

# Designing an Autonomous Vehicle Using Sensor Fusion Based on Path Planning and Deep Learning Algorithms

Bhakti Y. Suprpto, Member IEEE, Suci Dwijayanti, Member IEEE, Dimsyiar M.A. Hafiz, Farhan A. Ardandy and Javen Jonathan

**Abstract**— Autonomous electric vehicles use camera sensors for vision-based steering control and detecting both roads and objects. In this study, road and object detection are combined, utilizing the YOLOv8x-seg model trained for 200 epochs, achieving the lowest segmentation loss at 0.53182. Simulation tests demonstrate accurate road and object detection, effective object distance measurement, and real-time road identification for steering control, successfully keeping the vehicle on track with an average object distance measurement error of 2.245 m. Route planning for autonomous vehicles is crucial, and the A-Star algorithm is employed to find the optimal route. In real-time tests, when an obstacle is placed between nodes 6 and 7, the A-Star algorithm can reroute from the original path (5, 6, 7, 27, and 28) to a new path (5, 6, 9, 27, and 28). This study demonstrates the vital role of sensor fusion in autonomous vehicles by integrating various sensors. This study focuses on sensor fusion for object-road detection and path planning using the A\* algorithm. Real-time tests in two different scenarios demonstrate the successful integration of sensor fusion, enabling the vehicle to follow planned routes. However, some route nodes remain unreachable, requiring occasional driver intervention. These results demonstrate the feasibility of sensor fusion with diverse tasks in third-level autonomous vehicles.

**Index Terms**—A-star algorithm, object detection, path planning, road detection, sensor fusion, YOLOv8

## I. INTRODUCTION

THE convenience of autonomous vehicles for users originates from the automated features in vehicles. Autonomous vehicles have six levels of automation, ranging from no automation to full automation [1]. At the lowest level, known as "no automation," the driving tasks in the autonomous vehicle are entirely performed by the driver. Moving up to the first level, called "driver assistance," the steering and acceleration/deceleration systems start assisting the driver. The second level, "partial automation," involves the steering and acceleration/deceleration being fully controlled by the system. However, the driver remains responsible for recognizing the surroundings of the vehicle. In the third level, known as "conditional automation," steering, acceleration/deceleration, and environmental recognition are

entirely managed by the system. However, the driver's role is necessary to take over in case the self-driving system fails. The distinctions between the third and subsequent levels lie in the drivers' roles. In the third level, the driver is required to remain alert, whereas in the fourth and fifth levels, the driver's role is no longer required because the system fully controls the autonomous vehicle.

In this study, the autonomous vehicle developed is at the third level of conditional automation. In other words, the driving tasks and environmental recognition are performed by the system; however, driver vigilance is still necessary in case the self-driving feature fails. Research on the mechanisms of steering, acceleration, and braking in the autonomous mode has been conducted by several scholars, including the use of wormed-gear DC motors [2] and planetary gear DC motors [3]. Various methods for executing a steering system have been explored, including fuzzy logic [4][5], proportional-integral-derivative (PID) [6], a combination of fuzzy and PID, model predictive control (MPC), and adaptive control [7].

Recognizing the surrounding environment is another essential aspect that must be addressed to fulfill the requirements for entering level three of the taxonomy. Various sensors can be used for environmental recognition, including cameras, radar, LiDAR, global positioning system (GPS), and inertial measurement units (IMUs) [8]. A camera is a crucial sensor for autonomous electric vehicles, particularly for detecting objects around a vehicle [9]. The information obtained by the camera can then be used to predict the steering direction of the vehicle. Predicting the steering direction using a camera is a vision-based method [10].

Steering direction prediction for autonomous electric vehicles using vision-based methods can be accomplished through two approaches: computer vision-and imitation learning-based approaches [10]. A computer vision-based approach involves using road images from a dataset or capturing images from a camera installed in a vehicle. Subsequently, the images were preprocessed to enhance the accuracy of the relevant road features. Computer vision

This study was funded through the Directorate of Research, Technology, and Community Service Directorate General of Higher Education, Research, and Technology According to the Research Contract Number: 0192.09/UN9.3.1/PL/2023.

Bhakti Y. Suprpto is with the Electrical Engineering Department, Universitas Sriwijaya, 30662 Indonesia (e-mail: bhakti@ft.unsri.ac.id).

Suci Dwijayanti is with the Electrical Engineering Department, Universitas Sriwijaya, 30662 Indonesia (e-mail: sucidwijayanti@ft.unsri.ac.id).

Dimsyiar M. A. Hafiz is with the Electrical Engineering Department, Universitas Sriwijaya, 30662 Indonesia (e-mail: dimsyar417@gmail.com).

Farhan A. Ardandy is with the Electrical Engineering Department, Universitas Sriwijaya, 30662 Indonesia (e-mail: farhanabie14@gmail.com).

Javen Jonathan is with the Electrical Engineering Department, Universitas Sriwijaya, 30662 Indonesia (e-mail: javenjonathan99@gmail.com)

approaches aimed at predicting the steering direction of a vehicle using various methods have been extensively discussed. Tiago Almeida detected roads using deep learning [9]. Meanwhile, [11] and [12] detected unstructured roads using hue, saturation, value (HSV). Thiago Rateke employed a U-NET architecture with an ResNet34-based convolutional neural network (CNN) to detect and differentiate road surfaces while considering surface damage [13]. In another study, Marya Rasib performed semantic segmentation using a CNN with the deeplabv3+ architecture [14]. The second approach entails imitation learning, in which the machine learns the steering angles to navigate various scenarios through human driving demonstrations. In this case, images and steering angles are simultaneously captured while the vehicle is driven, serving as training data for an artificial neural network (ANN) model. Tianhao Wu used convolutional long short-term memory (Conv-LSTM) and Multi-scale Spatiotemporal Integration [15]. In other studies, neural networks [16], deep learning [17], and CNN [18][19], have been used. Both these approaches can be applied in autonomous electric vehicles, but the research methods using these approaches have primarily focused solely on road detection. In their implementation, autonomous electric vehicles must be capable of simultaneously recognizing roads and objects to predict the steering direction. Both aspects need to be considered to ensure that the vehicle stays within its lane and avoids surrounding obstacles.

Another critical aspect of self-driving cars is the time required to reach their destination [20]. Planning efficient movement requires careful consideration of the route the vehicle will take [21]. Route determination can also be beneficial for optimizing battery usage. In this regard, the real-time performance, algorithms, and mapping of relevant locations must be considered. In route determination, planning the path that a vehicle will traverse is necessary [22]. Various route selection methods exist, including genetic [22], Dijkstra, and A-Star [23] algorithms. The A-Star algorithm has been employed to determine the nearest route [24]. The algorithm was designed to compute the shortest distance from the target [25]. The A-Star algorithm is one of the most flexible methods [26]. It incorporates heuristic information into route searching, enabling it to evaluate the search outcomes and expedite problem solving [27]. When calculating the A-Star algorithm for route determination, it selects the smallest value from a heuristic distance [28].

Both are essential considerations in prior research on route determination and vehicle positioning. However, none of these studies have been implemented specifically for autonomous electric vehicles. Some studies implemented only the best route search method through simulation processes [20][22][23][26][29].

In autonomous vehicles, each sensor has its strengths and weaknesses. Therefore, multiple types of sensors are commonly used in autonomous vehicles to compensate for the shortcomings of individual sensors and improve accuracy. Previous research has discussed various sensor combinations, such as camera-LiDAR [30], GPS-IMU [31], radar-camera [32], and LiDAR-IMU-encoder [33]. However, in the

mentioned studies, sensor combinations were aimed at performing specific tasks, such as using cameras and lidar to detect roads [30] or radar and cameras to detect objects on the road [32].

Although various types of sensors have been combined, their use remains focused on a single function such as object or road detection. Ideally, autonomous vehicles should be able to integrate all tasks, including road and object detection, navigation, and other related tasks, by merging all sensors into a unified system. The process is known as sensor fusion and is crucial for simplifying the execution of tasks in autonomous vehicles. In this study, both detection methods are combined. For road and object detection, a computer vision approach based on the You Only Look Once (YOLO) algorithm, which is an extension of the CNN, is employed. The aim is to analyze roads without clear lines or distinct boundaries, such as the unstructured roads commonly found in Indonesia. Subsequently, real-time object detection is performed, enabling the vehicle to avoid objects around it and measure the distance between obstacles and the vehicle. Therefore, this study combines various sensors to perform multiple tasks in an autonomous vehicle, including path planning for localization and object-road detection for navigation. Additionally, an optimal route search method is developed using the A-Star algorithm for autonomous electric vehicles. The A-Star method can also be used to avoid obstacles along a path. Several factors must be considered in route searching, including the position of the vehicle during travel. GPS is employed to determine the position of the vehicle and target destination [34]. The contributions of this study are as follows:

- 1) Object-road detection implemented for unstructured roads commonly found in rural areas of Indonesia using YOLOv8.
- 2) Path planning implemented using the A-Star algorithm in real time, including rerouting for paths with obstacles.
- 3) Sensor fusion is implemented to develop a level three autonomous vehicle.

The remainder of this paper is organized as follows. Section II describes the materials and methods used in the study. The results and discussion are presented in Section III. Finally, section IV concludes the study.

## II. RELATED WORK

Various methods have been implemented for road and object detection. In [35], an RoIC-CNN was proposed to detect various road features, including intersections, left separation, right separation, crosswalks, and normal road segments. The accuracy of the proposed CNN surpasses that of VGGNet-5, LeNet, and AlexNet. Nevertheless, this method needs to be tested in actual highway environments. Meanwhile, [36] introduced a new segmentation model called a Segmentation Transformer (SETR). The method was tested on various datasets, and the results demonstrate its ability to eliminate the reliance on a fully connected network (FCN). In [37], a segmentation approach was employed. The proposed method has three components: an image encoder, flexible prompt encoder, and fast mask decoder. As in previous studies, this was tested using a dataset to assess its robustness. In [38], a

segmentation approach, specifically dynamic road region extraction, was employed using a Gaussian Mixture Model and Expectation Maximization algorithm. Subsequently, they computed the steering angle using the extracted road region.

Notably, previous methods rely on data segmentation for road detection, which is separate from object detection. In contrast to previous studies, this study combines road and object detection using YOLO. YOLO was chosen owing to its capability to perform both object and road detection, and its performance has been demonstrated in various fields [39] [40]. In addition, this study was conducted in real time.

### III. METHOD

A flowchart of the system used in this study is shown in Fig. 1. The system reads GPS data to obtain the coordinates of the autonomous vehicle. Subsequently, these values are used as inputs for the path-planning process. Once path planning determines the optimal route and generates movement commands, such as forward, left, right, and stop, the autonomous vehicle commences its movement toward the destination.

During the journey toward the destination, the autonomous vehicle simultaneously receives data from both the camera and GPS. The camera provides video data, which are then used as input for the object-road detection process using the YOLO algorithm. The output of this process is a steering angle setpoint that must be followed by the steering wheel. This value is adjusted using an encoder that reads the steering angle of the steering wheel. The other output is the distance to the objects. If this distance exceeds 2 m, the rear-wheel drive motor continues to move the wheels.

Furthermore, when the obtained GPS coordinates match the destination coordinates, the autonomous vehicle system stops completely.

#### A. Path planning system design

The path planning system in this study was designed by following the flowchart shown in Fig. 2. The path planning system begins with the microcontroller slave reading the GPS data to obtain the current coordinate values that represent the starting point position of the autonomous vehicle. These values are then sent to the main controller for route searching. The destination coordinates and A-Star algorithm are located within the main controller. Once the A-Star algorithm finds an optimal route, the program determines the movement commands for the autonomous vehicle, such as forward, right, left, and stop. Subsequently, the autonomous vehicle begins to move toward the destination.

During the journey to the destination, the autonomous vehicle continuously reads GPS data until the coordinates read from the GPS match the destination coordinates.

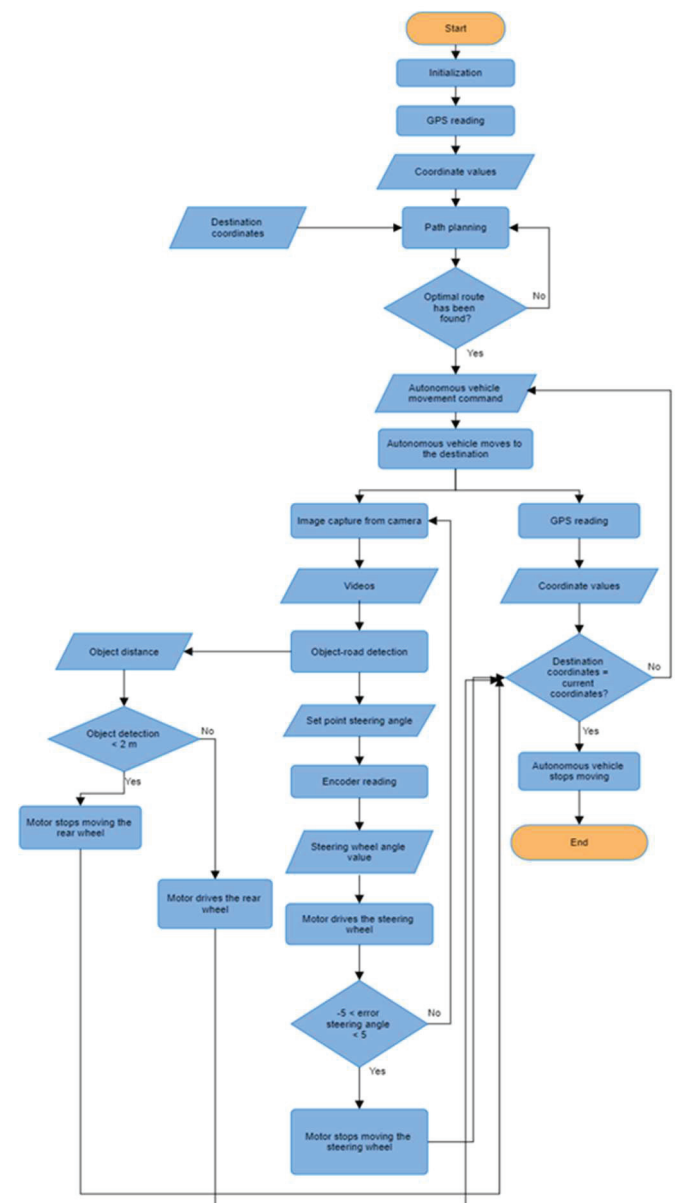


Fig. 1. Autonomous vehicle working system flowchart

#### B. Object and road detection

The object-road detection system designed in this setup follows the flowchart depicted in Fig. 3. The object-road detection system commences with the model training process. The dataset is obtained by video capturing using a camera. Once the model is trained, it is deployed in an autonomous vehicle. The vehicle captures video footage from the camera, segregating objects from the road. Objects are used for object detection, while the road is used for road detection. The objects and road are recognized using the YOLOv8 algorithm. The output of the road detection process is the steering angle setpoint. This set-point value is sent to a microcontroller, and the motor responsible for the steering wheel movement continues to operate until the encoder reading the steering angle matches the predetermined value.

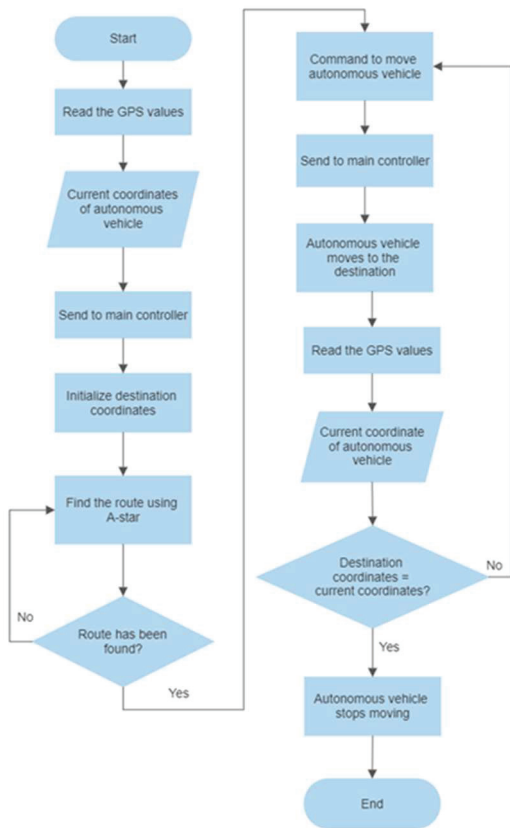


Fig. 2. Path planning flowchart

Conversely, the output of object detection is the distance from the object. If the distance to the object is less than 2 m, the motor responsible for the rear wheel movement ceases, resulting in the autonomous vehicle halting.

C. Hardware and software design

Sensors, such as encoders, are first accessed by a microcontroller. Subsequently, the data from these sensors are sent to the main controller via serial communication using the Robot Operating System (ROS) framework. GPS can also be directly accessed using the ROS. The webcam is accessed directly through the main controller. The programs within the main controller are developed using the Visual Studio Code (VS Code) software, while the embedded programs within the slave microcontroller are created using freely available software, namely the Arduino Integrated Development Environment (IDE). The STM32CubeIDE software was employed as the master microcontroller. In summary, the software design used in this study aligns with the diagram shown in Fig. 4.

In this phase, the design of hardware components that will be utilized in an autonomous vehicle is undertaken. This design encompasses the selection of hardware types, their placement, and layout.

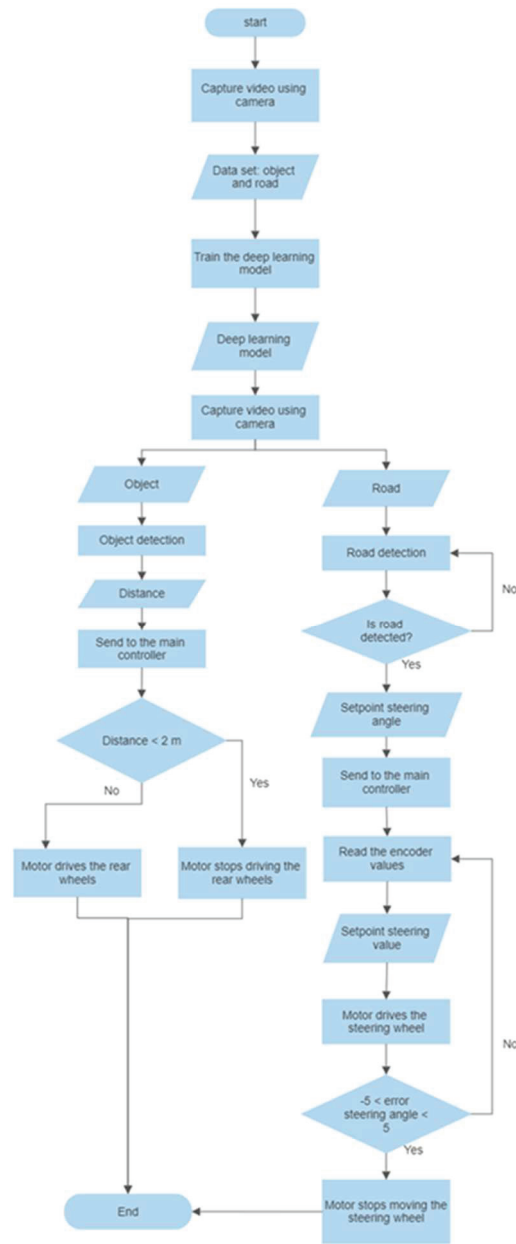


Fig. 3. Object-road detection flowchart

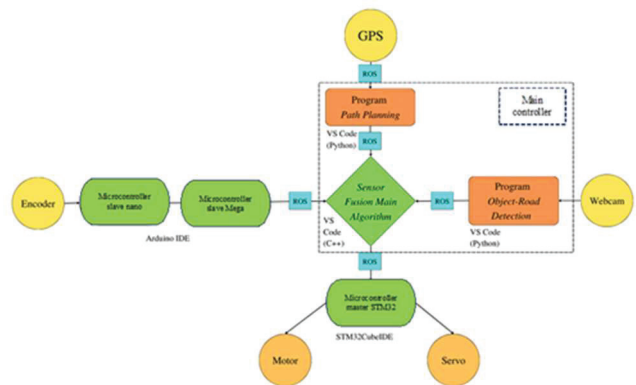


Fig. 4. Software design

#### D. Evaluation

Tests were conducted to assess the performance of the previously designed system. The tests involved integrating all developed programs and evaluating the core algorithm of sensor fusion. System testing was conducted in real time at the campus of Universitas Sriwijaya. Fig. 5 shows a map of the furthest route tested using the system. The determination of whether a model's performance is good can be observed through its performance measurement parameter, namely accuracy level. To calculate the accuracy level, an evaluation method for confusion metrics is employed, as listed in Table I.

TABLE I  
CONFUSION METRIC

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

As indicated in the table, the confusion matrix method has two variables: predicted values and actual values. The predicted values were the outcomes predicted by the YOLO model, while the actual values are the predetermined target values. Based on the confusion matrix table above, the accuracy of the model can be determined using the following equation [41]:

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

The precision and the recall can be calculated as

$$\text{Precision} = \frac{TP}{TP + FP} \quad (2)$$

$$\text{Recall} = \frac{TP}{TP + FN} \quad (3)$$

The mean average precision (mAP) can be calculated using the precision and recall values.

$$\text{mAP} = \frac{\sum_{q=1}^Q \text{AveP}(q)}{Q}, \quad (4)$$

where  $Q$  indicates the number of classes and  $\text{AveP}(q)$  represents the average precision for a given class  $q$ .

Furthermore, in this real-time testing, the model used in the simulation testing was validated against real-world conditions, specifically on roads within the Universitas Sriwijaya campus with predetermined routes, as illustrated in Fig. 5.

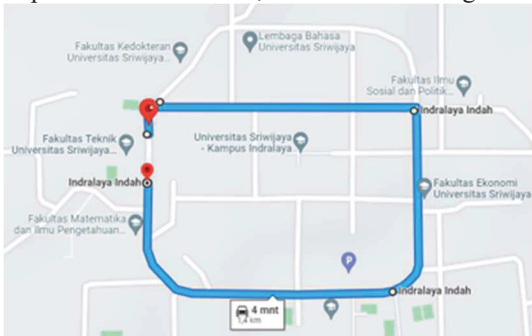


Fig. 5. Route for evaluation

In the first phase, route testing was conducted through simulations using longitudinal and latitudinal data obtained

from Google Maps. Upon acquiring the longitude and latitude data, the heuristic was tested from the current node to the destination node. Subsequently, real-time testing of the heuristic was conducted, and the final step involved testing the A-Star algorithm.

In this phase, following the collection of longitude and latitude coordinates for the destination using Google Maps, this research will proceed to conduct testing to identify the optimal route using the A-Star algorithm on an autonomous electric vehicle in real time, utilizing GPS sensors.

#### IV. RESULT AND DISCUSSIONS

In this study, the go-kart was adequately utilized as a prototype before being integrated into a car. It is important to ensure that the speed of this type of vehicle is not too high. The overall appearance of the developed autonomous vehicle is shown in Fig. 6a. Two encoders were utilized in the autonomous vehicles. One encoder was connected to a 48 V shunt motor using a belt. This was done to measure the rotations of the 48 V shunt motor, thereby obtaining the rotational speed of the wheel in rotations per minute (RPM). The other encoder is attached to the steering wheel using a chain to calculate the steering angle of the autonomous vehicle. The speed controller for the 48 V shunt motor is located at the rear right section of the autonomous vehicle, adjacent to the power source of the shunt motor. The positions of the hardware components are shown in Fig. 6.

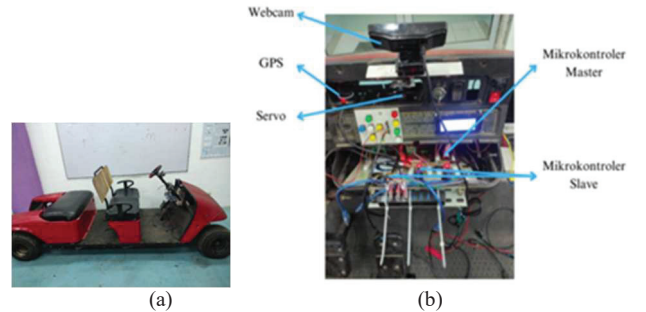


Fig. 6. (a) Autonomous electric design and (b) component positions

##### A. Evaluation of path planning algorithm

The route data acquisition for the simulation was conducted at the Universitas Sriwijaya Campus in Palembang. The collected data include longitudinal and latitudinal data retrieved from Google Maps. Nineteen nodes are obtained from the simulation route data, a total of 19 nodes were obtained. For real-time testing of the route data acquisition, longitudinal and latitudinal data were directly obtained using Google Maps, resulting in a route with 42 nodes. An example of route acquisition is shown in Fig. 7. Each node connects to the closest node, as shown in the figure. For example:  $0 \rightarrow 1$ ,  $1 \rightarrow 2$ ,  $1 \rightarrow 3$ , and  $30 \rightarrow 38$ .

The path planning system uses the A-Star algorithm as shown as follows:

$$F(n) = g(n) + h(n), \quad (5)$$

where  $F(n)$  represents the total cost between nodes,  $h(n)$  represents the calculated distance from the current node to the destination node, and  $g(n)$  represents the cost required from the

first node to the n-th node. The distance between the nodes is calculated using the Euclidean distance as follows [42]:

$$h = \left( \sqrt{(x_{destination} - x_{start})^2 + (y_{destination} - y_{start})^2} \right) \times 1 \text{ degree of earth} \quad (6)$$

where x denotes the latitude location, y denotes the longitude direction, and  $1^\circ$  of Earth is 111.319 km.



Fig. 7. Route in Universitas Sriwijaya campus in Inderalaya

The path planning system using the A\* algorithm, as in (5), was developed using the Python programming language. It comprises a list of waypoints or route nodes within the Universitas Sriwijaya campus and involves calculations to determine the quickest route for an autonomous vehicle. This program first acquires the current coordinates using GPS, and then proceeds to search for the optimal route. The output of the route search contains the nodes that must be traversed and their relative directions relative to the heading direction of an autonomous vehicle. Nodes are represented by numbers, as depicted in Fig. 7, where node 5 denotes the coordinates of the starting point, and node 28 corresponds to the destination coordinates. Consequently, the determined route is  $5 \rightarrow 6 \rightarrow 7 \rightarrow 27 \rightarrow 28$ . The direction sequence is from 5 to 6 (forward), 6 to 7 (right), 7 to 27 (left), and 27 to 28 (right), and upon reaching 28, the vehicle stops. Despite placing an obstacle to block the route between nodes 6 and 7, the A-Star algorithm could still reach the destination by navigating through nodes 5, 6, 9, 27, and 28.

The direction variable obtained can contain movement commands such as "forward," "right," "left," and "stop." These categorical values were converted into numerical values according to the values listed in Table II.

TABLE II : MOVEMENT COMMANDS

No	Category	Numerical
1	Forward	5
2	Right	1
3	Left	-1
4	Stop	0

### B. Evaluation of object-road detection

To ensure that YOLOv8 could be implemented in real time, a comparative experiment was conducted using the KITTI dataset [43]. In these experiments, objects, such as cars and roads, were detected. Some of the experimental results are shown in Fig. 8.



Fig. 8. Sample of road and car detection using the KITTI dataset.

Note: jalan = road and mobil = car

Based on the experiments, YOLOv8 can effectively detect objects with average accuracies of 78%, 82%, and 73% for car1, road, and car2, respectively. These results indicate that the proposed YOLOv8 algorithm is suitable for real-time data.

The proposed model YOLOv8 was implemented on the primary data. The primary data employed come in the form of videos captured using a webcam installed in front of both the autonomous electric vehicle and minibus. The webcam used was a NYK NEMESIS, with a resolution of 1080p at 30 fps. Videos were captured within the Universitas Sriwijaya campus in Inderalaya during the daylight. The video capturing process utilized a Windows video recording application set at a resolution of  $1080 \times 720$  pixels and a frame rate of 30 fps.

The video dataset was saved as. mp4 format and subsequently converted into image frames in PNG format. This conversion was conducted using the Shotcut software, lowering the frame rate to 10 fps. The training dataset was a combination of various objects around an autonomous electric vehicle that are identified during the object-detection phase. These images include roads, motorcycles, cars, pedestrians, and roadblocks. The total number of images obtained after the video conversion was 3803 images. Fig. 9 shows an example of the collected training data images.

Before they can be used in the training process, the image frames must be preprocessed through segmentation using polygon tools for both the road and object images (motorcycles, cars, pedestrians, and roadblocks). Segmentation was performed using a website called Roboflow. The website is designed to facilitate the creation of computer vision datasets. Fig. 9 illustrates an example of the segmentation process on the Roboflow website [44].

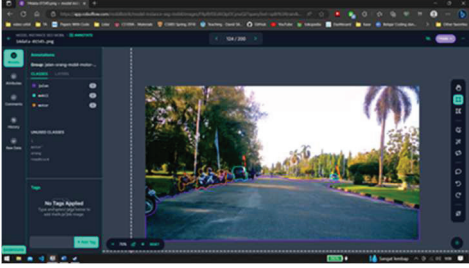


Fig. 9. Labelling process using Roboflow [44]

The training dataset must be labeled for the training process. A data labeling (annotation) process was performed on all objects and roads detected by an autonomous electric vehicle. The annotations obtained consisted of x- and y-coordinates, height, and width. Segmentation results in a collection of points that form lines. The outcomes were stored as raw data.

TABLE III  
LOSS AND METRIC FOR EACH MODEL

Loss	Final loss values		
	100 Epochs	200 Epochs	300 Epochs
Segmentation Loss	0.6411	0.53182	0.63428
Classification Loss	0.36241	0.36321	0.35549
Box Loss	0.46246	0.45688	0.45454
Metrics	Final metrics		
	100 Epochs	200 Epochs	300 Epochs
mAP Box	0.85481	0.85384	0.85715
mAP Masks	0.81334	0.81277	0.81808
Precision Box	0.78191	0.80046	0.79599
Precision Masks	0.79075	0.78085	0.78406
Recall Box	0.86880	0.85148	0.85828
Recall Masks	0.81034	0.80838	0.81313

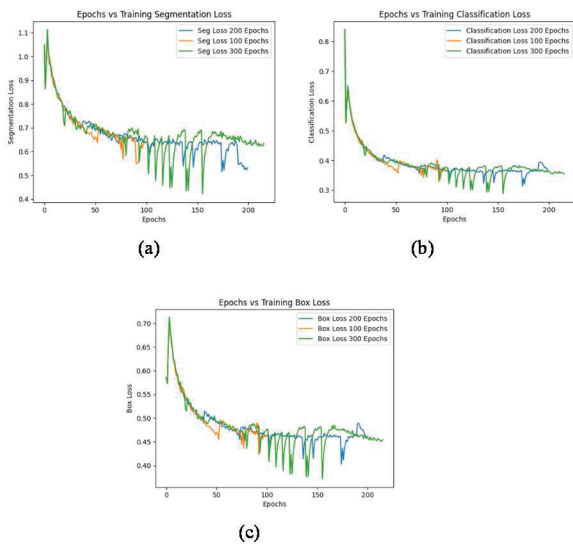


Fig. 10. Comparison of three models (a) Segmentation Loss, (b) Classification Loss, (c) Box Loss

The training process for identifying roads and objects is conducted using YOLO. This process was performed using the

Ultralytics library developed specifically for YOLOv8 [45]. The trained model resulting from this process was saved as .pt format. The training process was performed using Google Colab, a cloud-computing-based environment resembling a Jupyter notebook. Google Colab provides various support software packages for YOLO training including CUDA, CUDnn, OpenCV, and PyTorch. The hardware utilized for training on Google Colab consisted of a Tesla T4 GPU and an Intel Xeon processor with 13 GB of RAM. The training process on Google Colab was conducted for 100, 200, and 300 epochs using the YOLOv8x-seg model. This model was selected based on the initial test results of the YOLOv8 model. The specific differences between each YOLOv8 model are listed in Table III, and a comparison of the three models is presented in Fig. 10. The performance metrics were calculated using (2) to (4). The Precision Box assesses the alignment between the predicted boxes and ground truth boxes, determined through the bounding box coordinates (x, y, width, and height). The evaluation employs the IoU (Intersection over Union) metric, which is widely used in object detection assessments. In contrast, mask precision evaluates the correspondence between the predicted segmentation masks and ground truth masks. This aspect becomes relevant when the model supports segmentation involving pixel-wise classification of objects. Precision mask precision becomes crucial when the model not only detects objects, but also endeavors to outline the specific boundaries of the identified objects.

### C. Steering angle

The steering angle can be calculated as follows [14]:

$$\tan \theta = \left( \frac{X_{offset}}{Y_{offset}} \right) \times \frac{180}{\pi} + 90^\circ \quad (7)$$

where

$$X_{offset} = \frac{LX_2 + RX_2}{\frac{width}{2}} \quad (8)$$

$$Y_{offset} = \frac{Height}{2} \quad (9)$$

When implemented using the above equations, the steering wheel of the car consistently turns to the right and veers off the road, as shown in Fig. 11. Therefore, modifications must be made to the calculation of the car steering.

Therefore, to calculate the steering angle, we used the output from the road detection using YOLOv8, which utilizes the contour lines of the road (marked with blue lines, as shown in Fig. 10). Subsequently, we extracted the values of the first point, which represents the left end of the line, and the last point, which represents the right end of the line. Therefore, the length of the line that connects these two points can be calculated as follows:

$$length = \sqrt{(x_{left} - x_{right})^2 + (y_{left} - y_{right})^2} \quad (10)$$



Fig. 11. Steering angle is outside the road

Upon obtaining the length of that line, the  $x\_offset$  can be determined using the coordinates  $Lx$  at the left end of the contour line,  $Rx$  at the right end of the contour line, and half the length, as follows:

$$X_{offset} = Lx + \left(\frac{length}{2}\right) \quad (11)$$

Upon obtaining the length of that line, the  $x\_offset$  can be determined using the coordinates  $Lx$  at the left end of the contour line,  $Rx$  at the right end of the contour line, and half the length, as follows:

$$angle_{rad} = \tan^{-1}\left(\frac{x_{offset}}{y_{offset}}\right) \quad (12)$$

The above equation still uses radians as the unit, and to convert it into degrees, we use (10) as follows:

$$angle_{deg} = \frac{angle_{rad} \times 180}{\pi} \quad (13)$$

Equation (13) produces an output within the range of 0–90°. If the output approaches 0°, the steering wheel turn to the left. If the output approaches 90°, the steering wheel turns to the right. For straight steering, it has a value of 45°. Furthermore, steering to avoid objects was calculated when the object was at a distance of less than 4 m and positioned to the left. The steering angle is assigned as long as the object remains within the frame. Once the object exits the frame, the calculation is performed again based on (13).

#### D. Object distance measurement

Distance measurements are performed so that an autonomous vehicle can determine when it needs to avoid an object. Each detected object yields four variables, namely ( $x1$ ,  $y1$ ,  $x2$ , and  $y2$ ), where  $x1$  and  $y1$  represent the bottom-left point of the bounding box, and  $x2$  and  $y2$  represent the top-right points of the bounding box. Using these four variables, the height and width of the bounding box can be calculated as follows:

$$width = x2 - x1 \quad (14)$$

$$height = y2 - y1 \quad (15)$$

The values of the width and height can be used to measure the distance to an object based on depth information [46] as follows:

$$distance = \frac{(2 \times 3.14 \times 180)}{(w + h \times 360) \times 1000 + 3}. \quad (16)$$

This equation is a combination of the formula for the arc length and the width and height of each created bounding box. A value of 1000 was used to convert the final unit into inches, with three as the constant threshold added to approximate the measured distance to the actual distance.

The sample results of objects, steering angles, and the measured distances for the simulation based on the YOLOv8 model, using (13) and (16), are presented in Table IV.

TABLE IV  
SAMPLE TESTING RESULTS FOR SIMULATION

Image	Steering angle (degree)	Object detected and distance (m)
	55	Motorcycle - 4 meter
	60	Motorcycle - 1 meter
	47	Car - 0.6 meter
	43	Human - 5 meter






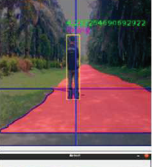



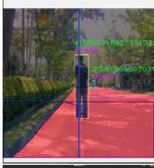
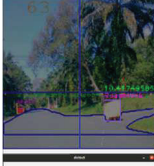



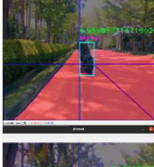


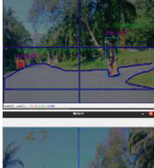

44 Human – 2 meter

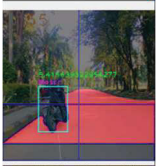
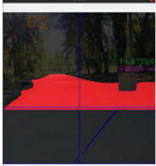



From the sample test results, it can be observed that the developed system is capable of segmenting the road, detecting objects, and determining the distance between an autonomous vehicle and the detected objects. The car steering system adjusts its movement based on the road and object detection results.

The results of the real-time experiments are presented in Table V. As indicated in the table, the system could detect distances with an average error of 2.245 m. The actual and detected distances obtained using the proposed method yielded zero errors in the 12th and 13th experiments. The highest errors occurred in the 9th and 19th experiments. This discrepancy can be attributed to the fact that the position of the object is extremely close to the corner of the image space, highlighting the significance of the position of the object from the camera's viewpoint for accurate detection.

TABLE V  
REAL-TIME RESULTS

Exp. no	Detected image	Actual distance (m)	Detected distance (m)	Error
1		3.7	3.6	0.1
2		4.5	8.5	4
3		5.8	5.9	0.1
4		6	4.2	1.8
5		9	5.2	3.8

6		6	4.1	1.9
7		10	10.4	0.4
8		7.8	5.1	2.7
9		13.3	7.6	5.7
10		6.2	9	2.8
11		7	6.9	0.1
12		7	7	0
13		4.9	4.9	0
14		10	6	4
15		9.4	5.5	3.9

16		3.8	5.4	1.6
17		9.1	11.8	2.7
18		7.3	5.1	2.2
19		13.3	7.6	5.7
20		3.8	5.2	1.4
Average error				2.245

*E. Evaluation of controller system*

The test results indicate that the developed path-planning program can create routes accurately and effectively. At point A, the path value or path score is one because of a right turn, specifically at the intersection after the FT mosque, toward the university library of Universitas Sriwijaya. Additionally, in three instances, where the path value is zero, indicating a stop command at points B, C, and D, corresponding to the FH-FE, FKIP, and FT Dean’s office intersections, respectively, as shown in Fig. 12. These values are evident in the graph that depicts the transmitted movement commands, as shown in Fig. 13. In the graph, point A represents a situation in which the path value is one, signifying a right turn command when approaching the intersection in front of the FT mosque. Furthermore, there are three instances with a value of zero, denoted as stop1 (B), stop2 (B), and stop3 (D).



Fig. 12. Route evaluation

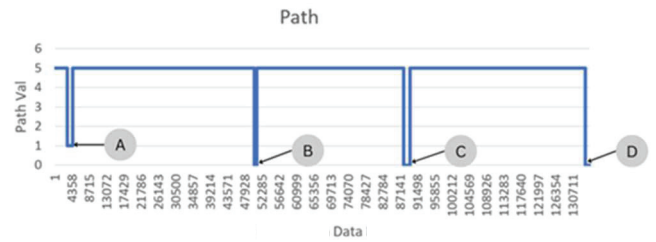


Fig. 13. Movement command execution results

*F. System Testing*

Based on the graph in Fig. 14, it can be observed that the autonomous vehicle is capable of following a straight path quite well. The detected steering angle values tend to be in the range of 200–300. However, significant deviations exist in the steering angle values, either above 300° or below 200°, or steering angle values below 40° and above 50°. These values indicate the detections that require sharp turns in autonomous vehicles. An example of such a sharp turn is shown in Fig. 15.

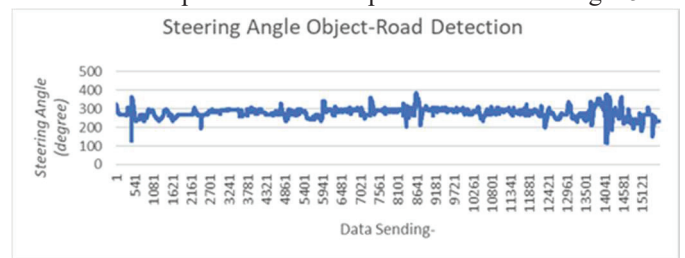


Fig. 14. Steering angle object-road detection

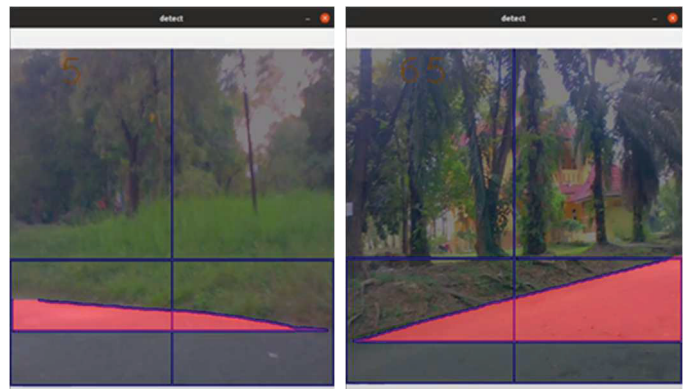


Fig. 15. Results of Detecting Sharp Turns: (a) Left 5° (b) Right 65°

This indicates that the program can perform well. The steering angle data obtained from the encoder readings are shown in Fig. 16.

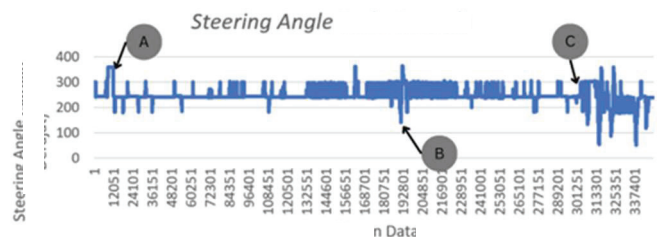


Fig. 16. Steering angle encoder wheel steering

Based on Fig. 16, point A with a steering value greater than 300 represents the response of the system to a turn at the FT mosque intersection, which requires the autonomous vehicle to turn to the right. The autonomous vehicle smoothly navigates this turn, as is evident from the graph, with the steering value consistently maintained in the range of 200–300 on straight roads, as per the provided route.

However, abnormal oscillations are observed at point B, where the autonomous vehicle cannot follow the road. This road condition is not straight, causing the vehicle to collide with road barriers and necessitating the driver's intervention to bring the autonomous vehicle back to its correct position. This condition is illustrated in Fig 17.



Fig. 17. Condition requiring driver intervention at the FH FE – FKIP intersection route

At point C, abnormal oscillations also occurred because the autonomous vehicle could not follow a non-straight road, as shown in Fig. 16. Therefore, driver intervention is necessary to realign an autonomous vehicle with the correct lane. Furthermore, the data of the shunt motor RPM readings can be observed in Fig. 18. Here, the speed is measured in RPM instead of km/h because it helps determine when to change gears. In addition, the current prototype has a low speed of approximately 20 km/h.



Fig. 18. Conditions Requiring Driver Intervention on Curved Road (a) and Intersection (b) on the FKIP – FT Dean's Office Intersection Route

Based on the graph in Fig. 19, it can be observed that the autonomous vehicle often stops, either suddenly or not, as indicated by revolutions per minute (RPM) of zero. This occurs because the shunt motor of an autonomous vehicle overheats, preventing the speed controller from driving the shunt motor. Another condition that causes the RPM to be zero is when the autonomous vehicle is too far from the road lane, causing it to touch the road barriers, as mentioned earlier. Additionally, this can occur when there are other users on the road, requiring the autonomous vehicle to stop to avoid collisions, as shown in Fig. 20.

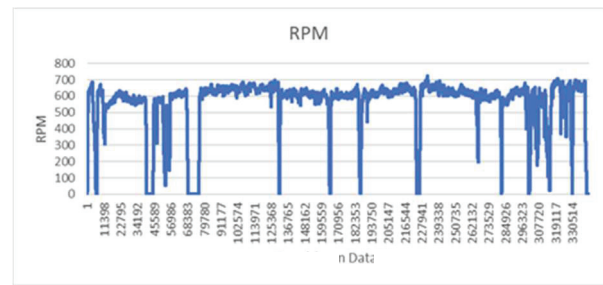


Fig. 19. RPM data



Fig. 20. Presence of other road users

Based on the conducted tests, the sensor fusion system of the autonomous vehicle, which combines key sensors, such as cameras and GPS, functioned. This can be observed in the test coordinates, which are aligned with the map. Additionally, the steering angle of the detected object and the steering angle from the steering wheel encoder are within the specified 0–540° range. The autonomous vehicle can travel smoothly on straight roads, as indicated by the steering values consistently maintained around the middle, i.e., 200–300°. However, errors still occur, such as vehicles moving off the road or touching road barriers. This can occur for various reasons, such as changes in the orientation of the camera requiring mid-road calibration, leading to steering angle detection that does not match the required steering angle. This is also a consequence of steering angle detection using a range of values from 0–9, making the sensitivity of the steering angle less responsive. This has significant consequences for curved roads and intersections. These false alarm conditions commonly occur when a picture is congested or when there is an illumination effect [47][48].

## V. CONCLUSION

The research model applied in this test used the YOLOv8x-seg model trained for 200 epochs. This model exhibited the lowest segmentation loss of 0.53182 compared with the training results for 100 and 300 epochs. This trained model was then used in simulation testing, where the system performed well in detecting objects and roads. In addition, the system could measure the distance between objects and an autonomous vehicle based on the bounding box results, and steering decisions are made based on road contours and object detection.

The real-time implementation of the instance segmentation algorithm using YOLOv8 to identify the road as an input for the steering control of an autonomous electric vehicle was successful. The system effectively maintained an autonomous electric vehicle on the road and allowed it to follow a predetermined route. Furthermore, the system could be used to identify objects and measure distances with an average error of 2.245 m. Distance measurements of objects can be used as inputs for the steering control of autonomous electric vehicles

to avoid obstacles. Based on our research, sensor fusion with a hybrid deep learning algorithm using YOLOv8 and path planning based on the A-star algorithm can be applied to autonomous vehicle systems. This study demonstrated that sensor fusion, which combines two tasks to improve autonomous vehicle performance, could be successfully implemented. Several parameters, such as the range values of the steering motor's pulse width modulation (PWM) and the range division for the steering angle, can make an autonomous vehicle more responsive to its movements.

However, in sensor fusion, which combines two tasks, several issues, such as unreachable nodes and blind spots on the autonomous vehicle, particularly at the rear and sides, remain. Therefore, sensor fusion can be performed separately for each task to enhance the performance of autonomous vehicles further. In addition, the use of a go-kart as a prototype at low speeds should be replaced with another car to ensure the performance of sensor fusion in future work.

#### ACKNOWLEDGMENT

This study was funded through the Directorate of Research, Technology, and Community Service Directorate General of Higher Education, Research, and Technology According to the Research Contract Number: 0192.09/UN9.3.1/PL/2023

#### REFERENCES

- [1] SAE J3016:JAN2014, "Taxonomy and Definitions for Terms Related to On-Road Motor Vehicle Automated Driving Systems," *Soc. Autom. Eng.*, 2014.
- [2] [M. Vignati, D. Tarsitano, M. Bersani, and F. Cheli, "Autonomous Steer Actuation for an urban Quadricycle," *2018 Int. Conf. Electr. Electron. Technol. Automotive, Automot.2018*, pp. 1–5, 2018, doi: 10.23919/EETA.2018.8493199.
- [3] W. Zhao, H. Zhang, and Y. Li, "Displacement and force coupling control design for automotive active front steering system," *Mech. Syst. Signal Process.*, vol. 106, pp. 76–93, 2018, doi: 10.1016/j.ymsp.2017.12.037.
- [4] H. Halin *et al.*, "Design Simulation of a Fuzzy Steering Wheel Controller for a buggy car," *2018 Int. Conf. Intell. Informatics Biomed. Sci. ICIBMS 2018*, vol. 3, pp. 85–89, 2018, doi: 10.1109/ICIBMS.2018.8550008.
- [5] B. Y. Suprpto, D. R. Dini, M. N. G. Iskandar, P. K. Wijaya, Rendyansyah, and S. Dwijayanti, "Position Control System on Autonomous Vehicle Movement Using Fuzzy Logic Methods," *2022 5th Int. Semin. Res. Inf. Technol. Intell. Syst. ISRITI 2022*, pp. 744–749, 2022, doi: 10.1109/ISRITI56927.2022.10052888.
- [6] T. D. Do, M. T. Duong, Q. V. Dang, and M. H. Le, "Real-Time Self-Driving Car Navigation Using Deep Neural Network," *Proc. 2018 4th Int. Conf. Green Technol. Sustain. Dev. GTSD 2018*, pp. 7–12, 2018, doi: 10.1109/GTSD.2018.8595590.
- [7] Z. Ercan, M. Gokasan, and F. Borrelli, "An adaptive and predictive controller design for lateral control of an autonomous vehicle," *2017 IEEE Int. Conf. Veh. Electron. Safety, ICVES 2017*, pp. 13–18, 2017, doi: 10.1109/ICVES.2017.7991894.
- [8] Z. Wang, Y. Wu, and Q. Niu, "Multi-Sensor Fusion in Automated Driving: A Survey," *IEEE Access*, vol. 8, pp. 2847–2868, 2020, doi: 10.1109/ACCESS.2019.2962554.
- [9] T. Almeida, B. Lourenço, and V. Santos, "Road detection based on simultaneous deep learning approaches," *Rob. Auton. Syst.*, vol. 133, p. 103605, 2020, doi: 10.1016/j.robot.2020.103605.
- [10] H. Saleem, F. Riaz, L. Mostarda, M. A. Niazi, A. Rafiq, and S. Saeed, "Steering Angle Prediction Techniques for Autonomous Ground Vehicles: A Review," *IEEE Access*, vol. 9, pp. 78567–78585, 2021, doi: 10.1109/ACCESS.2021.3083890.
- [11] A. Ghaida, H. Hikmarika, S. Dwijayanti, and B. Yudho Suprpto, "Road and Vehicles Detection System Using HSV Color Space for Autonomous Vehicle," 2020.
- [12] A. A. Mahersatillah, Z. Zainuddin, and Y. Yusran, "Unstructured Road Detection and Steering Assist Based on HSV Color Space Segmentation for Autonomous Car," *2020 3rd Int. Semin. Res. Inf. Technol. Intell. Syst. ISRITI 2020*, pp. 688–693, 2020, doi: 10.1109/ISRITI51436.2020.9315452.
- [13] T. Rateke and A. von Wangenheim, "Road surface detection and differentiation considering surface damages," *Auton. Robots*, vol. 45, no. 2, pp. 299–312, 2021, doi: 10.1007/s10514-020-09964-3.
- [14] M. Rasib, M. A. Butt, F. Riaz, A. Sulaiman, and M. Akram, "Pixel Level Segmentation Based Drivable Road Region Detection and Steering Angle Estimation Method for Autonomous Driving on Unstructured Roads," *IEEE Access*, vol. 9, pp. 167855–167867, 2021, doi: 10.1109/ACCESS.2021.3134889.
- [15] T. Wu, A. Luo, R. Huang, H. Cheng, and Y. Zhao, "End-to-End Driving Model for Steering Control of Autonomous Vehicles with Future Spatiotemporal Features," *IEEE Int. Conf. Intell. Robot. Syst.*, pp. 950–955, 2019, doi: 10.1109/IRROS40897.2019.8968453.
- [16] T. V. Samak, C. V. Samak, and S. Kandhasamy, "Robust Behavioral Learning for Autonomous Vehicles Using End-to-End Imitation Learning," *SAE Int. J. Connect. Autom. Veh.*, vol. 4, no. 3, pp. 1–17, 2021, doi: 10.4271/12-04-03-0023.
- [17] H. Haavaldsen, M. Aasbø, and F. Lindseth, "Autonomous vehicle control: end-to-end learning in simulated urban environments," *Commun. Comput. Inf. Sci.*, vol. 1056 CCIS, pp. 40–51, 2019, doi: 10.1007/978-3-030-35664-4\_4.
- [18] T. van Orden and A. Visser, "End-to-End Imitation Learning for Autonomous Vehicle Steering on a Single-Camera Stream," *Lect. Notes Networks Syst.*, vol. 412 LNNS, pp. 212–224, 2022, doi: 10.1007/978-3-030-95892-3\_16.
- [19] M. Hermawan, Z. Husin, H. Hikmarika, S. Dwijayanti, and B. Y. Suprpto, "Road Identification using Convolutional Neural Network on Autonomous Electric Vehicle," in *2021 8th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI)*, 2021, pp. 341–346.
- [20] J. Yu, J. Hou, and G. Chen, "Improved Safety-First A-Star Algorithm for Autonomous Vehicles," in *2020 5th International Conference on Advanced Robotics and Mechatronics (ICARM)*, Dec. 2020, pp. 706–710, doi: 10.1109/ICARM49381.2020.9195318.
- [21] K. Jo, M. Lee, W. Lim, and M. Sunwoo, "Hybrid Local Route Generation Combining Perception and a Precise Map for Autonomous Cars," *IEEE Access*, vol. 7, pp. 120128–120140, 2019, doi: 10.1109/ACCESS.2019.2937555.
- [22] Y. Li, D. Dong, and X. Guo, "Mobile robot path planning based on improved genetic algorithm with A-star heuristic method," vol. 2020, pp. 1306–1311, 2020, doi: 10.1109/ITAIC49862.2020.9338968.
- [23] A. Candra, M. A. Budiman, and K. Hartanto, "Dijkstra's and A-Star in Finding the Shortest Path: A Tutorial," *2020 Int. Conf. Data Sci. Artif. Intell. Bus. Anal. DATABIA 2020 - Proc.*, pp. 28–32, 2020, doi: 10.1109/DATABIA50434.2020.9190342.
- [24] R. Yang and L. Cheng, "Path Planning of Restaurant Service Robot Based on A-star Algorithms with Updated Weights," *Proc. - 2019 12th Int. Symp. Comput. Intell. Des. Isc. 2019*, vol. 1, pp. 292–295, 2019, doi: 10.1109/ISCID.2019.00074.
- [25] H. Liu, T. Shan, and W. Wang, "Automatic Routing Study of Spacecraft Cable based on A-star Algorithm," *Proc. 2020 IEEE 5th Inf. Technol. Mechatronics Eng. Conf. ITOEC 2020*, no. Itoec, pp. 716–719, 2020, doi: 10.1109/ITOEC49072.2020.9141822.
- [26] Z. Liu, H. Liu, Z. Lu, and Q. Zeng, "A Dynamic Fusion Pathfinding Algorithm Using Delaunay Triangulation and Improved A-Star for Mobile Robots," *IEEE Access*, vol. 9, pp. 20602–20621, 2021, doi: 10.1109/ACCESS.2021.3055231.
- [27] G. Tang, C. Tang, C. Claramunt, X. Hu, and P. Zhou, "Geometric A-Star Algorithm: An Improved A-Star Algorithm for AGV Path Planning in a Port Environment," *IEEE Access*, vol. 9, pp. 59196–59210, 2021, doi: 10.1109/ACCESS.2021.3070054.
- [28] M. Kusuma, Riyanto, and C. Machbub, "Humanoid Robot Path Planning and Rerouting Using A-Star Search Algorithm," *Proc. - 2019 IEEE Int. Conf. Signals Syst. ICSigSys 2019*, no. July, pp. 110–115, 2019, doi: 10.1109/ICSIGSYS.2019.8811093.
- [29] C. Ju, Q. Luo, and X. Yan, "Path Planning Using an Improved A-star Algorithm," *Proc. - 11th Int. Conf. Progn. Syst. Heal. Manag. PHM-Jinan 2020*, pp. 23–26, 2020, doi: 10.1109/PHM-Jinan48558.2020.00012.
- [30] L. Caltagirone, M. Bellone, L. Svensson, and M. Wahde, "LIDAR-camera fusion for road detection using fully convolutional neural networks," *Rob. Auton. Syst.*, vol. 111, pp. 125–131, 2019, doi: 10.1016/j.robot.2018.11.002.
- [31] Y. Liu, X. Fan, C. Lv, J. Wu, L. Li, and D. Ding, "An innovative

information fusion method with adaptive Kalman filter for integrated INS/GPS navigation of autonomous vehicles," *Mech. Syst. Signal Process.*, vol. 100, pp. 605–616, 2018.

- [32] F. Nobis, M. Geisslinger, M. Weber, J. Betz, and M. Lienkamp, "A Deep Learning-based Radar and Camera Sensor Fusion Architecture for Object Detection," *2019 Symp. Sens. Data Fusion Trends, Solut. Appl. SDF 2019*, 2019, doi: 10.1109/SDF.2019.8916629.
- [33] S. Zhang, Y. Guo, Q. Zhu, and Z. Liu, "Lidar-IMU and Wheel Odometer Based Autonomous Vehicle Localization System," *Proc. 31st Chinese Control Decis. Conf. CCDC 2019*, no. June 2019, pp. 4950–4955, 2019, doi: 10.1109/CCDC.2019.8832695.
- [34] R. Hussain and S. Zeadally, "Autonomous Cars: Research Results, Issues, and Future Challenges," *IEEE Commun. Surv. Tutorials*, vol. 21, no. 2, pp. 1275–1313, 2019, doi: 10.1109/COMST.2018.2869360.
- [35] V. Tümen and B. Ergen, "Intersections and crosswalk detection using deep learning and image processing techniques," *Phys. A Stat. Mech. its Appl.*, vol. 543, 2020, doi: 10.1016/j.physa.2019.123510.
- [36] S. Zheng *et al.*, "Rethinking Semantic Segmentation from a Sequence-to-Sequence Perspective with Transformers," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 6877–6886, 2021, doi: 10.1109/CVPR46437.2021.00681.
- [37] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, et al. "Segment anything." In Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 4015–4026. 2023..
- [38] M. Ranjith Rochan, K. Aarthi Alagammai, and J. Sujatha, "Computer vision based novel steering angle calculation for autonomous vehicles," *Proc. - 2nd IEEE Int. Conf. Robot. Comput. IRC 2018*, vol. 2018-Janua, pp. 143–146, 2018, doi: 10.1109/IRC.2018.00029.
- [39] S. Rahman, J. H. Rony, and J. Uddin, "Real-Time Obstacle Detection with YOLOv8 in a WSN Using UAV Aerial Photography," 2023.
- [40] P. Jiang, D. Ergu, F. Liu, Y. Cai, and B. Ma, "A Review of Yolo Algorithm Developments," *Procedia Comput. Sci.*, vol. 199, pp. 1066–1073, 2021, doi: 10.1016/j.procs.2022.01.135.
- [41] R.O. Duda, P.E. Hart, D.G. Stork, *Pattern Classification*, 2nd edn. New York, 2001.
- [42] E. Maria, E. Budiman, Haviluddin, and M. Taruk, "Measure distance locating nearest public facilities using Haversine and Euclidean Methods," *J. Phys. Conf. Ser.*, vol. 1450, no. 1, 2020, doi: 10.1088/1742-6596/1450/1/012080.
- [43] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "The KITTI vision Benchmark Suite," 2015, no. 5, pp. 1–13. [Online]. Available: <http://www.cvlibs.net/datasets/kitti> 2.
- [44] B. Dwyer, J. Nelson, T. Hansen, et. al., 2023. Roboflow (Version 1.0) [Online]. Available: <https://roboflow.com>. Accessed: Jul. 03, 2023.
- [45] G. Jocher, A. Chaurasia, and J. Qiu, 2023. Ultralytics YOLOv8 Docs. [Online]. Available: <https://github.com/ultralytics/ultralytics>. Accessed: Jul. 03, 2023.
- [46] I. V. Fanthony, Z. Husin, H. Hikmarika, S. Dwijayanti, and B. Y. Suprpto, "YOLO Algorithm-Based Surrounding Object Identification on Autonomous Electric Vehicle," *Int. Conf. Electr. Eng. Comput. Sci. Informatics*, vol. 2021-October, no. October, pp. 151–156, 2021, doi: 10.23919/EECSI53397.2021.9624275.
- [47] S. Usmanhujaev, S. Baydadaev, and K. J. Woo, "Real-time, deep learning based wrong direction detection," *Appl. Sci.*, vol. 10, no. 7, 2020, doi: 10.3390/app10072453.
- [48] Y. Wiseman, "Real-time monitoring of traffic congestions," *IEEE Int. Conf. Electro Inf. Technol.*, pp. 501–505, 2017, doi: 10.1109/EIT.2017.8053413.



**Bhakti Yudho Suprpto (M'14)** was born on February 11, 1975, in Palembang, South Sumatra, Indonesia. He became a Member (M) of IEEE in 2014. He graduated from Sriwijaya University of Palembang with a major in Electrical Engineering. His postgraduate and doctoral programs were in Electrical

Engineering at Universitas Indonesia (UI). He is an academic staff member in the Electrical Engineering department at Universitas Sriwijaya, where he lectures. He has authored 3-

chapter books and more than 67 articles. His research interests primarily focus on control and intelligent systems.



**Suci Dwijayanti (M'13)** received an M.S. degree in electrical and computer engineering from Oklahoma State University, Stillwater, OK, USA in 2013. She received the Fulbright scholarship for her Master's degree. Following this, in 2018, she received her Doctoral degree from the Graduate School of Natural Science and Technology, Kanazawa

University, Japan. Her research interests include signal processing and machine learning. She was previously an engineer with ConocoPhillips Indonesia Inc Ltd from 2007 to 2008. Since 2008, she has been with the department of Electrical Engineering at Universitas Sriwijaya, Indonesia. She is a member of IEEE.



**Farhan Abie Ardandy** was born in Tanjung Enim, Sumatera Selatan, Indonesia, on July 14, 2001. He obtained his bachelor's degree in Electrical Engineering from Universitas Sriwijaya, Indonesia, in 2023. His main research interests are in image processing and neural networks. For his thesis, he researched the application of

convolutional neural networks for image processing to detect roads and obstacles.



**Javen Jonathan** was as born in Palembang, South Sumatera, Indonesia, on October 1, 2001. He obtained his bachelor's degree in electrical engineering from Universitas Sriwijaya, Indonesia, in 2023. His main research interests are in path planning and robotics. For his thesis, he studied the application of the A-star algorithm to search for the best route for autonomous vehicles.

autonomous vehicles.



**Dimsyiar M Al Hafiz** was born in Palembang, South Sumatera, Indonesia, on October 1, 2002. He obtained his bachelor's degree in electrical engineering from Universitas Sriwijaya, Indonesia, in 2023. His main research interests are in control and sensors. For his thesis, he explored the application of deep learning

algorithms for controlling the movement of autonomous vehicles.