irvine

By Bhakti Suprapto

YOLO Algorithm-Based Surrounding Object Identification on Autonomous Electric Vehicle

Irvine Valiant Fanthony
Dept. of Electrical Engineering
Faculty of Engineering, Universitas
Sriwijaya
Palembang, Indonesia
irvine2501@gmail.com

Zaenal Husin
Dept. of Electrical Engineering
Faculty of Engineering, Universitas
Sriwijaya
Palembang, Indonesia
zaenalhusin@gmail.com

Hera Hikmarika
Dept. of Electrical Engineering
Faculty of Engineering, Universitas
Sriwijaya
Palembang, Indonesia
herahikmarika@gmail.com

Suci Dwijayanti
Dept. of Electrical Engineering
Faculty of Engineering, Universitas
Sriwijaya
Palembang, Indonesia
sucidwijayanti@ft.unsri.ac.id

Bhakti Yudho Suprapto
Dept. of Electrical Engineering
Faculty of Engineering, Universitas
Sriwijaya
Palembang, Indonesia
bhakti@ft.unsri.ac.id

Abstract – An autonomous vehicle must be equipped with a camera, which works by providing visual input that is used to detect objects around the autonomous electric vehicle. Currently, no method has been implemented in real-time. Thus, this study utilized the You Only Look Once (YOLO) algorithm to detect objects in real-time around the autonomous electric vehicle. The objects were limited to humans, motorcycles, and cars. The results showed that the most compatible YOLO model for the system was the Tiny YOLOv4 model which was built with the darknet framework. The simulation experiment showed that detection accuracy was 80% and was able to transmit information in a form of data location of the object to the microcontroller. A success rate of 100% was obtained from 10 tests. Hence, it showed that the YOLO was able to detect objects and provided input to the steering control system. Meanwhile, the depth information method was used to measure the distance of the object to the vehicle in real-time with an accuracy of 60%. Real-time testing was conducted to test whether the autonomous electric vehicle can avoid objects in front of it by providing input from the detection results of the Tiny-YOLOv4 model object. The success rate of the system in real-time experiments was 100%.

Keywords: Autonomous Electric Vehicle, YOLO, Object Detection, Image Processing

I. INTRODUCTION

Traffic accidents increase every year and have become the third killer in Indonesia after coronary heart disease and tuberculosis [1]. The highest accident-causing factor is the human factor (human error) caused by the carelessness of the driver, the driver's lack of understanding of driving techniques, traffic ethics, and communication on the road [2].

Along with technological developments in transportation, an autonomous electric vehicle was developed. By using an autonomous electric vehicle, it is expected to reduce the number of accidents caused by human negligence [3]. To work autonomously, the vehicle must be equipped with various sensors, one of which is a camera [4] which is in charge of providing visual input that is used to detect around the autonomous electric vehicle. Research for object detection has been carried out in various methods, such as the K-NN algorithm [5], MASK-RNN [6], Lane-Net based on Convolutional Neural Network (CNN) [7], and using Hough Transform to detect objects around the road [8].

Those studies showed good accuracy results. Nevertheless, they did not run in real-time [3,6-8]. Therefore, this study utilized the You Only Look Once (YOLO) algorithm, which is the developed version of the convolutional neural network (CNN) method. The object detection using the YOLO algorithm has previously been carried out by Hendawan Soebakti [9], YOLO algorithm was used in soccer robots that had a webcam to detect the goal and the ball. That study obtained an accuracy of 87.07% with frames per second of 28.3 fps based on that study [9], YOLO has better fps than the other methods [3, 6-7]. The Tiny-YOLOv4 is the fastest detection with 371 fps and is more accurate than other real-time detections [10].

Thus, the YOLO algorithm was used in this study. This algorithm was then applied to run in real-time to detect road objects that were later traversed by autonomous electric vehicles.

The rest of this paper is organized as follows. In Section 2, a brief description of YOLO is explained, followed by the research method used in this study as shown in Section 3. Section 4 represents the results and discussion. Finally, the paper is concluded in Section 5.

II. YOU ONLY LOOK ONCE (YOLO)

You Only Look Once or commonly abbreviated as YOLO is a deep learning algorithm used to perform real-time detection. To detect objects, YOLO uses a detection system by using a repurpose classifier or localization which is a model applied to an image at several locations and scales.

YOLO uses an artificial neural network approach to detect an object in an image or video. It divides the image into several parts and predicts every boundary and possible object in each region [10]. Each bounding box is compared with every predicted possibility. The YOLO detection algorithm has several advantages compared to other classifier-oriented algorithms, it can be seen in all images during testing with predictions obtained globally on the image. This makes predictions with neural network synthesis unlike the Region Convolutional Neural Network (R-CNN) algorithm which requires thousands for an image, thus making YOLO several times faster than R-CNN [11]. YOLO has convolutional layers and 4 architectures max pool layer as can be seen in Fig. 1.
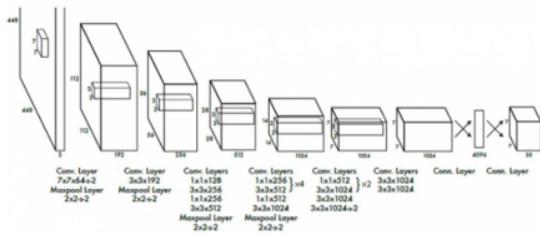
Fig. 1.   The architecture of the YOLO algorithm[12]

YOLO uses a method to apply a single neural network to the entire image. This network divides the image into regions and predicts bounding boxes and probabilities. For each bounding box, it will calculate the probability needed to classify it as an object. In the final stage, the bounding box with the highest value is used as a separator of objects from one another as shown in Fig. 2.
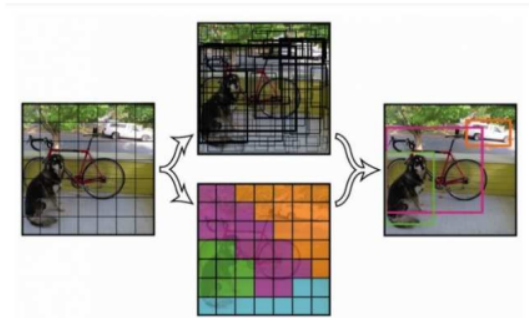

Fig. 2.   Illustration of the YOLO algorithm architecture [12]

III. RESEARCH METHOD

A. The System Design

System design in this study consists of two parts, which are software design and hardware design. The hardware in this study was composed of several devices used to adjust the speed, the direction of the steering wheel, and sensors. The hardware design can be seen in Fig. 3.
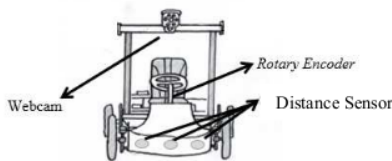

Fig. 3.   The design of the position of the hardware

Dataset used in this study is a video that was taken by the autonomous electric vehicle in the Universitas Sriwijaya, Indralaya. The video image was recorded using a webcam on an autonomous electric vehicle that was driven manually. The webcam has a resolution of 1080p with 30 fps. The video was taken twice on the same road. Then, the video was converted into an image to be annotated to separate the objects to be identified.

After the object is annotated, the training process is carried out using the Darknet repository with the YOLO network. The training process is done using python 3.7 as the programming language with the library of TensorFlow 1.15.0 and OpenCV-python 4.5.1.18.

B. System Testing

The system was tested using a model obtained from the training process. The test consists of two types. The first test is a simulation and the second one is testing the model in real-time. In the simulation test, the purpose is to see whether the YOLO algorithm can truly identify objects with the input of a video of the road segment of the Universitas Sriwijaya, Indralaya campus.

The evaluation method used is a confusion metric as can be seen in Table I.

TABLE I.   CONFUSION METRIC

	Actual Value	
Predicted Values	True Positive (TP)	False Positive(FP)
	False Negative(FN)	True Negative(TN)

As shown in the table, the predicted value is the *confidence* value given by the YOLO model and the actual value is the specified target value. So from the table, it can be determined the value of accuracy with the equation as,

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} . \qquad (1)$$

True Positive (TP) is the number of positive data that the system has classified as true, and False Positive (FP) is the number of positive data that the system has been classified as invalid. On the other hand, True Negative (TN) is the number of negative data correctly classified by the system, and False Negative (FN) is the number of negative data that is misclassified by the system.

If the model provides good accuracy, testing is then carried out in real-time using a webcam located on an autonomous electric vehicle. The results of the identification using the YOLO algorithm send the coordinates of the road to the Arduino using serial communication, where the microcontroller then determines the movement of the steering wheel.

IV. RESULT AND DISCUSSION

A. Train Data Collection

This study used primary data in the form of a video image captured by the webcam attached to the autonomous electric vehicle. The Logitech C925e webcam with a resolution of 1080p 30 fps was placed on the front side of the autonomous electric vehicle with an altitude of 1 meter above ground level. The dataset was taken in the environment around the Universitas Sriwijaya, Indralaya campus in daytime conditions. The video capture process used software written in the programming language python. The video was taken with a resolution of 640pixels x 480pixels with a 4:3 ratio and a framerate of 30 fps. Some additional secondary data originating from youtube were added with a composition of 31% secondary data and 69% primary data due to the lack of human objects in the primary data taken.

The video dataset was saved in .mp4 format which was then be divided into several images using the Free Video to JPG Converter software. Images from the converted video were saved in .jpg format. The dataset used for training consisted of various objects identified around the autonomous electric vehicle. The number of training images was 2000 images, which consist of 1200 human objects, 1231 motorcycles, and 1200 cars. Fig. 4 is an example of an image of the training data that has been collected.

Fig. 4. Example of Training Data Image

B. Processing Training Data

Training data images were annotated for all objects during the training process. The annotation was performed using software called labeling which is written in python. Fig. 5 is an example of the process labeling using a bounding box.
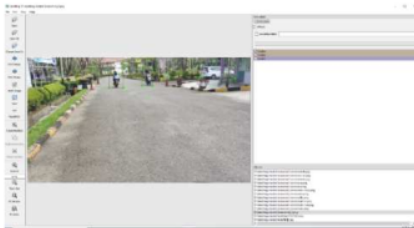

Fig. 5. Example of labeling process

Annotation was performed on all objects in the training data that were going to be identified. In this process, the coordinates of each were stored bounding box according to its class against the image. Annotation using coco dataset saved in .txt format which can be seen in fig. 6. The output of the coco dataset annotation has the format

<object-class> <x_center> <y_center> <width>

In the coco dataset bounding box format, there are classes, x position, y position, width, and height of the class object that has been bounding box. In this study, 0 represents the class of cars, 1 represents the class of motorcycles and 2 represents the class of humans.

```
File  Edit  Format  View  Help
1 0.465885 0.349537 0.081771 0.217593
1 0.564323 0.373611 0.072396 0.334259
0 0.630469 0.302778 0.089063 0.129630
```

Fig. 6. Example of output .txt boundingbox

C. YOLO Training

The training process to identify objects around the autonomous electric vehicles was performed using the YOLO architecture through the Darknet repository. The YOLO network used has a .cfg format where the file can be modified to match the number of classes to be identified. In this training, two types of initial weights were used, Tiny-YOLO v3 and Tiny YOLO v4 in .weight format. Each model was then compared in the number of epochs in the training process. The number of epochs carried out in this study was 100, 200, and 400 epochs. At each training result, a test was carried out to get the accuracy value of the object detection and YOLO confidence value in identifying the object.

In this study, there were two types of YOLO, namely Tiny YOLOv3 and Tiny YOLOv4. Both architectural models used the Tiny type model because they have a simpler architecture than the older YOLO architecture and use less GPU memory which results in higher fps being obtained.

Tiny YOLOv4 an improvement over YOLOv3 with some modifications to the original YOLO network structure. The simple structure of the Tiny-YOLOv4 network can be seen in Table II.

TABLE II. TINY YOLOV4 ARCHITECTURE

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

The training was also conducted using Tiny YOLOv3, which is simplification of YOLOv3 with some modifications to the tiny YOLOv3 network structure. The Tiny YOLOv3 network structure can be seen in Table III. The Tiny YOLOv3 has fewer layers than Tiny YOLOv4

Tiny YOLOv3 and Tiny YOLOv4 training were conducted in batches of 64 with a total epoch of 100, 200, and 400 epochs. Each model consists of three classes consisting of humans, cars, and motorcycles with a threshold of 0.6.

TABLE III. TINY YOLOV3 ARCHITECTURE

Layer	Type	Size/Stride	Input	Output
0	Convolutional +bnorm leaky	3 x 3 / 1	416 x 416 x 3	416 x 416 x 16
1	Maxpool	2 x 2 / 2	416 x 416 x 16	208 x 208 x16
2	Convolutional +bnorm leaky	3 x 3 / 1	208 x 208 x16	208 x 208 x 32
3	Maxpool	2 x 2 / 2	208 x 208 x 32	104 x 104 x 32
4	Convolutional +bnorm leaky	3 x 3 / 1	104 x 104 x 32	104 x 104 x 64
5	Maxpool	2 x 2 / 2	104 x 104 x 64	52 x 52 x 64
6	Convolutional +bnorm leaky	3 x 3 / 1	52 x 52 x 64	52 x 52 x 128
7	Maxpool	2 x 2 / 2	52 x 52 x 128	26 x 26 x 128
8	Convolutional +bnorm leaky	3 x 3 / 1	26 x 26 x 128	26 x 26 x 256
9	Maxpool	2 x 2 / 2	26 x 26 x 256	13 x 13 x 256
10	Convolutional +bnorm leaky	3 x 3 / 1	13 x 13 x 256	13 x 13 x 512
11	Maxpool	2 x 2 / 1	13 x 13 x 512	13 x 13 x 512
12	Convolutional +bnorm leaky	3 x 3 / 1	13 x 13 x 512	13 x 13 x 1024
13	Convolutional +bnorm leaky	1 x 1 / 1	13 x 13 x	13 x 13 x 256
14	Convolutional +bnorm leaky	3 x 3 / 1	13 x 13 x 256	13 x 13 x 512
15	Convolutional+ linear	1 x 1 / 1	13 x 13 x 512	13 x 13 x 255
16	YOLO			

D. Training Tiny YOLOv3 dan Tiny YOLOv4

TABLE IV. COMPARISON OF AVERAGE LOSS FOR EACH EPOCH

Epoch	Average Loss	
	Tiny YOLOv3	Tiny YOLOv4
100	1.469462	0.664295
200	1.178938	0.523478
400	0.997723	0.366202

Table IV shows the training results of Tiny YOLOv3 and tiny YOLOv4 with batches 64 and the number epoch of 100,200, and 400 epoch. As shown in the table, at 100 epochs, YOLOv4 got the initial loss of 394.9009 and the final average loss was 0.664295 with an iteration of 1578. Meanwhile, YOLOv3 got an average initial loss of 566.8707 and a final average loss of 1.469462 in the iteration of 1578.

The test results using YOLOv3 Tiny model detection and Tiny YOLOv4 with 100, 200, and 400 epochs can be seen in Table V. Tiny YOLOv4 with 400 training epochs has the best accuracy, which is 83% while the Tiny YOLOv3 model with 100 training epochs got the worst accuracy with 8% during 8 tests. Therefore, based on the test results and refer to [13-16], the model used in this study is Tiny YOLOv4 with the epoch of 400.

TABLE V. COMPARISON OF TINY YOLO MODEL DETECTION ACCURACY

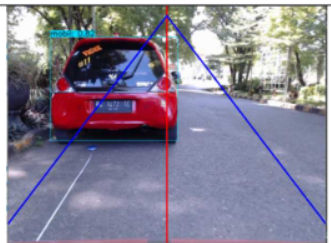| Epoch	Accuracy	
	Tiny YOLOv3	Tiny YOLOv4
100	8%	33%
200	25%	50%
400	42%	83%

E. Simulation Testing

The simulation test was aimed to see whether the YOLO algorithm can identify objects properly with the input of road segment video of the Universitas Sriwijaya, Indralaya campus. Simulation testing was carried out using the Tiny YOLOv4 network model with an epoch of 400 and a threshold of 0.5.

The detection results were then sent serially to the microcontroller which is processed into a control signal and calculate how accurate the distance measurement is with the actual distance measurement results using Depth information. The test is carried out by identifying the object in front of the autonomous electric vehicle and send the object location data serially to the microcontroller. It is later processed into a control signal. This test was carried out 10 times. However, if the object is not in the front of the electric vehicle, the object is detected by YOLO but is not sent to the microcontroller. The simulation test results can be seen in Table VI.

TABLE VI. SIMULATION TEST RESULTS

Image Detection	
Actual Distance	3.5 Meters
Object Location and Distance	distance : 4.06942006287816 object on the left
Object Location Accepted	Object On The Left

Based on the results of simulation tests that were carried out 10 times, Tiny YOLOv4 with 400 training epochs obtained a detection accuracy of 80%. The percentage of success of serial communication to the microcontroller is 100%. If the tolerance for measuring distance using depth information was 50 cm, the accuracy of distance measurement was 66.77%.

F. Comparison of YOLO and HSV in Detecting Objects

TABLE VII. COMPARISON RESULTS OF TINY YOLO AND HSV

Model	Figure	Description
Tiny YOLOv 4 400 Epoch		Detected
HSV		Detected

The YOLO algorithm was also compared to the HSV. The results showed that YOLO was able to detect object with an accuracy of 86.66% while HSV only got an accuracy of 33.33% in the experiment that performed 9 times as shown in Table VII.

G. Real-time Testing

Real-time testing was conducted on roads around the Universitas Sriwijaya, Indralaya campus. The path taken by the autonomous electric vehicle is 1.7 km as can be seen in Fig. 7.
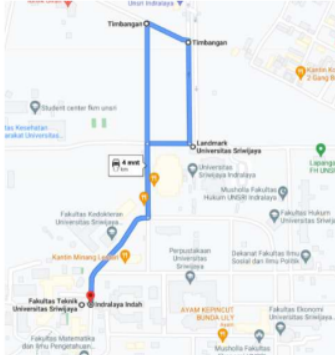
Fig. 7. Map for Real-time Testing

This test aims to examine whether the autonomous electric vehicle can avoid objects in front of it with the input from the detection results of the YOLOv4 Tiny model with an epoch of 400 and a threshold of 0.5. The results in real-time can be seen in Table VIII. Since the autonomous electric vehicle has a 14 km/hour speed, which is below the average speed of a vehicle, the test was only carried out on stationary objects or in the opposite direction to the autonomous electric vehicle.

TABLE VIII.   REAL-TIME TESTING RESULTS

Image detection and Location	Frame Car Movement	Distance(m)
		5
		4.1
		3.7
		-
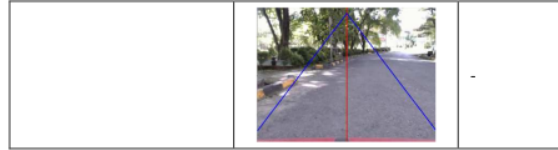
The real-time experiment using the implementation of the YOLO algorithm-based to identify the objects as input to the steering control system has been successfully carried out with an accuracy of 100%.

V. CONCLUSION

Based on the research conducted, the implementation of YOLO algorithm-based object identification has been successfully implemented as an input to the steering control system. The YOLO can detect an object instantly as long as the object is recognized by YOLO. The Tiny YOLOv4 model is superior to the Tiny YOLOv3 model because it has an object detection success rate of 83% and the average loss is the smallest with an average loss of 0.366202. Thus, it becomes the model used in the system. The system can detect all objects in real-time with a success rate of avoiding objects has an accuracy of 100%.

For future work, updated Graphic Processor Unit (GPU) with more Video Random Access Memory is needed to speed up the length of the model training, and more learning data is needed for the training process due to changes conditions that affect the system.

REFERENCES

[1] A. Hidayati and L. Y. Hendrati, "Analisis Risiko Kecelakaan Lalu Lintas Berdasar Pengetahuan, Penggunaan Jalur, dan Kecepatan Berkendara," [Traffic Accident Risk Analysis Based on Knowledge, Line Usage, and Driving Speed] (in Indonesian) J. Berk. Epidemiol., vol. 4, no. 2, pp. 275–287, 2016, doi: 10.20473/jbe.v4i2.2016.275.

[2] Katsuya Matsuzaki, Masuhiro Nitta and Kiyotaka Kato, "Development of an intelligent traffic light for reducing traffic accidents," 2008 International Conference on Control, Automation and Systems, 2008, pp. 443-447, doi: 10.1109/ICCAS.2008.4694681.

[3] M. Taufiqurrahman, S. Sumardi, and M. A. Riyadi, "Perancangan Self Driving Dengan Metode Kontrol PD Pada Sistem Tracking Autonomous Car," [Self Driving Design With PD Control On Autonomous Car Tracking System] (in Indonesian) Transient J. Ilm. Tek. Elektro, vol. 5, no. 2, pp. 173–179, 2016, [Online]. Available: https://ejournal3.undip.ac.id/index.php/transient/article/view/13717.

[4] M. Silva, L. Garrote, F. Moita, M. Martins, and U. Nunes, "Autonomous electric vehicle: Steering and path-following control systems," Proc. Mediterr. Electrotech. Conf. - MELECON, pp. 442–445, 2012, doi: 10.1109/MELCON.2012.6196468.

[5] A. Dairi, F. Harrou, Y. Sun, and M. Senouci, "Obstacle Detection for Intelligent Transportation Systems Using Deep Stacked Autoencoder and k-Nearest Neighbor Scheme," IEEE Sens. J., vol. 18, no. 12, pp. 5122–5132, 2018, doi: 10.1109/JSEN.2018.2831082.

[6] B. Liu, H. Liu, and J. Yuan, "Lane Line Detection based on Mask R-CNN," vol. 87, no. Icmeit, pp. 696–699, 2019.

[7] Z. Wang, "LaneNet: Real-Time Lane Detection Networks for Autonomous Driving."

[8] W. Song, X. Hu, J. Fu, Q. Zhou, T. Zhou and P. Si, "The method of hybrid-laser image spot extracts based on HSV space SVD for power

transmission line detection," *2016 IEEE International Conference on Information and Automation (ICIA)*, 2016, pp. 1361-1364, doi: 10.1109/ICInfA.2016.7832031.

[9]   H. Soebhakti, S. Prayoga, R. A. Fatekha, and M. B. Fashla, "The Real-Time Object Detection System on Mobile Soccer Robot using YOLO v3," Proc. 2019 2nd Int. Conf. Appl. Eng. ICAE 2019, pp. 3–8, 2019, doi: 10.1109/ICAE47758.2019.9221734.

[10] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017, vol. 2017-Janua, pp. 6517–6525, 2017, doi: 10.1109/CVPR.2017.690.

[11] R. Girshick, "Fast R-CNN," Proc. IEEE Int. Conf. Comput. Vis., vol. 2015 Inter, pp. 1440–1448, 2015, doi: 10.1109/ICCV.2015.169.

[12] R. Joseph, S. Divvala, R. Girschick, and F. Ali, "You Only Look Once: Unified, Real-Time Object Detection," J. Chem. Eng. Data, 2016, doi: 10.1021/je00029a022.

[13] P. Pias, "Object Detection and Distance Measurement," 2019, [Online]. Available: https://github.com/paul-pias/Object-Detection-and-Distance-Measurement (accessed Jun. 04, 2021).

[14] A. Bochkovskiy, C. Y. Wang, and H. Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," arXiv, 2020.

[15] A. Sarda, S. Dixit and A. Bhan, "Object Detection for Autonomous Driving using YOLO [You Only Look Once] algorithm," *2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV)*, 2021, pp. 1370-1374, doi: 10.1109/ICICV50876.2021.9388577.

[16] Alexey, "Darknet YOLOv3 & YOLOv4," 2018. https://github.com/AlexeyAB/darknet (accessed Jun. 01, 2021).

irvine

PRIMARY SOURCES

1	stackoverflow.com Internet	209 words —	6%
2	Markus Hermawan, Zaenal Husin, Hera Hikmarika, Suci Dwijayanti, Bhakti Yudho Suprapto. "Road Identification using Convolutional Neural Network on Autonomous Electric Vehicle", 2021 8th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI), 2021 Crossref	138 words —	4%
3	bdm.unb.br Internet	136 words —	4%
4	Ike Bayusari, Albert Mario Alfarino, Hera Hikmarika, Zaenal Husin, Suci Dwijayanti, Bhakti Yudho Suprapto. "Position Control System of Autonomous Underwater Vehicle using PID Controller", 2021 8th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI), 2021 Crossref	58 words —	2%
5	Davide Adami, Mike O. Ojo, Stefano Giordano. "Design, Development and Evaluation of an Intelligent Animal Repelling System for Crop Protection based on Embedded Edge-AI", IEEE Access, 2021 Crossref	55 words —	1%

6   Deni Kristin Manase, Zahir Zainuddin, Syafruddin Syarif, Arsan Kumala Jaya. "Car Detection in Roadside Parking for Smart Parking System Based on Image Processing", 2020 International Seminar on Intelligent Technology and Its Applications (ISITIA), 2020
Crossref   49 words — 1%

7   ijfac.unsri.ac.id
Internet   37 words — 1%

8   Muhammad Yusup, Suci Dwijayanti, Hera Hikmarika, Zaenal Husin, Bhakti Yudho Suprapto. "Gyroscope Control System in Wave Power Plant Using PID Controller", 2021 International Conference on Converging Technology in Electrical and Information Engineering (ICCTEIE), 2021
Crossref   27 words — 1%

9   I Gede Susrama Mas Diyasa, Ni Made Ika Marini Mandenni, Mohammad Idham Fachrurrozi, Sunu Ilham Pradika et al. "Twitter Sentiment Analysis as an Evaluation and Service Base On Python Textblob", IOP Conference Series: Materials Science and Engineering, 2021
Crossref   25 words — 1%

10   A Yudono, Surjono, C Meidiana, I Nurika, E Martati, S Wibowo. "Humanitarian engineering approach for re-designing pedestrian traffic inside the lecture building during the new normal COVID-19 pandemic", IOP Conference Series: Earth and Environmental Science, 2021
Crossref   24 words — 1%

11   www.coursehero.com
Internet   24 words — 1%

12   Sutrisno, Widowati, Redemtus Heru Tjahjana. "Optimization-based Decision-Making Support for Fuzzy and Probabilistic Order Allocation Planning", 2021 8th   17 words — < 1%

International Conference on Electrical Engineering, Computer Science and Informatics (EECSI), 2021
Crossref

13   www.asp.eurasipjournals.com
Internet   16 words — < 1%

14   Eka Nuryanto Budisusila, Bustanul Arifin, Sri   13 words — < 1%
Arttini Dwi Prasetyowati, Bhakti Yudho Suprapto,
Zainuddin Nawawi. "Artificial Neural Network Algorithm for
Autonomous Vehicle Ultrasonic Multi-Sensor System", 2020
10th Electrical Power, Electronics, Communications, Controls
and Informatics Seminar (EECCIS), 2020
Crossref

15   Meiga Isyatan Mardiyah, Tuti Purwaningsih.   13 words — < 1%
"Developing deep learning architecture for image
classification using convolutional neural network (CNN)
algorithm in forest and field images", Science in Information
Technology Letters, 2020
Crossref

16   Matheus Mello Jacques, Joao Brum, Henrique   11 words — < 1%
Martins, Daniel Pinheiro Bernardon et al.
"Application of Neural Network to Locate Non-Technical Losses
in Optical Satellite Images", 2021 IEEE PES Innovative Smart
Grid Technologies - Asia (ISGT Asia), 2021
Crossref

17   Erwin, Heranti Reza Damayanti. "Supervised   10 words — < 1%
Retinal Vessel Segmentation Based Average Filter
and Iterative Self Organizing Data Analysis Technique",
International Journal of Computational Intelligence and
Applications, 2020
Crossref

18   eecsi.org
Internet

10 words — < 1%

19 export.arxiv.org
    Internet

    10 words — < 1%

20 "Artificial Neural Networks in Pattern Recognition", Springer Science and Business Media LLC, 2018
    Crossref

    8 words — < 1%

21 Wenbo Lan, Jianwu Dang, Yangping Wang, Song Wang. "Pedestrian Detection Based on YOLO Network Model", 2018 IEEE International Conference on Mechatronics and Automation (ICMA), 2018
    Crossref

    8 words — < 1%

22 jurnal.umt.ac.id
    Internet

    8 words — < 1%

23 Ge Wang, Pengbo Pu, Tingyan Shen. "An efficient gene bigdata analysis using machine learning algorithms", Multimedia Tools and Applications, 2020
    Crossref

    7 words — < 1%

24 Zhang Yi, Shen Yongliang, Zhang Jun. "An improved tiny-yolov3 pedestrian detection algorithm", Optik, 2019
    Crossref

    7 words — < 1%

EXCLUDE QUOTES   ON   EXCLUDE SOURCES   OFF
EXCLUDE BIBLIOGRAPHY   ON   EXCLUDE MATCHES   OFF