Obstacle Detection Using Monocular Camera with Mask R-CNN Method

Ari Santoso
Department of Electrical Engineering
Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia
santoso@ee.its.ac.id

Rafif Artono Darmawan
Department of Electrical Engineering
Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia
rafifadarmawan@gmail.com

Mohamad Abdul Hady
Department of Electrical Engineering
Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia
hady@bme.its.ac.id

Ali Fatoni
Department of Electrical Engineering
Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia
fatoni@ee.its.ac.id

***Abstract*—An autonomous car is a car that can operate without being controlled by humans. Autonomous cars must be able to detect obstacles so that the car does not hit objects that are on the path to be traversed. Therefore, it takes a variety of sensors to determine the surrounding conditions. The sensors commonly used in autonomous cars are cameras and LiDAR. Compared to LiDAR, the camera has a relatively long detection distance, lower cost, and can be used to classify objects. In this research, the monocular camera and Mask R-CNN algorithm are used to create a system that can detect obstacles in the form of cars, motorcycles, and humans. The system will generate segmentation instances, bounding boxes, classifications, distance, and width estimation for each detected object. Custom dataset that is created manually are used to get tresults that fits the environtment. The system used can produce a Mean Average Precision of 0.81, a Mean Average Recall of 0.89, an F1 score of 0.86, and a Mean Absolute Percentage Error of 13.4% for the distance estimator. The average detection speed of each image is 0.29 seconds.**

Keywords—autonomous car, mask R-CNN, monocular camera, obstacle detection

I. INTRODUCTION

An autonomous car is a car that can operate without the involvement of people behind its wheel[1]. Nowadays, the improvement of autonomous car technology has increased rapidly. This rapid improvement was influenced by technological developments, both from electronic technology, telecommunications, and transportation. Autonomous cars provide many benefits in life. With autonomous cars, road congestion can be reduced, transportation costs are reduced, and can reduce the number of road accidents[2][3]. With the increasing use of autonomous cars, vision zero programs will be more likely to be achieved. Vision Zero is a program that aims to reduce the number of fatal road accidents to zero[4].

Autonomous cars can be either partially or fully autonomous. According to the Society of Automotive Engineers (SAE), the level of automation of autonomous cars is divided into 6. With the first three levels, the driver still controls the car with the assist feature and the last three levels are automatic. For the autonomous cars in level 0, the automatic features are still limited to temporary warnings and assistance, such as blind spot warnings and automatic emergent breaks. Level 1 provides steering or brake assistance, as in line centering assistance and adaptive cruise control, whereas level 2 can do both. At levels 3 and 4 the car can run automatically under limited conditions, where several conditions must be met. And at the last level or level 5 the car can run automatically in any condition.

Autonomous cars do not require human intervention to operate. Therefore, autonomous cars must be able to identify the surrounding environment and calculate the steps that will be taken. Some of the features that exist in autonomous cars are obstacle detection, empty space detection, path planning, tracking, and others. In detecting obstacles several sensors can be used such as LiDAR and cameras. LiDAR works by emitting a focused beam of light and the reflection time detected by the sensor is measured, so that the distance to the object can be calculated. Because it uses light, LiDAR cannot penetrate rain, clouds, and fog[5]. In addition, LiDAR has other weaknesses, including having a short detection distance, having an expensive price and not being able to distinguish between types of objects[6]. In contrast to the use of a camera that can detect objects at a relatively long distance, the use of a camera also has a lower cost. With the use of the camera, image processing and computer vision applications can be carried out, so that object detection can be carried out and classify the types of objects that have been detected.

Detection of objects using the camera has been done a lot. Detection of objects is done by using a neural network. Some of the algorithms used are Mask R-CNN[7], YOLO[8][9], etc. In research [10], the detection of car obstacles was carried out using the faster R-CNN method and the predictions of object classification and 3D bounding boxes of objects were generated with an average detection speed of 0.81 seconds for each image. In research [11], 5 object types were detected using the YOLOv4 method and obtained object classification and object bounding boxes with a processing time of 0.05 seconds for image input and 0.03 for video input. In research [12], traffic signs were detected using the mask R-CNN method. By using the Mask R-CNN method, prediction results are obtained in the form of bounding boxes, classifications, and segmentation instances of objects. In addition, the mask R-CNN also has an average detection speed of 0.38 seconds for each frame.

II. METHODS

A. Dataset

The dataset used for training in the Mask R-CNN algorithm consists of two input data, namely training data and

How to cite: Santoso, A., Darmawan, R. A., Hady, M. A., & Fatoni, A. (2022). Obstacle Detection Using Monocular Camera with Mask R-CNN Method. *JAREE (Journal on Advanced Research in Electrical Engineering), 6(2).*

Fig. 1. Training Data Sample.

TABLE I. COMPARISON OF THE AMOUNT OF TRAINING DATA AND VALIDATION DATA

Train/Val	Total Data	
	Pictures	Annotations
Training	594	2660
Validation	291	1460

validation data. The input data is in the form of an image as shown in Fig. 1 and annotations in the form of the coordinates of the segmentation of the object. In the picture and annotations in the form of the segmentation coordinates of the object. Annotations are given in the image so that when doing the training model you can find out the location of the object and for validation when the validation process occurs.

This custom dataset is created manually. The pictures were taken manually using a monocular camera. Each image is given annotations manually. Annotations were given using the VGG Image Annotator (VIA) program [13]. Fig. 1 shows an example of annotations using VIA.

This dataset consists of three classes, namely cars, motorcycles, and people. The three classes are three objects that are often encountered on the road. The use of custom datasets is due to using custom datasets, the objects that are better suited to the surrounding environment, and only using object data that is only needed. The comparison of the amount of training and validation data can be seen in Table I.

B. Mask R-CNN Architecture

Broadly speaking, Mask R-CNN architecture has two main parts. The first part is the backbone network and head network. The backbone network consists of a Residual Neural Network[14] with Feature Pyramid Network[15] (ResNet 101- FPN) and Region Proposal Network (RPN) and the head network consists of a Fully Connected layer and Fully Convolutional Network.

ResNet-101 FPN functions to detect or extract features in the image, which is then continued by the RPN process, where the feature map area that has the potential to have objects will be identified to produce the proposed region. The

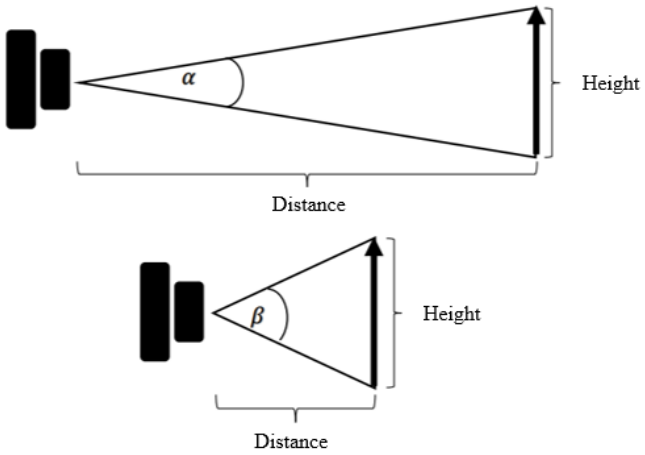

Fig. 2. Effect of Distance on FOV Angle.

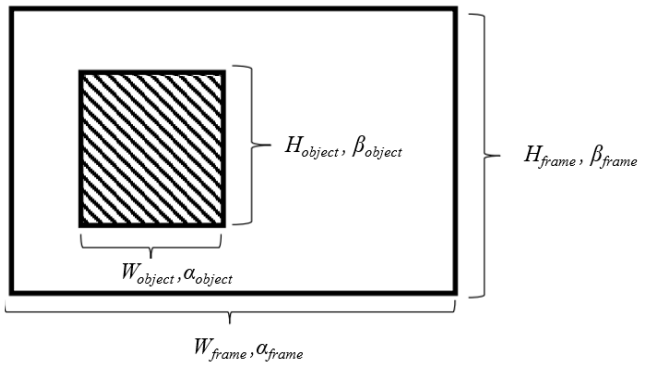

Fig. 1. Representation of Comparison of Dimensions and FOV of Objects to Frames from Cameras.

head network has two branches, namely the Fully Connected Layer (FCL) which has a function to classify and predict bounding boxes from the region that has been proposed by RPN, and the Fully Convolutional Network to create segmentation instances from objects. Thus, the Mask R-CNN algorithm will process the input image and resulting in object prediction, bounding box, and instance segmentation.

C. Object Distance and Width Estimation

Distance estimation requires some data, including the Field of View of the camera used, object size reference, and bounding box data. The distance that is estimated is the distance from the camera to the detected object. Estimating the object distance can be done by comparing the size and angle of the FOV relative to the object. In Fig. 2 it can be seen that an object that has the same height has a different relative

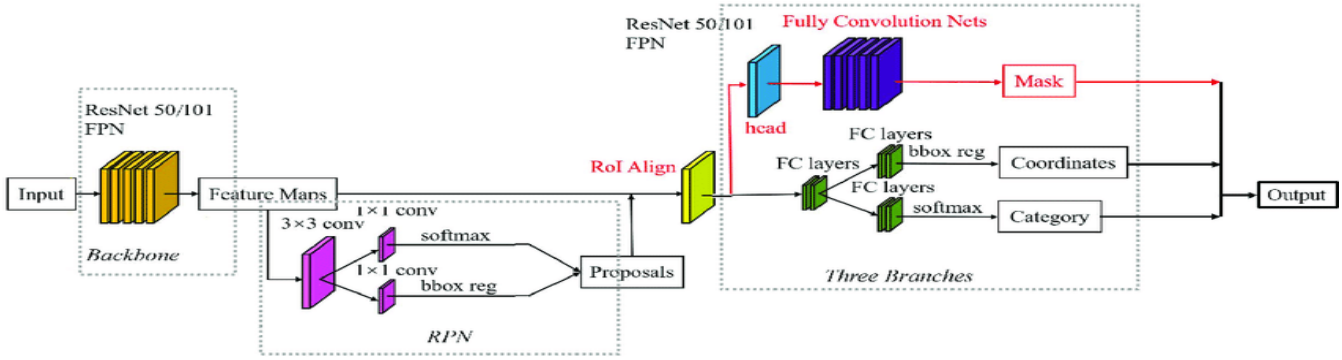

Fig. 2. Mask R-CNN Architecture.

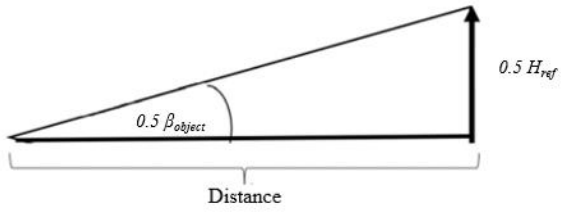

Fig. 4. Use of trigonometric rules to obtain the estimated distance between the camera and the object.

angle, depending on the distance from the object to the camera.

The representation of the object against the frame can be seen in Fig. 3, where the length and height of the object are directly proportional to the relative FOV of the object. By using the variable H (Height) the relative FOV angle can be calculated by equations (1) and (2) where H_{object} is the height of the bounding box.

$$\frac{H_{object}}{H_{frame}} = \frac{\beta_{object}}{\beta_{frame}} \quad (1)$$

$$\beta_{object} = \frac{H_{object} \times \beta_{Frame}}{H_{frame}} \quad (2)$$

From the angle obtained, an isosceles triangle is generated, which can be divided into right triangles, as shown in Fig 5. The distance of the object to the camera can be calculated using a trigonometric equation as in equation (3) with H_{ref} as the object's reference height or the actual object's height.

$$Distance = \frac{0.5W_{ref}}{\tan(0.5\beta_{object})} \quad (3)$$

The estimated width of the detected object is calculated by comparing the reference size and the size of the bounding box. In equation (4) the reference used is the height of the object because unlike the length and width of the object, it changes more when compared to the height of the object which has a relatively constant value.

$$W_{object} = \frac{H_{ref} \times W_{object}}{H_{object}} \quad (4)$$

III. System Testing

A. Training

The system training process is carried out using a cloud computing platform from Google, with the name Google Collaboratory Platform. To do the training, the Google Colab Pro plan was used. Google colab pro is very capable of doing heavy machine learning training. GPU specifications from google colab pro can be seen in the Table II. The training

TABLE II. Specification of the used GPU

Parameter	Specification
GPU Architecture	NVIDIA Pascal
Cores	3584
Memory Size	16 GB
Memory Type	HBM2
Bandwidth	732 GB/s

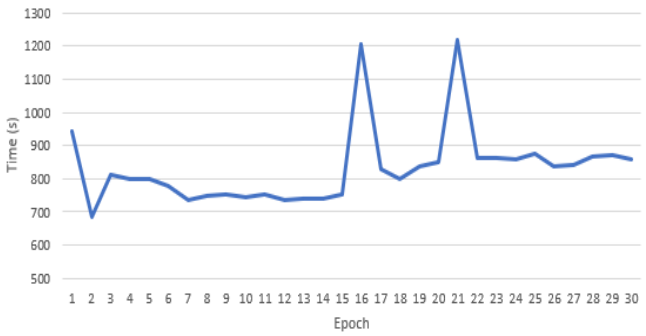

Fig. 5. Graph of Training Duration for each Epoch.

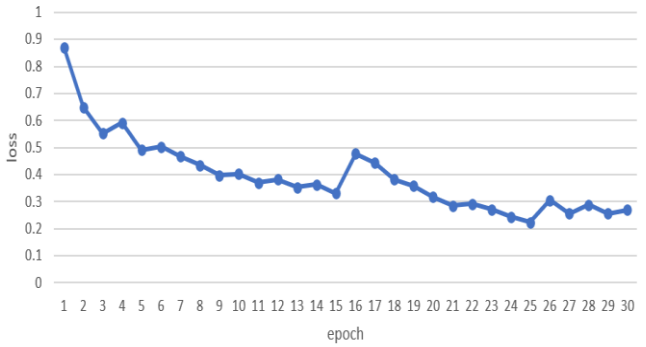

Fig. 3. Training Loss Chart for each Epoch.

model is carried out for 30 epochs and is divided into three stages, where each stage has a different number of epochs. The first stage is done to do training heads and is done for 15 epochs. The second stage is the ResNet stage 4 training layer until the end, including the heads, do 5 epochs. The third is fine-tuning all layers and 10 epochs are done. Every time you finish an epoch, one weight will be generated and then save it to Google Drive. Fig. 7 shows the results of the training loss for each epoch, where epochs 1-15 are the first stage, epochs 16-20 are the second stage, and epochs 21 to 30 are the third stage.

From Fig. 6, it can be seen that the training process in each of the first epochs of each stage has a higher time than in other epochs. The 16th epoch is the beginning of the second stage, and the 21st epoch is the beginning of the third stage. In the second epoch, the detection time decreases and then increases again before decreasing in epoch 5.

B. Validation

Validation is carried out to determine the performance of the model that has been trained. Validation is done by testing the model to the validation data that has been prepared. Validation data contains data that is different from training

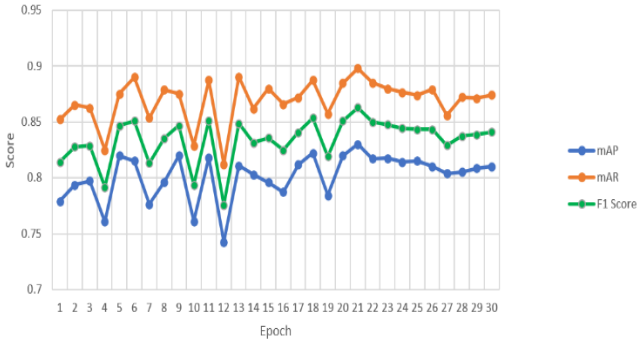

Fig. 6 Graph of Evaluation Results for each Epoch.

data. There are three parameters used to evaluate the model, namely mean average precision (mAP), mean average recall (mAR), and f1 scores. Where mAP is the average of Average precision which states how precise a model is to make predictions on data with positive categories and mAR is the average of Average Recall which states how well the model can predict positive data from all categories. While the F1 score is a calculation between mAP and mAR. The three parameters have values between 0 to 1, where 1 is a very good value and 0 is a very bad value. The selection of the best epoch can be determined by looking at the F1 score.

Fig. 8 shows that the mAP score is in the range of 0.7 and 0.85, with the highest score is 0.83 at the 21st Epoch with the lowest score is 0.74 at the 12th Epoch. The mAR score is in the range of 0.8 and 0.9 with the highest score being 0.89 at the 21st Epoch and the lowest being 0.81 at the 12th Epoch. The F1 score is in between the mAR and mAP scores in the range of 0.75 and 0.9 with the highest score of 0.86 at the 21st epoch and the lowest score of 0.77 at the epoch 12th

C. Object distance and width estimation

The demonstration of distance and width estimation is done by estimating objects that have been detected. To perform the test, used images were taken using a camera at a distance of 1 to 30 meters from the object at 1-meter intervals. Testing the estimated distance and width of the object, the results are shown in Table IV. and the width of the object is 2 meters.

Evaluation can be calculated using the Mean Absolute Percentage Error (MAPE) method in equation (5) where x is the actual value (actual distance) and y is the predicted value (the estimated distance).

$$MAPE = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{x_i - y_i}{x_i}\right| \quad (5)$$

TABLE III. CAMERA SPECIFICATION FOR OBJECT DISTANCE AND WIDTH ESTIMATION

Parameter	Spesification
Frame Height	1280 Pixels
Frame Width	720 Pixels
Vertikal FOV	50°
Horizontal FOV	77°

From the test, parameters are obtained in the form of estimated distance, error, and estimated width of the object. In distance estimation, the largest error is obtained at a distance of 28 meters, amounting to 3.23 meters, with the smallest error obtained at a distance of 7 meters, with an error of 0.05 meters. The obtained MAPE of distance estimation is 13.2% and for the width estimation with an average estimated width of 2.093, the MAPE is 16.4%.

TABLE IV. DISTANCE AND WIDTH ESTIMATION RESULT

True Distance (m)	Estimated Distance (m)	Estimated Width (m)
1	2.58	2.95
2	2.52	2.1
3	3.52	2.13
4	4.43	2.09
5	5.3	2.08
6	6.3	2.11
7	7.05	2.08
8	7.81	2.05
9	8.6	2.04
10	9.4	2.02
11	10.4	2.05
12	11.6	2.06
13	12.52	1.99
14	12.84	2.03
15	13.92	2.02
16	14.74	2.15
17	15.67	2.04
18	16.53	2.08
19	17.3	2.03
20	18.59	2.05
21	19.07	2.09
22	20.1	2.14
23	20.94	2.08
24	21.55	2.08
25	22.87	1.96
26	23.98	2.14
27	24.77	2
28	24.77	2.05
29	26.54	2.08
30	27.02	2.02

D. Demonstration

The data used for the demonstration process is data that is different from the dataset. There are two types of data used as

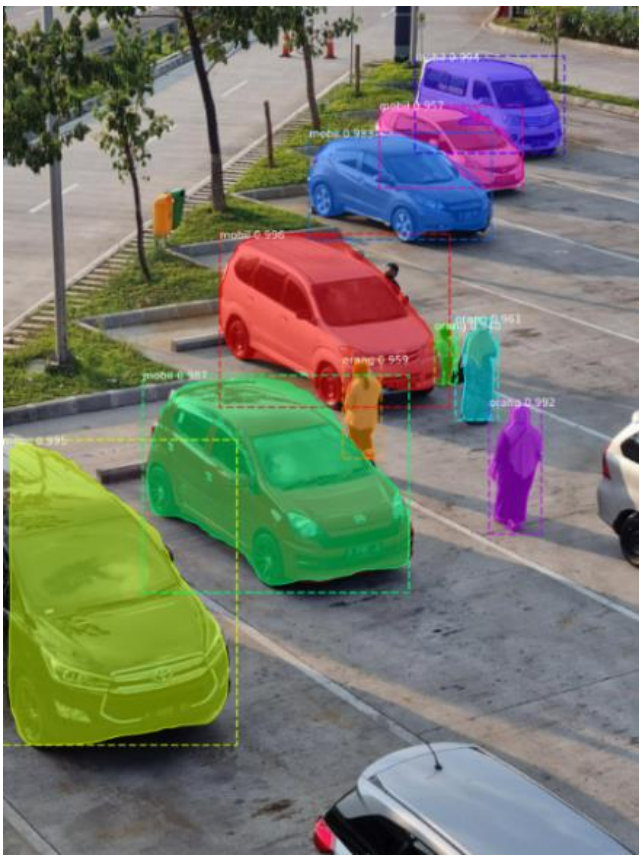

Fig. 97 Demonstration of photo detection results.


Fig. 10 Demonstration video detection result.

TABLE V. DETECTION TIME

Input	Processing Time (second)		
	Average	Fastest	Slowest
A Picture	0.286924	0.27964	0.30534
Pictures	0.290395	0.27905	0.30566
Video	0.290181	0.2840	0.33475

input data, namely photos and videos. The results will be displayed in the form of photos and snippets of images from the video. Testing the Mask R-CNN algorithm for photo produces four outputs, namely bounding box, segmentation instance, class, and confidence level of the detected object and for video it also produce the estimation of distance and width of detected object. Each photo will be processed by the model to get object detection in each photo. The results of the demonstration are shown in Fig. 9, for the demonstration on photo and the results of the video detection demonstration are shown in Fig. 10.

E. Detection rate

The detection speed test is carried out using the hardware used in the training process with the specifications in Table V. This test is carried out with three different inputs, the speed of detecting images with single objects, images with multiple objects, and video. Pictures and videos are taken using a smartphone camera. The average detection speed is 0.29 which is equivalent to 3.45 frames per second (fps).

IV. RESULT ANALYSIS

In the training process, the value of the training loss decreases constantly. However, at epochs 1, 16, and 21 the training loss is very high. This is because there is a change in the training stage, so there is an increase in loss before it decreases again. The best epoch is the 21st epoch because it has the highest score for all the validation parameters, with 0.81 for mAP, 0.89 for mAR, and 0.86 for F1 score. The mAP, mAR, and f1 scores obtained show good values because the dataset used was created manually. To detect video, there is no big difference compared to photo detection, where in photo detection, the detection speed value is 0.2869 seconds and object detection in the video is 0.2901 seconds per frame.

V. CONCLUSION

After conducting a series of tests on the Mask R-CNN algorithm, several conclusions can be drawn:

1. In certain frames, there are some false positive. Some of the factors can be in the form of similarity of features from one class to another.
2. Estimation are influenced by the size of the bounding box, where the bounding box can be larger than the object size, thereby reducing the accuracy of estimating the distance and width of the object.
3. The detection speed of the algorithm is in the range of 0.29 s, equivalent to 3.45 fps. The detection speed is influenced by the hardware specifications used to perform the detection and the frame size of the detected input.

There are rooms for improvement in this research. For future work, the writer suggests to consider the following things:

1. Addition of detected object class.

   Currently, the algorithm is only trained to detect three detected objects, cars, motorcycles, and people, so it still cannot detect other objects. The addition of detection objects can be in the form of traffic signs, potholes in the road, speed bumps, and so on.

2. Sensor integration

   The focus of the current algorithm is to detect objects so that the position of the sensor with the detected object cannot be carried out. So the addition and integration with other sensors such as RADAR, LiDAR, or ultrasonic can be used to determine the location of objects.

3. Dataset

   Although a good evaluation value has been obtained, with an mAP of 0.89, the test and training data were made under normal conditions with sunny weather and good lighting. So it can be added data and tested with different conditions.

4. Addition of detected parameters

   Currently, the algorithm can only generate detection in the form of a bounding box, class, segmentation instance, object distance from the camera, and object width. So that in the future it can be added to produce position, speed, and dimension parameters in the form of length and width.

ACKNOWLEDGMENT

This research was carried out thanks to the support of the Electrical Engineering Department of Institut Teknologi Sepuluh Nopember.

REFERENCES

[1] K. Jo, J. Kim, D. Kim, C. Jang, and M. Sunwoo, "Development of autonomous car-part i: Distributed system architecture and development process," *IEEE Trans. Ind. Electron.*, vol. 61, no. 12, pp. 7131–7140, 2014, doi: 10.1109/TIE.2014.2321342.

[2] S. Pettigrew, "Why public health should embrace the autonomous car," *Aust. N. Z. J. Public Health*, vol. 41, no. 1, pp. 5–7, 2017, doi: 10.1111/1753-6405.12588.

[3] C. P. H. Ernst and P. Reinelt, "Autonomous car acceptance: Safety vs. Personal driving enjoyment," *AMCIS 2017 - Am. Conf. Inf. Syst. A Tradit. Innov.*, vol. 2017-August, pp. 1–8, 2017.

[4] E. Kim, P. Muennig, and Z. Rosen, "Vision zero: a toolkit for road safety in the modern era," *Inj. Epidemiol.*, vol. 4, no. 1, pp. 1–9, 2017, doi: 10.1186/s40621-016-0098-z.

[5] N. O. and A. A. N. C. S. Center, "Lidar 101 : An Introduction to Lidar Technology , Data , and Applications," *NOAA Coast. Serv. Cent.*, no. November, p. 76, 2012.

[6] G. Sharabok, "Why Tesla Won't Use LIDAR. And which technology is ideal for… | by German Sharabok | Towards Data Science," *Towards Data Science*, 2020. https://towardsdatascience.com/why-tesla-wont-use-lidar-57c325ae2ed5 (accessed Jun. 30, 2021).

[7] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask R-CNN," *Proc. IEEE Int. Conf. Comput. Vis.*, vol. 2017-Octob, pp. 2980–2988, 2017, doi: 10.1109/ICCV.2017.322.

[8] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," 2020, [Online]. Available: http://arxiv.org/abs/2004.10934.

[9] E. Prasetyo, N. Suciati, and C. Fatichah, "A Comparison of YOLO and Mask R-CNN for Segmenting Head and Tail of Fish," *ICICoS 2020 - Proceeding 4th Int. Conf. Informatics Comput. Sci.*, pp. 2–7, 2020, doi: 10.1109/ICICoS51170.2020.9299024.

[10] A. Firmansyah, A. A. Firmansyah, R. E. AK, and A. Santoso, "Deteksi Halangan Menggunakan Metode Stereo R-CNN pada Mobil Otonom," *J. Tek. ITS*, vol. 9, no. 2, pp. E160–E166, 2021, [Online]. Available: http://ejurnal.its.ac.id/index.php/teknik/article/view/53687%0Ahttps://ejurnal.its.ac.id.

[11] A. O. Kurniawan *et al.*, "Pendeteksi Objek untuk Kendaraan Otonom Menggunakan Single Kamera Berbasis YOLOv4," vol. 1, no. 1, pp. 1–6, 2021.

[12] K. N. Banjarnahor, R. Effendi, and Y. Bilfaqih, "Sistem Pengenalan Dan Estimasi Jarak Rambu Lalu Lintas Berbasis Mask R-CNN Dan Kamera Monokuler Untuk Kendaraan Otonom," vol. 1, no. 1, pp. 1–7, 2021.

[13] A. Dutta and A. Zisserman, "The VIA annotation software for images, audio and video," *MM 2019 - Proc. 27th ACM Int. Conf. Multimed.*, pp. 2276–2279, 2019, doi: 10.1145/3343031.3350535.

[14] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2016-Decem, pp. 770–778, 2016, doi: 10.1109/CVPR.2016.90.

[15] X. Li, T. Lai, S. Wang, Q. Chen, C. Yang, and R. Chen, "Weighted feature pyramid networks for object detection," *Proc. - 2019 IEEE Intl Conf Parallel Distrib. Process. with Appl. Big Data Cloud Comput. Sustain. Comput. Commun. Soc. Comput. Networking, ISPA/BDCloud/SustainCom/SocialCom 2019*, pp. 1500–1504, 2019, doi: 10.1109/ISPA-BDCloud-SustainCom-SocialCom48970.2019.00217.