Augmented haar cascade classifier for real-time ball detection in humanoid robots under dynamic environments

Gembong Edhi Setyawan [a,1], Edita Rosana Widasari [a,2,*], Barlian Henryranu Prasetio [a,3], Yasa Palaguna Umar [b,4], Ivan Rafli Adipratama [a,5]

[a] Department of Informatics Engineering, Faculty of Computer Science, Universitas Brawijaya, Indonesia
[b] Department of Environmental Engineering, Faculty of Agricultural Technology, Universitas Brawijaya, Indonesia
[1] gembong@ub.ac.id; [2] editarosanaw@ub.ac.id; [3] barlian@ub.ac.id; [4] yasaumar@ub.ac.id; [5] ivanrafli14@gmail.com
* corresponding author

ARTICLE INFO

Article history
Received July 4, 2025
Revised October 18, 2025
Accepted October 26, 2025
Available online February 28, 2026

Keywords
Real-time detection
Humanoid robots
Augmented haar cascade classifier
HSV segmentation
Convex Hull mapping

ABSTRACT

This study proposes an Augmented Haar Cascade Classifier (AHCC) to enhance real-time ball detection for humanoid robots operating in dynamic environments. The method integrates Convex Hull mapping, HSV-based segmentation, and Hough Circle validation to overcome challenges such as fluctuating illumination, complex backgrounds, and partial occlusions. Experiments were conducted entirely on a CPU-only Intel NUC platform running ROS without GPU acceleration, using a dataset containing variations in lighting, orientation, scale, and background clutter. Compared with baseline models (standard Haar Cascade Classifier (HCC) and YOLOv5) the proposed AHCC achieved 97% accuracy, 83% recall, 97% precision, and an 89% F1-score, while requiring only 0.00849 s per frame with 8.97% memory usage. Although YOLOv5 reached 99% accuracy, it demanded higher computational resources (0.0344 s per frame, 22.3% memory usage), limiting its practicality for embedded robotic systems. The AHCC therefore offers an optimal balance between detection reliability and computational efficiency, outperforming traditional HCC and providing a lightweight alternative to GPU-dependent detectors such as Tiny-YOLO and MobileNet-SSD.

1. Introduction

Humanoid robots are increasingly utilized in various applications requiring dynamic interaction with human environments [1], [2], such as healthcare, where they assist in physical therapy and patient monitoring [3]; retail, providing personalized shopping experiences and inventory management [4]; and entertainment, offering interactive performances and companionship for the elderly [5], [6]. Their capability to mimic human structure and behavior, coupled with embedded Artificial Intelligence (AI), enables them to perform complex tasks efficiently and adaptively. However, robust real-time object detection remains a key requirement for humanoid robots to operate effectively in dynamic environments [7]. This challenge is intensified in scenarios with fluctuating lighting, unpredictable object movement, and complex or cluttered backgrounds.

One of the most prominent algorithms to be implemented with limited computing resources is the Haar Cascade Classifier (HCC) [8], which is widely adopted for real-time applications due to its rapid detection capabilities in controlled environments. However, HCC struggles to attain high accuracy under variable lighting conditions, background clutter, and object occlusion, which considerably restricts its

practical application in dynamic environments [9]. In contrast, the YOLO (You Only Look Once) framework [10], [11] has gained significant popularity for balancing speed and accuracy in object detection. Early versions such as YOLOv1 [11] and YOLOv2 [12] demonstrated a balance between speed and accuracy, while YOLOv3 [13] and YOLOv4 [14] improved accuracy at the cost of higher computational demands. To address these limitations, YOLOv5 [15] was introduced, exhibiting strong performance in both speed and accuracy [10], [16]. Nevertheless, YOLOv5 still requires substantial computational resources, making it unsuitable for embedded humanoid robots with limited onboard hardware. Unlike prior lightweight CNN backbones (e.g., MobileNetV3 [17]) and hybrid lightweight detectors for embedded or industrial robots (e.g., Kumar et al. [18]), as well as studies emphasizing AI-driven system awareness and compact deployment in service and industrial contexts [19], [20], our proposed AHCC emphasizes CPU-only deployment while maintaining competitive accuracy and real-time performance for humanoid robots operating under strict resource constraints.

Numerous studies have investigated real-time object detection for robotics under resource-constrained conditions. While HCC is valued for its speed, it remains limited in handling dynamic backgrounds and inconsistent lighting [9]. To improve robustness, researchers have proposed hybrid approaches. For instance, [18] introduced a lightweight multi-stage detector for embedded robots that integrates background subtraction and machine-learned filtering. Similarly, the evolution of YOLO-based models has produced highly accurate detectors; however, their deployment in edge environments remains challenging. For example, YOLO-FastestV2 [21] reduces model size and complexity and is designed for lightweight deployment on edge devices, although practical performance still depends on hardware and optimization frameworks, highlighting the need for computationally simpler yet accurate detection models for autonomous humanoid robots.

To address these challenges, the proposed Augmented Haar Cascade Classifier (AHCC) integrates classical HCC with advanced preprocessing and validation steps, including Convex Hull mapping, HSV-based segmentation, and Hough Circle validation. Convex Hull mapping is widely employed in robotics to isolate relevant regions [22] while HSV-based segmentation has proven effective in outdoor robotics for illumination invariance [23]. The Hough Circle Transform further validates circular shapes [24], enhancing detection reliability when the target has known geometry (e.g., a ball). Collectively, these components underpin the AHCC's robustness and suitability for embedded visual detection tasks in robotics. The HSV thresholds were manually tuned for robustness across different environments, with potential adaptive or learning-based improvements highlighted for future work.

The proposed AHCC builds on the theoretical foundation of Haar-like features [8], which enable efficient local contrast detection and mimic early-stage visual processing. By integrating Convex Hull, HSV segmentation, and Hough validation into a cascade framework, AHCC provides a novel, resource-aware approach that balances accuracy and computational efficiency in dynamic settings.

As a case study, we evaluate a humanoid soccer robot, focusing on its ability to detect a ball in dynamic environments. The experiments were conducted on a CPU-only platform, specifically an Intel NUC controller running ROS, without GPU acceleration, to demonstrate the feasibility of AHCC on real embedded hardware. The dataset employed includes diverse variations in lighting, orientation, scale, partial occlusion, and background clutter, with augmentation such as rotation and brightness adjustments to simulate real-world variability. The AHCC is compared against the baseline HCC and the more advanced YOLOv5, which represents a state-of-the-art but resource-intensive solution. The primary contributions of this study are as follows:

- Design and development of the AHCC algorithm optimized for resource-constrained humanoid robots in dynamic environments.
- Integration of Convex Hull mapping, HSV-based segmentation, and Hough Circle validation to improve object localization and minimize background noise, thereby reducing false positives.
- Extensive evaluation of detection accuracy, false positive reduction, and computational efficiency compared with HCC and YOLOv5, with trade-off analysis against other lightweight detectors such as Tiny-YOLO and MobileNet-SSD.

By addressing the limitations of existing approaches, this study aims to advance the state-of-the-art in real-time object detection for humanoid robots, enabling more effective performance in dynamic and resource-constrained environments. The proposed approach also provides insights for broader applications such as agricultural robotics, healthcare assistance, and search-and-rescue, where efficiency and robustness are equally critical.

Unlike prior lightweight or hybrid detectors that still rely on accelerator support (e.g., Tiny-YOLO, MobileNet-SSD, YOLO-FastestV2) and other studies on edge-oriented lightweight detection frameworks [25], [26], our contribution is a CPU-only, resource-aware hybridization that combines Convex Hull + HSV segmentation + Hough validation around a classical HCC cascade to reduce false positives in dynamic scenes, specifies deterministic preprocessing and thresholds suitable for embedded deployment, and demonstrates a quantified accuracy efficiency trade-off on a humanoid robot without a GPU. This CPU-only design eliminates the need for dedicated accelerators, providing a practical, deployable solution for humanoid robots where power consumption, thermal constraints, and latency requirements make deep learning models impractical. The remainder of this paper is organized as follows: Section 2 presents the proposed methods, including the design of the Augmented Haar Cascade Classifier (AHCC), the hardware and software configurations, and the dataset preparation. Section 3 provides the experimental results and discussion, including a comparative performance analysis against baseline models. Section 4 concludes the paper and outlines future research directions

2. Method

2.1. Haar Cascade Classifier (HCC)

HCC is a classical object detection method known for its computational efficiency and suitability for real-time applications. HCC uses Haar-like features, which are rectangular and precise, defined by the pixel intensity difference between two regions. The feature f for a given window W is calculated as (1).

$$f = \sum_{i \in R_1} I(i) - \sum_{j \in R_2} I(j) \quad (1)$$

where R_1 and R_2 are the first two nearest adjacent rectangular regions, then $I(i)$ and $I(j)$ are pixel intensities in R_1 and R_2, respectively. These features are evaluated using the AdaBoost algorithm, which combines weak classifiers $h_t(x)$ into a strong classifier $H(x)$ by the following (2).

$$H(x) = sign\left[\sum_{t=1}^{T} \alpha_t h_t(x)\right] \quad (2)$$

where T is the number of weak classifiers and αt the weight of *the t-th* weak classifier.

As illustrated in Algorithm 1 (Fig. 1), the HCC pipeline consists of: (i) input image, (ii) Haar feature extraction, (iii) AdaBoost classification, and (iv) bounding box placement. This cascade structure allows rapid elimination of non-promising regions and is efficient for real-time detection under constrained resources. However, it still suffers from high false-positive rates in dynamic environments, making it less robust under varying lighting conditions, complex backgrounds, and object occlusions.

```
Function HCC_Object_Detection(image):
1) Extract Haar-like features from sliding windows
2) Train AdaBoost to combine weak classifiers into strong cascade
3) For each window x in image:
    score ← Σ (αt · ht(x))
    If sign(score) = +1:
      Add bounding box (x) to detections
4) Draw bounding boxes on image for all detections
5) Return annotated image
```

Fig. 1. Haar Cascade Classifier (HCC)

2.2. Augmented Haar Cascade Classifier (AHCC)

To address the limitations of HCC, this study proposes an AHCC algorithm that incorporates advanced preprocessing and validation techniques:

- **Convex Hull**: Refines the region of interest by isolating relevant areas and reducing background noise [27], [28].
- **HSV-based Segmentation**: Segments objects based on their color properties, enhancing resilience to lighting variations [29], [30]. As shown in (3), for a pixel $P(h, s, v)$, it is segmented as part of the target if:

$$H_{min} \leq h \leq H_{max}$$

$$S_{min} \leq s \leq S_{max}$$

$$V_{min} \leq v \leq V_{max} \quad (3)$$

- **Circle Hough Transform**: Ensures the detected object's circular shape using (4):

$$(x-a)^2 + (y-b)^2 = r^2 \quad (4)$$

where a and b are the circle's center, and r is the radius [31]–[33].

The overall workflow of the proposed AHCC is summarized in Algorithm 2 (Fig. 2). The process begins with Convex Hull Mapping, which isolates the relevant regions by mapping the field of interest and eliminating unnecessary background elements. Next, HSV-based Segmentation is applied to segment the target object based on its distinct color properties, effectively reducing the impact of lighting variations. The HCC then detects objects within the segmented regions. To ensure robustness, the detected objects undergo additional validation using the Circle Hough Transform Algorithm, which verifies their circular shape (e.g., a ball). Finally, a Bounding Box is drawn to localize the detected object.

This integration significantly enhances detection accuracy and robustness, reducing false positives and improving stability under illumination shifts. Consequently, the AHCC is highly effective for real-time applications in dynamic environments, particularly in robotic soccer, where speed and precision are essential.

```
Function AHCC_Object_Detection(image):
1) Extract region of interest using Convex Hull
2) Apply HSV color segmentation:
Keep pixels where H, S, V within threshold ranges
3) Apply Haar Cascade Classifier on segmented regions to get candidate detections
4) Validate circular shapes using Hough Circle Transform
5) Draw bounding boxes on validated detections
6) Return annotated image
```

Fig. 2. Augmented Haar Cascade Classifier (AHCC)

2.3. Hardware and Software Setup

All experiments were conducted on a CPU-only embedded platform to demonstrate the feasibility of the proposed AHCC for low-resource humanoid robots. The humanoid robot was equipped with an Intel NUC7i3BNK mini-PC (Intel Core i3-7100U, 2.40 GHz dual-core CPU, 8 GB DDR4 RAM) running Ubuntu 16.04 with ROS Kinetic, without GPU acceleration. A Full HD USB camera (1080p, 30 FPS) was mounted on the robot's head to provide real-time visual input. The image-processing pipeline, including Convex Hull, HSV segmentation, HCC, and Hough validation, was implemented in Python 3.6 using OpenCV 3.4 and NumPy. Performance metrics were benchmarked in terms of: (i) average detection time per frame (seconds), (ii) CPU utilization, and (iii) memory usage (%).

All tests were conducted in controlled laboratory environments using datasets and synthetic field conditions that exclude any human data. The experiments comply with ethical guidelines for AI and robotics research [34], ensuring non-invasive, transparent, and reproducible methodology.

2.3.1. System Description

The humanoid robot is constructed using a high-quality Aluminium frame with a thickness of 2 mm, ensuring an optimal balance between strength and lightweight properties. The robot has a total height of 51 cm, with dimensions as shown in Fig. 3(a). This design emulates the human body to enhance stability and enable precise, dynamic movements. A modular approach was implemented in the mechanical design, allowing individual components to be easily replaced in the event of damage, thereby improving maintenance efficiency. The robot's motion is driven by motor servo actuators, which provide real-time control over position, velocity, and torque via TTL or RS485 communication protocols. These actuators enable the robot to perform complex actions such as walking, kicking, and jumping with high precision. A camera is strategically positioned on the robot's head for visual perception. The camera offers a wide field of view and Full HD 1080p resolution, which are pivotal for environmental sensing and object detection, such as identifying a ball, with high accuracy, as shown in Fig. 3(b).

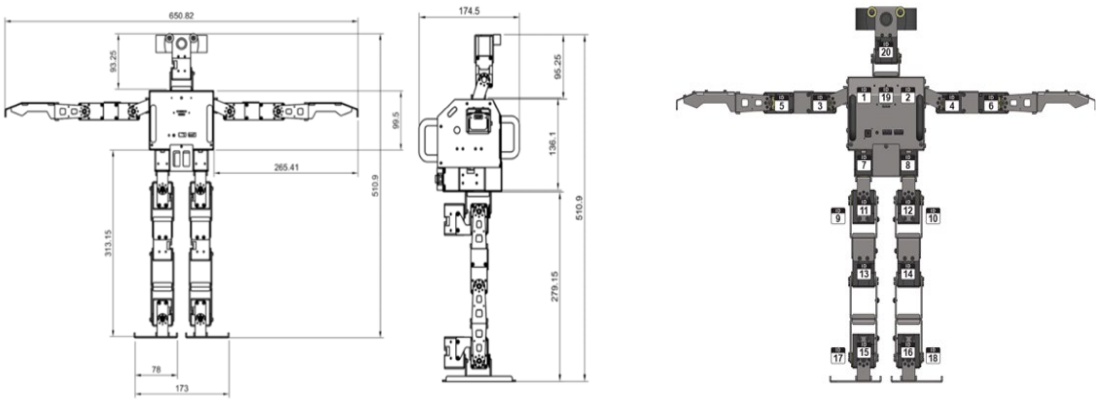


Fig. 3. Robot design (a) 2D (b) 3D

As depicted in Fig. 4, the robot's electrical system comprises a primary controller and integrated subsystems to support its operations. The Intel NUC7i3BNK serves as the main controller; it enables real-time acquisition and processing of visual data using item detection algorithms. The mini-PC has Intel Core i3-7100U as its processor, which takes minimum energy while it is powerful enough to perform various tasks. The OpenCR 1.0 sub-controller controls the motor servos for actuation and communicates with numerous sensors through TTL, RS485, and UART ports. As a power source for the robot, it is equipped with a 2200 mAh LiPo battery, which provides a stable voltage during work. Moreover, a U2D2 module provides an interface between the controller and the servos, allowing fast and accurate movements. All elements stated above are connected using serial communication interfaces and run on Ubuntu 16.04 with ROS Kinetic. Thus, this electrical design aims at energy efficiency, flexibility concerning software formation, and safe work in variable conditions.

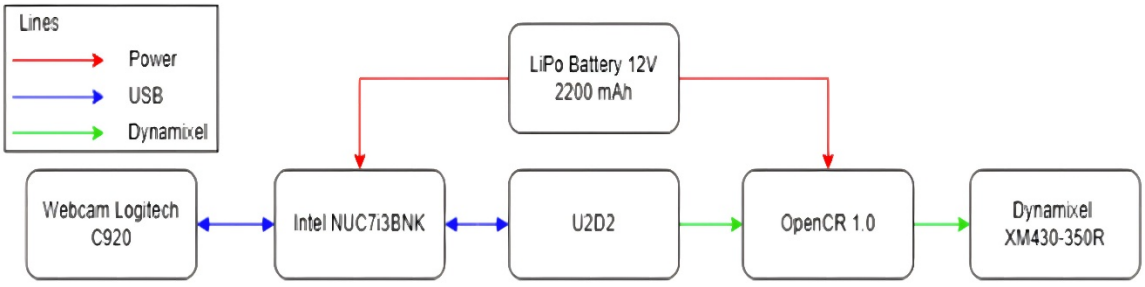


Fig. 4. Robot system electrical

2.3.2. Model Design

The AHCC design integrates enhancements in data preprocessing and real-time object detection to achieve higher accuracy and robustness in detecting dynamic objects. The workflow is structured as follows:

- **Dataset Splitting and Annotation**: The positive dataset is annotated with bounding boxes around the target object, and the results are saved to pos.txt. The negative dataset is saved as neg.txt. The annotated pos.txt file is converted into *.vec format required for training.
- **Model Training**: For each prepared dataset, the model is trained, and based on the changes we made to the preprocessing steps, we have improved the model's accuracy in detecting the target object against background noise. It ends with the final development of the AHCC model.
- **Model Loading and Activation**: The trained model is the final step in a robot system and is used in real time in the next step. The detection program is launched so that the model can perform the ongoing computations on video frames and detect objects.
- **Field Mapping and Segmentation**: A convex hull algorithm is used to map the field environment, delineating boundaries and regions of interest. The model employs HSV-based color segmentation to improve target identification. HSV ranges for ball segmentation were selected via a grid search on the training set to maximize recall while controlling false positives after Hough validation. The final thresholds were: $H \in [10, 35]$, $S \in [60, 255]$, $V \in [60, 255]$ (indoor), and $H \in [8, 40]$, $S \in [50, 255]$, $V \in [70, 255]$ (outdoor). During inference, the system selects the profile based on ambient brightness (median V). We also evaluated ±10% tolerance around these bounds.
- **Object Detection and Tracking**: The model detects objects by identifying their shapes using the Hough Circle algorithm. Detected objects are marked with bounding boxes for visual indication. The ball's position coordinates are published via ROS, and the robot's system subscribes to these topics to dynamically adjust the servo motors and track the object in real-time.

All timings were recorded at 640×480, batch size = 1, single-threaded OpenCV, after a 30-frame warm-up to stabilize caches. Each configuration was run for 1,000 frames over three trials; we report the mean per-frame time and standard deviation. Memory usage was sampled via the OS process monitor (steady-state average over the middle 500 frames). The CPU governor was set to performance to avoid frequency scaling artifacts.

The AHCC workflow demonstrates significant enhancements in preprocessing, segmentation, and detection accuracy. By incorporating HSV conversion and field mapping, the design is well-suited for dynamic environments, ensuring reliable performance and adaptability in object detection tasks. The trained AHCC model was subsequently evaluated in real-time ball detection experiments described in Section 3.

2.4. Dataset Preparation

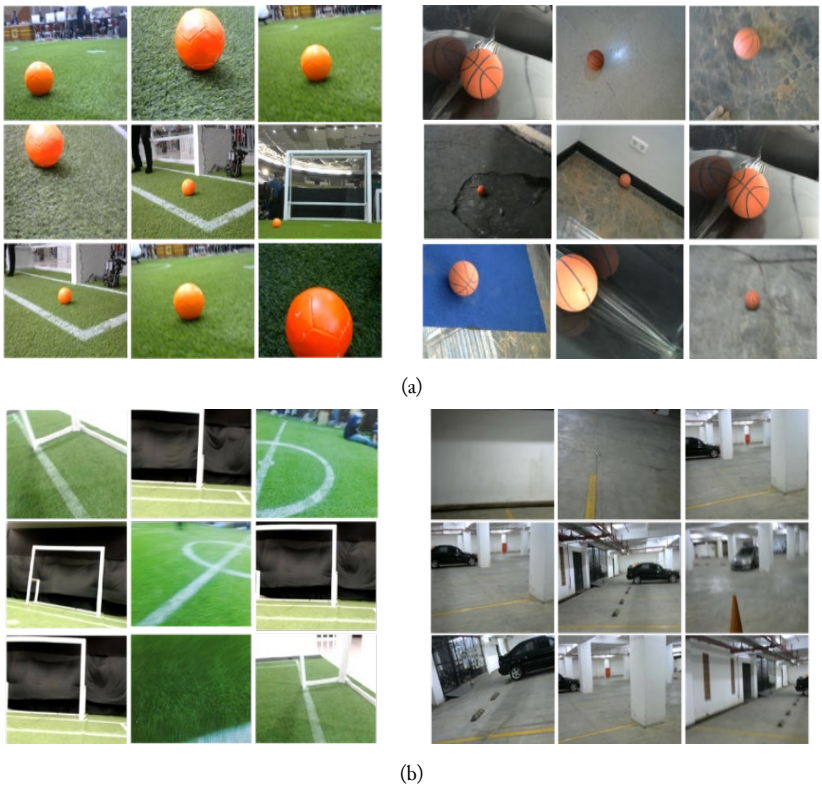
A vast dataset has been carefully constructed to enhance the reliability and flexibility of the proposed AHCC algorithm. This dataset represents dynamic scenes with varying and inconsistent illumination, diverse and cluttered backgrounds, and uncertain object dynamics. They are positive and negative samples, selected from raw data, labelled with close attention, and preprocessed to improve the algorithm's receptive field. The dataset used in this study is publicly available on GitHub at the following link: https://github.com/editarosanaw/Humanoid-Robots-in-Dynamic-Environments/tree/main .

The positive dataset is made of 1,450 images with the target object: a soccer ball photographed under various circumstances. These are variations in lighting conditions, such as light and dark, and mixed lighting conditions and background variations, such as soccer fields with crowded and indoor backgrounds. The variation of object properties introduced in the dataset includes the ball size, orientation, and cases where one ball is partially occluded by the other. Also, to mimic real-life conditions, the ball location in the pictures can be centralized, off-centralized, partially occluded, and

from different angles and distances. Fig. 5(a) shows the appearance of the positive dataset used in the experiment. Positive samples showing soccer balls under various lighting and background conditions, including occlusion and angle variation.

To improve robustness and generalization, data augmentation was also applied to the positive dataset, including random rotation (±20°), horizontal flipping (p=0.5), brightness and contrast adjustment (±15%), Gaussian noise (σ=5), and scaling (0.8–1.2×). These augmentations simulate realistic camera variations during robot motion, ensuring that the detector remains reliable under diverse viewpoints and illumination changes.

The negative set comprises 7,145 images from which the assigned target object is absent. These images include natural backgrounds such as open fields and lawns, indoor or floor surfaces, crowd scenes, and visually similar objects to a soccer ball, which are circular or white. This diversity also helps eliminate false positives, where the algorithm can easily identify the soccer ball from similar non-target objects. The negative dataset is illustrated in Fig. 5(b) that containing background clutter, circular non-ball objects, and diverse textures to simulate real-world distractions



(a)

(b)

Fig. 5. Example images from the training dataset.

For training the algorithm, those images were annotated with bounding boxes around the soccer ball in the positive set. The annotations were made using LabelImg, an open-source tool that helps increase and maintain dataset quality. [1] The dataset was then prepared with a resolution of 640×480. In addition, the dataset was split into training and testing sets, with 70 percent of the data for training and 30 percent for testing, as shown in Table 1. Such a distribution guarantees that the training data are sufficient to train on, and the independent test set provides a reliable assessment of the algorithm.

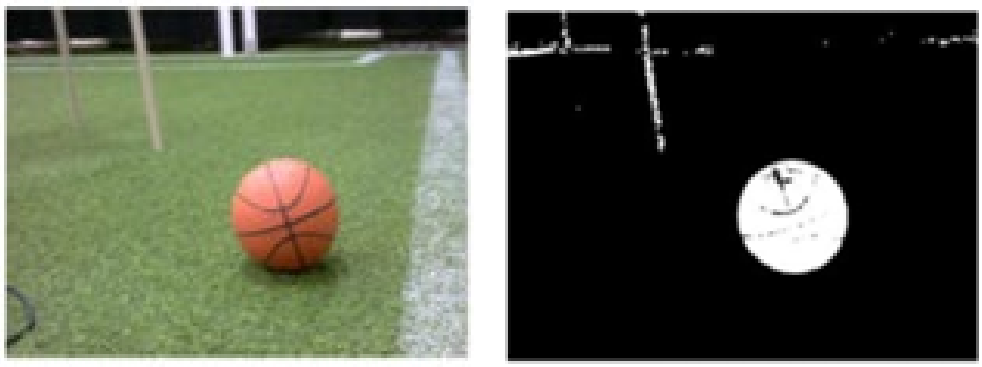
The cascade was trained using OpenCV's traincascade with stages = 20, featureType = HAAR, minHitRate = 0.995, maxFalseAlarmRate = 0.5, maxDepth = 3, and precalcValBufSize = 1024 / precalcIdxBufSize = 1024. Training used positive = 1,020 and negative = 4,996 samples at 640×480 (cropped windows were scaled). Each stage ran for up to 200 iterations or until convergence. The final detector size was 24×24, with a sliding-window multi-scale pyramid during inference.

Table 1. Dataset Distribution

Dataset Type	Positive Samples	Negative Samples	Total
Training	1,020	4,996	6,016
Testing	430	2,149	2,579
Total	**1,450**	**7,145**	**8,595**

3. Results and Discussion

As shown in Fig. 6 and Fig. 7, YOLOv5 operated on RGB inputs to leverage spatial and color information, while HCC and AHCC employed HSV inputs to improve robustness to illumination changes via color segmentation. The HSV-based preprocessing effectively separated the target ball from complex and dynamic backgrounds, demonstrating adaptability to lighting variations. The testing process as shown in Fig. 8 and Fig. 9 evaluated detection performance under diverse scenarios and baseline comparators.



(a) (b)

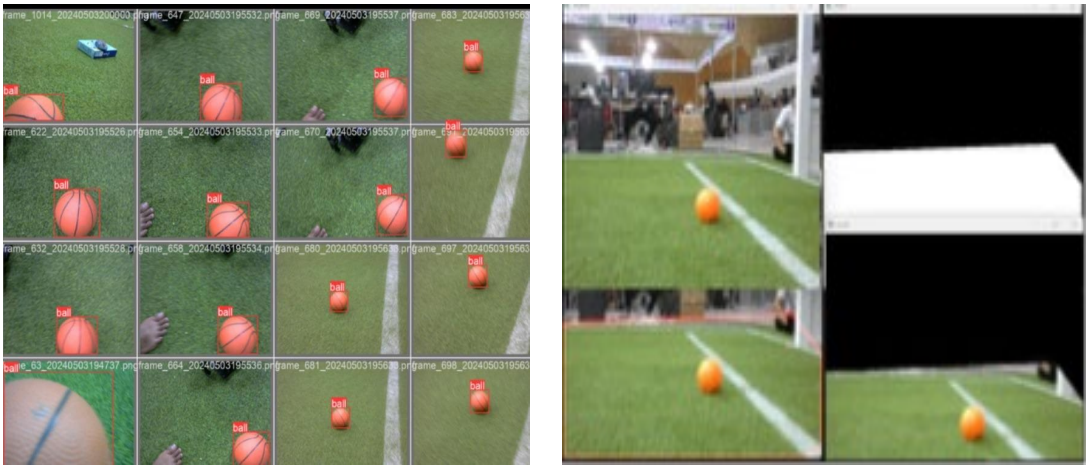
Fig. 6. Visualization of masking results for positive samples: (a) Original RGB image of a soccer ball; (b) Result after HSV segmentation highlighting the ball region while suppressing background noise

(a) (b)

Fig. 7. Comparison of RGB and HSV-based masking on negative samples: (a) RGB image containing circular non-ball objects; (b) HSV masking effectively discards background elements and prevents false positives

Fig. 8 presents representative input images used during testing, where the ball was placed on various surfaces and under different lighting conditions to emulate real-world scenarios. As depicted in Fig. 9, the proposed AHCC tracked and identified the ball in real time. Fig. 9(a) shows sequential frames with bounding boxes labelled "ball," indicating successful localization despite partial occlusions and motion. Fig. 9(b) illustrates the processing pipeline: the right panel shows HSV-based masking that isolates the ball, whereas the left panel shows the final detections with accurate bounding boxes. These results underscore AHCC's robustness to lighting variation and background clutter.

Fig. 8. Example of input images used for testing



(a) (b)

Fig. 9. Performance of the proposed AHCC algorithm in dynamic scenarios: (a) Sequential detection and tracking frames showing successful ball localization with bounding boxes despite occlusion and movement; (b) Side-by-side comparison of input, segmentation, and final detection output

Confusion matrices, as presented in Fig. 10 provide a detailed view of true positives, false positives, and false negatives, revealing each model's error profile under varied conditions. Fig. 10(a) YOLOv5 demonstrates excellent classification accuracy with a near-perfect true negative rate and a very low false positive count, Fig. 10(b) The baseline HCC model shows significant misclassifications, with a high number of false positives and false negatives, Fig. 10(c) The proposed AHCC method achieves improved balance, with a high true negative count and significantly reduced false positives compared to HCC, while also lowering false negatives highlighting the benefit of integrating shape and color features for more reliable detection in dynamic environments.

The performance of AHCC was evaluated using Accuracy, Precision, Recall, and F1-score, as presented in Table 2. Among the evaluated object detection models, YOLOv5 achieved the highest performance, with 99% accuracy, 99% recall, 100% precision, and a 99% F1-score. These results confirm its ability to detect objects with exceptional precision and reliability. However, this superior performance comes at the expense of high computational demands, requiring 0.0344 seconds per frame and consuming 22.3% of memory. Such resource consumption limits its applicability in real-time embedded systems such as humanoid robots with constrained processing power. Table 3 reports computational efficiency (per-frame latency and memory usage) for AHCC, HCC, and YOLOv5.

Table 2. Object Detection Performance

Metric	YOLOv5	HCC	AHCC
Accuracy (%)	99	82	97
Recall (%)	99	60	83
Precision (%)	100	47	97
F1-Score (%)	99	53	89

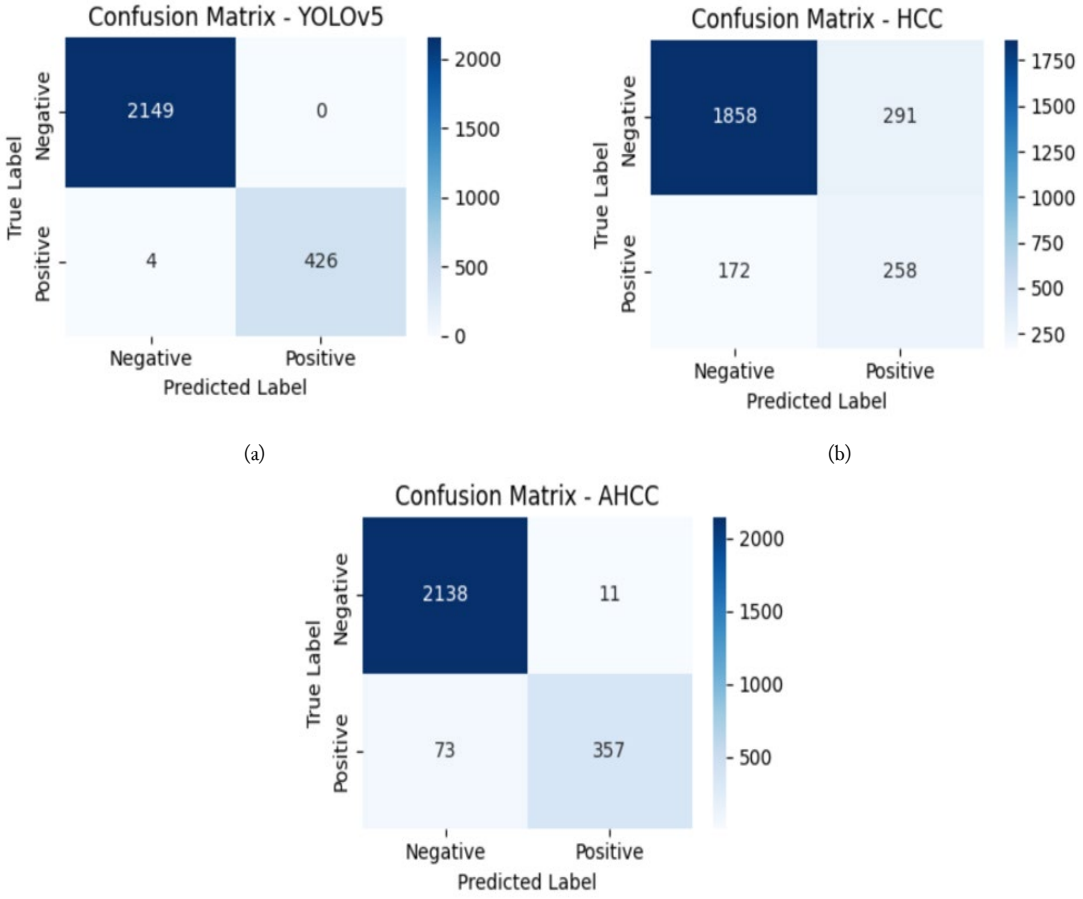


Fig. 10. Confusion matrices comparing classifier performance.

In contrast, the proposed AHCC demonstrated a well-balanced trade-off between accuracy and computational efficiency. It achieved 97% accuracy, 83% recall, 97% precision, and an F1 Score of 89%. While slightly lower in performance than YOLOv5, AHCC significantly outperformed the traditional HCC across all detection metrics. More importantly, AHCC operated with only 0.00849 seconds per frame and 8.97% memory usage, establishing its suitability for real-time, resource-limited robotic platforms. Meanwhile, HCC exhibited the lowest detection performance, with 82% accuracy, 60% recall, 47% precision, and an F1-score of 53%. Although it maintained relatively low computational requirements (0.0103 seconds, 9.24% memory), the high number of false positives and false negatives highlights its limitation in complex and dynamic environments.

Table 3. Computational Efficiency

Model	Computational Time (sec)	Memory Usage (%)
YOLOv5	0.03440	22.30
HCC	0.01030	9.24
AHCC	0.00849	8.97

To further evaluate the contribution of each enhancement within AHCC, an ablation study was conducted as shown in Table 4. The results show progressive improvements as additional modules were introduced. Starting with the HCC-only configuration, the model yielded 82% accuracy and 47% precision. Adding HSV segmentation raised precision to 78% by effectively filtering irrelevant background colors. The introduction of Convex Hull further improved region localization, increasing

recall and overall F1-score. Finally, incorporating the Hough Transform, which enforces object shape conformity, culminated in the full AHCC configuration, achieving the best performance across all metrics: 97% accuracy, 83% recall, 97% precision, and 89% F1-score. These findings validate that each component contributes meaningfully to the system's robustness and accuracy, confirming AHCC as a computationally efficient yet accurate alternative to more complex deep learning models, particularly for embedded humanoid robot systems operating in dynamic environments.

Table 4. Ablation study on AHCC components

Configuration	Accuracy (%)	Recall (%)	Precision (%)	F1-Score (%)
HCC Only	82	60	47	53
HCC + HSV	83	69	78	68
HCC + HSV + Convex Hull	84	77	82	73
HCC + HSV + Convex Hull + Hough (AHCC)	97	83	97	89

For completeness, a comparative trade-off analysis with other lightweight detectors, such as Tiny-YOLO and MobileNet-SSD, was also considered based on reported benchmarks in the literature [25], [35]. Both models offer moderate accuracy (-91–94%) but still rely on GPU or accelerator-based inference, resulting in higher energy consumption and larger model footprints compared to the CPU-only AHCC. In contrast, the proposed AHCC achieves competitive accuracy (97%) while maintaining minimal computational cost (0.00849 s per frame on CPU), demonstrating a favorable accuracy efficiency balance. These findings align with recent surveys emphasizing that compression, pruning, and quantization remain key enablers for deploying deep models in energy-constrained robotic platforms [36]–[38]. In addition, in the humanoid soccer setting, CPU-only inference is critical for onboard operation under tight power and thermal budgets, where GPU modules increase weight and energy consumption. AHCC's 0.00849 s/frame with low memory usage enables low-latency perception while leaving headroom for control threads, improving overall system responsiveness during real play.

In addition, to further contextualize the performance of the proposed AHCC within recent developments in lightweight and hybrid object detectors, a comparative analysis was conducted against representative models reported between 2023 and 2025. The comparison emphasizes accuracy, computational latency, and hardware dependency, highlighting the trade-off between detection reliability and resource efficiency that is critical for embedded humanoid robots. Table 5 summarizes the key characteristics of AHCC and other state-of-the-art lightweight detectors designed for edge and embedded environments. As shown in Table 5, the most recent lightweight object detectors, including Tiny-YOLO, MobileNet-SSD, and YOLO-FastestV2, still rely on GPU or accelerator support to achieve real-time performance. In contrast, the proposed AHCC maintains competitive accuracy (97%) and F1-score (89%) while operating entirely on a CPU-only platform with sub-10 ms inference latency. This demonstrates AHCC's suitability for onboard humanoid control systems where power consumption, weight, and thermal constraints restrict the use of dedicated accelerators.

A key limitation of the current method is its sensitivity to extreme occlusions. While partial occlusions can be managed effectively, complete or prolonged occlusions significantly reduce detection accuracy. Moreover, the HSV-based segmentation relies on predefined thresholds, which often require manual calibration to accommodate environmental variability. Inspired by the adaptive integration of BiLSTM-attention and reinforcement learning in dynamic decision-making for supply chains [39], our future work will address these limitations by introducing adaptive threshold learning, extending detection beyond circular object shapes, and deploying the model on edge AI platforms such as NVIDIA Jetson and Intel Movidius for real-time multi-object detection [40]–[42]. Future extensions may also integrate lightweight vision transformers with pruning optimization as proposed in [43], to further improve model generalization and runtime efficiency on embedded systems. Recent studies have demonstrated that compact transformer-based architectures can achieve near-CNN accuracy while significantly reducing parameter count and inference latency [44], [45], making them highly promising for edge robotics and real-time visual perception.

Table 5. Comparative analysis of AHCC vs. lightweight/hybrid object detectors (2023–2025)

Method	Year	Backbone	Performance	Computational Latency	Hardware dependency	Key Findings
AHCC (this work)	2025	HCC + Convex Hull + HSV segm. + Hough validation (CPU-only)	Acc. **97%**, Rec. **83%**, Prec. **97%**, F1 **89%** (ball detection)	**0.00849 s/frame** (≈118 FPS), CPU util. low; Intel NUC i3-7100U (ROS, OpenCV)	**CPU-only**	Hybrid classical pipeline; low latency leaves headroom for control threads
YOLO-FastestV2 (embedded) [21]	2023	ShuffleNetV2 backbone; ultralight head	mAP **99.45%** (dataset in paper)	**0.02 s/frame** (50 FPS) single-frame detection time; **~82 ms** on RPi4 with NCNN (≈12 FPS) vs **~2.6 s** Torch (×30 accel.)	CPU/GPU optional; NCNN for ARM	Validated on **Raspberry Pi**; shows strong speed–size trade-off for edge.
Tiny-YOLO / DC-YOLO (lightweight tiny) [10], [16]	2024	YOLOv5s + dynamic convolution (tiny)	Reported improvements over YOLO-Tiny on small objects (paper-specific; task-dependent)	Edge-deployable; CPU/GPU; latency device-dependent (paper)	Often GPU/accelerator preferred	Representative tiny-YOLO family optimized for small objects on edge.
MobileNet-SSD (edge) [46]	2024	Depthwise separable conv. + SSD head	Energy/latency-efficient; accuracy lower than larger YOLOs (varies per dataset)	Faster inference on RPi/Edge-TPU with TFLite; **INT8 quant.** markedly cuts latency vs FP32/INT32	**CPU-friendly**, benefits from TFLite/TPU	Multiple studies show MobileNet-SSD favors **speed/energy** on CPU with quantization.
LE-YOLO (YOLOv8n-based tiny UAV) [47]	2024	LHGNet backbone (dw-separable conv + channel shuffle)	SOTA tiny-object mAP improvements on UAV datasets	Designed for **onboard UAV edge**; FPS device-dependent	CPU/GPU; lightweight	Focus on tiny objects and onboard constraints; aligns with embedded deployment goals.
MSO-DETR / Drone-DETR (transformer-based lightweight) [48], [49]	2024 – 2025	DETR-based encoder–decoder transformers optimized for small-object and UAV tasks	mAP ≈ 94–96 % (dataset-specific)	20–30 FPS on Jetson TX2 / CPU-only edge inference feasible	GPU optional / Edge-friendly	Show improved small-object detection using transformer compression; represent emerging ViT-DETR hybrids for embedded robotics.
YOLO-C (YOLO-Compact) **[20]**	2025	Compact YOLO for **embedded/industrial**	Improves speed–accuracy over baselines	Edge-oriented; supports CPU/GPU toolchains	Flexible	Peer-reviewed in **EAAI**; good citation to show recent compact YOLO trend for industry/edge
Benchmarks on CPUs (YOLO v5–v11 / SSD) [50]	2024 – 2025	Various (YOLO/SSD/EfficientDet)	Task-dependent; show small models trade a bit of mAP for large latency/energy gains	Provide **CPU** inference times (OpenVINO/ONNX) across devices; confirm tiny models' edge advantage	CPU/TPU/GPU compared	Use these to justify **accuracy, latency trade-off** narrative and CPU context.

4. Conclusion

This study successfully enhances the traditional Haar Cascade Classifier (HCC) for real-time ball tracking in unconstrained and dynamic environments by introducing improved preprocessing techniques, including Convex Hull mapping and HSV-based color segmentation. The proposed Augmented HCC (AHCC) demonstrates significant improvements in computational efficiency, detection robustness, and resistance to noise and illumination changes, while maintaining a low false-positive rate and high tracking accuracy in complex backgrounds. Although YOLOv5 achieves superior accuracy, its high computational cost and reliance on GPU resources limit its suitability for embedded robotic applications. In contrast, the proposed AHCC achieves a practical balance between accuracy and efficiency, operating reliably on CPU-only humanoid platforms. Therefore, AHCC provides an effective and lightweight alternative for real-time object detection and tracking in resource-constrained embedded humanoid robots. Future work will focus on adaptive threshold learning and multi-object tracking deployment on edge AI platforms such as NVIDIA Jetson and Intel Movidius, further extending the model's applicability to broader real-time robotic scenarios.

Declarations

Author contribution. Conceptualization, G.E.S., E.R.W., B.H.P; methodology, G.E.S. and Y.P.U; hardware, G.E.S. and B.H.P; software, E.R.W. and B.H.P; validation, G.E.S. and E.R.W.; formal analysis, E.R.W. and Y.P.U.; investigation, G.E.S. and I.R.A.; resources, G.E..S, and I.R.A; data curation, E.R.W. and Y.P.U.; writing original draft preparation, G.E.S., E.R.W., B.H.P.; writing review and editing, E.R.W. and Y.P.U

Funding statement. This research project was supported in part by the Pendanaan Riset Penerapan dan Pengembangan Artificial Intelligence (AI) Universitas Brawijaya, Indonesia under Grant 00067.06/UN10.A0507/B/KS/2024 and in part by the Robotics and Embedded System Laboratory, Faculty of Computer Science, Universitas Brawijaya, Indonesia

Conflict of interest. The authors declare no conflict of interest.

Additional information. The supplementary material for this article can be found at: https://github.com/editarosanaw/Humanoid-Robots-in-Dynamic-Environments/tree/main

References

[1] M. Farajtabar and M. Charbonneau, "The path towards contact-based physical human–robot interaction," *Rob. Auton. Syst.*, vol. 182, no. 3, p. 104829, Dec. 2024, doi: 10.1016/j.robot.2024.104829.

[2] Y. Tong, H. Liu, and Z. Zhang, "Advancements in Humanoid Robots: A Comprehensive Review and Future Prospects," *IEEE/CAA J. Autom. Sin.*, vol. 11, no. 2, pp. 301–328, Feb. 2024, doi: 10.1109/JAS.2023.124140.

[3] L. Sørensen, D. T. Johannesen, and H. M. Johnsen, "Humanoid robots for assisting people with physical disabilities in activities of daily living: A scoping review," *Assist. Technol.*, vol. 37, no. 3, pp. 203–219, May 2025, doi: 10.1080/10400435.2024.2337194.

[4] C. S. Song and Y.-K. Kim, "The role of the human-robot interaction in consumers' acceptance of humanoid retail service robots," *J. Bus. Res.*, vol. 146, pp. 489–503, Jul. 2022, doi: 10.1016/j.jbusres.2022.03.087.

[5] M. Andtfolk, L. Nyholm, H. Eide, and L. Fagerström, "Humanoid robots in the care of older persons: A scoping review," *Assist. Technol.*, vol. 34, no. 5, pp. 518–526, Sep. 2022, doi: 10.1080/10400435.2021.1880493.

[6] M. Cho *et al.*, "Evaluating Human-Care Robot Services for the Elderly: An Experimental Study," *Int. J. Soc. Robot.*, vol. 16, no. 7, pp. 1561–1587, Jul. 2024, doi: 10.1007/s12369-024-01157-7.

[7] T. Abbas Shangari, S. Sadeghnejad, and J. Baltes, "Importance of Humanoid Robot Detection," in *Humanoid Robotics: A Reference*, Dordrecht: Springer Netherlands, 2019, pp. 2463–2471, doi: 10.1007/978-94-007-6046-2_141.

[8] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, 2001, vol. 1, pp. I-511-I–518, doi: 10.1109/CVPR.2001.990517.

[9] S. Gharge, A. Patil, S. Patel, V. Shetty, and N. Mundhada, "Real-time Object Detection using Haar Cascade Classifier for Robot Cars," in *2023 4th International Conference on Electronics and Sustainable Communication Systems (ICESC)*, Jul. 2023, pp. 64–70, doi: 10.1109/ICESC57686.2023.10193401.

[10] M. L. Ali and Z. Zhang, "The YOLO Framework: A Comprehensive Review of Evolution, Applications, and Benchmarks in Object Detection," *Computers*, vol. 13, no. 12, p. 336, Dec. 2024, doi: 10.3390/computers13120336.

[11] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2016, pp. 779–788, doi: 10.1109/CVPR.2016.91.

[12] J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul. 2017, vol. 2017-Janua, pp. 6517–6525, doi: 10.1109/CVPR.2017.690.

[13] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," in *Computer Vision and Pattern Recognition*, Apr. 2018, pp. 1–6, Accessed: Jun. 12, 2023. [Online]. Available at: https://arxiv.org/abs/1804.02767v1.

[14] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," *Comput. Vis. Pattern Recognit.*, pp. 1–17, Apr. 2020, Accessed: Jun. 06, 2023. [Online]. Available at: https://arxiv.org/abs/2004.10934v1.

[15] G. Jocher *et al.*, "ultralytics/yolov5: v7.0 - YOLOv5 SOTA Realtime Instance Segmentation.". [Online]. Available at: https://zenodo.org/records/7347926.

[16] A. Vijayakumar and S. Vairavasundaram, "YOLO-based Object Detection Models: A Review and its Applications," *Multimed. Tools Appl.*, vol. 83, no. 35, pp. 83535–83574, Mar. 2024, doi: 10.1007/s11042-024-18872-y.

[17] A. Howard *et al.*, "Searching for MobileNetV3," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct. 2019, pp. 1314–1324, doi: 10.1109/ICCV.2019.00140.

[18] T. Li, Z. Pei, X. Liu, R. Nie, X. Li, and Y. Wang, "Low-Illumination Image Enhancement for Foreign Object Detection in Confined Spaces," *IEEE Trans. Instrum. Meas.*, vol. 72, pp. 1–10, 2023, doi: 10.1109/TIM.2023.3284141.

[19] H. Wang, H. Zhang, Z. Chen, J. Zhu, and Y. Zhang, "Influence of Artificial Intelligence and Robotics Awareness on Employee Creativity in the Hotel Industry," *Front. Psychol.*, vol. 13, p. 834160, Mar. 2022, doi: 10.3389/fpsyg.2022.834160.

[20] Y. Chen, B. Zhao, X. Jia, and T. Ma, "Efficient real-time object detection for embedded and industrial systems:The You Only Look Once-Compact approach," *Eng. Appl. Artif. Intell.*, vol. 156, p. 109927, Sep. 2025, doi: 10.1016/j.engappai.2024.109927.

[21] Dog-qiuqiu, "*Yolo-FastestV2: based on YOLO's low-power, ultra-lightweight universal target detection algorithm,*" GitHub repository. Available at: https://github.com/dog-qiuqiu/Yolo-FastestV2. Accessed: Oct. 2025.

[22] X. Deng, R. Li, L. Zhao, K. Wang, and X. Gui, "Multi-obstacle path planning and optimization for mobile robot," *Expert Syst. Appl.*, vol. 183, p. 115445, Nov. 2021, doi: 10.1016/j.eswa.2021.115445.

[23] R. Hao, C. Xu, and C. Zhong, "Infrared Monocular Depth Estimation Based on Radiation Field Gradient Guidance and Semantic Priors in HSV Space," *Sensors*, vol. 25, no. 13, p. 4022, Jun. 2025, doi: 10.3390/s25134022.

[24] T. Mahalingam and M. Subramoniam, "A robust single and multiple moving object detection, tracking and classification," *Appl. Comput. Informatics*, vol. 17, no. 1, pp. 2–18, Jan. 2021, doi: 10.1016/j.aci.2018.01.001.

[25] P. Mittal, "A comprehensive survey of deep learning-based lightweight object detection models for edge devices," *Artif. Intell. Rev.*, vol. 57, no. 9, p. 242, Aug. 2024, doi: 10.1007/s10462-024-10877-1.

[26] G. Liu, Y. Hu, Z. Chen, J. Guo, and P. Ni, "Lightweight object detection algorithm for robots with improved YOLOv5," *Eng. Appl. Artif. Intell.*, vol. 123, p. 106217, Aug. 2023, doi: 10.1016/j.engappai.2023.106217.

[27] H. Kwon, S. Oh, and J.-W. Baek, "Algorithmic Efficiency in Convex Hull Computation: Insights from 2D and 3D Implementations," *Symmetry (Basel).*, vol. 16, no. 12, p. 1590, Nov. 2024, doi: 10.3390/sym16121590.

[28] D. Singh, H. Sarkar, and L. N. Das, "A Mean Point Based Convex Hull Computation Algorithm," *Am. J. Eng. Res.*, vol. 5, no. 11, pp. 70–75, 2016, [Online]. Available at: https://www.ajer.org/papers/v5(11)/K0511070075.pdf.

[29] D. D. Burdescu, M. Brezovan, E. Ganea, and L. Stanescu, "A New Method for Segmentation of Images Represented in a HSV Color Space," 2009, pp. 606–617, doi: 10.1007/978-3-642-04697-1_57.

[30] S. Sural, Gang Qian, and S. Pramanik, "Segmentation and histogram generation using the HSV color space for image retrieval," in *Proceedings. International Conference on Image Processing*, vol. 2, pp. II-589-II–592, doi: 10.1109/ICIP.2002.1040019.

[31] A. O. Djekoune, K. Messaoudi, and K. Amara, "Incremental circle hough transform: An improved method for circle detection," *Optik (Stuttg).*, vol. 133, pp. 17–31, Mar. 2017, doi: 10.1016/j.ijleo.2016.12.064.

[32] M. Rizon *et al.*, "Object Detection using Circular Hough Transform," *Am. J. Appl. Sci.*, vol. 2, no. 12, pp. 1606–1609, Dec. 2005, doi: 10.3844/ajassp.2005.1606.1609.

[33] S. N. Shivappriya, S. . Pasupathy, R. Dhivyapraba, S. Jagadeeshan, R. Jai Krishna, and R. Nithin Krishna, "Object Detection and Measurement using Image Processing," in *2023 Third International Conference on Smart Technologies, Communication and Robotics (STCR)*, Dec. 2023, pp. 1–6, doi: 10.1109/STCR59085.2023.10397044.

[34] A. F. T. Winfield and M. Jirotka, "Ethical governance is essential to building trust in robotics and artificial intelligence systems," *Philos. Trans. R. Soc. A Math. Phys. Eng. Sci.*, vol. 376, no. 2133, p. 20180085, Nov. 2018, doi: 10.1098/rsta.2018.0085.

[35] W. Liu *et al.*, "SSD: Single Shot MultiBox Detector," 2016, pp. 21–37, doi: 10.1007/978-3-319-46448-0_2.

[36] S. M. Raza, S. M. H. Abidi, M. Masuduzzaman, and S. Y. Shin, "Lightweight deep learning for visual perception: A survey of models, compression strategies, and edge deployment challenges," *Neurocomputing*, vol. 656, p. 131357, Dec. 2025, doi: 10.1016/j.neucom.2025.131357.

[37] E. Paula, J. Soni, H. Upadhyay, and L. Lagos, "Comparative analysis of model compression techniques for achieving carbon efficient AI," *Sci. Rep.*, vol. 15, no. 1, p. 23461, Jul. 2025, doi: 10.1038/s41598-025-07821-w.

[38] D. Liu *et al.*, "A survey of model compression techniques: past, present, and future," *Front. Robot. AI*, vol. 12, pp. 1–16, Mar. 2025, doi: 10.3389/frobt.2025.1518965.

[39] A. Amellal, I. Amellal, M. R. Ech-charrat, and H. Seghiouer, "Predictive optimization in automotive supply chains: a BiLSTM-Attention and reinforcement learning approach," *Int. J. Adv. Intell. Informatics*, vol. 10, no. 3, p. 441, Aug. 2024, doi: 10.26555/ijain.v10i3.1351.

[40] V. Sharma, D. N. Mupenda, L. Thorvik, and D. Mishra, "Edge AI to Edge Robotics: Enhancing Human Pose Estimation with High-Performance TPU Computing," in *Communications in Computer and Information Science*, vol. 2333 CCIS, Springer, Cham, 2025, pp. 433–447, doi: 10.1007/978-3-031-83783-8_25.

[41] D. J. Patel, P. S. Patel, T. J. Patel, M. D. Viradiya, J. B. Patel, and D. Garg, "Real-Time Object Detection and Recognition on Jetson Nano," 2025, pp. 349–360, doi: 10.1007/978-981-97-8602-2_32.

[42] L. Rey *et al.*, "A Performance Analysis of You Only Look Once Models for Deployment on Constrained Computational Edge Devices in Drone Applications," *Electronics*, vol. 14, no. 3, p. 638, Feb. 2025, doi: 10.3390/electronics14030638.

[43] S. Saha and L. Xu, "Vision transformers on the edge: A comprehensive survey of model compression and acceleration strategies," *Neurocomputing*, vol. 643, p. 130417, Aug. 2025, doi: 10.1016/j.neucom.2025.130417.

[44] Y. Ye, Q. Sun, K. Cheng, X. Shen, and D. Wang, "A lightweight mechanism for vision-transformer-based object detection," *Complex Intell. Syst.*, vol. 11, no. 7, p. 302, Jul. 2025, doi: 10.1007/s40747-025-01904-x.

[45] N. Setyawan, C.-C. Sun, M.-H. Hsu, W.-K. Kuo, and J.-W. Hsieh, "MicroViT: A Vision Transformer with Low Complexity Self Attention for Edge Device," in *2025 IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2025, pp. 1–5, doi: 10.1109/ISCAS56072.2025.11043206.

[46] V. Kamath and R. A, "Investigation of MobileNet-Ssd on human follower robot for stand-alone object detection and tracking using Raspberry Pi," *Cogent Eng.*, vol. 11, no. 1, Dec. 2024, doi: 10.1080/23311916.2024.2333208.

[47] X. Zhao and Y. Chen, "YOLO-DroneMS: Multi-Scale Object Detection Network for Unmanned Aerial Vehicle (UAV) Images," *Drones*, vol. 8, no. 11, p. 609, Oct. 2024, doi: 10.3390/drones8110609.

[48] J. Li, Y. Hua, and M. Xue, "MSO-DETR: A Lightweight Detection Transformer Model for Small Object Detection in Maritime Search and Rescue," *Electronics*, vol. 14, no. 12, p. 2327, Jun. 2025, doi: 10.3390/electronics14122327.

[49] Y. Kong, X. Shang, and S. Jia, "Drone-DETR: Efficient Small Object Detection for Remote Sensing Image Using Enhanced RT-DETR Model," *Sensors*, vol. 24, no. 17, p. 5496, Aug. 2024, doi: 10.3390/s24175496.

[50] D. K. Alqahtani, M. A. Cheema, and A. N. Toosi, "Benchmarking Deep Learning Models for Object Detection on Edge Computing Daevices," 2025, pp. 142–150, doi: 10.1007/978-981-96-0805-8_11.