

Real-time object detection and distance measurement for humanoid robot using you only look once

Suci Dwijayanti, Bhakti Yudho Suprpto, Mutiyara, Rendyansyah

Department of Electrical Engineering, Faculty of Engineering, Universitas Sriwijaya, Palembang, Indonesia

Article Info

Article history:

Received Aug 28, 2023

Revised May 3, 2024

Accepted May 17, 2024

Keywords:

Accuracy

Humanoid robot

Object detection

Real-time

You only look once

ABSTRACT

Humanoid robots are designed to mimic human structures and utilize cameras to process visual input to identify surrounding objects. However, previous studies have focused solely on object detection, overlooking both the complexities of real-world implementation and the significance of calculating the distance between objects and the robot. This study proposes a system that employs the you only look once (YOLO) algorithm to detect various objects in the proximity of a robot. Using a dataset of primary data collected in a laboratory, the detected objects are from 12 classes, including humans, chairs, tables, cabinets, computers, books, doors, bottles, eggs, learning modules, cups, and hands, with each class comprising 1500 data points. Two YOLO architectures, namely tiny YOLOv3 and tiny YOLOv4, are assessed for their performance in object detection, with the tiny YOLOv4 demonstrating a superior accuracy of 82.99% compared to tiny YOLOv3. Evaluation under simulated conditions yields an accuracy of 74.16%, while in real-time scenarios, accuracies are 61.66% under bright conditions and 38.33% under dim conditions, affirming tiny YOLOv4's efficacy. Moreover, this study reveals an average error distance of 31% between an object and the robot in real-time conditions. The developed system enhances human-robot interaction capabilities via data transmission.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Suci Dwijayanti

Department of Electrical Engineering, Faculty of Engineering, Universitas Sriwijaya

Palembang, Indonesia

Email: sucidwijayanti@ft.unsri.ac.id

1. INTRODUCTION

Humanoid robots, defined as robots whose form mirrors the human body, can interact with equipment and environments designed for human use. These mechanical or virtual intelligent devices possess human-like structure, movement, and functionality, enabling them to execute tasks automatically or under guidance. Integrated with a variety of sensors such as radar, lidar, sonar, GPS, odometry, and object detection, these robots can interpret their surrounding environment. In Indonesia, the utilization of humanoid robots is relatively new, and their usage is sparse. Challenges arise in applying humanoid robots as they must perform tasks according to programmed functions. To prevent accidents, humanoid robots require a navigation system that mimics human vision to recognize surrounding conditions and detect surrounding objects, discerning whether they are moving or stationary. Object detection involves creating computational models and methods that furnish a fundamental piece of information crucial for computer vision applications: identifying the presence and location of objects. The two primary benchmarks for evaluating object detection include precision (comprising classification precision and localization precision) and processing speed [1].

Object detection plays a crucial role in enabling humanoid robots to interact effectively with their environment. Previous studies have explored various methods for detecting objects around humanoid robots,

primarily via digital image processing. For instance, Cofield *et al.* [2] utilized a red, green, blue, and depth (RGB-D) camera to estimate the position of an object relative to the robot. The camera captured point cloud data, which were then cropped to focus on the region of interest (ROI). Additionally, a random sample consensus (RANSAC) algorithm was employed to estimate the inlier plane, and the RGB-D sensor was used for object recognition. While the facile approach facilitates the recognition of unknown objects, it struggles to accurately identify small objects, posing challenges in determining the precise boundaries of specified objects. Another method involves leveraging the hue saturation value (HSV) color space, as demonstrated in [3]. The study showed that the HSV color space provides more distinguishable data for isolating the ROI compared to the RGB-derived opponencies color model. Studies [4]-[6] further explored object recognition using the HSV model to differentiate objects based on color disparities. However, focusing solely on color differences can reduce the accuracy when the color of an object closely matches its background or surrounding colors. To address this limitation, scholars enhanced the algorithm by automatically deriving the HSV color of the reference ROI from the vision of the robot, linked to an internet-of-things (IoT) device. Moreover, the algorithm selectively combines ROIs with significant HSV resemblance to the reference ROI via random sampling. The approach ensures the automatic identification of the spatial placement of a product within the 3D printer, focusing exclusively on ROIs that closely match the reference HSV [7].

Another method for object detection utilizes the histogram of oriented gradient (HOG). Sultana *et al.* [8], the HOG technique was employed to locate the target object within a test cluster image, using a template image patch. Additionally, An *et al.* [9] introduced a processor for object detection that synchronizes the image sensor within a field programmable gate array (FPGA). The processor integrates a cell-based HOG feature descriptor and a support vector machine (SVM) classifier. Ahmed *et al.* [10] proposed a method that combines the K-means clustering method with a generalized hough transform and genetic algorithm. However, the HOG algorithm relies on local gradient information, which may reduce the accuracy when dealing with small or partially distorted objects.

Another commonly used feature for object detection is the scale invariant feature transform (SIFT). Gupta *et al.* [11] SIFT was combined with the fast-rotated and BRIEF (ORB) features for object detection. Meanwhile, Jiang *et al.* [12] employed SIFT in conjunction with principal component analysis (PCA) to track moving objects. Moreover, SIFT has been implemented in real-time on a graphics processing unit (GPU) [13]. Putri *et al.* [14] used SIFT for feature extraction and K-nearest neighbor (KNN) as the classifier to detect an object. However, the SIFT algorithm has high computational complexity. In contrast, Kaymak and Ucar [15] combined various feature descriptors, such as SIFT, speeded up robust features (SURF), oriented features from accelerated segment test (FAST), and rotated BRIEF (ORB), and FAST. Bansal employed the blob detection method, which recognizes objects based on the clumps of segments they generate, unaffected by the object size [16]. If different objects generate segment blobs with identical values, the system will recognize the objects based on previously stored data. However, tracking can be problematic.

The aforementioned methods highlight the importance of features; however, feature extraction requires complex computation. Previous studies have demonstrated that object detection methods for humanoid robots are limited in terms of suboptimal accuracy and have slow processing speed, primarily owing to the use of distinct feature extractions for each object. Thus, deep learning has been implemented to overcome this challenge. Chatterjee *et al.* [17] conducted research on humanoid robots using a deep-learning method by applying the region-based fully convolutional network (R-FCN), a variant of the convolutional neural network (CNN) algorithm. The method detects objects using an object position score map and removes fully connected layers after incorporating an ROI that is sensitive to the position of the region proposal network (RPN), subsequently performing class delimiter box regression on the recognized object. The detection results can be visualized from the perspective of the robot based on the preprogrammed class label. The R-FCN model yielded comparable accuracy scores owing to the lower test image resolution of the detection results, indicating that deep learning can effectively detect objects with low computational power, making the approach more power-efficient. However, the detection process was slow with sluggish responses and decreased humanoid robot response accuracy owing to variations in feature extraction and classifiers.

In another study [18], two types of CNN architectures, single shot multi-box detector (SSD) with MobileNetV1 and faster region-based CNN (Faster-RCNN), were utilized. CNNs have also been employed in detecting hand gestures for human-robot interaction [19], [20]. Meanwhile, [21] utilized deep neural networks to detect hands. Faster-RCNN was employed for detecting marine objects in an underwater robot [22]. Additionally, Dairi *et al.* [23] utilized deep boltzmann machines and autoencoders to detect obstacles in driving environments.

This study aims to address the computational limitations of previous research by proposing a method for detecting objects around humanoid robots using an intelligent algorithm, you only look once (YOLO) [24]. YOLO, a CNN algorithm used in image processing tasks such as image classification, is one among several developed CNN algorithms, including R-CNN, Fast R-CNN, and Faster R-CNN. Previous research utilizing YOLO includes a study by Melek *et al.* [25], which employed YOLO to detect objects on a shelf

using a grocery dataset containing 354 images of ten different cigarette brands and a simulated database of 200 Coca-Cola samples obtained in various environments. The results demonstrated that the YOLO algorithm could detect objects with a loss rate of 8.22%. Corovic *et al.* [26] used YOLO to detect road objects such as cars, trucks, pedestrians, traffic lights, and traffic signs in real time. Despite a low accuracy rate (<50%) owing to the presence of numerous small, closed objects, all objects closest to the camera position were successfully detected and classified, indicating that YOLO can be effectively used for object detection around humanoid robots. Adarsh *et al.* [27] YOLOv3 tiny is proposed for object detection and recognition, while [28] utilized deep learning to detect a ball and a goal.

The aforementioned studies were not conducted in the context of humanoid robots capable of interacting with humans. Therefore, this study presents a comparative analysis of using YOLOv3 [29] and YOLOv4 [30] for detecting various objects commonly found in laboratories. These two architectures were chosen because YOLOv3 has a higher mean average precision (mAP) compared to Faster R-CNN [31]–[33] and YOLOv4 exhibits higher accuracy [30], [34], [35]. YOLOv5 is not considered in this study because YOLOv4 has a higher mAP than YOLOv5 [34]. Additionally, this study proposes a method for measuring the distance between objects and robots to facilitate human–robot interactions. Furthermore, a new dataset containing 12 classes is proposed. The contributions of this study are as follows:

- Real-time object detection is implemented for humanoid robots.
- The distance from the object to the robot is measured.
- A comprehensive comparison is presented between YOLOv3 and YOLOv4 is presented in object detection for the humanoid robot.
- Provides a new dataset for the objects commonly found in the laboratory.

The remainder of this paper is organized as follows. Section 2 describes the YOLO method. Section 3 presents the results and discussions. Finally, section 4 presents the conclusion and future work.

2. METHOD

2.1. You only look once

The YOLO algorithm utilizes neural networks for real-time object detection, featuring a relatively straightforward architecture. It employs a single CNN applied to the entire image during both training and testing phases, providing information on object appearance and class probability. The YOLO detection system processes input images into a 448×448-pixel size, followed by a sequence of operations involving a single neural network. Detection results are then constrained based on a confidence model. The system trains a single neural network to detect objects while simultaneously generating all bounding boxes and class probabilities, embodying the concept of “YOLO.”

This study employs two YOLO architectures: tiny YOLOv3 and YOLOv4. Both models are smaller owing to the simplified architecture of tiny YOLO compared to other YOLO variants, resulting in lower GPU memory usage and higher frames per second (FPS). Tiny YOLOv4, an enhancement of YOLOv3, incorporates modifications to the original YOLO network structure, as outlined in Table 1 (in Appendix). The training process involves tiny YOLOv3, a streamlined version of YOLOv3 with adjustments to the tiny network structure, as detailed in Table 2. Notably, tiny YOLOv3 has fewer layers compared to tiny YOLOv4.

Table 2. Tiny YOLOv3 architecture

Layer	Type	Size/stride	Input	Output
0	Convolutional+bnorm leaky	3×3/1	416×416×3	416×416×16
1	Maxpool	2×2/2	416×416×16	208×208×16
2	Convolutional+bnorm leaky	3×3/1	208×208×16	208×208×32
3	Maxpool	2×2/2	208×208×32	104×104×64
4	Convolutional+bnorm leaky	3×3/1	104×104×32	104×104×32
5	Maxpool	2×2/2	104×104×64	52×52×64
6	Convolutional+bnorm leaky	3×3/1	52×52×64	52×52×128
7	Maxpool	2×2/2	52×52×128	26×26×128
8	Convolutional+bnorm leaky	3×3/1	26×26×128	26×26×256
9	Maxpool	2×2/2	26×26×256	13×13×256
10	Convolutional+bnorm leaky	3×3/1	13×13×256	13×13×512
11	Maxpool	2×2/2	13×13×512	13×13×512
12	Convolutional+bnorm leaky	3×3/1	13×13×512	13×13×1024
13	Convolutional+bnorm leaky	1×1/1	13×13×1024	13×13×256
14	Convolutional+bnorm leaky	3×3/1	13×13×256	13×13×512
15	Convolutional+linear	1×1/1	13×13×512	13×13×255
16	YOLO			

2.2. Hardware design

In this study, specific hardware components were utilized to support the operation of humanoid robots, including a 3 MP Webcam Camera module OV3660 with a wide-angle 110° field of view, dot matrix, JX servo, and an Arduino Mega 2560 microcontroller. A schematic depicting the configuration of the humanoid robot is presented in Figure 1, highlighting the positions of the components used in this investigation. Meanwhile, the design of the object detection system integrated into the humanoid robot for object recognition is depicted in Figure 2. Initially, the data acquired from the webcam are divided into training and testing datasets. Subsequently, a model is developed through training. The model was then incorporated into the humanoid robot for real-time testing, which includes measuring the distance between the object and the robot.

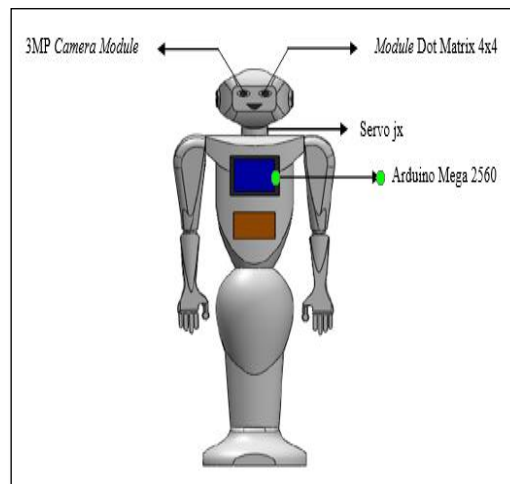


Figure 1. Schematic of components in a humanoid robot

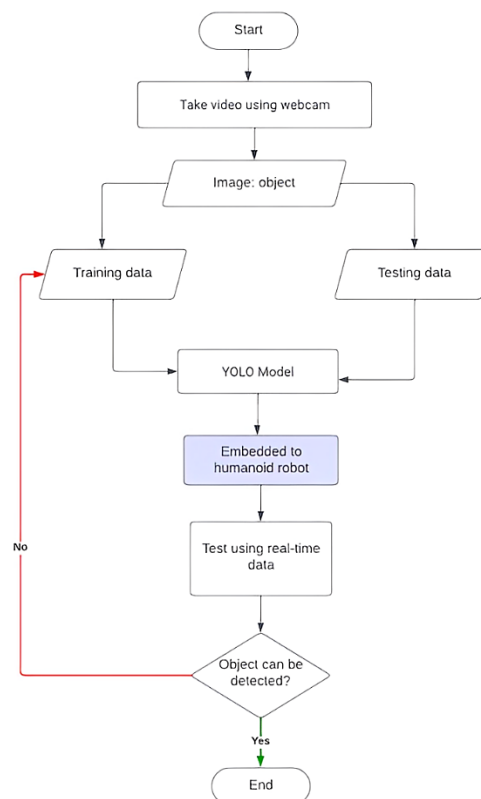


Figure 2. System design flow diagram

2.3. System testing

System tests were conducted utilizing variables within an evaluation method known as the confusion matrix model, which is detailed in Table 3. Additionally, this section provides the equations used to determine their values and accuracies.

Table 3. Confusion metric

Predicted value	True value	
	True positive (TP)	False positive (FP)
	False negative (FN)	True negative (TN)

Table 3 indicates that the predicted value corresponds to the confidence value produced by the YOLO model, while the actual value is derived from the target objective. The accuracy is defined as (1):

$$accuracy = \frac{TP+TN}{TP+FP+FN+TN} \quad (1)$$

Here, TP represents the count of positive data correctly classified by the system, FP signifies the quantity of positive data incorrectly classified by the system, TN denotes the total of negative data accurately classified by the system, and FN refers to the quantity of negative data misclassified by the system.

3. RESULTS AND DISCUSSION

This section discusses the results obtained from implementing real-time object detection using humanoid robots. Specifically, the analysis compares the two models of tiny YOLO: tiny YOLOv3 and tiny YOLOv4.

3.1. Robot humanoid design

The design of the humanoid robot encompasses both the mechanical structure and overall wiring of the components. These components include a dot matrix for displaying the visual appearance of the robot's eyes, JX servo for maneuvering the robot's head, and camera modules linked to laptops used for object detection. The camera was positioned on the head of the humanoid robot at a height of 1.5 m from the ground. The outcomes of the humanoid robot design are illustrated in Figure 3(a) for the front view and Figure 3(b) for the side view.

3.2. Training and test data collection

In this study, the object dataset was derived from a video recorded in the Control and Robotics Laboratory and the Robotics Secretariat at the Universitas Sriwijaya. An example of the object data is displayed in Figure 4. The data were captured using a webcam with a resolution of 640×480 pixels. The collected data underwent processing and annotation for object class recognition in the images. The dataset comprises 12 classes: humans, chairs, tables, cupboards, computers, books, doors, bottles, eggs, modules, cups, and hands. This dataset is relatively different from common datasets like the COCO dataset because it contains objects that are commonly found in the laboratory. Additionally, we included eggs and hands as objects for evaluating the capability of the robot arm.

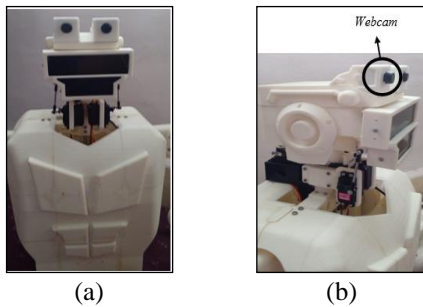


Figure 3. Humanoid robot with components; (a) front view and (b) side view



Figure 4. Samples of classes of objects containing a cupboard, module, and computer

The total dataset contains 5447 entries, with 4072 primary and 1375 secondary data points. Secondary data were incorporated to introduce variation into the object dataset. Each object was annotated with the class, x-center position, y-center position, width, and height. Following processing and annotation, the dataset expanded to 18000 entries, with each class contributing 1500 entries. Of these, 70% (or 12600 entries) were allocated for training data, while the remaining 30% (or 5400 entries) served as test data.

3.3. Training and test data processing

The collected training and test data, obtained from real-world sources, underwent a preprocessing stage. This critical step involved annotating and labeling each object class within every image. In addition to enhancing the clarity of the dataset, the annotation also serves as the initial step for the class initialization of every object contained within an image.

3.4. Object detection results

Two types of YOLO architectures, namely tiny YOLOv3 and tiny YOLOv4, were employed to evaluate the robustness of each model in detecting objects. These architectures were selected owing to their simpler architectures and lower GPU memory consumption compared to other architectures. Each YOLO model was assessed over 50, 100, 200, 500, and 1000 epochs. The training losses for each class are depicted in Figure 5. As illustrated, the tiny YOLOv4 model architecture exhibits lower training loss than the tiny YOLOv3 model. The best model performance is achieved when trained over 1000 epochs, resulting in the smallest loss value compared to other training sessions. The average loss for tiny YOLOv4 is smaller than that for tiny YOLOv3, with values of 0.3215 and 0.5133, respectively.

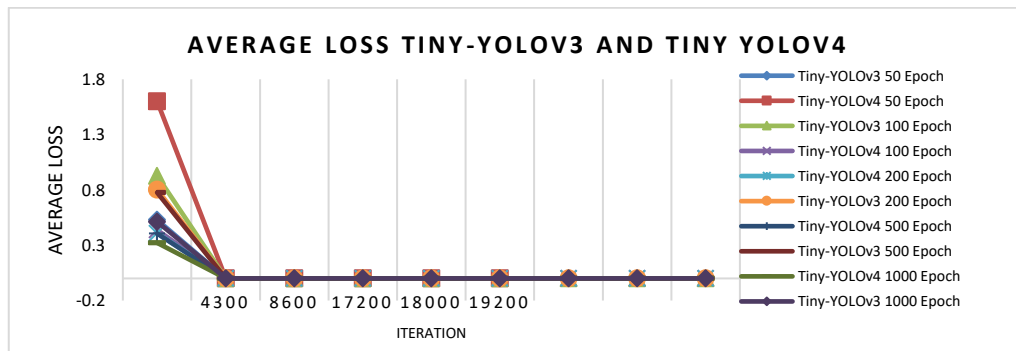


Figure 5. Average loss of the YOLO model for object detection

Two models of tiny YOLO are compared: tiny YOLOv3 and tiny YOLOv4. Both models were trained using a batch size of 64 over 50, 100, 200, 500, and 1000 epochs. Table 4 lists the smallest average loss ratios for each epoch. As indicated in Table 4, Tiny YOLOv3 exhibits a high computation time, requiring 70 h to complete training for 1000 epochs. In contrast, tiny YOLOv4 requires less computation time and demonstrates a lower average loss compared to Tiny YOLOv3. The results depicted in Figure 5 and Table 4 illustrate that with a larger number of epochs, the loss value decreases. However, this entails a longer training duration. Additionally, the training time is influenced by the batch size. The training outcomes indicate that the average loss of tiny YOLOv4 in 1000 epochs is lower compared to Tiny YOLOv3.

Table 4. Comparison of each epoch

Epoch	Smallest average loss		Training time (hours)	
	Tiny YOLOv3	Tiny YOLOv4	Tiny YOLOv3	Tiny YOLOv4
50	1.6012	0.5297	10	6
100	0.9272	0.4378	17	9
200	0.8034	0.4095	20	13
500	0.7764	0.4057	23	17
1000	0.5133	0.3215	70	54

Table 5 presents the test results of the tiny YOLOv3 and tiny YOLOv4 models trained over 50, 100, 200, 500, and 1000 epochs for object class detection. As observed in the table, the tiny YOLOv4 model significantly outperforms the tiny YOLOv3 model. Specifically, the tiny YOLOv4 model achieves the highest accuracy of 82.99% when trained over 1000 epochs. Conversely, the tiny YOLOv3 model, when trained over

50 epochs, yields the lowest accuracy of 8.33% for ten attempts across 40 objects. Among all the detected object categories, some are relatively easy to identify for both tiny YOLO models, including chairs, humans, hands, and eggs. This ease of detection is attributed to the distinctive characteristics and sizes of these objects compared to others. Moreover, the results indicate that tiny YOLOv4 is better suited for implementation in object detection for real-time environments owing to its better computation time and accuracy.

Table 5. Tiny YOLOv3 and tiny YOLOv4 model comparison results

Test #	Epoch 50 model tiny YOLOv3	Epoch 100 model tiny YOLOv3	Epoch 200 model tiny YOLOv3	Epoch 500 model tiny YOLOv3	Epoch 1000 model tiny YOLOv3	Test #	Epoch 50 model tiny YOLOv4	Epoch 100 model tiny YOLOv4	Epoch 200 model tiny YOLOv4	Epoch 500 model tiny YOLOv4	Epoch 1000 model tiny YOLOv4
1	(0/5)×1 00%= 0%	(0/5)×1 00%= 0%	(0/5)×1 00%= 0%	(0/5)×1 00%= 0%	(0/5)×1 00%= 0%	1	(0/5)×10 0%=0%	(0/5)×1 00%= 0%	(0/5)×1 00%= 0%	(4/5)×1 00%= 80%	(4/5)×1 00%= 80%
2	(0/2)×1 00%= 0%	(0/2)×1 00%= 0%	(0/2)×1 00%= 0%	(0/2)×1 00%= 0%	(0/2)×1 00%= 0%	2	(1/2)×10 0%=50%	(2/2)×1 00%= 0%	(2/2)×1 00%= 100%	(2/2)×1 00%= 100%	(2/2)×1 00%= 100%
3	(0/6)×1 00%= 0%	(2/6)×1 00%= 33.33%	(2/6)×1 00%= 33.33%	(2/6)×1 00%= 33.33%	(2/6)×1 00%= 33.33%	3	(1/6)×10 0%= 16.66%	(2/6)×1 00%= 33.33%	(4/6)×1 00%= 66.66%	(5/6)×1 00%= 83.33%	(5/6)×1 00%= 83.33%
4	(0/4)×1 00%= 0%	(1/4)×1 00%= 25%	(2/4)×1 00%= 50%	(3/4)×1 00%= 75%	(3/4)×1 00%= 75%	4	(1/4)×10 0%=25%	(3/4)×1 00%= 75%	(3/4)×1 00%= 75%	(3/4)×1 00%= 75%	(4/4)×1 00%= 100%
5	(0/3)×1 00%= 0%	(0/3)×1 00%= 0%	(0/3)×1 00%= 0%	(1/3)×1 00%= 33.33%	(1/3)×1 00%= 33.33%	5	(0/3)×10 0%=0%	(0/3)×1 00%= 0%	(2/3)×1 00%= 66.66%	(2/3)×1 00%= 66.66%	(2/3)×1 00%= 66.66%
6	(0/4)×1 00%= 0%	(2/4)×1 00%= 50%	(2/4)×1 00%= 50%	(2/4)×1 00%= 50%	(2/4)×1 00%= 50%	6	(0/4)×10 0%=0%	(0/4)×1 00%= 0%	(1/4)×1 00%= 25%	(2/4)×1 00%= 50%	(3/4)×1 00%= 75%
7	(0/3)×1 00%= 0%	(0/3)×1 00%= 0%	(0/3)×1 00%= 0%	(0/3)×1 00%= 0%	(1/3)×1 00%= 33.33%	7	(0/3)×10 0%=0%	(0/3)×1 00%= 0%	(0/3)×1 00%= 0%	(0/3)×1 00%= 0%	(2/3)×1 00%= 66.66%
8	(0/3)×1 00%= 0%	(0/3)×1 00%= 0%	(0/3)×1 00%= 0%	(0/3)×1 00%= 0%	(0/3)×1 00%= 0%	8	(0/3)×10 0%=0%	(2/3)×1 00%= 66.66%	(2/3)×1 00%= 66.66%	(2/3)×1 00%= 66.66%	(3/3)×1 00%= 100%
9	(0/4)×1 00%= 0%	(0/4)×1 00%= 0%	(0/4)×1 00%= 0%	(0/4)×1 00%= 0%	(2/4)×1 00%= 50%	9	(0/4)×10 0%=0%	(3/4)×1 00%= 75%	(3/4)×1 00%= 75%	(3/4)×1 00%= 75%	(3/4)×1 00%= 75%
10	(5/6)×1 00%= 83.33%	(5/6)×1 00%= 83.33%	(5/6)×1 00%= 83.33%	(5/6)×1 00%= 83.33%	(5/6)×1 00%= 83.33%	10	(5/6)×10 0%= 83.33%	(5/6)×1 00%= 83.33%	(5/6)×1 00%= 3.33%	(5/6)×1 00%= 83.33%	(5/6)×1 00%= 83.33%
Ave. acc	8.33%	19.17%	21.67%	27.49%	35.83%	Ave. acc	17.49%	43.33%	55.83%	67.99%	82.99%

3.5. Object distance estimation using depth information

In addition to object detection, this research also involves measuring the distance to objects detected by the humanoid robot. This distance measurement is crucial for the humanoid robot to determine the positions of objects around it, enabling the robot to approach or avoid the objects.

In object detection, for each frame, there exists four variables used in the distance measurement: (x, y, w, h), where x and y represent adjustments of the bounding box on the top-left corner for x and the bottom-right corner for y, respectively. Moreover, w and h represent the width and height of the bounding box. When the bounding box is formed, these four variables can be used in the distance formula as they are known.

$$Distance = \frac{2 \times 3.14 \times 180}{(w + h \times 360) \times 1000} + 3 \quad (2)$$

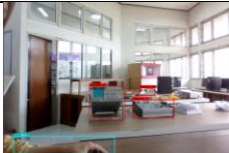
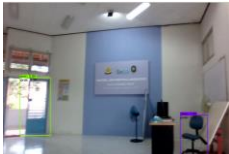



The equation combines the formula for the length of an arc with the values of w and h from each bounding box created. The value 1000 is used to convert the final unit to inches, and three is a constant threshold added to obtain a measured distance that approximates the actual distance.

3.6. Simulation testing on humanoid robot

Simulations were performed to assess the performance of the YOLO algorithm in identifying objects using video inputs within the Control and Robotics Laboratory at Sriwijaya University's Indralaya Campus.

The test, which utilized pre-recorded videos instead of real-time footage, was necessary before introducing the humanoid robot into the laboratory. The simulation test was carried out using the Tiny YOLOv4 network model, with 1000 training epochs and a threshold of 0.5. The resulting detections were utilized for distance measurements, leveraging depth information. The data for class, distance, and location were then serially transmitted to the microcontroller for further processing, enabling the humanoid robot to move toward or away from the object. The classes of objects sent to the microcontroller include eggs, humans, bottles, and cups, should the YOLO detect such objects. The results of five trials are presented in Table 6.

Table 6. Five samples of object detection by simulation in humanoid robots using tiny YOLOv4 1000 epoch


Test #	Display	Accuracy	Actual distance	Object detection, location, and measured distance	LED display
1		4/6×100% =66.66%	Hand: 0.45 m Module1: 1 m Module 2: 1.2 m Module 3: 2.2 m	Hand: object found in the left 0.3120404520962019 m Module1: object found in the left 1.1940895122942006 m Module2: object found in the right 1.0187875709924166 m Module3: object found in the right 1.570924347961893 m	Unsent 
2		2/3×100% =66.66%	Door: 7 m Chair: 3 m	Door: object found in the left 0.3950822745446469 m Chair: object found in the right 0.4239943189388893 m	Unsent 
3		4/4×100% =100%	Table: 3.1 m Chair: 3.24 m Door: 3 m Module: 0.3 m	Table: object found in the left 0.3120404520962019 m Chair: object found in the right 0.5390007736963907 m Door: object found in the right 0.5390007736963907 m Module: object found in the right 0.3883803137877421 m	Unsent 
4		3/4×100% =75%	Module 1: 0.44 m Module 2: 0.47 m Cupboard: 4.3 m	Module1: object found in the left 0.4740682186655581 m Module2: object found in the left 0.495111008754013 m Cupboard: object found in the left 0.5532491966703773 m	Unsent 
5		4/4×100% =100%	Door: 6.2 m Table: 3.76 m Chair: 3.74 m Human: 3.74 m	Door: object found in the left 0.4740682186655581 m Table: object found in the right 0.380038811403416 m Chair: object found in the right 0.380038811403416 m Human: object found in the right 0.3986310211232019 m	

As depicted in Table 6, the architecture of the tiny YOLOv4 model exhibits high accuracy in detecting nearby objects. Overall, the model achieves an accuracy of 74.16% and is capable of transmitting the class, position, and distance data to the microcontroller. The key advantage of the detection system lies in its ability to identify objects that surround the humanoid robot. Additionally, the proposed object detection system can determine distances and positions, offering valuable information regarding the location of detected objects.

3.7. Real-time testing on humanoid robots

Real-time testing involves direct assessment by humanoid robots equipped with a tiny YOLOv4-based object detection system. The tests were conducted in the classroom of the Department of Electrical Engineering, Unsri Palembang. The tests were performed under two conditions: bright and dim room conditions. A bright room condition entails open windows that provide natural light, supplemented by the internal lighting of the room. Conversely, the dim room condition simulates a closed window environment without additional lighting from a lamp in the room. The objective of this test is to evaluate the humanoid robots' capability to detect objects in their surroundings under both well-lit and dark conditions, utilizing the tiny YOLOv4 model trained for 1000 epochs with a threshold set at 0.5. The results of object detection for the bright and dim room conditions are presented in Tables 7 and 8, respectively. Based on the real-time testing conducted ten times in each room condition, the tiny YOLOv4 model demonstrates the ability to detect objects around the humanoid robot at a speed of 30 fps. The model showcases proficient object detection capabilities. Furthermore, in distance testing using real-time depth information, it can be observed that the system effectively measures the distance of detected objects. Subsequently, the detected data can be transmitted to the microcontroller for further processing by the humanoid robot.











Table 7. Five sample results of real-time testing for tiny YOLOv4 training 1000 epochs when the room has open windows and the 'lamp in the room is turned on

Test#	Display	Accuracy	Actual distance	Object detection, location, and measured distance	LED display
1		$1/2 \times 100\% = 50\%$	Human: 0.75 m	Human: object found in the left 0.7991550547652021	
2		$1/1 \times 100\% = 100\%$	Door: 3.02	Door: object found in the left 0.7696131910	Unsent 
3		$1/3 \times 100\% = 33.33\%$	Hand: 2	Hand: object found in the right 2.2780527607361910	Unsent 
4		$2/3 \times 100\% = 66.66\%$	Human: 1.45 Computer: 1.4	Human: object found in the left 1.33318975571316 Computer: object found in the right 1.2953482314976013	
5		$1/2 \times 100\% = 50\%$	Table: 1.25	Table: object found in the right 1.190197051292	Unsent 

Tables 7 and 8 demonstrate the object detection capabilities of tiny YOLOv4 under different lighting conditions, including both bright and dim settings. However, the results reveal that object detection performs more effectively in well-lit conditions, achieving an average accuracy of 61.66% out of 25 objects across 12 classes. This discrepancy is attributed to certain objects being closely clustered and positioned,

posing challenges for accurate detection by the tiny YOLOv4 model. Conversely, in experiments conducted in dimly lit environments, the accuracy of object detection decreases to 38.33% out of 17 objects. Notably, the tiny YOLOv4 model frequently generates false detections, observed in tests 1 and 3, where it either misidentifies objects or fails to recognize them. Moreover, it impacts the distance calculation accuracy of the system. The influence of room lighting intensity on the outcomes of the tiny YOLOv4 detection system is evident. The lighting level directly affects the accuracy of object detection; dim lighting obscures object features, making detection more challenging. Across all experiments conducted under both lighting conditions, hands and humans exhibit the highest accuracy in detection.

Table 8. Five sample results of real-time testing for Tiny YOLOv4 training 1000 epochs when the room has closed windows and the lamp is off

Test #	Display	Accuracy	Distance measured	Object detection and location	LED display
1		0/2×100%=0%	Egg: 0.32 human: 0.51	None	
2		1/2×100%=50%	Bottle: 0.58 Human: 0.72	Bottle object found in the right 0.6967455380913 878 None	
3		0/2×100%=0%	Cup: 0.62 Hand: 0.62	Cup object found in the left 0.6875272991021 184 None	Unsent 
4		1/2×100%=50%	Hand: 0.70 Human: 0.71	Hand object found in the right 0.7056686356313 768 None	Unsent 
5		1/1×100%=100%	Book: 0.74	Book object found in the left 0.7423084707034 439	Unsent 

Furthermore, the testing conducted under bright and dim lighting conditions compares the measured distance by the detection system using depth information to the actual distance. Readings obtained from depth information indicate that the distance calculation of the system remains reasonably accurate for both lighting conditions, with an average error of 0.31 or 31%. The accuracy is attributed to the objects being positioned at distances greater than two meters. However, it is noteworthy that distance reading errors are more pronounced in dimly lit rooms, suggesting that light intensity also influences distance readings.

4. CONCLUSION

This study showed that YOLO can be implemented to detect objects in humanoid robots, particularly for the objects commonly found in the laboratory. The tiny YOLOv4 model, trained over 1000 epochs, surpasses the tiny YOLOv3 model in object detection, achieving an impressive detection accuracy of 82.99%. This study implemented both tiny YOLO models for a humanoid robot to detect objects, including the distance and location. Real-time testing utilizing the tiny YOLOv4 model demonstrated the effectiveness in using the YOLO algorithm to detect objects on humanoid robots in real time. Results from the tests indicate that the YOLO algorithm performs admirably in detecting objects on humanoid robots, achieving an accuracy of 74.16% in simulated scenarios and 61.66% accuracy in real-time scenarios under bright conditions, and 38.33% accuracy under dim conditions. Additionally, the study indicated that the performance of object detection is influenced by the lighting intensity within a room, with objects in dimly lit environments presenting challenges for recognition compared to those in adequately lit environments. Furthermore, the YOLO algorithm is capable of estimating the distance between objects and humanoid robots using depth information. In comparing the observed distance reading errors against the actual distance in real-time tests across both lighting conditions, an error rate of 31% was found. Object distances can be accurately measured and communicated to the microcontroller, providing input for the humanoid robot to either approach or maintain a distance from detected objects. However, this study did not consider YOLOv8. Therefore, in future work, real-time object detection will be implemented in humanoid robots using YOLOv8.

ACKNOWLEDGEMENTS

This study was funded by the Directorate of Research, Technology, and Communicaty Service Directorate General of Higher Education, Research, and Technology according to the Research Contract Number: 090/E5/PG.02.00.PL/2024.

APPENDIX

Table 1. Tiny YOLOv4 architecture

Layer	Type	Size/stride	Input	Output
0	Convolutional+bnorm leaky	3×3/2	416×416×3	208×208×32
1	Convolutional+bnorm leaky	3×3/2	208×208×32	104×104×64
2	Convolutional+bnorm leaky	3×3/1	04×104×64	104×104×64
3	Route 2		1/2	104×104×32
4	Convolutional+bnorm leaky	3×3/1	104×104×32	104×104×32
5	Convolutional+bnorm leaky	3×3/1	104×104×32	104×104×32
6	Router 5 4			104×104×64
7	Convolutional+bnorm leaky	1×1/1	104×104×64	104×104×64
8	Route 2 7			104×104×128
9	Maxpool		104×104×128	52×52×128
10	Convolutional+bnorm leaky	3×3/1	52×52×128	52×52×128
11	Route 10		1/2	52×52×64
12	Convolutional+bnorm leaky	3×3/1	52×52×64	52×52×64
13	Convolutional+bnorm leaky	3×3/1	52×52×64	52×52×64
14	Route 13 12			52×52×128
15	Convolutional+bnorm leaky	1×1/1	52×52×128	52×52×128
16	Router 10 15			52×52×256
17	Maxpool		52×52×256	26×26×256
18	Convolutional+bnorm leaky	3×3/1	26×26×256	26×26×256
19	Router 18		1/2	26×26×128
20	Convolutional+bnorm leaky	3×3/1	26×26×128	26×26×128
21	Convolutional+bnorm leaky	3×3/1	26×26×128	26×26×128
22	Route 21 20		26×26×256	26×26×256
23	Convolutional+bnorm leaky	1×1/1	26×26×256	26×26×256
24	Router 18 23			26×26×512
25	Maxpool		26×26×512	13×13×512
26	Convolutional+bnorm leaky	3×3/1	13×13×512	13×13×512
27	Convolutional+bnorm leaky	1×1/1	13×13×512	13×13×256
28	Convolutional+bnorm leaky	3×3/1	13×13×256	13×13×512
29	Convolutional+linear	1×1/1	13×13×512	13×13×24
30	YOLO			




REFERENCES

- [1] Z. Zou, K. Chen, Z. Shi, Y. Guo and J. Ye, "Object Detection in 20 Years: A Survey," in *Proc. of the IEEE*, vol. 111, no. 3, pp. 257-276, Mar. 2023, doi: 10.1109/JPROC.2023.3238524
- [2] A. Cofield, Z. El-Shair, and S. A. Rawashdeh, "A humanoid robot object perception approach using depth images," *2019 IEEE National Aerospace and Elec. Conf. (NAECON)*, Dayton, OH, USA, 2019, pp. 437-442, doi: 10.1109/NAECON46414.2019.9057808.
- [3] A. Ajmal, C. Hollitt, M. Frean, and H. Al-Sahaf, "A Comparison of RGB and HSV Colour Spaces for Visual Attention Models," *2018 Int. Conf. Image Vis. Comput. New Zeal.*, vol. 2018, pp. 1-6, 2019, doi: 10.1109/IVCNZ.2018.8634752.
- [4] H. Kang et al., "HSV Color Space Based Robot Grasping for Personalized Manufacturing Services," *ICTC 2019 - 10th Int. Conf. ICT Conver. ICT Conver. Lead. Auton. Futur.*, 2019, pp. 1010-1012, doi: 10.1109/ICTC46691.2019.8939796.
- [5] K. Cameron and M. S. Islam, "Multiple Objects Detection using HSV," *NAECON 2018 - IEEE National Aerospace and Electronics Conference*, Dayton, OH, USA, 2018, pp. 270-273, doi: 10.1109/NAECON.2018.8556711.
- [6] J. Song, L. Liu, W. Huang, Y. Li, X. Chen, and Z. Zhang, "Target detection via HSV color model and edge gradient information in infrared and visible image sequences under complicated background," *Opt. Quantum Electron.*, vol. 50, no. 4, pp. 1-13, 2018, doi: 10.1007/s1082-018-1442-z.
- [7] H. C. Kang, H. N. Han, H. C. Bae, M. G. Kim, J. Y. Son, and Y. K. Kim, "Hsv color-space-based automated object localization for robot grasping without prior knowledge," *Appl. Sci.*, vol. 11, no. 16, 2021, doi: 10.3390/app11167593.
- [8] M. Sultana, T. Ahmed, P. Chakraborty, M. Khatun, M. R. Hasan, and M. S. Uddin, "Object detection using template and HOG feature matching," *Int. J. Adv. Comput. Sci. Appl.*, vol. 11, no. 7, pp. 233-238, 2020, doi: 10.14569/IJACSA.2020.0110730.
- [9] F. An, P. Xu, Z. Xiao, and C. Wang, "FPGA-based object detection processor with HOG feature and SVM classifier," *Int. Syst. Chip Conf.*, vol. 2019, pp. 187-190, 2019, doi: 10.1109/SOCC46988.2019.1570558044.
- [10] A. Ahmed, A. Jalal, and A. A. Rafique, "Salient Segmentation based Object Detection and Recognition using Hybrid Genetic Transform," *2019 Int. Conf. Appl. Eng. Math. ICAEM 2019 - Proc.*, 2019, pp. 203-208 doi: 10.1109/ICAEM.2019.8853834.
- [11] S. Gupta, M. Kumar, and A. Garg, "Improved object recognition results using SIFT and ORB feature detector," *Multimed. Tools Appl.*, vol. 78, no. 23, pp. 34157-34171, 2019, doi: 10.1007/s11042-019-08232-6.
- [12] D. H. Jiang, L. Dai, D. Li, and S. Y. Zhang, "Moving-Object Tracking Algorithm Based on PCA-SIFT and Optimization for Underground Coal Mines," *IEEE Access*, vol. 7, pp. 35556-35563, 2019, doi: 10.1109/ACCESS.2019.2899362.
- [13] K. A. Acharya, R. V. Babu, and S. S. Vadhiyar, "A real-time implementation of SIFT using GPU," *J. Real-Time Image Process.*, vol. 14, no. 2, pp. 267-277, 2018, doi: 10.1007/s11554-014-0446-6.
- [14] D. I. H. Putri, Martin, Riyanto, and C. Machbub, "Object detection and tracking using SIFT-KNN classifier and Yaw-Pitch servo motor control on humanoid robot," *2018 Int. Conf. Signals Syst. ICSSigSys 2018 - Proc.*, 2018, pp. 47-52, doi: 10.1109/ICSIGSYS.2018.8373566.
- [15] C. Kaymak and A. Ucar, "Implementation of Object Detection and Recognition Algorithms on a Robotic Arm Platform Using Raspberry Pi," *2018 Int. Conf. Artif. Intell. Data Process. IDAP 2018*, pp. 1-8, 2019, doi: 10.1109/IDAP.2018.8620916.
- [16] G. Bansal, "Simultaneous Image Segmentation and Object Recognition," *2018 7th Int. Conf. Reliab. Infocom Technol. Optim. Trends Futur. Dir. ICRITO 2018*, 2018, pp. 873-876, doi: 10.1109/ICRITO.2018.8748407.
- [17] S. Chatterjee, F. H. Zunjani, and G. C. Nandi, "Real-Time Object Detection and Recognition on Low-Compute Humanoid Robots using Deep Learning," *2020 6th Int. Conf. Control. Autom. Robot. ICCAR 2020*, 2020, pp. 202-208, doi: 10.1109/ICCAR49639.2020.9108054.
- [18] G. A. E. S. Kumar and R. Sumalatha, "Object Detection using Deep Convolutional Neural Network," *Mach. Learn. Methods Signal, Image Speech Process.*, pp. 181-204, 2021, doi: 10.1201/9781003338789-9.
- [19] O. Mazhar, B. Navarro, S. Ramdani, R. Passama, and A. Cherubini, "A real-time human-robot interaction framework with robust background invariant hand gesture detection," *Robot. Comput. Integr. Manuf.*, vol. 60, no. May, pp. 34-48, 2019, doi: 10.1016/j.rcim.2019.05.008.
- [20] Q. Gao, J. Liu, and Z. Ju, "Robust real-time hand detection and localization for space human-robot interaction based on deep learning," *Neurocomputing*, vol. 390, pp. 198-206, 2020, doi: 10.1016/j.neucom.2019.02.066.
- [21] Q. Gao, J. Liu, Z. Ju, and X. Zhang, "Dual-hand detection for human-robot interaction by a parallel network based on hand detection and body pose estimation," *IEEE Trans. Ind. Electron.*, vol. 66, no. 12, pp. 9663-9672, 2019, doi: 10.1109/TIE.2019.2898624.
- [22] F. Xu et al., "Real-time detecting method of marine small object with underwater robot vision," *2018 Ocean. - MTS/IEEE Kobe Techno-Oceans, Ocean. - Kobe 2018*, pp. 1-4, 2018, doi: 10.1109/OCEANSKOB.2018.8558804.
- [23] A. Dairi, F. Harrou, M. Senouci, and Y. Sun, "Unsupervised obstacle detection in driving environments using deep-learning-based stereovision," *Rob. Auton. Syst.*, vol. 100, pp. 287-301, 2018, doi: 10.1016/j.robot.2017.11.014.
- [24] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once : Unified, Real-Time Object Detection," *2016 IEEE Conf. on Comp. Vis. and Patt. Rec. (CVPR)*, 2016, doi: 10.1109/CVPR.2016.91.
- [25] C. G. Melek, E. B. Sonmez, and S. Albayrak, "Object Detection in Shelf Images with YOLO," *EUROCON 2019 - 18th Int. Conf. Smart Technol.*, pp. 1-5, 2019, doi: 10.1109/EUROCON.2019.8861817.
- [26] A. Corovic, V. Ilic, S. Duric, M. Marijan, and B. Pavkovic, "The Real-Time Detection of Traffic Participants Using YOLO Algorithm," *2018 26th Telecommun. Forum, TELFOR 2018 - Proc.*, 2018, doi: 10.1109/TELFOR.2018.8611986.
- [27] P. Adarsh, P. Rath, and M. Kumar, "YOLO v3-Tiny: Object Detection and Recognition using one stage improved model," *2020 6th Int. Conf. Adv. Comput. Commun. Syst. ICACCS 2020*, pp. 687-694, 2020, doi: 10.1109/ICACCS48705.2020.9074315.
- [28] Susanto, E. Rudiawan, R. Analia, P. D. Sutopo, and H. Soebakti, "The deep learning development for real-time ball and goal detection of barelang-FC," *Proc. IES-ETA 2017 - Int. Electron. Symp. Eng. Technol. Appl.*, vol. 2017, pp. 146-151, 2017, doi: 10.1109/ELECSYM.2017.8240393.
- [29] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," *arXiv*, 2018, doi: 10.48550/arXiv.1804.02767
- [30] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," *arXiv*, 2020, 10.48550/arXiv.2004.10934
- [31] B. Benjdira, T. Khursheed, A. Koubaa, A. Ammar, and K. Ouni, "Car Detection using Unmanned Aerial Vehicles: Comparison between Faster R-CNN and YOLOv3," *2019 1st Int. Conf. Unmanned Veh. Syst. UVS 2019*, pp. 1-6, 2019, doi: 10.1109/UVS.2019.8658300.
- [32] M. Li, Z. Zhang, L. Lei, X. Wang, and X. Guo, "Agricultural greenhouses detection in high-resolution satellite images based on convolutional neural networks: Comparison of faster R-CNN, YOLO v3 and SSD," *Sensors (Switzerland)*, vol. 20, no. 17, pp. 1-14, 2020, doi: 10.3390/s20174938.




- [33] K. Zhao and X. Ren, "Small Aircraft Detection in Remote Sensing Images Based on YOLOv3," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 533, no. 1, 2019, doi: 10.1088/1757-899X/533/1/012056.
- [34] E. Ur Rahman, Y. Zhang, S. Ahmad, H. I. Ahmad, and S. Jobaer, "Autonomous vision-based primary distribution systems porcelain insulators inspection using UAVs," *Sensors (Switzerland)*, vol. 21, no. 3, pp. 1–24, 2021, doi: 10.3390/s21030974.
- [35] X. Long *et al.*, "PP-YOLO: An Effective and Efficient Implementation of Object Detector," *arXiv*, 2020, doi: 10.48550/arXiv.2007.12099

BIOGRAPHIES OF AUTHORS






Suci Dwijayanti    received her M.S. in Electrical and Computer Engineering from Oklahoma State University, Stillwater, OK, USA, in 2013. She received a Fulbright scholarship for her master's degree. In 2018, she received her Doctoral degree from the Graduate School of Natural Science and Technology, Kanazawa University, Japan. Her research interests include signal processing and machine learning. She also worked as an engineer at ConocoPhillips Indonesia Inc. Ltd. from 2007 to 2008. Since 2008, she has been with the Department of Electrical Engineering at Universitas Sriwijaya, Indonesia. She became an IEEE member in 2014. She can be contacted at email: sucidwijayanti@ft.unsri.ac.id.






Bhakti Yudho Suprpto    was born on February 11, 1975 in Palembang (South Sumatra, Indonesia). He is a graduate of Universitas Sriwijaya with a major in Electrical Engineering. His Master's and Doctoral degrees were in Electrical Engineering from Universitas Indonesia (UI). He is an academic staff of Electrical Engineering of Sriwijaya University of Palembang. His research interests include control and intelligent systems. He became an IEEE member in 2014. He can be contacted at email: bhakti@ft.unsri.ac.id.



Mutiyara    was born on July 16, 2000 in Sekayu, South Sumatra, Indonesia. She received her B.E. (S.T.) degree with cum laude honors in Electrical Engineering from the Universitas Sriwijaya, Indralaya, in 2022. Her research interests include robot and image processing. She can be contacted at email: mutiyara85@gmail.com.



Rendyansyah    was born on September 22, 1988 in South Sumatra, Indonesia. He is a graduate of Universitas Sriwijaya with a major in Computer System. His Master's degree in Electrical Engineering was obtained from the Institut Teknologi Sepuluh Nopember (ITS). He is an academic staff of Electrical Engineering of Universitas Sriwijaya. His research interests include robotics and automation. He can be contacted at email: rendyansyah.ilkom@gmail.com.