IJEIS (Indonesian Journal of Electronics and Instrumenations Systems)
Vol.16, No.1, April 2026, pp. 1~12
ISSN (print): 2088-3714, ISSN (online): 2460-7681
DOI: 10.22146/ijeis.75869

Boundary Balls Detection System for Unmanned Surface Vehicle Using Yolov4-Tiny

Diva Diansari Hanggraeni*[1], Ika Candradewi[2], Muhammad Auzan[3]
[1]Electronics and Instrumentation Undergraduate Program, DCSE, FMIPA UGM, Indonesia
[2,3]Department of Computer Science and Electronics, FMIPA UGM, Yogyakarta, Indonesia
e-mail: ***[1] diva.d@mail.ugm.ac.id**, [2] ika.candradewi@ugm.ac.id,
[3] muhammadauzan@mail.ugm.ac.id

Abstrak

Teknologi Unmanned Surface Vehicle (USV) membutuhkan sebuah acuan seperti kamera untuk penentuan arah geraknya. Pengolahan citra digital klasikal masih kurang akurat untuk mendeteksi objek batas dan halangan lintasan karena perubahan intensitas cahaya dan pengaruh lingkungan perairan. Sistem deteksi yang robust dibangun untuk meningkatkan akurasi navigasi kapal. Pada penelitian ini dilakukan analisis hyperparameter subdivision dan learning rate YOLOv4-tiny untuk mendeteksi bola pembatas dan halangan lintasan. Model hasil pelatihan digunakan dalam pengujian video pada berbagai kondisi untuk mengetahui ketahanan sistem. Pengujian dilakukan pada perangkat embedded yang digunakan dalam perangkat otonom.

Pelatihan dilakukan pada 2.000 data dengan subdivision 8 dan learning rate 0,00261 yang menghasilkan mAP sebesar 97,02%, F1-score 95,1% dan avg IoU 87,73%. Sistem deteksi ini robust untuk berbagai waktu pengujian yaitu pagi, siang dan sore dari segi keberagaman intensitas cahaya dan refleksi objek serta kondisi saat ada dan tidaknya gelombang. Sistem mampu bekerja optimal pada siang hari dengan perolehan rata-rata performa F1-score 98,13%, mAP 99,18%, presisi 98,83%, recall 97,45% dan avg IoU 88,75%. Dalam kondisi terdapat gelombang sistem menunjukkan performa tidak jauh berbeda saat tidak ada gelombang, yaitu mAP 91,93%, presisi 96,93%, recall 92,41%, F1-score 94,62% dan avg IoU 87,33%. Rata-rata kecepatan pemrosesan video uji pada Jetson Nano 4GB sebesar 14,2 FPS.

Kata kunci*— Kapal tanpa awak, YOLOv4-tiny, SBC*

Abstract

Unmanned Surface Vehicle (USV) technology requires a reference such as a camera to determine the direction of its motion. The classical digital image processing is still less accurate for detecting boundary objects due to changes in light intensity and the influence of the aquatic environment. This study used hyperparameter subdivision and learning rate analysis of YOLOv4-tiny to detect boundary balls and trajectory obstacles. The training model is used in video testing under various conditions to determine the system's robustness.

The training was carried out with subdivision 8 and a learning rate of 0.00261, which obtained a mAP of 97.02%, 95.1% F1-score, and 87.73% avg IoU. This system is robust for various testing times in terms of the diversity of light intensity and object reflections as well as conditions when there are waves or not. The system can work optimally during the day with an average performance of 98.13% F1-score, 99.18% mAP, and 88.75% avg IoU. In the conditions where there are waves, the system shows that the performance is not much different when there

Received January 4th,2026; Revised April 11th, 2026; Accepted April 29th, 2026

are no waves, namely mAP 91.93%, F1-score 94.62%, and avg IoU 87.33%. The average detection processing speed on the Jetson Nano 4GB is 14.2 FPS.

Keywords— *Unmanned Surface Vehicle, YOLOv4-tiny, SBC*

1. INTRODUCTION

Unmanned Surface Vehicle (USV) technology is widely used to monitor the aquatic environment, surveillance, and data retrieval, which is superior in terms of mobility and high autonomy [1]. Camera-based unmanned ship technology is one of the popular categories of competition in the KKCTBN (Kontes Kapal Cepat Tak Berawak Nasional), boundary balls and trajectory obstacles must be detected for reference to ship motion. This ship is an automatic vehicle that requires a reference such as a camera to determine the direction of its motion.

Robust track limit and obstacle detection systems are crucial for unmanned ship navigation to avoid potential hazards. Detection of trajectory boundary balls using image segmentation methods based on Hue (H), Saturation (S), Value (V) values and morphological operations are unable to adjust lighting changes, adjacent boundary balls cannot be separated and the shadow of the ball in the clear water environment that is formed causes object detection is less accurate [2]. Problems in the application of classical digital image processing, namely the lack of detection performance on USV due to changes in light intensity and environmental influences such as waves and background complexity was also founded on the study [3]. To deal with this problem, it is proposed to use the YOLOv4-tiny algorithm which is able to overcome the decrease in detection accuracy due to changes in light intensity and certain environmental influences. Besides being quite accurate, YOLOv4-tiny is suitable for developing embedded devices widely used in autonomous vehicles [4].

The development of a waterline detection system for USV with a learning approach on the results of multi-stage adaptive segmentation research [5] shows that the system is robust and adaptive in dynamic environments but cannot handle reflections on the water surface. In addition to detecting water boundaries, floating objects can also be considered obstacles the ship must avoid when maneuvering. The YOLOv2 [6] algorithm detects images and videos of floating objects in the sea on USV, and the IOU (Intersection over Union) performance and the resulting recall are 66.69% and 77.12%, respectively. As an upgraded version of YOLOv2, YOLOv3 is also used for target detection for USV in various weather conditions [1]. The results of these studies show that YOLOv3 is quite robust to changes in the natural aquatic environment but is not real-time on USV hardware and has not been able to handle data in complex environments and object data that is only partially visible.

2. METHODS

This study aimed to analyze the performance of YOLOv4-tiny compared to YOLOv3-tiny in detecting objects that block USV passage in the aquatic environment under previously described problem conditions. The video detection system in this study was implemented on a Jetson Nano 4GB embedded device. The system block diagram is shown in Figure 1.

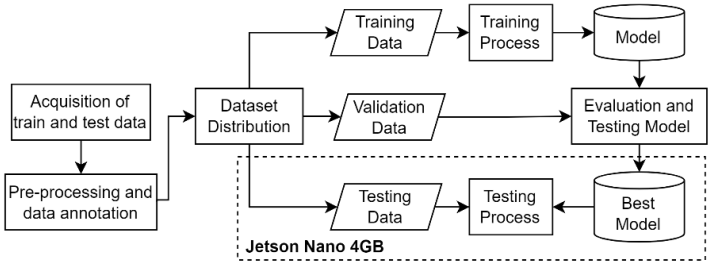

Figure 1 Object detection block diagram

In the training phase, the subdivision value and learning rate gradually vary for analysis. The training model was evaluated on the validation data and tested for processing speed in the Raspberry Pi 4B and Jetson Nano 4GB embedded devices. In the testing process, the system will process the acquisition video, which is taken when the ship passes through the trajectory. Then on each video frame, the object is detected using the best model. The best model is the model that produces the highest F1-score, mAP, and average IOU values.

2.1 Dataset Acquisition

The data used in this study came from the direct image acquisition in two different swimming pools using a Logitech HD C270 camera with a resolution of 720×1280 pixels, the default mode of 30 FPS. The acquisition at the first location was carried out from 02.30 PM - 04.00 PM in cloudy and drizzling weather conditions. Meanwhile, the acquisition at the second location was carried out from 09.30 AM - 10.30 AM, from 00.30 PM - 01.30 PM, and from 04.00 PM - 05.00 PM in sunny, cloudy conditions. The data taken are in the form of images and videos of each ball on the surface of the water and videos of the ship's perspective as it passes through the trajectory according to the provisions of the competition. The camera position is in the center of the ship, about 45 cm above the water surface, and the camera tilt angle is about 5˚.

2. 2 Pre-processing and Data Annotation

The video data for training and testing is frame-extracted to produce an image in ".jpg" format. The image data then goes through a pre-processing stage by providing brightness variations using Microsoft Photo to increase the datasets and represent the diversity of lighting in the virtual environment. The variations in the brightness values given are -70%, -40%, 30%, 50%, and 70%. The total dataset produced is 2,000 images with a distribution of 80% for training and 20% for validation. The test data is from videos that have never been used for training or validation. Table 1 below shows the distribution of data in this study. The amount of test data in various test variations is the number of frames from each test video.

Table 1 Dataset Distribution

Train Data	Validation Data	Test Data							
1,600	400	Morn	Day	Evening	There is a Wave	No Waves	Morn Ref	Day Ref	Evening Ref
		1,356	1,388	1,371	1,282	1,224	351	350	345

This study uses three classes of objects, namely red balls and green balls as the left and right boundary balls of the ship's trajectory and yellow balls as obstacles in the trajectory. In addition to the three objects in the class, there are several non-class objects on the surface of the water, such as bottles, jar lids, and plastic objects with colors and shapes similar to objects in the class, to determine the reliability of the applied detection system.

Data annotation uses the LabelImg application and is done by correctly creating a bounding box on each object which want to detect in the image and the object class manually. The annotation results are saved in a text file format (. txt) containing the object id and object coordinates (x_center, y_center, width, and height). After all, images are annotated, the image files and their annotations are divided into 80% training data and 20% validation data in train.txt and valid.txt files and test.txt files containing the path of each image.

2. 3 System Training Process

The training process is carried out so that the system can detect objects contained in an image or video. Images that have been annotated in the previous stage are trained to get a feature map (convolved feature). The feature maps will be weighted as a classifier in the testing

phase. This stage requires several files that need to be prepared, such as a YOLO configuration file in ".cfg" format containing the network architecture and YOLO hyperparameters, a dataset configuration file with a. "data" format used to connect external data with the training system containing the address of the training data storage, data validation files, backup files, and class types files. In addition, pre-trained weights are also prepared for darknet53.conv.74 for YOLOv3-tiny and yolov4-tiny.conv.29 for YOLOv4-tiny. The training flowchart is shown in Figure 2.

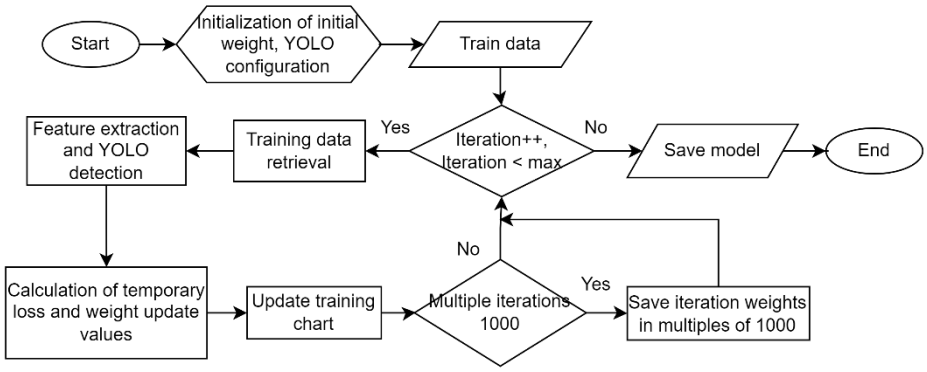

Figure 2 YOLO training flowchart

The YOLOv4-tiny and YOLOv3-tiny model training begins with cloning the object detector from the AlexeyAB Github repository (https://github.com/AlexeyAB/darknet), then the prepared files are uploaded to the Darknet directory. The incoming training data is resized to 416×416 and then forwarded to the feature extractor network and YOLO detection. The feature extraction on YOLOv3-tiny uses Darknet-53 with seven convolution layers and six max-pooling layers. Meanwhile, the feature extractor on YOLOv4-tiny uses CSPDarknet53-tiny, consisting of a CSPBlock module and a convolution block. The system stores the weighted value obtained every 1000 iterations. The training will continue until the iteration reaches the maximum (max_batch) at 6,000 iterations.

Other training parameter configurations used are batch 64, steps 4,800 and 5,400, filter before YOLO layer of 24, classes 3, the learning rate of 0.00261 for YOLOv4-tiny and 0.001 for YOLOv3-tiny which will be varied after obtaining the best subdivision value from the variation. . The training process is carried out using Google Collaboratory (Colab).

2. 4 System Testing Process

The training process produces weights in multiples of 1,000 iterations. These weights are then evaluated on the validation data to obtain the best weights with the F1-score, mAP (mean Average Precision), and maximum Avg IOU values. Furthermore, the best models were tested on the Raspberry Pi 4B and Jetson Nano 4GB to select embedded devices capable of performing detection tasks using the YOLOv3-tiny and YOLOv4-tiny models. The detection system testing phase on the selected device is carried out based on Figure 3. At this stage, each device's FPS (Frames per second) performance is compared.

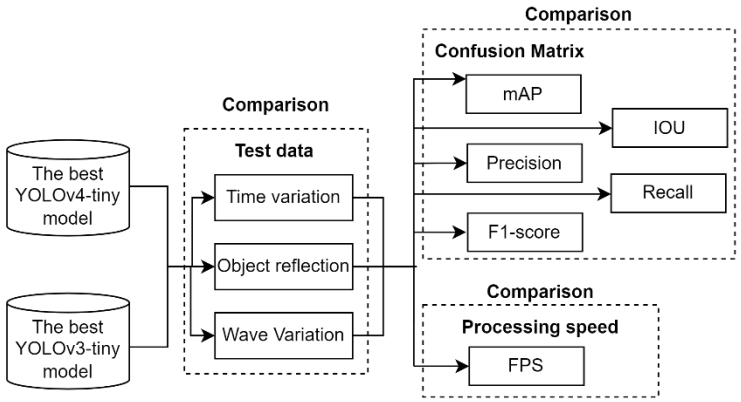

Figure 3 System testing block diagram

System testing using the best model evaluation results on selected embedded devices. The test data is in the form of a video of the perspective of the ship passing through the trajectory, there are 3 test videos for testing time variations, 2 test videos for testing wave variations and 3 test videos for testing object reflections at three different times. Through system testing, we get mAP, IOU, confusion matrix values that produce precision values, recall, and F1-score systems, and processing speed in the form of FPS values on related embedded devices.

2. 5 System Performance Analysis

The performance of the boundary ball and trajectory obstacle detection system can be seen based on the mAP, IOU values , and the performance of the results of the confusion matrix [7]. Mean Average Precision is the average value of Average Precision (AP) for each class, representing the detection system's accuracy. Intersection over Union compares the overlap area between the predicted bounding box and the ground truth bounding box. The greater the IOU value, the more similar the predicted bounding box to ground truth, which shows the object's exact position.

Confusion Matrix is used to compare the results of system detection with the detection that should be. In the case of detection, three possibilities can occur, namely True Positive (TP), False Positive (FP), and False Negative (FN). These possibilities obtained system performance in the form of precision, recall, and F1-score values. Precision shows the number of correct predictions from all predicted data, recall shows the number of data that are predicted correctly from the total number of correct data (ground truth box), and F1-score is the average harmonic precision and recall. The selection of models for systems with an uneven distribution of object classes, as in this study, is recommended to use the highest F1-score value [8]. A system with a high F1 score indicates high precision and sensitivity. The calculation of the evaluation component based on the confusion matrix is as follows:

1. Precision (Pr): TP/(TP+FP)
2. Recall (Re): TP/(TP+FN)
3. F1-score: 2((Pr.Re) / (Pr+Re))

3. RESULTS AND DISCUSSION

3. 1 Model Training Results

In the model training process, hyperparameter variations are carried out in stages to improve the performance of the detection algorithm. The type of hyperparameter that is varied refers to research [9], namely the value of subdivision (8, 16, and 32) and the value of the learning rate.

3. 1.1 Subdivision Value Variations

The subdivision determines the number of mini_batch images that the GPU processes simultaneously. Table 2 shows the effect of subdivision values on YOLOv4-tiny and YOLOv3-tiny.

Table 2 Results of subdivision variations on YOLOv4-tiny and YOLOv3-tiny

Sub division Var	YOLOv4-tiny				YOLOv3-tiny			
	mAP	*Avg* IOU	F1-score	Train Time	mAP	*Avg* IOU	F1-score	Train Time
8	97.02%	87.73%	94.94%	125’	96.95%	78.93%	93.80%	172’
16	96.83%	87.37%	94.80%	196’	96.71%	78.89%	93.67%	224’
32	96.71%	87.36%	94.68%	200’	96.50%	78.74%	93.63%	280’

It can be seen in Table 2 that the smaller the subdivision value, the higher the F1-score, mAP, and average IOU values , which indicate the optimal performance of the model. In addition, the subdivision value also affects the computational speed (training); the smaller the subdivision value, the faster the training time required because, with a subdivision value of 8, the system can learn more images (8 images) in each batch. From the results of this variation, it is determined that the value of subdivision eight will be used in the learning rate variation process on YOLOv4-tiny and YOLOv3-tiny because it produces the best performance.

3. 1.2 Variation of Learning Rate Value

The learning rate determines the speed at which the neural network learns and updates the weights during the training process. A significant learning rate value results in fast model learning, but the resulting weight can be less than optimal. On the other hand, a small learning rate makes learning slow but can produce optimal weights. The learning rate variation in this study consisted of a variety of the rank from 0.1 to 0.00001 and a variation of the multiplier or constant value. The graph in Figure 4 below shows the effect of variations in the rank value on the learning rate on the mapped value.

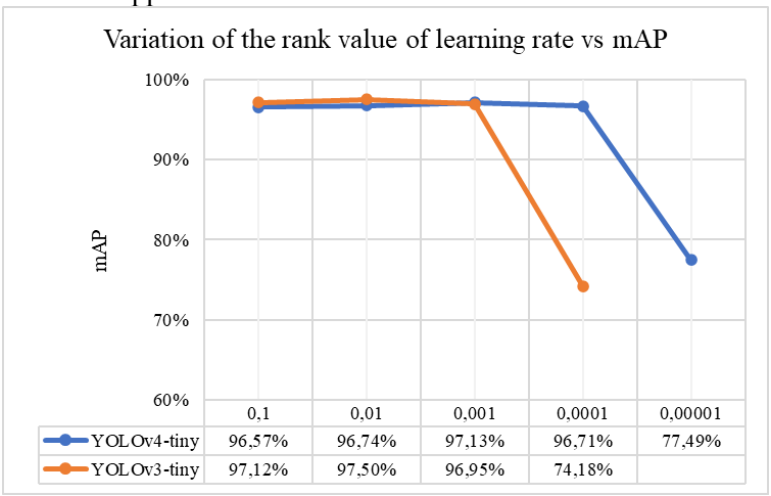

Figure 4 Graph of variation in the value of the rank of learning rate to the value of mAP

The graph above shows the mAP value, which continues to increase as the learning rate decreases to a certain maximum point, after which the mAP value will begin to decrease until, at a specific learning rate value, there is a drastic decrease in the mAP. Variations in rank on YOLOv4-tiny show that the mAP increase continues from a learning rate of 0.1 to 0.001 (maximum map 97.13%). A slight decrease occurs in the next value, and a learning rate of 0.00001 satiations occurs during the training process so that the resulting mAP decreased

drastically by 77.49%. This condition also applies to YOLOv3-tiny, as shown in the graph above. Satiation in the training process occurs because the limitations of the given iterations make it difficult for the model to achieve optimal conditions, and the training loss value is constantly high. In addition to the variation of the rank value, the YOLOv4-tiny also varies the value of the learning rate constant, as shown in Table 3. At this learning rate value, it is possible for learning to be trapped in a local minimum during the training, a local minimum is a point that appears as a minimum value but is not a point that produces an optimal value. The most ideal learning rate value is the value that produces the optimal level of accuracy and does not require a long training time [10].

Table 3 The results of variations in the value of the learning rate constant on YOLOv4-tiny

Learning rate Var	mAP	Average IoU	F1-score
0,001	97.13%	87.01%	94.90%
0,00261	97.02%	87.73%	95.10%
0,005	96.55%	88%	95.09%
0,009	96.48%	88.31%	94.99%

Based on Table 3, it can be seen that the greater the value of the constant causes a decrease in the mAP value, although it is not significant. The greater the value of the learning rate constant, the greater the value causes the network accuracy to decrease because the range for determining weight changes in the neural network is getting bigger. Considering the balance of each resulting performance, the hyperparameter value on YOLOv4-tiny to train the test model is subdivision eight, and the learning rate is 0.00261. Meanwhile, in YOLOv3-tiny, the selected hyperparameter value is subdivision eight, and the learning rate is 0.01. Figure 5 and Figure 6 are YOLOv4-tiny and YOLOv3-tiny training graphs, respectively.

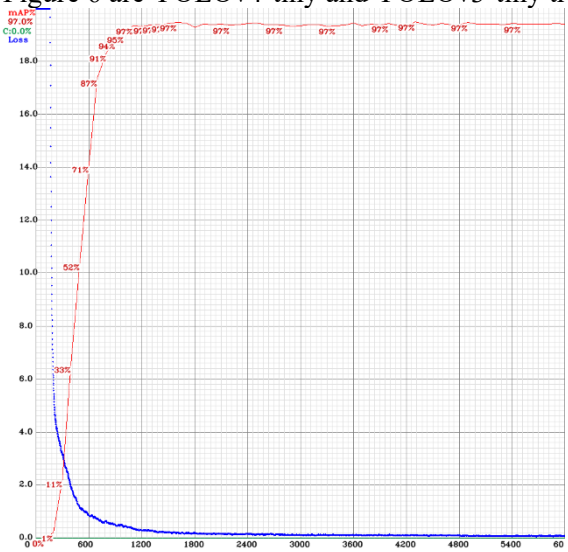

Figure 5 YOLOv4-tiny training graph

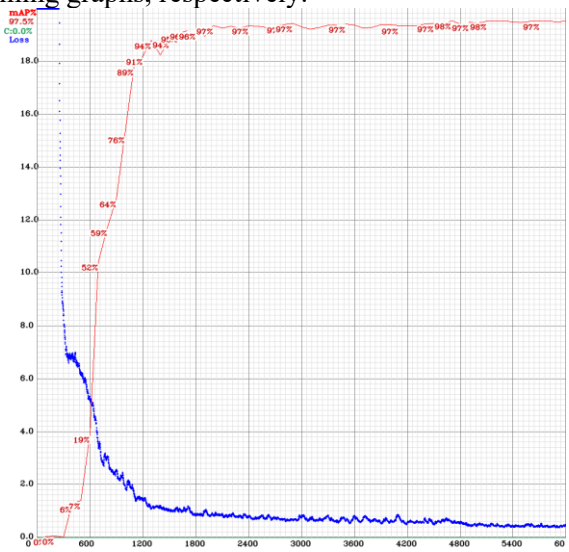

Figure 6 YOLOv3-tiny training graph

The results of the YOLOv4-tiny training in Figure 5 show that as the number of iterations increases, the Avg loss value decreases until it converges to the smallest value obtained. Meanwhile, the mAP value continues to increase with increasing iterations. Although there are some decreases, the mAP value is still in the range of 96% - 97% until the last iteration is obtained, a value of 97.02%. The Avg loss value in the 1st iteration is still quite significant. With increasing iterations, this value decreases until the last iteration is 0.0767. In YOLOv3-tiny, shown in Figure 6, the Avg loss value generally moves down as the number of iterations continues to increase, but this movement is sometimes inconsistent. This erratic movement also

occurs in the mAP parameter, where there is a momentary decrease in value at a particular iteration.

3. 2 Model Evaluation and Testing

The model evaluation was conducted to select the best model from a total of 6 weights multiples of 1,000 training results, while model testing aimed at selecting embedded devices suitable for carrying out detection tasks with YOLOv4-tiny and YOLOv3-tiny. The data in Table 4 below is the result of the performance of the training model that has been carried out.

Table 4 Model performance results at the 1000th iteration and its multiples

Iteration	YOLOv4-tiny			YOLOv3_tiny		
	mAP	Avg IOU	F1-score	mAP	Avg IOU	F1-score
1000	95.02%	73.46%	92.62%	60.44%	29.71%	82.93%
2000	96.90%	82.98%	94.10%	92.50%	48.55%	91.96%
3000	96.60%	84.43%	94.51%	91.10%	52.38%	93.59%
4000	97.03%	84.47%	94.63%	90.45%	44.72%	94.35%
5000	96.98%	87.56%	95.08%	97.17%	80.79%	94.17%
6000	97.02%	87.73%	95.10%	97.50%	81.03%	94.48%

Based on Table 4, the optimal model taken for the test data testing process is the weight of the 6,000th iteration for each method. The determination of this model considers the highest F1-score at the end of the iteration, namely 95.1% (YOLOv4-tiny) and 94.48% (YOLOv3-tiny). In addition to achieving the highest F1-score performance, the last iteration also produces the highest Avg IOU value. The best model in each method was then tested on a Raspberry Pi 4B and Jetson Nano 4GB. The embedded device that can produce the highest fps was chosen to perform detection tests on video. The framework used for testing the Raspberry Pi 4B is Darknet-NNPACK, while the Darknet framework is used for the Jetson Nano 4GB. The frame rate test results are shown in Table 5.

Table 5 Frame rate test results on embedded devices

Platform	Performance	YOLOv4-tiny	YOLOv3-tiny	YOLOv3
Google Colab	mAP	97.02%	97.50%	99.25%
	Avg IOU	87.73%	81.03%	84.71%
	FPS video	40.7	33.6	14.2
Raspberry Pi 4B	FPS video	1.3	0.5	0.2
Jetson Nano 4GB	FPS video	14.6	14.7	2.2

Based on Table 5, the high mAP value in YOLOv3 is not supported by the optimal frame rate in the case of video object detection, in contrast to YOLOv4-tiny and YOLOv3-tiny, which have slightly lower mAP performance of around 1.75% - 2.23% but still produces the optimal frame rate for the same case. This shows that the lightweight version of the YOLO algorithm can achieve optimal frame rates while maintaining its accuracy, especially when the model is applied to embedded devices with computational limitations. The Raspberry Pi 4B can process object detection per frame for 0.77 seconds to 5 seconds, while the Jetson Nano 4GB can detect objects for 0.068 seconds to 0.45 seconds for each frame so that the testing process for detecting barrier balls and track barriers on This research uses a Jetson Nano 4GB device.

3. 3 Time Variation Test

This time variation test is carried out to determine the detection system's accuracy in various weather conditions in the natural environment, especially related to various light

intensities. The time variation test data that has been annotated with the amount according to Table 1 is processed as a ground truth box and compared with the predicted results of the bounding box to produce detection performance, as shown in Table 6.

Table 6 Time variation test detection performance

Performance	Morning		Afternoon		Evening	
	YOLO v4-tiny	YOLO v3-tiny	YOLO v4-tiny	YOLO v3-tiny	YOLO v4-tiny	YOLO v3-tiny
Precision	97.69%	59.54%	98.16%	60.97%	95.22%	51.32%
Recall	94%	87.76%	96.55%	93.33%	92.45%	81.86%
F1-score	95.81%	63.33%	97.35%	73.76%	93.82%	63.09%
Avg IOU	88.27%	32.52%	88.81%	41.27%	71.54%	33.62%
mAP @0,5	97.83%	76.36%	98.99%	90.06%	93.13%	67.01%

Based on Table 6, during the day, the system can detect the ball optimally with an F1-score of 97.35% for YOLOv4-tiny and 73.76% for YOLOv3-tiny. The prediction results on YOLOv3-tiny have large enough FP and FN values , so the resulting precision is only 60.97%, and recall is 93.33%. In YOLOv4-tiny, the high TP value and low FP and FN values make the precision and recall value 98. 16% and 96.55%, respectively. This method can maintain detection performance when testing is carried out in the morning and evening. If the mAP value of the test results is compared with the mAP value of the training results, there is an indication of overfitting on YOLOv3-tiny because the difference in values is quite significant. Overfitting on YOLOv3-tiny affects the magnitude of the FP value, which causes the precision value to be relatively small. Figure 7 and Figure 8 are examples of detection results with both methods on the same frame.

Figure 7 Example of YOLOv4-tiny result

Figure 8 Example of YOLOv3-tiny result

Figure 7 shows that YOLOv4-tiny can predict the object's bounding box correctly. Figure 8 shows the presence of multiple detections of a single object, resulting in YOLOv3-tiny having up to 3 times the number of object detections compared to YOLOv4-tiny. From the figure, it is also known that the predicted results of the bounding box are not right around the object, causing the combined area between the ground truth box and the bounding box of the prediction results to be larger, resulting in the small IOU value obtained.

Based on the data obtained, it can be said that the barrier ball detection system and ship trajectory barrier work optimally with the YOLOv4-tiny method during the day. This method also has high performance on tests in the morning and evening which shows the method is robust enough to be used at various times and weather conditions in this study.

3. 4 Wave Variation Test

The wave variation test is carried out to determine the response of the detection system when condition A: There is a wave and condition B: There is no wave. In this test, the IOU

value is used to determine how accurate the prediction results of the bounding box are to the ground truth box for each object that moves dynamically on the surface of the water.

Table 7 Wave variation detection performance with YOLOv4-tiny and YOLOv3-tiny

Performance		TP	FP	FN	Pr	Re	F1-score	mAP	Avg IOU
A	YOLOv4-tiny	7,855	249	645	97%	92%	95%	91.93%	87.33%
	YOLOv3-tiny	7,093	8.106	1.407	47%	83%	60%	30.11%	69.63%
B	YOLOv4-tiny	6,879	165	451	98%	94%	96%	97.80%	88.13%
	YOLOv3-tiny	6,415	6.593	915	49%	88%	63%	32.31%	76.05%

In addition to maintaining the Avg IOU value in the presence or absence of waves, the YOLOv4-tiny model also has a high F1-score value of 94.62% in the first condition and 95.71% in the second condition. Based on Table 7, the average Avg IoU value detected with YOLOv4-tiny in condition A reached a value of 87.33%. Not much differs when condition B was 88.13%. In contrast to YOLOv4-tiny, the YOLOv3-tiny model only produces Avg IOU values of 30.11% of condition A and 32.31% of condition B, these values indicate that the model is not accurate enough to predict the bounding box. The reason is that there are multiple detections in several objects, resulting in a prediction area of the bounding box that is wider than the ground truth, thereby lowering the average IOU score. Figure 9 shows the processing of IOU values from the ground truth box and the predicted results of the bounding box.

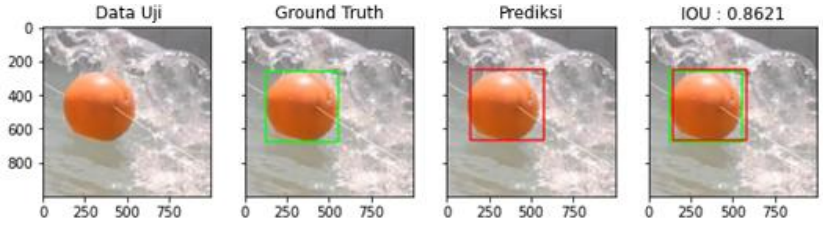

Figure 9 An illustrative example of processing IOU values

3. 5 Object Reflection Test

The object reflection test on the water surface is carried out to determine the system's accuracy in detecting objects that are included in the class, not the reflection of objects that are formed. The following table 8 shows the detection performance results on object reflection testing.

Table 8 Object reflection test detection performance

Performance	Morning		Afternoon		Evening	
	YOLOv 4-tiny	YOLOv 3-tiny	YOLOv 4-tiny	YOLOv 3-tiny	YOLOv 4-tiny	YOLOv 3-tiny
Precision	96.22%	46.59%	99.49%	45.46%	96.90%	62.92%
Recall	98.32%	84.04%	98.34%	81.09%	95.08%	92.04%
F1-score	97.26%	59.94%	98.91%	58.26%	95.98%	74.74%
Avg IOU	81.24%	29%	88.69%	28.30%	76.43%	41.61%
mAP @0,5	95.80%	60.87%	99.36%	68.63%	95.45%	72.91%

The object reflection test on the test video with YOLOv4-tiny shows that no ball reflection on the water surface is detected as a ball in the morning, afternoon, or evening. Based on Table 8, the predicted mAP values with YOLOv4-tiny reached 95.8% for the morning reflection, 99.36% for the afternoon reflection and 95.45% for the afternoon reflection. This value is also supported by the Avg IOU value which shows the prediction results are quite close

to the ground truth box. The low mAP value of the YOLOv3-tiny test results in the test with the smaller amount of data strengthens the indication of overfitting in the related model.

3. 6 Testing Performance under Various Conditions

Based on the results of testing the detection system, it can be seen that the method that produces the best precision, recall, and F1-score values is YOLOv4-tiny. This method can be said to be robust for various testing times, namely morning, afternoon, and evening in terms of the diversity of light intensity and object reflections as well as conditions when there are waves or not. The system can work optimally during the day with an average mAP performance of 96.82%, 98.83% precision, 97.45% recall, 98.13% F1-score, and 84.76% Avg IOU. Moreover, the system can also maintain this performance in the morning and evening, which applies to object reflection testing. The system works optimally during the day, and performance does not decrease much in the morning and evening. In the condition where there is a wave, the system shows a performance that is not much different when there is no wave, with an mAP value of 91.93%, a precision of 96.93%, and recall of 92.41%, F1-score 94.62%, and avg IoU 87.33 %.

On the other hand, there are still some errors in the system, such as misclassification, misdetection of non-class objects, and multi-label classification. These errors do not occur continuously but only in a few frames. The misclassification was caused by the lack of training data for the yellow ball class compared to the other two classes, the system was still less than optimal in studying the features of the class during training, resulting in a misclassification of the yellow ball with a fairly low confidence score. The misdetection are caused by the position and part of the object caught on camera. In some frames the object is only partially visible, has the same color as other classes, especially in the afternoon, the shape of the object looks like a ball. Multi label classification can occur because YOLOv4-tiny uses a logistic classifier binary cross entropy with a sigmoid activation function which classifies each class in terms of yes or no probability in each class, thus allowing a bounding box to issue two object classes at once. The classification system for this case will be suitable if you use the softmax function in the logistic classifier, this can happen because the object classes have a fairly high degree of similarity, especially in terms of the shape of the same object.

3. 7 Processing Speed Test

This test is needed to determine the computational time required by the model when detecting objects in frames per second. The faster the model successfully detects, the more optimal the model's performance.

Table 10 Test video processing FPS results

FPS Video	1	2	3	4	5	6	7	8	**Average**
YOLOv4-tiny	13.8	14.2	14.3	14.3	14.2	14.2	14.3	14.1	**14.2**
YOLOv3-tiny	15.5	15.5	14	14.7	14.4	14.8	14.7	14.8	**14.8**

Based on Table 10, it can be seen that the test video processing with YOLOv4-tiny produces an average of 14.2 FPS, which means that 1 video frame can be processed in 0.07 seconds, the minimum frame rate value is 13.8 FPS, and the maximum frame rate achieved is 13.8 FPS. 14.3 FPS. YOLOv3-tiny is slightly superior in processing time because this model has fewer convolution layers than YOLOv4-tiny. However, YOLOv3-tiny has much lower precision, recall, and F1-score than YOLOv4-tiny. By considering the balance of performance in terms of processing speed and system accuracy, YOLOv4-tiny remains the most optimal method in the detection system for ball boundary and trajectory obstacles in this study. Through this research, the processing speed obtained from testing on the Jetson Nano 4GB is compared with the camera frame rate during the acquisition of the test video to evaluate the real-time performance of the detection system. The camera frame rate used by default is 30 FPS. This value is used as a reference for the real-time system in this study. From the test results and references, it can be said that the system has not been included in the real-time category.

4. CONCLUSIONS

Based on the research that has been done, it can be concluded that on YOLOv4-tiny, the best subdivision and learning rate values are 8 and 0.00261, respectively, which can produce an mAP value of 97.02%, F1-score 95.1%, and Avg IOU 87, 73%. The barrier ball detection system and ship trajectory barrier with YOLOv4-tiny are robust for various testing times, namely morning, afternoon, and evening in terms of the diversity of light intensity and object reflection as well as conditions when there is or is not a wave. The system can work optimally during the day with an average mAP performance of 99.18%, 98.83% precision, 97.45% recall, 98.13% F1-score, and 88.75% Avg IOU. In the condition where there is a wave, the system shows performance that is not much different when there is no wave, namely mAP 91.93%, precision 96.93%, recall 92.41%, F1-score 94.62%, and Avg IOU 87.33%. The average processing speed of the test video detection with YOLOv4-tiny on the Jetson Nano 4GB is 14.2 FPS, compared to the camera FPS when the system's video acquisition cannot be said to be real-time.

REFERENCES

[1] Y. Li *et al.*, "A novel target detection method of the unmanned surface vehicle under all-weather conditions with an improved yolov3," *Sensors (Switzerland)*, vol. 20, no. 17, pp. 1–14, 2020, doi: 10.3390/s20174885.

[2] C. Zhang, J. Liu, J. Xiao, and J. Xiong, "Review on light vision detection of surface obstacles for USV," *2021 36th Youth Acad. Annu. Conf. Chinese Assoc. Autom.*, pp. 390–395, 2021, doi: doi: 10.1109/YAC53711.2021.9486488.

[3] E. Gershikov, "Efficient horizon line detection using an energy function," *Proc. 2017 Res. Adapt. Converg. Syst. RACS 2017*, vol. 2017-Janua, pp. 110–115, 2017, doi: 10.1145/3129676.3129732.

[4] Z. Jiang, L. Zhao, L. I. Shuaiyang, and J. I. A. Yanfei, "Real-time object detection method for embedded devices," *arXiv*, vol. 3, no. October, pp. 1–11, 2020.

[5] W. Zhan *et al.*, "Autonomous visual perception for unmanned surface vehicle navigation in an unknown environment," *Sensors (Switzerland)*, vol. 19, no. 10, 2019, doi: 10.3390/s19102216.

[6] S. J. Lee, M. Il Roh, H. W. Lee, J. S. Ha, and I. G. Woo, "Image-based ship detection and classification for the unmanned surface vehicle using real-time object detection neural networks," *Proceedings of the International Offshore and Polar Engineering Conference*, vol. 2018-June. pp. 726–730, 2018.

[7] F. J. P. Montalbo, "A computer-aided diagnosis of brain tumors using a fine-tuned Yolo-based model with transfer learning," *KSII Trans. Internet Inf. Syst.*, vol. 14, no. 12, pp. 4816–4834, 2020, doi: 10.3837/tiis.2020.12.011.

[8] R. Padilla, W. L. Passos, T. L. B. Dias, S. L. Netto, and E. A. B. Da Silva, "A comparative analysis of object detection metrics with a companion open-source toolkit," *Electron.*, vol. 10, no. 3, pp. 1–28, 2021, doi: 10.3390/electronics10030279.

[9] A. P. Saputra, "Waste Object Detection and Classification using Deep Learning Algorithm: YOLOv4 and YOLOv4-tiny," *Turkish J. Comput. Math. Educ.*, vol. 12, no. 14, pp. 1666–1677, 2021.

[10] Sahrul, S. Hadinisa, M. Koyimati, A. Irawan, and H. Nugroho, "Analisis Learning Rate pada Metode Transfer Learning untuk Sistem Pendeteksi Api Sahrul, Sabila Hadinisa, Muhamad Koyimatu, Ade Irawan, Herminarto Nugroho," *Semin. Nas. Microw.*, pp. 8–11, 2018.