive neural network inference, it achieves a nearly threefold increase in frame rate compared to edge-deep learning models, while simultaneously boosting accuracy to 82.81%. This throughput ensures seamless real-time operation well within autonomous wheeled soccer robot requirements. Furthermore, unlike deep learning approaches that remain restricted to simulation or suffer on edge computing modules, the proposed method successfully translates its high computational efficiency into a fully functional, validated real prototype system. The proposed method results the average computational latency per frame: HSV Segmentation and CHT validation (7.2 ms), Kalman filter state update (1.1 ms), and third-order polynomial prediction (3.6 ms), resulting in a total average per-frame latency of approximately 11.9 ms (supporting the 79.04 fps processing rate). The experimental results show that the third-order polynomial model results very good distance conversion accuracy under convex mirror distortion, yielding an RMSE of only 0.54 cm and an MAE of 0.48 cm across the 20–350 cm testing range. This precision is below 1 cm and especially significant because convex mirrors produce complex nonlinear distortion patterns that challenge conventional optical models. The analysis was compared with lower-order models (linear and second-order polynomial) and revealed that both failed fatally, with RMSE values more than 68 cm, as they cannot capture the saturation characteristics where pixel distance increases at a reducing rate relative to actual distance. In contrary, the fourth-order polynomial showed small overfitting (RMSE = 1.31 cm), while the power law (RMSE = 347.82 cm) and rational function (RMSE = 2.15 cm) proved unsuitable for this specific convex mirror configuration. These parameter values verify that the third-order polynomial shows the optimal trade-off between model complexity and generalization capability for convex mirror-based distance estimation.

5. CONCLUSION

This study showed a comprehensive approach for ball detection, distance conversion, and trajectory prediction in wheeled soccer robots that used a camera system reflected to convex mirror. The proposed method combines a third-order polynomial model to convert a distance and integrated HSV, CHT and Kalman filter that used to get real-time ball detection and trajectory prediction. According to the experimental

results and discussion, the following conclusions can be obtained.

The standard tracking accuracy achieved by conventional object detection methods under heavy convolutional models remains unsatisfactory, yielding only 51% for YOLOv8 and 48% for YOLO-NAS. By employing traditional color segmentation combined with circular feature extraction and utilizing Kalman filter prediction, the proposed HSV-CHT-Kalman method demonstrates a strong balance between high tracking accuracy (82.81%), good computational throughput (79.04 FPS), and physical deployment feasibility, vastly outperforming existing deep learning-based frameworks in terms of overall suitability for real-time robotic tracking systems.

These results validate the effectiveness of the proposed approach for real-time applications, particularly in scenarios where onboard computational resources are limited and highly reliable target localization is critical. The experiment results show that the wheeled soccer robot can track the target appropriately according to field scenarios, including varying angular displacements. Based on the benchmarking and experimental results, it was concluded that the proposed low-overhead system can be used as an alternative high-performance vision tracking method without requiring expensive GPU acceleration. In future research, we will investigate this optimized vision framework to control multiple collaborative autonomous robots to support advanced product innovation as stated in the ninth target of SDGs.

ACKNOWLEDGMENT

The authors would like to express their gratitude and acknowledge to the University of Trunodjoyo Madura (UTM), Indonesia and INTI International University, Malaysia for supporting the international collaboration research in 2025 (Grant 346/UN46.4.1/PT.01.03/RISMAN/2025).

REFERENCES

[1] Costa, E. (2024). Industry 5.0 and SDG 9: A symbiotic dance towards sustainable transformation. Sustainable Earth Reviews, 7: 4. https://doi.org/10.1186/s42055-024-00073-y

[2] Othman, U., Yang, E. (2023). Human–robot collaborations in smart manufacturing environments: Review and outlook. Sensors, 23(12): 5663. https://doi.org/10.3390/s23125663

[3] Jiang, T.Y., Zhang, S.L., Wang, R., Wang, S. (2023). Development and verification of an autonomous and controllable mobile robot platform. Mechatronics and Intelligent Transportation Systems, 2(1): 11-19. https://doi.org/10.56578/mits020102

[4] Stone, P., Behnke, S., Cohen, J.S., Kruijff, G.J.M., Visser, D.L. (2024). The human in the loop: Perspectives and challenges for RoboCup 2050. Autonomous Robots, 48(1): 25-40. https://doi.org/10.1007/s10514-024-10159-3

[5] Rodriguez, N., Chen, C.L.P., Almeida, M.G.S. (2025). Designing offensive robot soccer strategies through tactical performance indicators: A human-inspired approach. Journal of Intelligent & Robotic Systems, 111(2): 88-105. https://doi.org/10.1007/s10846-025-02334-0

[6] Han, X., Wang, Q., Wang, Y. (2024). Ball tracking based on multiscale feature enhancement and cooperative trajectory matching. Applied Sciences, 14(4): 1376. https://doi.org/10.3390/app14041376

[7] Shilpa, V., Swetha, N., Thippeswamy, B.K., Vidhyashree, V., Srija, D. (2025). Autonomous soccer robot using AI technology. International Journal of Advanced Research in Computer and Communication Engineering, 14(12): 64-70. https://doi.org/10.17148/IJARCCE.2025.141211

[8] Liu, T., Wang, Z., Hu, J., Zeng, S., Liu, X., Zhang, T. (2025). Adaptive motion planning leveraging speed-differentiated prediction for mobile robots in dynamic environments. Applied Sciences, 15(13): 7551. https://doi.org/10.3390/app15137551

[9] Bonar, B., Ambrożkiewicz, M., Wawro, M., Buratowski, T., Małka, P. (2025). Predictive navigation of mobile robots in dynamic environments: A UKF–APF approach. Electronics, 14(19): 3810. https://doi.org/10.3390/electronics14193810

[10] Kang, H.C., Han, H.N., Bae, H.C., Kim, M.G. (2021). HSV color-space-based automated object localization for robot grasping without prior knowledge. Applied Sciences, 11(16): 7593. https://doi.org/10.3390/app11167593

[11] Oleynikov, A.A., Palchevsky, E.V. (2026). Adaptive HSV segmentation for real-time object detection under varying lighting conditions. Information Technology, 32(1): 46-56. https://doi.org/10.17587/it.32.46-56

[12] Hikmahwan, B., Hario, F., Mudjirahardjo, P. (2023). Ball detection based on color and shape features captured by omni-directional camera. In 2023 International Seminar on Intelligent Technology and Its Applications (ISITIA), Surabaya, Indonesia, pp. 642-647. https://doi.org/10.1109/ISITIA59021.2023.10221097

[13] Yang, Y., Kim, D., Choi, D. (2023). Ball tracking and trajectory prediction system for tennis robots. Journal of Computational Design and Engineering, 10(3): 1176-1184. https://doi.org/10.1093/jcde/qwad054

[14] Khan, S., Ali, S., Iqbal, K.F., et al. (2025). Active interception of moving ball: A multi-player strategy for humanoid soccer robots. Multimedia Tools and Applications, 84(27): 32487-32503. https://doi.org/10.1007/s11042-024-20491-6

[15] Archana, D., Radhika, V. (2026). Visual intelligence in resource-constrained edge devices: A review of Raspberry Pi. International Journal of Innovative Science and Research Technology, 11(3): 187-199. https://doi.org/10.38124/ijisrt/26mar114

[16] Kao, S.T., Ho, M.T. (2021). Ball-catching system using image processing and an omni-directional wheeled mobile robot. Sensors, 21(9): 3208. https://doi.org/10.3390/s21093208

[17] Rahmat, F., Wibisana, A., Fauzi, M.R., Aliman, M., Firmansyah, N. (2024). Implementation of intercept ball algorithm for wheeled soccer robot Barelang63. In 7th International Conference on Applied Engineering (ICAE 2024), Batam, Indonesia, pp. 115-121. https://doi.org/10.2991/978-94-6463-620-8_15

[18] Farhan, T.M., Candra, F. (2024). CNN-based ball and goal detection for KRSBI robot with omnidirectional camera. International Journal of Electrical, Energy and Power System Engineering, 8(2): 86-98.

https://doi.org/10.31258/ijeepse.8.2.1-13
[19] Habibi, A., Badri, F., Faisal, A.Q., Maulana, B.R. (2025). Implementation and analysis of a computer vision-based ball detection system for robot soccer in ROS simulation. International Journal of Artificial Intelligence & Robotics, 7(1): 26-34. https://doi.org/10.25139/ijair.v7i1.10502
[20] Bordado, M., Silveria, D., Laureano, G.T. (2023). Ball detection and tracking with different embedded systems in the RoboCup soccer context. In 2023 Latin American Robotics Symposium (LARS), 2023 Brazilian Symposium on Robotics (SBR), and 2023 Workshop on Robotics in Education (WRE), Salvador, Brazil, pp. 514-519. https://doi.org/10.1109/LARS/SBR/WRE59448.2023.10333050
[21] Jati, H., Ilyasa, N.A., Dominic, D.D. (2024). Enhancing humanoid robot soccer ball tracking, goal alignment, and robot avoidance using YOLO-NAS. Journal of Robotics and Control, 5(3): 829-838. https://doi.org/10.18196/jrc.v5i3.21839
[22] Long, Y., Pan, L. (2024). Research on fast recognition method for robot soccer visual images. In 2024 International Conference on Computers, Information Processing and Advanced Education (CIPAE), pp. 130-134. https://doi.org/10.1109/CIPAE64326.2024.00130
[23] Sridharan, H.M.G. (2024). A lightweight convolutional network for deep learning-based ball manipulation by a soccer playing robot. In 2024 3rd International Conference on Automation, Computing and Renewable Systems (ICACRS), Ottawa, ON, Canada, pp. 1-6. https://doi.org/10.1109/ICACRS62842.2024.10841538
[24] Lin, H.Y., He, C.H. (2021). Mobile robot self-localization using omnidirectional vision with feature matching from real and virtual spaces. Applied Sciences, 11(8): 3360. https://doi.org/10.3390/app11083360
[25] Pratama, F.P., Widiarti, A.R. (2025). Enhancing visibility in low-illumination street images using HE, AHE, and CLAHE techniques. Journal of Informatics and Telecommunication Engineering, 9(1): 1-10. https://doi.org/10.31289/jite.v9i1.15451
[26] Zhang, Q.L., Li, S.L., Duan, J.G., Qin, J.Y., Zhou, Y. (2024). Moving object detection method based on the fusion of online moving window robust principal component analysis and frame difference method. Neural Processing Letters, 56: 55. https://doi.org/10.1007/s11063-024-11463-w
[27] Aji, S.A., Pradana, E.Y., Abdurrozaq, M.A., Risnumawan, A., Pitowarno, E., Sudaryo, A. (2024). Rapid goalpost detection through candidate generation and Hough Transform in humanoid soccer robots. In 2024 International Electronics Symposium (IES), Denpasar, Indonesia, pp. 248-253. https://doi.org/10.1109/IES63037.2024.10665837
[28] He, Y., Kang, S.H., Morel, J.M. (2023). Topology- and perception-aware image vectorization. Journal of Mathematical Imaging and Vision, 65(6): 874-893. https://doi.org/10.1007/s10851-023-01149-8
[29] Steffi, D., Mehta, S., Venkatesh, V. (2022). Object detection on robosoccer environment using convolution neural network. Indonesian Journal of Electrical Engineering and Computer Science, 29(1): 286-294. https://doi.org/10.11591/ijeecs.v29.i1.pp286-294
[30] Huang, Z., Xu, Q., Sun, M., Zhu, X., Fan, S. (2025). Adaptive Kalman filtering localization calibration method based on dynamic mutation perception and collaborative correction. Entropy, 27(4): 380. https://doi.org/10.3390/e27040380
[31] Lai, N., Dewi, D.A., Maidin, S.S., Xiao, W., Zhao, S., Hu, Q. (2026). A comprehensive review of lightweight deep learning models for edge computing with future directions. Discover Computing, 29: 110. https://doi.org/10.1007/s10791-026-10021-3