

Neural network training for serial multisensor of autonomous vehicle system

Eka Nuryanto Budisusila¹, Sri Arttini Dwi Prasetyowati², Bhakti Yudho Suprpto³,
Zainuddin Nawawi³

¹Department of Electrical Engineering, Faculty of Engineering, Universitas Sriwijaya on leave Universitas Islam Sultan Agung, Semarang, Indonesia

²Department of Electrical Engineering, Faculty of Industrial Engineering, Universitas Islam Sultan Agung, Semarang, Indonesia

³Department of Electrical Engineering, Faculty of Engineering, Universitas Sriwijaya, Palembang, Indonesia

Article Info

Article history:

Received Dec 6, 2021

Revised May 27, 2022

Accepted Jun 14, 2022

Keywords:

Autonomous vehicle

Backpropagation

Multisensor

Neural network

Ultrasonic

ABSTRACT

This study aims to find the best artificial neural network weight values to be applied to the autonomous vehicle system with ultrasonic multisensor. The implementation of neural network in the system required long time process due to its training process. Therefore, this research is using offline training before implementing to online training by embedding the best network weight values to obtain the outputs faster according to desired targets. Simulink were used to train the system offline. Eight ultrasonic sensors are used on all sides of the vehicle and arranged in a serial multisensory configuration as inputs of neural network. With eight inputs, one sixteen-depth hidden layer, and five outputs, it was trained using the back-propagation algorithm of artificial neural network. By 100000 iterations, the output values and the target values are almost the same, indicating its convergency with minimum of errors. The result of this training is the best weights of the networks. These weight values can be implemented as fixed-weight in online training.

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



Corresponding Author:

Zainuddin Nawawi

Department of Electrical Engineering, Faculty of Engineering, Universitas Sriwijaya

Indralaya, Palembang, Indonesia

Email: nawawi_z@unsri.ac.id

1. INTRODUCTION

In the recent decade, unmanned transportation has grown in popularity. Several businesses produce goods to demonstrate their latest technology, particularly how their devices can automatically follow the path and fulfil their objectives. Certainly, they will face impediments on the road to their destination that must be avoided in order to avoid mishaps [1]. The system needs sensors placed in it to scan the scenario and conditions around the car in order to recognize items that could become vehicle obstacles. Despite their color and light intensity limitations, cameras and Lidar are increasingly being employed as sensors to detect objects around cars [2]. Radiofrequency radars are also used for this purpose, however they are ineffective in detecting non-metallic items [3].

Due to technical advancements and its range, ultrasonic sensors are thought to be able to replace or improve detection systems to overcome the concerns mentioned above. Ultrasonic sensors have been employed in automobile sensors for a long time, but their employment has been confined to the parking system due to the old technology's short detection distance [4], [5]. Ultrasonic sensors have improved their detection, signal quality, and reliability as a result of substantial advancements in technology. It includes their resistance to light intensity, object color, and climatic parameters [3], [4], [6].

Several algorithms were used in past studies to regulate the vehicle system and avoid impediments around the vehicle. Although the research uses a modest prototype robot model, the majority of them use neural network techniques to solve their system challenges due to the advantages of neural networks and their compatibility in vehicle systems [7]-[10]. It is common practice to use artificial intelligence in a system to improve system performance and reliability. Such as the application of deep learning using artificial neural networks for various types of systems, both forecasting and system control. An artificial neural network has three important elements, there are input layer, hidden layer, and output layer. The input layer may consist of several neurons, as well as the output layer. The hidden layer may also consist of several hidden layers with a number of neurons in each. The wider and deeper artificial neural network system is, the more complex the system will be [11]. Artificial neural networks (ANNs) take a long time in the training process, especially when implementing deep learning in it with large iterations. The amount of data that must be trained is also quite large to obtain precise output results according to the given target [12]. Likewise with the weight of each network that is resulted for each iteration. All processes require a large amount of time and memory [13].

With this fact, the application of ANNs for real time systems encounters many constraints, especially the problem of length of time, so it is necessary to pursue strategies and methods for their implementation [14]. Especially for the requirements of autonomous vehicle systems that require processing speed and precision of action. For this reason, this paper proposes a new method by conducting offline training with a number of iterations to obtain network weights, and then pinning those weights for online application, so that the process becomes shorter and faster because in online implementation there is no more iteration process and the network weights have been obtained during offline training.

Medina-Santiago and his team describe in their research “Neural control system in obstacle avoidance in mobile robots using ultrasonic sensors” about the design and realization of robots using ultrasonic sensors. There are three ultrasonic sensors of the SRF02 type that are placed on the front of the robot and function as object detectors to avoid obstacles in front of the robot. An Arduino mega microcontroller unit with a liquid-crystal display (LCD) screen was used as the central processing unit (CPU). To move forward, the robot actuator uses two motor gears and their drive as a robot mover. The back-propagation method is used in this study’s algorithm, which is a neural network with three input components and four output elements. To acquire the optimum performance from the system, learning this system is done offline using the neural network training (nntool) tools in the MATLAB software. Medina’s implementation was limited to testing mini robots and has not been implemented in real terms for real vehicles, according to the results of the experiments. It only provided ultrasonic detectors at the front at close range without equipping it with side or rear sensors, and its implementation was limited to testing mini robots and has not been implemented in real terms for real vehicles [7]. Considering the limitations of Medina’s research, the proposed method to upgrade the research is described here, that there are eight sensors installed on vehicles, and their implementation for real vehicles in real conditions.

Sugathan *et al.* [15] created a tiny robot based on Medina’s research, as detailed in his publication “Collision avoidance using neural networks.” All of the procedures utilized are nearly identical to Medina’s research, but Shilpa chose a robotic detector with three infrared (IR) sensors on the front side and another sensor on the back side, all powered by an ARM Cortex-R3 processor [15]. This infrared sensor’s flaw is that, in addition to its limited range, it also struggles to detect dark-color objects, resulting in frequent detection failures.

In the study “Obstacle avoidance system for unmanned ground vehicle by using ultrasonic sensors,” Marco and his colleagues use the SRF05 sensor in the construction of a small robot that can avoid obstacles. The approach is also based on a neural network, but it is weighted for offline learning in MATLAB [8]. However, because the path model has already been established, certain obstacle items have been identified. So that the robot can record the trace and location of the obstruction, so that when the robot is tested to cross again, it will automatically detect and avoid the obstacle. The robot’s design is also unique in terms of wheel location, with the driving force located in front of the three existing ultrasonic sensors.

As a response of previous research limitation, the long range sensors of Maxbotix MB7383HRXL and MB7380HRXL ultrasonic sensors are used in this research, which have a detection range of up to 10 meters and are designed to be deployed outside [16]–[18]. By these sensors the limitations range detections are upgraded to be implemented in real condition without light or weather problems. Also, the intelligent of the system is using an artificial neural network algorithm to analyze all sensor input signals and determine what action to perform, such as braking, turning, or continuing forward [15], [16], and compared to the Medina’s training method. This study is trained in a more complete neural network diagram and shows the training weights in real time as they are processed. This study aims to initialize the neural network weights that will be used in online training by implementing them on a neural network with fixed weights, so that the neural network process can be faster.

2. ALGORITHM

The back-propagation method of an artificial neural network algorithm is incorporated into the system. This algorithm will take several inputs and process them to make a decision and produce certain outputs. The output forecast will be assessed and compared to the set of target data. If errors occur, the back-propagation algorithm will update the weight value and re-process the neural network method until the errors are reduced to a minimum [9], [10].

A neural network approach with a back-propagation weight updating procedure is shown in Figure 1. A system with a neural network algorithm processes the data from the input. The system’s output predictions are based on the data that has been processed. This result will be compared to the data that has been specified as the target. If any errors are identified, the back propagation method will adjust the weight values until the output has a minimal error or no error.

As shown in Figure 2, input data will be provided by eight inputs, they are left-front (LF), right-front (RF), left-rear (LR), right-rear (RR), left-side-front (LSF), left-side-rear (LSR), right-side-front (RSF), and right-side-rear (RSR). The inputs are fed into the deep hidden layer, then sent to the output layer to issue some outputs of the system. As the system’s inputs, they deliver detection data of its each position. When delivering data to the hidden layer, each node has a weight value that determines the strength of the propagation signal. The hidden layer has approximately 16 neurons to get the greatest network performance due to the input number [19]-[21]. The system’s actions are described to five outputs, whether to stop or brake (S/B), soft brake (SB), turn right (TR) or turn left (TL), or continue onward (GO).

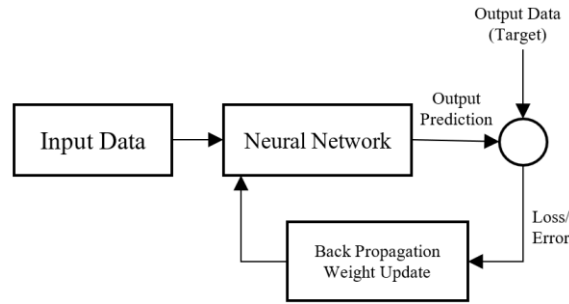


Figure 1. Neural network with back propagation algorithm

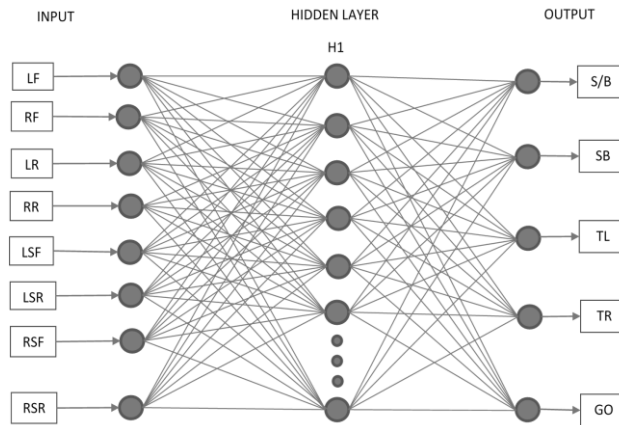


Figure 2. Neural network configuration

There are two main parts of ANN, the forward propagation and the backward propagation, as it shown in Figure 3. In forward propagation, ANN will try to produce a value of y , while in backward propagation, ANN will improve itself (update weights). So, in the next forward propagation it is expected to produce a better y value or closer to the original label [21]–[24]. Forward propagation is the calculation process “forward” from the input (symbolized x) until the output model is obtained (symbolized y). For example, as illustrated in Figure 4, if there are four inputs (x) and three outputs (y), the values of y_1 , y_2 , and y_3 are obtained by calculating as mentioned in formula (3).

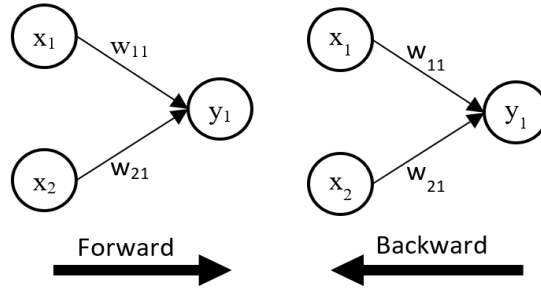


Figure 3. Forward and backward propagation

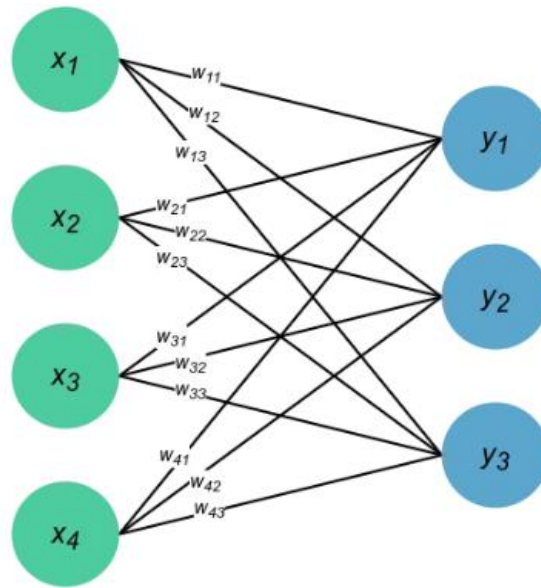


Figure 4. Input to output network

$$\begin{aligned}
 y_1 &= \sigma(w_{11}x_1 + w_{21}x_2 + w_{31}x_3 + w_{41}x_4 + b_1) \\
 y_2 &= \sigma(w_{12}x_1 + w_{22}x_2 + w_{32}x_3 + w_{42}x_4 + b_2) \\
 y_3 &= \sigma(w_{13}x_1 + w_{23}x_2 + w_{33}x_3 + w_{43}x_4 + b_3)
 \end{aligned}
 \tag{1}$$

It is abbreviated by the formula:

$$y_j = \sigma(\sum_{i=1}^4 w_{ij}x_i + b_j)
 \tag{2}$$

The sigma (σ) is the activation function. That is, after the process of multiplying the input x and w weights and then adding all of them, the next step is regarding the results of these calculations with the activation function. There are many activation functions to choose from, one of which is the sigmoid activation function which looks like (3).

$$\sigma(x) = \frac{1}{1+e^{-x}}
 \tag{3}$$

After getting the error value, we can start to improve our ANN with backpropagation. The weight fixing formula is

$$w_{new} = w_{old} - \alpha \frac{\partial E}{\partial w}
 \tag{4}$$

Likewise, the bias can be written as

$$b_{new} = b_{old} - \alpha \frac{\partial E}{\partial w} \quad (5)$$

The symbol alpha (α) in the above formula is the learning rate, a constant usually between 0 to 1 that determines how fast the learning process of the model is. In this learning, the value of 0.001 is used [25]. While the symbol $E/\partial w$, or partial derivative of E against w, is the process of finding the value of the derivative of E for the variable to be updated, in this example is w. The process of looking for this new derivative is more accurately called backpropagation. Since there are many w values, we specify to update the w_{11} values first.

3. RESEARCH METHOD

3.1. The system

There are two methods implemented in this study. They are offline training and online training. Here, only the offline training is discussed in this paper, and the online training can be presented next. However, the system conducted for the methods is described here to give general view.

The system's inputs consist of eight ultrasonic sensors. Two sensors are in front of the car, and two sensors are in the back. Two additional sensors are located on the right side, and two sensors are located on the left side. The front and back sensors are the MB7383 HRXL ultrasonic sensors, which have detection range up to 10 meters. The range of these sensors can be regulated by the sensors' voltage source, which is affected by changes in vehicle speed. The sensor's range increases as the speed increases. This method is intended to conserve the energy absorbed by the sensors from the power supply [26]. On the vehicle's side, an MB7380 HRXL sensor is employed, which can detect objects up to a distance of five meters, and this device is configured to the greatest range it can achieve without any input voltage fluctuations [27].

An Arduino due microcontroller is used to handle sensor data and is programmed with an ANN algorithm that can obtain five outputs based on sensor detection. If the front sensors detect no object, it can instruct the system to proceed forward. If all of the car's front sensors detect an object in front of it, the system will automatically brake the vehicle to bring it to a complete stop. However, when no object is detected behind the car, full braking is used to stop the vehicle, and soft braking is utilized when something is detected by the back sensors. The goal of this strategy is to avoid an accident caused by rapid brake shocks. Meanwhile, if only one front sensor detects an object in the left or right, the system will attempt to turn right or left to avoid it, assuming no other object is detected by the left-side or right-side sensors. However, if an item is detected, the system is programmed to apply the brakes and bring the vehicle to a complete stop [28]. Figure 5 shows a detailed description of the scheme.

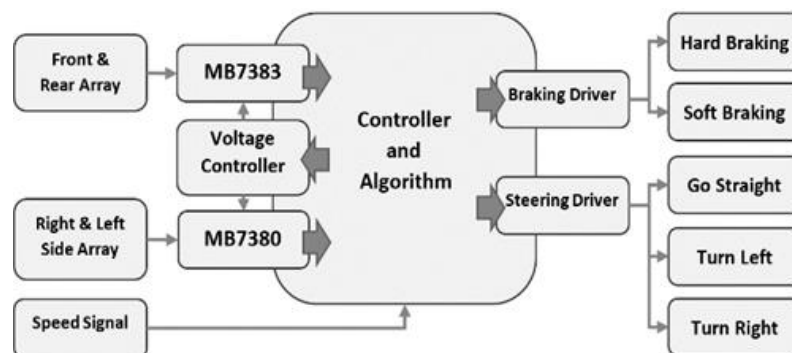


Figure 5. System configuration

To create a fully functional system, the system prototype is needed. On behalf of the vehicle that will be implemented in real sets, this study employs a length of 50 cm, a width of 75 cm, and a height of 50 cm of prototype with four wheels connected. Each side's sensors are located in the center. The distance between the front and rear sensors is around 25 cm. The sensors on the left and right are about 30 cm apart. This prototype also has a power source and a microprocessor in the center. Figure 6 depicts the system design of sensor placement as shown in Figure 6(a) and the prototype of the system as shown in Figure 6(b). Front and rear sensors are Maxbotix MB7383 HRXL that have 10 meters maximum range detection, and for side sensor Maxbotix MB7380 HXRRL are used, which its maximum detection range are five meters. The flow diagram of the system is shown in Figure 7.

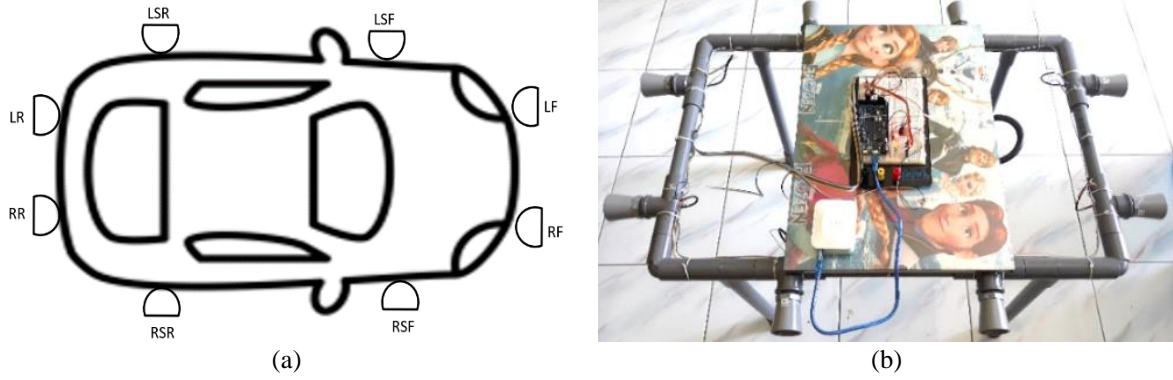


Figure 6. The system design (a) placement of sensors and (b) prototype of the system

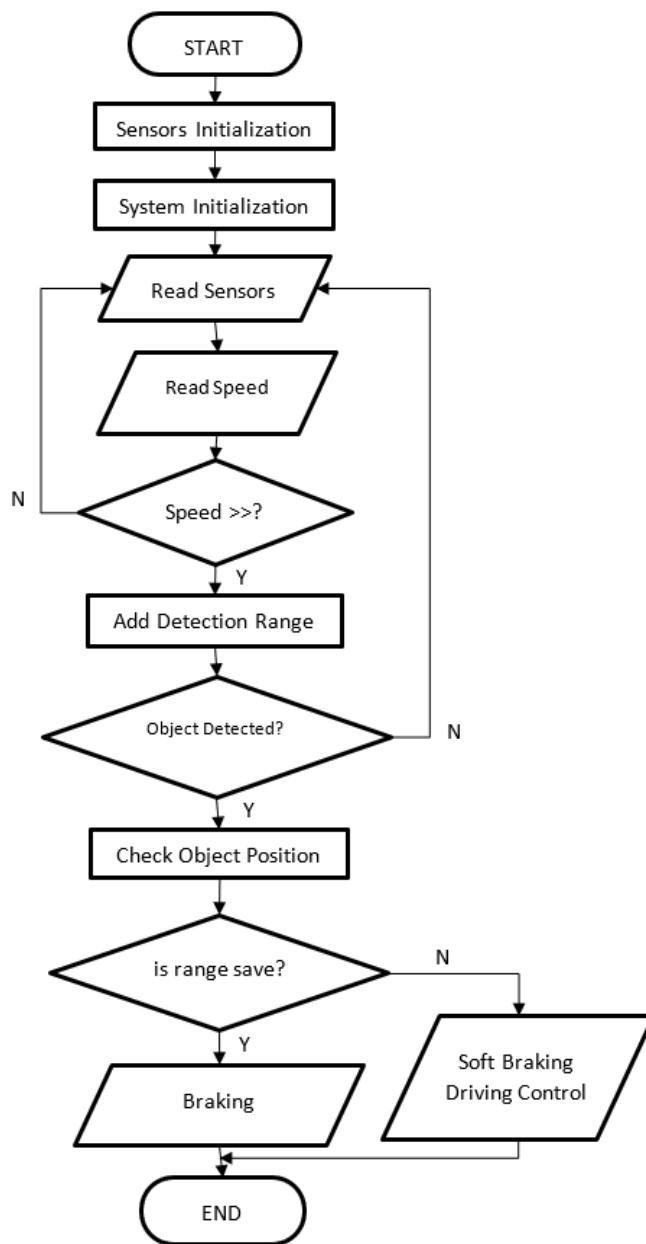


Figure 7. System flow chart

As shown in Figure 7, to acquire a thorough system setup and verify that the system is ready to run, the system begins with system initialization and sensor initialization. After all of the setting up has been completed correctly, the system is told to read data from all sensors. The system records the information and calculates the distance in millimeter. If no object is detected, the ultrasonic sensor sends maximum range detection data and if an object is identified, the sensor sends a shorter detection distance.

The technology also scans the vehicle’s speed. The higher speed, the higher voltage delivered to the sensor and extends the detection range. If an object is detected, the system will determine its location. Its objective is to carry out the braking, forwarding, or turning actions of the system. This action also considers the safety of other cars or itself by applying soft braking if there are any objects in the front and rear of vehicle.

3.2. The training

This study was done using the Simulink application. Its goal is to train the system’s neural network algorithm and to find the best network weights. In this training, there are numerous blocks that are meant to perform system performance [29]. The blocks are referred to in the Figure 5 diagram and then transformed to the Simulink diagram in Figure 8. The processed output is $Y(n)$, the desired output is $D(n)$, the weight between the input and the hidden layer is $Wx(n+1)$, and the weight between the hidden layer and the output is $Wy(n+1)$.

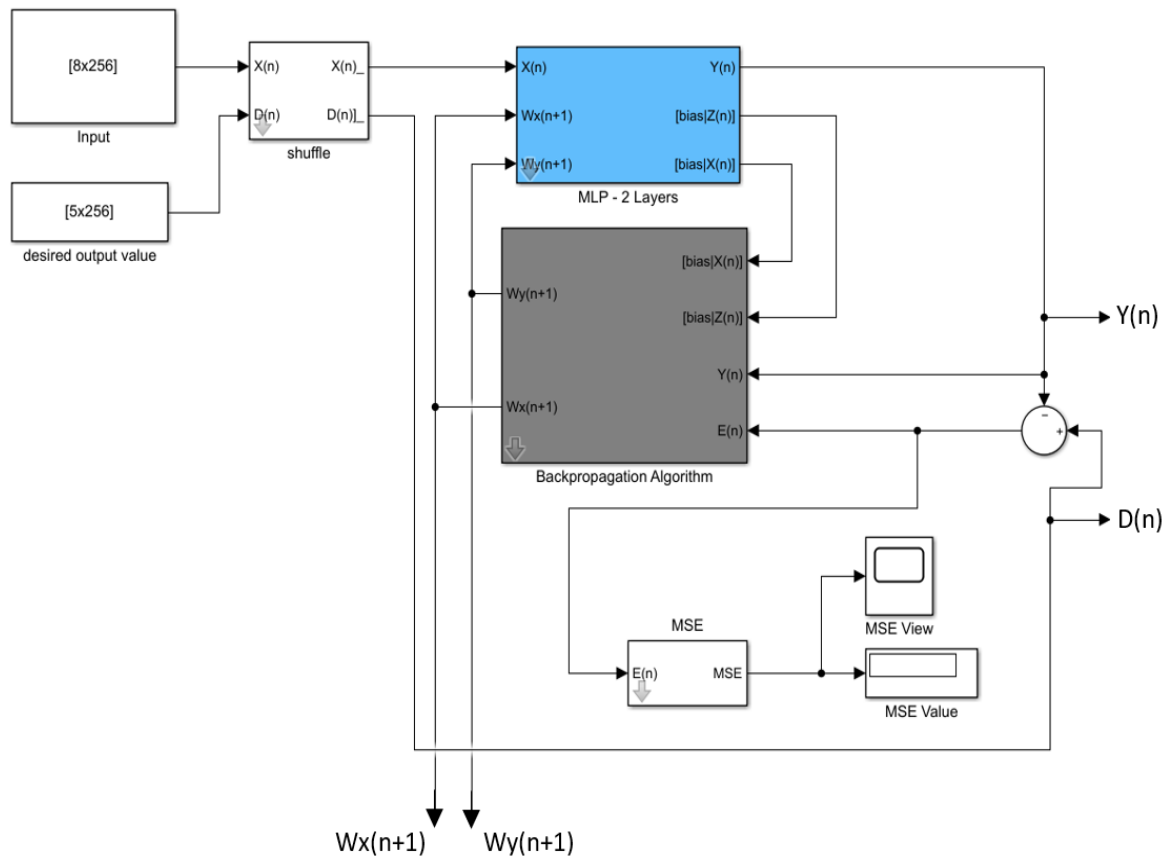


Figure 8. The configuration of offline training diagram

The diagram’s input is derived from the variation of sensor value after normalization using the “one hot encoding” approach, in which if the sensor detects an item, the value is noted as “1”, and if the sensor detects no object, the value is noted as “0”. The values of variation are determined by the quantity of inputs. Because there are eight inputs in this system, the input values will vary by 2^8 or 256 possibilities [30]. The sample of these various inputs are given in Table 1. The targets or desired outputs are assigned to five outputs in proportion to input changes, with each state containing only one action. As indicated in outputs section of Table 1, each action is marked with a “1” to indicate that it is active and a “0” to indicate that it is inactive.

Table 2 shows the neural network parameters, where mu is the weight correction value in the back-propagation process and alpha is the learning rate value. In addition, sixteen neurons are concealed in a layer. As the epoch of the neural network method, the simulation is configured to run for up to 100,000 iterations.

Table 1. Variation of inputs and desired targets samples

I/O POS	INPUTS								TARGETS				
	LF	RF	LR	RR	LS	LSR	RSF	RSR	S/B	SB	TL	TR	GO
SAMPLE	0	1	1	0	0	0	0	1	0	0	1	0	0
VALUE	1	1	1	0	0	0	0	1	0	1	0	0	0
	0	0	0	1	0	0	0	1	0	0	0	0	1
	1	0	0	1	0	0	0	1	0	1	0	0	0
	0	1	0	1	0	0	0	1	0	0	1	0	0
	1	1	0	1	0	0	0	1	0	1	0	0	0
	0	0	1	1	0	0	0	1	0	0	0	0	1
	1	0	1	1	0	0	0	1	0	1	0	0	0
	0	1	1	1	0	0	0	1	0	0	1	0	0
	1	1	1	1	0	0	0	1	0	1	0	0	0
	0	0	0	0	1	0	0	1	0	0	0	0	1
	1	0	0	0	1	0	0	1	1	0	0	0	0
	0	1	0	0	1	0	0	1	1	0	0	0	0
	1	1	0	0	1	0	0	1	1	0	0	0	0
	0	0	1	0	1	0	0	1	0	0	0	0	1
	1	0	1	0	1	0	0	1	0	1	0	0	0
	0	1	1	0	1	0	0	1	0	1	0	0	0
	1	1	1	0	1	0	0	1	0	1	0	0	0
	0	0	0	1	1	0	0	1	0	0	0	0	1
	1	0	0	1	1	0	0	1	0	1	0	0	0
	0	1	0	1	1	0	0	1	0	1	0	0	0
	1	1	0	1	1	0	0	1	0	1	0	0	0
	0	0	1	1	1	0	0	1	0	0	0	0	1
	1	0	1	1	1	0	0	1	0	1	0	0	0
	0	1	1	1	1	0	0	1	0	1	0	0	0
	1	1	1	1	1	0	0	1	0	1	0	0	0
	0	0	0	0	0	1	0	1	0	0	0	0	1
	1	0	0	0	0	1	0	1	1	0	0	0	0
	0	1	0	0	0	1	0	1	1	0	0	0	0

Table 2. Parameters of neural network

Parameters	Value
Training gain (mu)	0.75
Learning rate (alpha)	0.001
Training size	256
Number of inputs	8
Number of hidden neurons	16
Number of output neurons	5

4. RESULT AND DISCUSSION

Due to the extreme input generating procedure, the values of the tables appear randomly when the simulation started. When the optimum value or iteration is attained, the simulation will come to an end. It is important to show the output results during iteration process to know the training is running well. So, the 6,000th iteration is taken and can be compared to the last 100,000th iteration.

Table 3 displays the final producing outcomes for 6,000 iterations in about 10 seconds. The required target values and the processing output values are listed on two lines in the result. It can be shown that the outputs are almost similar to the targets but it is not exactly the same. While in Table 4, both data are exactly the same, indicating that the goal has been met in about 35 seconds for 100,000 iterations.

As shown in Table 5, the weight values are presented. They are the best values found by using the back-propagation method with a neural network algorithm. Between the input layer and the hidden layer, the weight range values are from -0.1510 to 0.4512. The significant value of the backward process is smaller, and the significant value of the forward process is higher. It signifies that a positive weight value increases output while a negative weight value reduces output. In the table, there is no value of zero. It means that the system is significantly affected by all weight values in the network [9]. It also occurs in the weight value between the hidden and output layers, which ranges are from -0.3880 to 0.6135, as shown in Table 6.

When in the 6000 iterations training process, the mean square error (MSE) is shown in Figure 9. The figure shows that the line is still unstable. This means that the output and target results have not converged enough, which is indicated by the difference in results as in the previous Table 3. Then the iteration continued to the higher to reach optimum results. The MSE is nearly zero in the final 100,000 iteration phase, as shown in Figure 10. It is around 0.0078, which means there's no significant difference between the output and the target.

Table 3. Comparison of result output to desired output (target) for 6000 iterations

OUTPUTS															
S/B	0	0	0	0	1.52E-293	0	1	0	0	0	1	0	0	1.02E-234	0
SB	1	1	0	1	1	0	0	1	1	1	1.06E-243	0	2.14E-23	2.18E-68	1
TL	2.18E-107	6.66E-78	3.50E-202	2.44E-156	1.91E-74	5.89E-83	2.04E-306	1.65E-29	2.44E-156	1.65E-29	1.84E-43	4.88E-201	2.17E-224	0.999588	1.65E-29
TR	4.62E-176	3.84E-188	5.13E-122	1.25E-145	1.13E-292	0	3.38E-219	0	1.25E-145	0	0	1.14E-113	1	0	0
GO	0	0	1	0	0	1	2.64E-282	1.68E-228	0	1.68E-228	0	0	0	0	1.68E-228
TARGETS															
S/B	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0
SB	1	1	0	1	1	0	0	1	1	1	0	0	0	0	1
TL	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
TR	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
GO	0	0	1	0	0	1	0	0	0	0	0	1	0	0	0

Table 4. Comparison of result output to desired output (target) for 100000 iterations

OUTPUTS																						
S/B	1	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	
SB	0	1	0	0	0	0	1	0	1	1	0	0	0	0	1	0	1	1	0	0	1	1
TL	0	0	1	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
TR	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
GO	0	0	0	0	0	0	1	0	0	0	1	1	1	0	0	0	0	1	1	0	0	1
TARGETS																						
S/B	1	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0
SB	0	1	0	0	0	0	1	0	1	1	0	0	0	0	1	0	1	1	0	0	1	1
TL	0	0	1	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
TR	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
GO	0	0	0	0	0	0	1	0	0	0	1	1	1	0	0	0	0	1	1	0	0	1

Table 5. The weight values between input layer and hidden layer

Wx	1	2	3	4	5	6	7	8	b
1	0.1666	-0.032	-0.151	0.3371	0.3372	0.0915	0.1179	-0.02	-0.02
2	0.272	0.2346	-0.074	0.1456	0.1948	0.1883	0.1347	0.1547	0.1327
3	0.1435	0.1221	-0.017	0.1228	0.1641	0.0917	-0.011	-0.015	0.0366
4	0.2086	0.2498	0.2079	0.1711	0.0421	0.0891	0.0295	0.1716	0.1826
5	0.2743	0.3139	0.213	0.0732	-0.012	0.1197	0.048	0.0274	-0.056
6	-0.072	-0.139	0.0478	0.0712	0.0811	0.0439	0.1392	0.1745	-0.015
7	0.2724	0.4512	0.2726	-0.057	-0.0003	0.1282	0.1286	-0.054	-0.046
8	0.0789	0.2098	-0.093	0.0418	0.0084	0.226	0.2039	0.0517	0.0288
9	0.4054	0.0572	0.142	0.4275	0.4021	0.0252	0.0389	0.0745	0.0928
10	0.3097	0.0743	0.0018	0.0245	0.0055	0.1432	0.1398	0.1921	0.1399
11	0.1928	0.2716	0.2057	0.0449	0.1032	-0.005	0.0464	-0.05	0.0477
12	0.3584	0.2055	0.3268	0.0362	0.0397	0.0199	0.0571	0.19	0.1049
13	0.2971	0.1698	0.0448	0.1249	-0.012	0.1082	0.17	0.1521	0.1706
14	0.1396	-0.016	0.157	0.034	0.0393	0.1562	0.1464	0.1966	0.156
15	0.3249	0.2728	0.2168	0.0402	0.0413	0.1298	0.0668	-0.007	0.0331
16	0.2112	0.0983	0.319	0.019	0.0082	-0.005	-0.021	0.1728	0.1922

Table 6. The weight values between hidden layer and output layer

Wy	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	b
1	0.2982	-0.34	-0.044	-0.128	-0.033	0.0167	-0.167	0.1579	0.0938	-0.373	0.0915	0.0546	0.0437	0.0913	0.1356	0.0749	0.0793
2	0.6135	0.1681	-0.018	0.0102	-0.115	0.0345	-0.206	0.1872	0.0082	0.3531	0.0616	0.0102	0.2827	-0.042	-0.158	0.1108	0.1652
3	0.1534	-0.1	-0.113	-0.039	0.1361	-0.075	0.0885	-0.125	-0.23	0.1284	-0.042	0.0132	0.1468	-0.097	-0.043	0.0313	0.1091
4	0.208	0.1077	0.0859	-0.037	0.0643	0.1107	-0.06	0.1403	0.0928	-0.021	-0.066	0.0929	-0.06	-0.044	-0.179	0.0249	-0.267
5	-0.07	0.2891	0.0368	0.0038	-0.09	-0.294	0.2554	-0.388	0.146	-0.087	0.0809	-0.162	-0.178	-0.046	0.0363	-0.228	-0.131

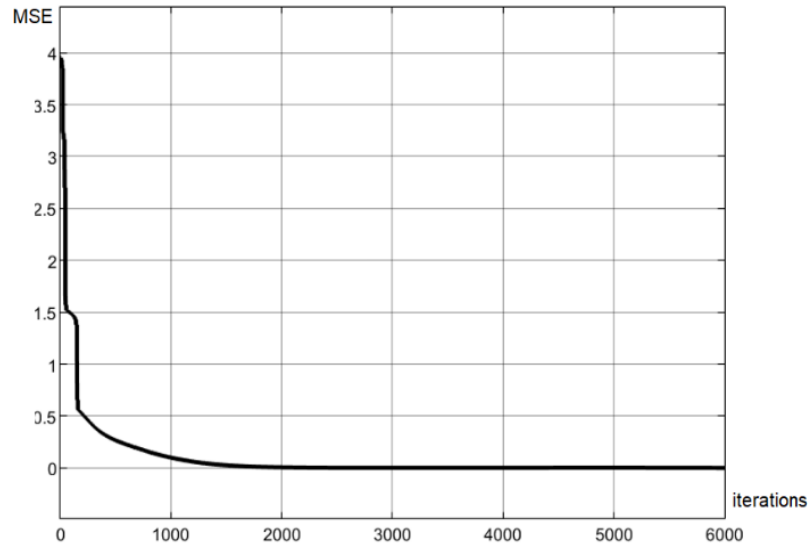


Figure 9. The MSE result of 6000 iterations

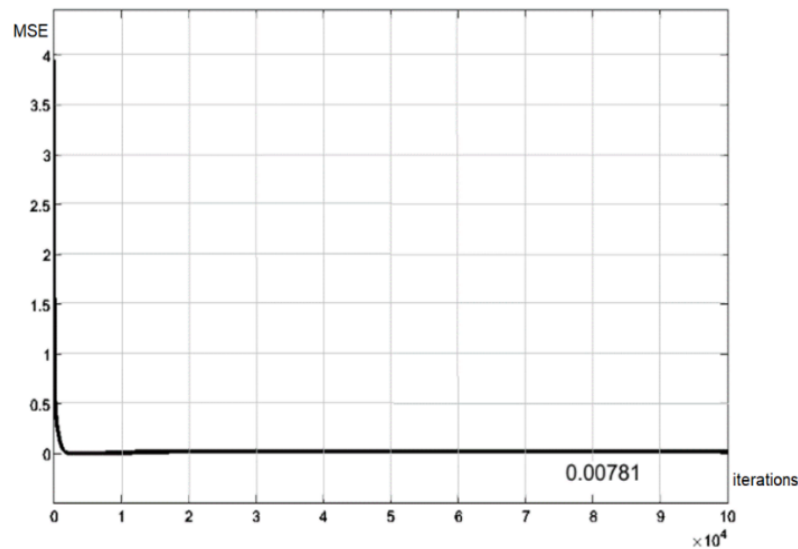


Figure 10. The MSE result of 10000 iterations

5. CONCLUSION

The offline training was successfully conducted by implementing an artificial neural network algorithm using the back-propagation method. The inputs of the training represent the sensors of the autonomous vehicle arranged in multisensory circuits. The training processes eight neuron nodes in input layer, sixteen neuron nodes in hidden layer, and five neuron nodes in output layer. The output layer represents the action taken by the vehicle as the reaction of processed signal of the sensors. In the end of training, the processed outputs and the desired targets have similar values, indicating that the system operation is working properly with minimum error and convergency of output and target. It means, the network weights resulted in this training become the best weights and can be taken as the fixed-weight for the online training.

In the 6,000th iteration or lower, the results are not convergent already. And in iteration of 100,000, there are no differences between output and target, that means the convergencies are reached. The iteration process of 100,000 epochs yielded a weight value of -0.15 to 0.61 with an MSE of 0.0078 in about 35 minutes. In future studies, conducting online training will be advantageous. The weights of the network can be used to be implemented as fixed-weight neural network or as known as adaptive neural network. By this

method the neural network implementation will be faster process, because the neural network weights have been initialed in offline training. The network' fixed-weights will be set as arrays and embedded in Arduino Due system to process the inputs from the sensors. The process of neural network in it will obtain the outputs in line to desired target due to the fixed-weight values pinned.

ACKNOWLEDGEMENT

This project is collaboration research in autonomous vehicle between Universitas Islam Sultan Agung Semarang and Universitas Sriwijaya Palembang.

REFERENCES




- [1] Q. Long, Q. Xie, S. Mita, K. Ishimaru, and N. Shirai, "A real-time dense stereo matching method for critical environment sensing in autonomous driving," in *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, Oct. 2014, pp. 853–860, doi: 10.1109/ITSC.2014.6957796.
- [2] T. Kim and T. Park, "Calibration method between dual 3D lidar sensors for autonomous vehicles," in *56th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE)*, 2017, pp. 1075–1081, doi: 10.23919/SICE.2017.8105583.
- [3] S.-J. Babak, S. A. Hussain, B. Karakas, and S. Cetin, "Control of autonomous ground vehicles: a brief technical review," *IOP Conference Series: Materials Science and Engineering*, vol. 224, no. 1, Jul. 2017, doi: 10.1088/1757-899X/224/1/012029.
- [4] D. J. Pradeep, P. Venugopal, and N. R. Pankaj, "Adaptive cruise control using ultrasonic sensor," *International Journal of Pharmacy and Technology*, vol. 8, no. 3, pp. 15949–15960, 2016.
- [5] G. Farias *et al.*, "A neural network approach for building an obstacle detection model by fusion of proximity sensors data," *Sensors*, vol. 18, no. 3, Feb. 2018, doi: 10.3390/s18030683.
- [6] T. A. Andreeva and W. W. Durgin, "Wind tunnel investigation of sound attenuation in turbulent flow," *Ultrasonics*, vol. 61, pp. 15–19, Aug. 2015, doi: 10.1016/j.ultras.2015.03.008.
- [7] A. Medina-Santiago, J. L. Camas-Anzueto, J. A. Vazquez-Feijoo, H. R. Hernández-de León, and R. Mota-Grajales, "Neural control system in obstacle avoidance in mobile robots using ultrasonic sensors," *Journal of Applied Research and Technology*, vol. 12, no. 1, pp. 104–110, Feb. 2014, doi: 10.1016/S1665-6423(14)71610-4.
- [8] M. De Simone, Z. Rivera, and D. Guida, "Obstacle avoidance system for unmanned ground vehicles by using ultrasonic sensors," *Machines*, vol. 6, no. 2, Apr. 2018, doi: 10.3390/machines6020018.
- [9] G. Ognjanovski, "Everything you need to know about neural networks and backpropagation-machine learning easy and fun," *towardsdatascience*. pp. 1–5, 2019.
- [10] F. Zhang, C. M. Martinez, D. Clarke, D. Cao, and A. Knoll, "Neural network based uncertainty prediction for autonomous vehicle application," *Frontiers in Neurorobotics*, vol. 13, pp. 1–17, May 2019, doi: 10.3389/fnbot.2019.00012.
- [11] S. Afaq and S. Rao, "Significance of epochs on training a neural network," *International Journal of Scientific and Technology Research*, vol. 19, no. 6, pp. 485–488, 2020.
- [12] H. Kanan, "Reduction of neural network training time using an adaptive fuzzy approach in real time applications," *International Journal of Information and Electronics Engineering*, 2012, doi: 10.7763/IJIEE.2012.V2.140.
- [13] N. O. Hodas and P. Stinis, "Doing the impossible: why neural networks can be trained at all," *Frontiers in Psychology*, vol. 9, pp. 1–7, Jul. 2018, doi: 10.3389/fpsyg.2018.01185.
- [14] J. Tan, D. Xia, S. Dong, B. Xu, and Y. Li, "Research on the solution of BP neural network training problem," in *Proceedings of the 2018 2nd International Conference on Artificial Intelligence: Technologies and Applications (ICAITA 2018)*, 2018, vol. 146, pp. 41–45, doi: 10.2991/icaita-18.2018.11.
- [15] S. Sugathan, B. V. S. Shree, M. R. Warriar, and C. M. Vidhyapathi, "Collision avoidance using neural networks," *IOP Conference Series: Materials Science and Engineering*, vol. 263, no. 5, Nov. 2017, doi: 10.1088/1757-899X/263/5/052041.
- [16] Maxbotix, "HRXL-MaxSonar®-WR™ Series," 2012. https://www.maxbotix.com/ultrasonic_sensors/mb7383.htm (accessed Aug. 18, 2021).
- [17] Maxbotix, "HRLV-MaxSonar®-EZ™ Series," 2014. https://www.maxbotix.com/ultrasonic_sensors/mb1043.htm (accessed Sep. 05, 2021).
- [18] E. N. Budisusila, M. Khosyi'In, S. A. D. Prasetyowati, B. Y. Suprpto, and Z. Nawawi, "Ultrasonic multi-sensor detection patterns on autonomous vehicles using data stream method," *International Conference on Electrical Engineering, Computer Science and Informatics (EECSI)*, vol. 2021, pp. 144–150, 2021, doi: 10.23919/EECSI53397.2021.9624313.
- [19] W. Bruce and E. V. O. N. Otter, "Artificial neural network autonomous vehicle artificial neural network controlled vehicle," KTH Royal Institute of Technology, School of Industrial Engineering and Management, 2016.
- [20] C. Tirnăuică, J. L. Montaña, and Z. Mediavilla, "Neural networks in autonomous driving," *AAAI Spring Symposium-Technical Report*, pp. 520–523, 2017.
- [21] P. Baldi, P. Sadowski, and Z. Lu, "Learning in the machine: Random backpropagation and the deep learning channel," *Artificial Intelligence*, vol. 260, no. 1, pp. 1–35, Jul. 2018, doi: 10.1016/j.artint.2018.03.003.
- [22] G. A. Santiago and M. Favre, "Analysis and evaluation of recurrent neural networks in autonomous vehicles," KTH Royal Institute of Technology, School of Industrial Engineering and Management, 2017.
- [23] Y. Pan, C.-A. Cheng, K. Saigol, and K. Lee, "Learning deep neural network control policies for agile off-road autonomous driving," in *31st Conference on Neural Information Processing Systems (NIPS 2017)*, 2017, pp. 1–13.
- [24] A. Budianto *et al.*, "Analysis of artificial intelligence application using back propagation neural network and fuzzy logic controller on wall-following autonomous mobile robot," in *2017 International Symposium on Electronics and Smart Devices (ISESD)*, Oct. 2017, no. 1, pp. 62–66, doi: 10.1109/ISESD.2017.8253306.
- [25] J. Kocic, N. Jovicic, and V. Drndarevi, "An end-to-end deep neural network for autonomous driving designed for embedded automotive platforms," *Sensors*, 2019, doi: 10.3390/s19092064.
- [26] E. N. Budisusila, S. A. D. Prasetyowati, B. Y. Suprpto, and Z. Nawawi, "Review and design of environmental smart detector for autonomous vehicle in urban traffic," in *AIP Conference Proceedings*, 2019, vol. 2173, doi: 10.1063/1.5133923.
- [27] R. Gutierrez-Osuna, J. A. Janet, and R. C. Luo, "Modeling of ultrasonic range sensors for localization of autonomous mobile robots," *IEEE Transactions on Industrial Electronics*, vol. 45, no. 4, pp. 654–662, 1998, doi: 10.1109/41.704895.
- [28] E. N. Budisusila, B. Arifin, S. A. D. Prasetyowati, B. Y. Suprpto, and Z. Nawawi, "Artificial neural network algorithm for

Neural network training for serial multisensor of autonomous vehicle system (Eka Nuryanto Budisusila)




- autonomous vehicle ultrasonic multi-sensor system,” in *2020 10th Electrical Power, Electronics, Communications, Controls and Informatics Seminar (EECCIS)*, Aug. 2020, pp. 128–131, doi: 10.1109/EECCIS49483.2020.9263459.
- [29] D. Nichols, “Arduino-based data acquisition into excel, LabVIEW, and MATLAB,” *The Physics Teacher*, vol. 55, no. 4, pp. 226–227, Apr. 2017, doi: 10.1119/1.4978720.
- [30] D. Erwanto, S. A. D. Prasetyowati, and E. N. B. Susila, “Utilization of digital image processing in process of quality control of the primary packaging of drug using color normalization method,” *Proceeding of the Electrical Engineering Computer Science and Informatics*, vol. 3, no. 1, Dec. 2016, doi: 10.11591/eecsi.v3.1118.

BIOGRAPHIES OF AUTHORS






Eka Nuryanto Budisusila    was born on October 19th 1973 in Teluk Cendrawasih, Irian Jaya (West Papua, Indonesia). He is a graduate of Universitas Muhammadiyah Malang (UMM) Indonesia in major of electrical engineering. His post graduate program is in Electrical Engineering of Universitas Brawijaya Malang Indonesia. And in recent year he takes the doctoral program of Electrical Engineering in Universitas Sriwijaya Palembang Indonesia. He is an academic staff of Electrical Engineering of Universitas Islam Sultan Agung Semarang as a lecture. He concerns in major of sensors and actuators, medical instrumentation, and digital image processing. He can be contacted at email: ekanbs@unissula.ac.id.






Sri Arttini Dwi Prasetyowati    was born on February 20th 1965 in Yogyakarta (Indonesia). She is a graduate of Gadjah Mada University (UGM) of Yogyakarta, Indonesia in major of mathematics, and also her post graduate program was taken there. Her doctoral program is in Electrical Engineering of Gadjah Mada University of Yogyakarta, Indonesia. She is an academic staff of Electrical Engineering of Sultan Agung Islamic University of Semarang as a lecture. She concerns in major of signal processing. She can be contacted at email: arttini@unissula.ac.id.



Bhakti Yudho Suprpto    was born on February 11th 1975 in Palembang (South Sumatra, Indonesia). He is a graduate of Sriwijaya University of Palembang in major of electrical engineering. His post graduate and doctoral program is in Electrical Engineering of Indonesia University (UI) of Jakarta. He is an academic staff of Electrical Engineering of Sriwijaya University of Palembang as a lecture. He concerns in major of control and intelligent system. He can be contacted at email: bhakti@ft.unsri.ac.id.



Zainuddin Nawawi    was born on March 3rd 1959 in Lubuklinggau (South Sumatra, Indonesia). He is a graduate of Sriwijaya University of Palembang in major of electrical engineering. His post graduate and doctoral program is in Electrical Engineering of Technology University of Malaysia (UTM). He is an academic staff of Electrical Engineering of Sriwijaya University of Palembang Indonesia as a lecture. He concerns in major of electrical power and protection material. He is one of Electrical Engineering Professor in Sriwijaya University. He can be contacted at email: nawawi_z@unsri.ac.id.