sciendo

VOLUME 18, N° 1 2024
Journal of Automation, Mobile Robotics and Intelligent Systems

Application of Multilayer Neural Networks for Controlling a Line-Following Robot in Robotic Competitions

Submitted: 6th September 2023; accepted: 27th October 2023

Cesar Minaya, Ricardo Rosero, Marcelo Zambrano, Pablo Catota

DOI: 10.14313/JAMRIS/1-2024/4

Abstract:
The paper presents an approach for controlling a line-following robot using artificial intelligence algorithms. This study aims to evaluate and validate the design and implementation of a competitive line-following robot based on multilayer neural networks for controlling the torque on the wheels and regulating the movements. The configuration of the line-following robot consists of a chassis with a set of infrared sensors that can detect the line on the track and provide input data to the neural network. The performance of the line-following robot on a running track with different configurations is then evaluated. The results show that the line-following robot responded more efficiently with an artificial neural network control algorithm than with a PID control or fuzzy control algorithm. At the same time, the reaction and correction time of the robot to errors on the track is earlier by about 0.1 seconds. In conclusion, the capabilities of a neural network allow the line-following robot to adapt to environmental conditions and overcome obstacles on the track more effectively.

Keywords: *Robotics, Line-following robot, Artificial neural networks*

1. Introduction

Autonomous line-following robots in the last decades have been of increasing interest for their involvement in various fields ranging from industry to healthcare, education, logistics, transportation, and robotic competitions [1, 2]. Nowadays, robot development focuses on achieving high precision, speed, and stability levels. Intelligent mobile robots combine control engineering, electronics, computer science, software, and mechanics.

An autonomous line-following robot recognizes and follows a path traced by a black line on a flat white surface. The control system detects the line and regulates the robot to keep it on its path while constantly correcting for deviations [3]. These robots are often implemented in academic settings, such as teaching techniques in robotics, control systems, or artificial intelligence [4]. The architecture shared by most line-following robots includes a chassis, line detection system, locomotion system, and control unit. Various sensors can detect these black lines describing the traced route for this—these range from low-cost detection modules to expansive vision systems [5].

Numerous studies [6–8] have highlighted the usefulness of infrared sensors for line detection systems. These are located on the underside of the robot base and emit a beam of infrared light, which allows us to detect the amount of infrared light reflected from the ground surface. The main reason for choosing this sensor is its range for line detection from a minimum of 100cm to a maximum of 500cm. In addition, they consume low power and can be placed in small spaces [9]. However, several studies [10–12] have shown that the use of cameras can be an alternative for line detection by capturing images and describing the ground environment while sending them to an image processing system to detect the line. However, in the last decades, authors have experimented with new techniques such as color segmentation, edge detection, or more advanced methods with convolutional neural networks to identify and separate the background line and allow more accurate tracking [13].

The robot control system allows monitoring and taking action on the collected data or determining what action the robot should take to stay on the line, such as adjusting direction or speed. In [14], the authors describe the implementation of a field programmable gate array known as FPGA in the control system of the line-following robot to develop the sensor data processing and control algorithm efficiently. In addition, the FPGA can be easily reprogrammed and adjusted to suit different scenarios or specific requirements of the line-following robot. Most of these FPGA implementations are task-oriented for the entire robotic system or are used for particular applications [15].

Several authors have investigated algorithms applied to the control of a line-following robot. Kader et al. have tried to explain the application of a PID control algorithm to correct the current error between the robot position and the traced line by calculating a control signal that rectifies the robot trajectory in real time [16]. On the other hand, Nikolov et al. highlight the need to apply a histogram filtering of the Markov process effectively to the velocity and length measurements, thus mitigating the current position error [17]. In a different study, Wu et al. highlight the implementation of a new fuzzy sliding mode controller and backtracking algorithm for trajectory tracking.

This backtracking control technique eliminates pose deviations of the robot based on its mathematical model [18]. Recently, an intelligent technique for robot speed control using a combination of fuzzy logic and supervised machine learning has been proposed using numerical simulations [19]. However, there is little progress in the discussion of intelligent control tools. Therefore, an investigation focused on using artificial intelligence algorithms for the control and performance of a line-following robot is relevant.

This study aims to evaluate and validate the design and implementation of a line-following robot based on neural networks to control the torque on the wheels and regulate the movements. The line-following robot is an autonomous guided vehicle (AVG) that follows a trajectory determined by a black or white line. Using a set of analog reflectance sensors incorporated in the competition robot, it can detect the line on the track, and based on the values acquired by the controller, the neural network incorporated in the programming will interpret these signals and set the best speed parameters to follow the trajectory in a straight line or the curves, in order to guarantee the best performance of the robot on the track.

The article is organized into six sections: Section 1 Introduction, Section 2 Line-following robot architecture, Section 3 Implementation of the neural control network, Section 4 Tests, Section 5 Results, and Section 6 Conclusions.

2. Line-following Robot Architecture

In this section, we describe the architecture of the line-following robot that consists of several essential components which work together to achieve its functionality. Figure 1 provides information on the critical elements of the line-following robot structure. These elements are systematic and complement each other; in this figure, it can be seen that it consists of seven blocks that can vary according to their application [20].

The first element focuses on the environment in which it is immersed. Then, there is the second element that incorporates the physical components in charge of capturing the signals of the variables. These signals are then directed to the third element, which consists of a control board in charge of interpreting them and issuing corresponding actions.

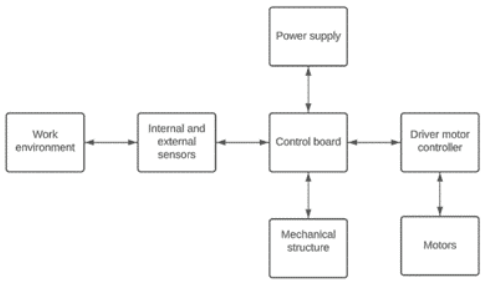

Figure 1. Block diagram of the operation of the line-following robot

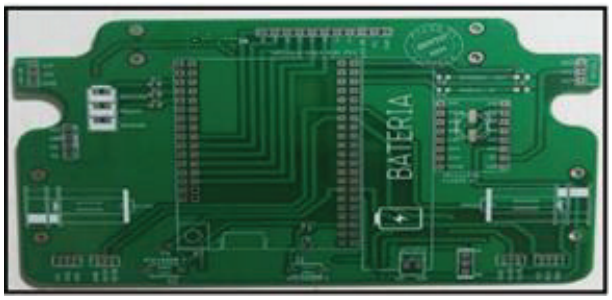

Figure 2. Robot chassis construction – PCB

It then moves on to the fourth element, representing the point of control interaction in synchronization with the motors in the fifth element. The sixth element covers the general power supply of the robot. Finally, the seventh element encompasses the mechanical structure that holds all the electrical and mechanical components of the robot [21].

2.1. Embedded Line-following Robot Platform

The following parts were used in the construction of the robot: 2 wheels, 2 DC motors, a base structure, a control board consisting of a microcontroller, a motor control circuit, a line follower module, a Bluetooth connection module, and a power supply circuit. The locomotion used for its construction is of differential type. For this reason, it is essential to consider particularities such as the robot's chassis, the sensors' dimensioning concerning the chassis, and the dimensioning of the motor-res [22].

The chassis is the physical structure that supports all the elements of the robot. For its design, the implementation of a printed circuit board (PCB) is considered, where the electrical schematic that shows all the components and their connections with each other is integrated. The schematic includes the sensors, motor controller, microcontroller, communication, and power supply. The structure's design is visualized in Figure 2, where the primary consideration of the chassis design is the need for a structure that ensures a solid, functional base that can accommodate all components, ensuring smooth and precise movements along the route. It is essential to consider the weight of the chassis to avoid excessive energy consumption and flexibility to improve robot performance on different surfaces and maintain traction [23].

2.2. Control Unit

Microcontrollers are very important in constructing the line-following robot because they help monitor, control, and take action with the data obtained. These devices integrate a single chip's central processing unit, memory, and peripherals. The most common controllers are Arduino, Raspberry Pi, and PIC. Thanks to microcontrollers, robots can perform various tasks, from controlling basic movements to executing more complex functions in changing environments [24].

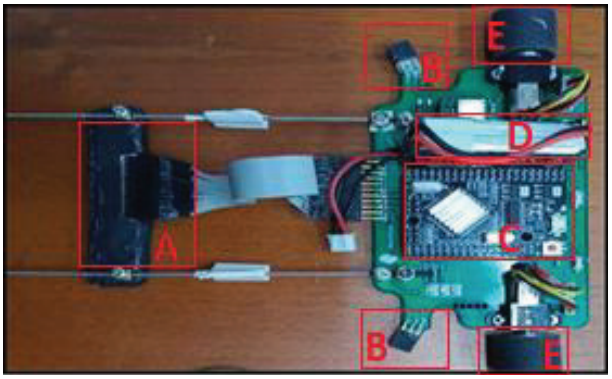

Figure 3. Microcontroller and robot platform

In this study, the Arduino Mega 2560 microcontroller forms the basis of the line follower robot, which receives the signals from each connected component and consequently provides the desired output. It is the brain of the entire system and is coded as required. The microcontroller interprets the data received by the sensors and generates control commands that drive the robot's motors. These commands allow adjusting the speed and direction of the movement to keep the robot following the line precisely; Figure 3 shows the connection of the elements connected to the microcontroller [25]. Figure 3 also shows the components of the robot: QTR-8A analog reflectance sensor (A), the QTR-1RC Reflectance Sensor (B), Arduino Mega 2560 (C), Battery (D), and Wheels (E).

2.3. Lane Detection System

The lane detection system of a line-following robot is critical to its correct operation. This system is based on optical sensors, such as phototransistors or reflection sensors, which can detect the difference between reflectivity between the line and the background [26]. In this research, the QTR-8A analog reflectance sensor was implemented for the track detection system. This electronic device has eight infrared sensors, i.e. a phototransistor LED mounted 9.5 mm from each other, allowing a more extended range when detecting the traced route. The selected sensor belongs to the type of exteroceptive sensor that detects changes in the robot's external environment. It has an LED that emits radiation in the infrared spectrum, which hits the ground and causes a reflection, which is captured by the phototransistor; the amount of reflection depends on the color of the ground [27]. To determine the line's position and generate control commands, a structure was fabricated, as shown in Figure 4. with the analog reflectance sensor QTR-8A (A), which allows the robot to follow the line precisely and continuously.

2.4. Locomotion System

This section mentions the dimensioning of the engines and their electronic control system. The motors are responsible for the correct displacement of the robot on the track, so it is essential to carry out correct dimensioning, taking into account the motor torque that the robot needs for each wheel.

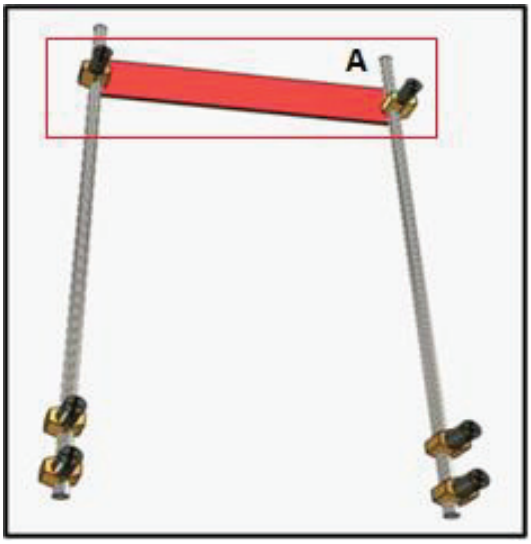

Figure 4. Lane detection system bracket design

The behavior of the torque about the wheels is directly proportional, i.e. a high torque is needed when the radius of the wheels is large, thus reducing the revolutions of the wheel and, therefore, the speed of the robot; on the other hand, if the motor torque is small and the radius of the wheels is small, the revolutions of the wheel would be faster. Consequently, the speed of the robot would increase, which in our case, is ideal for looking for a small torque [28]. Equation (1) can be used to calculate the required motor torque; the acceleration to be achieved is a matter of judgment.

$$T = M \cdot (a + g \cdot \sin(\theta)) \cdot r \tag{1}$$

Where:

T: Torque of the motor.

M: total Mass of the robot.

a: Acceleration.

θ: Angle of the plane.

g: Gravity.

r: Radius of the wheels

The data we have are the mass of the robot and its components, which is equal to 170 grams; this mass was obtained by weighing the robot on a scale. In addition to this, we are looking to manage an acceleration around 2 m/s^2, and the radius of the wheels to be implemented 1 (cm). Finally, the angle of the track is zero. From Equation (1), it is obtained that the required torque is equivalent to 0.0042 Nm. To control the motors that allow the movement of the line follower robot in a more precise, more efficient way in the direction and speed of the motors. Finally, a controller was selected TB6612FNG because of its dimensions and its applications in similar studies [29] were chosen for this case study.

3. Neural Network for Line-following Robot

The controller of the line-following robot is a fundamental part of its correct operation; for this reason, a multilayer neural network is used to improve the accuracy of decision-making.

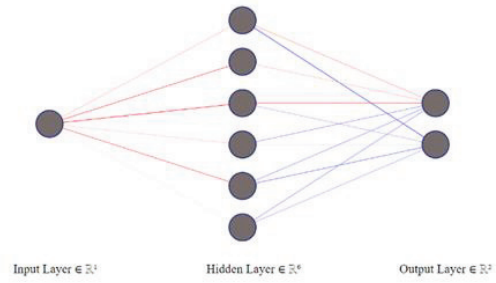

Figure 5. Neural network architecture implemented

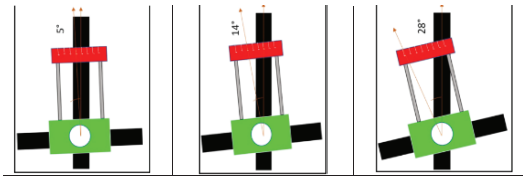

Figure 6. Position of the robot on the lane

The neural network predicts the position of the robot on the route and provides a control signal that allows the robot to control the movement of the motors [30]. The structure of the implemented neural network is visualized in Figure 5. This network was constructed using multiple hidden layers. The input layer consists of one neuron, a hidden layer of six neurons, and an output layer of two neurons. Each layer of the neural network receives a value of the loss function in the current state, and the connection weight of each neuron is adjusted accordingly.

3.1. Implementation of the Neural Network Controller

This section describes input data collection to the neural network and the output data. Figure 6. shows how the input data X are obtained using the sensors in each position where the robot can be on the line so that the neural network can respond correctly to any situation and make decisions to maintain the trajectory that describes the route. The data obtained from the QTR-8A analog reflectance sensors are fed to the input layer of the multilayer neural network.

The output data y corresponds to the PWM value required in the motors, which depends on the position of the robot on the line and tends to change constantly.

Some input data and output data are shown in (2).

$$X = \begin{bmatrix} 0 \\ 50 \\ 100 \\ 150 \\ 200 \\ 300 \end{bmatrix} \tag{2a}$$

$$y = \begin{bmatrix} 185 & 190 \\ 190 & 195 \\ 195 & 200 \\ 200 & 205 \\ 205 & 210 \\ 210 & 215 \end{bmatrix} \tag{2b}$$

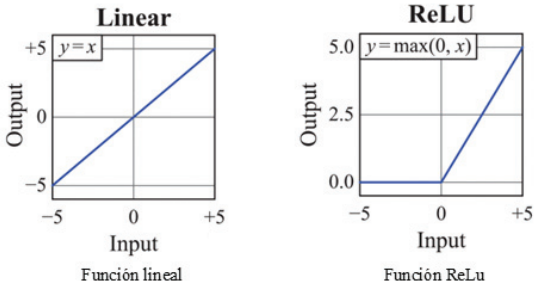

Figure 7. Activation functions

Table 1. Hyperparameters of the prediction model

Repetitions	**2500**
Number of hidden layers	6
Learning rate	0.002

The extraction of weights and biases are fundamental components in the neural network. These values allow the behavior to be adjusted and represent more complex non-linear functions.

3.2. Activation Function

The activation function makes the non-linear relationship between the input and the output more effective. Different activation functions can be used for the different neural network layers [31]; the most commonly used options are shown in Figure 7. Rectified Linear Unit (ReLU) neurons are used for the hidden layers; in machine learning, ReLU and Linear are the most popular activation functions, expressed in Figure 7. In our study, the ReLU activation function passes the information from the input layer to the hidden layer, with the peculiarity that the negative values are canceled, letting the positive values pass without modifying or canceling them. The linear activation function is used at the neural network's output; this has the characteristic of letting the values it has at its input pass through without modifying them.

3.3. Model Evaluation

The learning of the neural network was accomplished off-line by scanning data in the work environment by Keras library in Python. The full dataset was used to train the proposed neural network, and the performance of the network was determined for that same dataset. The evaluation of the implemented prediction model was performed by changing the training parameters and epochs in order to achieve satisfactory results. The hyperparameters used in this model can be visualized in Table 1. The loss function is a measure that evaluates how well the model makes predictions based on the predicted outputs and the actual outputs. Our study used the mean absolute error loss function and obtained a low error but with many interactions or epochs. The SDG (Stochastic Downward Gradient) optimizer was used to reduce the error and the number of interactions or epochs.

Additionally, a mean absolute error regression metric was used, which did not minimize the error but served to evaluate the training results. The optimal network structure is created by comparing the loss function values over the generated samples.

4. Test

Once the assembly of the robot has been completed and it is working correctly, the software and hardware part of the robot is tested. Once the assembly of the robot has been completed and the software and hardware are working correctly, the necessary tests are carried out to evaluate its operation, for which the track used in the SUCREBOT 2022 robotics competition, which can be seen in Figure 8, is taken as a reference.

This item also explains the modifications and improvements made to solve errors when the line follower robot was on track. This is the result of different participation in robotics competitions.

The first tests carried out on the robot were the use of a printed circuit board of drilling technology, with a thickness of 1 mm, which increased the weight of the robot. At this stage, the final prototype was also modified using surface mount technology with a thickness of 0.8 mm. The following tests were carried out about the adherence of the line-following robot on the track. Therefore, the width of the wheels was changed from 2 cm and 3.7 cm, respectively. Finally, another essential point to observe the correct operation of the robot on the track was to consider the distance the sensors should have concerning the chassis, for which tests were carried out with different distances of the sensors ranging from 6.8 cm to 10.5 cm. Figure 9 shows the final prototype of the line-following robot.

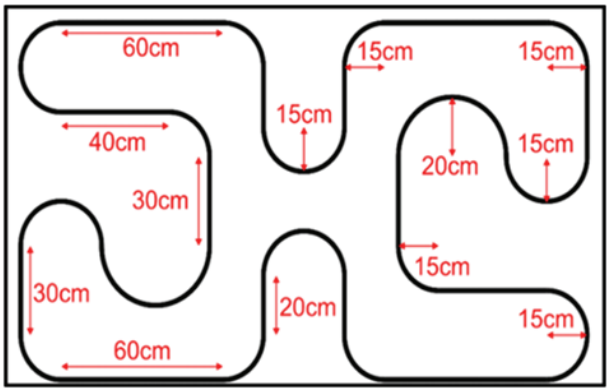

Figure 8. Line-following robot lane dimensions

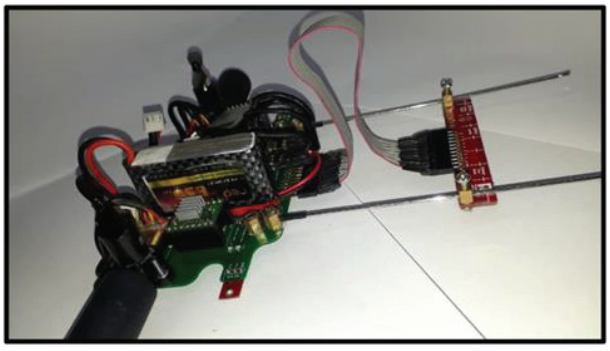

Figure 9. Final prototype of the line-following robot

5. Results and Discussions

5.1. Model Training

During the neural network training, the best performance achieved was at epoch 2500 with an absolute mean square error of 2.2355. The training performance plot is shown in Figure 10.

The optimized weight and bias matrixes obtained during the training process are shown in (3).

$$w_{HI} = \begin{bmatrix} -2,321 \\ 0,384 \\ -0.105 \\ -2.151 \\ -7.617 \\ -0.431 \end{bmatrix} \tag{3a}$$

$$w_{OH} = \begin{bmatrix} 7,128 & 9,884 & 0,032 & 2,260 \\ 7,780 & 9,164 & -0,714 & 3,822 \\ -5,152 & 2,493 \\ -5,363 & 1,372 \end{bmatrix} \tag{3b}$$

$$b_H = [10,327 \quad 13,463 \quad -0,171 \quad 3,898 \quad -0,084 \quad 2,701] \tag{3c}$$

$$b_O = [2,756 \quad 2,757] \tag{3d}$$

where w_{HI} is the weight vector for the weights from the input to the hidden layer, w_{OH} is the weight vector for the weights from the hidden layer to the output layer, b_H is the bias vector from the input to the hidden layer, and b_O is the vector from input bias to pa output layer.

In the graph of Figure 11, it is evident that the real output is close to the estimated one, which is enough to consider that the model works. After training and testing the neural network, the obtained weights and bias parameters are implemented for torque and motion control in the Arduino IDE in C++.

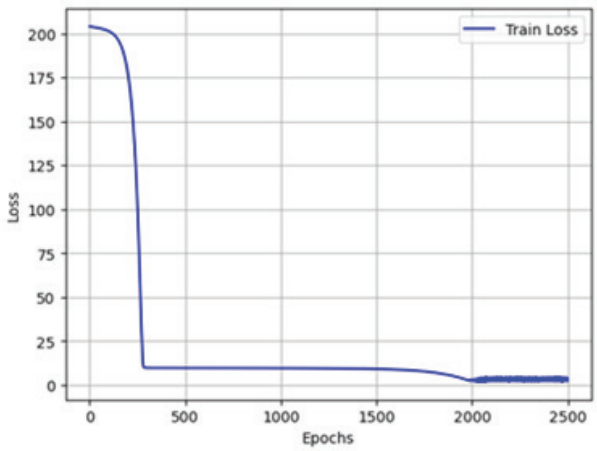

Figure 10. Training performance plot

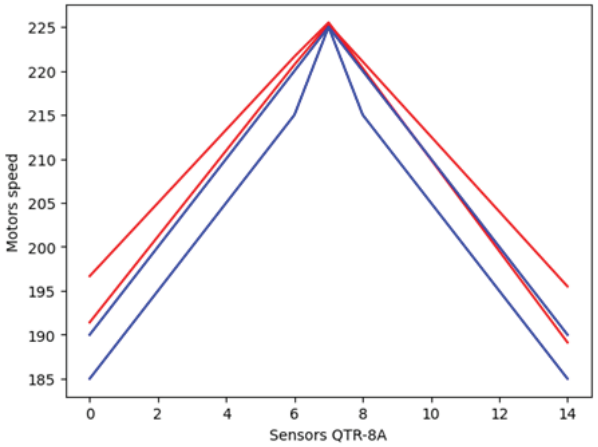

Figure 11. Actual and estimated output

Table 2. Results of prototypes with different assembly elements

Prototype	Weight of the robot	Time on lane
Prototype with elements Trough-Hole and PCB of 1 mm	178 grams	7.08 seconds
Prototype with elements SMD and PCB 0.8 mm	170 grams	6. 38 seconds

Table 3. Time results with different tire dimensions

Rim type	Time on lane
2 cm wide rim	6.38 seconds
3.7 cm wide rim	6.11 seconds

5.2. Robot Prototype Setup

Four tests were performed during each configuration of the prototype robot, and the average measured value was taken. From the data in Table 2, the line-following robot showed high speed and maneuverability on the lane by using surface mount technology in the chassis due to its low weight.

Table 3 shows that the line-following robot has excellent track stability when the tire width increases. This makes it a suitable option when there are irregularities in the lane.

The difference between the distances of the sensors from the chassis is highlighted in Table 4. By equipping the sensors at a distance of 6.8 cm, the line-following robot showed accurate abilities in scanning the track while avoiding significant deviations or irregularities.

Table 5 shows the results obtained between the two controllers by performing four track tests with the best functioning prototype in the modifications.

Table 4. Time results with different optical sensor distances from the chassis

Distance of the optical sensors from the chassis	Time on lane
6.8 cm	5. 75 seconds
8.8 cm	5. 9 seconds
10.5 cm	6.4 seconds

Table 5. Timing results with PID controller and neural network

Test number	Time on lane	
	PID controller	**Neural network**
1	6.10 seconds	5.75 seconds
2	6.15 seconds	5.70 seconds
3	6.05 seconds	5.61 seconds
4	6.07 seconds	5.58 seconds

This study indicates that the line-following robot had a more effective response with an artificial neural network control algorithm because of its ability to learn complex patterns and adapt in different environments compared to the PID control algorithm, which is simple and effective in predictable systems, but does not work well in non-linear situations. On the other hand, the fuzzy control algorithm tends to be complicated by configuration and tuning and needs to be more robust in predictable environments. The second important finding was that, with the implementation of this control algorithm, the reaction time and correction of the robot to errors on the track is faster.

The present findings also support the studies of Farkh et al. [22], who conclude that neural networks are well suited for mobile robots because they can operate with imprecise information, i.e. different environments. When processing signals from sensors, the neural network-based control algorithm responds faster to take action.

The results of the present study also suggests the use of artificial neural networks to improve performance, both in accuracy and ability to adapt to various situations. The learning capability of the neural network enabled the robot to face real-time challenges and effectively follow complicated routes.

Results of Kader and other authors in their research [16] propose a PID control algorithm that also allows a smooth and stable response of the robot as it follows the line, but with a later correction time compared to the results obtained with a neural network, considering that this time is indispensable in robotic competitions.

On carrying out tests on the line-following robot between the PID controller and the neural network, it was established that the times taken by the robot to travel along the track are almost the same for the two controllers; the variation between the two is in milliseconds. However, the PID controller presents specific errors when the robot travels along the track in a straight line; at that moment, it shows very constant

zigzagging movements. In the same way, in the curves, the robot makes abrupt movements, unlike the neural networks that make them smoother, which causes the movements on the track to take less time.

6. Conclusion

As part of the mechanical design, it was found that the chassis of the line follower robot plays a vital role since this is where all the electronic and mechanical elements are located. This part was carried out considering the smallest possible size because in the different robotics competitions, there are limitations regarding this parameter, and it is sought that the robot can pass the homologation without any problem. Considering the size and design of the line follower robot in this study, the ideal weight is 170 grams, which guarantees the robot's good performance and stability on the track. In addition, it allowed for greater agility and responsiveness when changing direction on obstacles.

The infrared sensors that make up the line-following robot, both lateral and frontal, are fundamental, as they will be responsible for keeping the robot on track. For this reason, the distance of the sensors concerning the robot chassis will depend on the radius of the curves on the different tracks; in our study, the ideal distance is 6.8 cm, representing a more precise and faster performance in detecting the route. If an incorrect location of these sensors, no matter how well the mechanical, electronic, and programming parts are working, the robot will not be able to fulfill its function properly.

The neural network was implemented as a multilayer perceptron model with linear regression, which proved that it could work successfully on a line-following robot, taking into account that it is not necessary to add more than one hidden layer in the neural network, depending on the complexity of the problem and the amount of training data available. The hidden layer used the ReLU activation function to introduce non-linearities into the network and allowed the capture of complex relationships between the sensors and the output.

AUTHORS

Cesar Minaya* – Department of Electronics, Instituto Tecnológico Superior Rumiñahui, Sangolquí, 171103, Ecuador, e-mail: cesar.minaya@ister.edu.ec.

Ricardo Rosero – Department of Electronics, Instituto Tecnológico Superior Rumiñahui, Sangolquí, 171103, Ecuador, e-mail: ricardo.rosero@ister.edu.ec.

Marcelo Zambrano – Department of Electronics, Instituto Tecnológico Superior Rumiñahui, Sangolquí, 171103, Ecuador, e-mail: marcelo.zambrano@ister.edu.ec.

Pablo Catota – Department of Electronics, Instituto Tecnológico Superior Rumiñahui, Sangolquí, 171103, Ecuador, e-mail: pablo.catota@ister.edu.ec.

*Corresponding author

References

[1] O. Gumus, M. Topaloglu, and D. Ozcelik, "The Use of Computer Controlled Line Follower Robots in Public Transport," *Procedia Comput. Sci.*, vol. 102, no. August, 2016, pp. 202–208, doi: 10.1016/j.procs.2016.09.390.

[2] A. Latif, H. A. Widodo, R. Rahim, and K. Kunal, "Implementation of Line Follower Robot Based Microcontroller atmega32a," *J. robot. Control*, vol. 1, no. 3, 2020, pp. 70–74, doi: 10.18196/jrc.1316.

[3] M. Antony, M. Parameswaran, N. Mathew, V.S. Sajithkumar, J. Joseph, and C.M. Jacob, "Design And Implementation Of Automatic Guided Vehicle For Hospital Application," *Proc. 5th Int. Conf. Commun. Electron. Syst. ICCES 2020*, no. Icces, 2020, pp. 1031–1036, doi: 10.1109/ICCES48766.2020.09137867.

[4] O.F. Gómez and U. E. Gómez, "Kinematic Simulation Of A Line Follower Robot For The Creation Of The Programming Videogame Rusty Roads In The Unity Framework," *Inf. Tecnol.*, vol. 28, no. 5, 2017, pp. 55–64, doi: 10.4067/s0718-07642017000500008.

[5] A. Aharari and Y. Ueda, "Low Pass Filter Applied to Color Sensor of Line Follower Robot," *Procedia Computer Science*, vol. 154, 2018, pp. 693–698, doi: 10.1016/j.procs.2019.06.108.

[6] V.G.R. Caitite, D.M.G. Dos Santos, I.C. Gregorio, W.B. Da Silva, and V.F. Mendes, "Diffusion Of Robotics Through Line Follower Robots," *Proc. – 15th Lat. Am. robot. Symp. 6th Brazilian robot. Symp. 9th Work. robot. Educ. LARS/SBR/WRE 2018*, 2018, pp. 604–609, doi: 10.1109/LARS/SBR/WRE.2018.00109.

[7] H. A. Calderón, C. Mejía, and L. Cobo, "Implementación De Un Robot Móvil Seguidor De Línea Y Detector De Obstáculos Con Comunicación Bluetooth," *Review Ontare*, vol. 4, no. 2, 2017, pp. 99–118, doi: 10.21158/23823399.v4.n2.2016.1639.

[8] S. Kokare, "Using ZigBee," *2018 Fourth Int. Conf. Comput. Commun. Control Autom.*, 2018, pp. 1–5.

[9] J. Chaudhari, A. Desai, and S. Gavarskar, "Line following Robot Using Arduino For Hospitals," *2019 2nd Int. Conf. Intell. Commun. Comput. Tech. ICCT 2019*, 2019, pp. 330–332, doi: 10.1109/ICCT46177.2019.8969022.

[10] W.K. Born and C.J. Lowrance, "Application of Convolutional Neural Network Image Classification for a Path-Following Robot," *2018 IEEE MIT Undergrad. Res. Technol. Conf. URTC 2018*, 2018, pp. 11–14, doi: 10.1109/URTC45901.2018.9244781.

[11] C.F. Hsu, C.T. Su, W.F. Kao, and B.K. Lee, "Vision-Based Line-Following Control of a Two-Wheel Self-Balancing Robot," *Proc. - Int. Conf. Mach. Learn. Cybern.*, vol. 1, 2018, pp. 319–324, doi: 10.1109/ICMLC.2018.8526952.

[12] J. Sarwade, S. Shetty, A. Bhavsar, M. Mergu, and A. Talekar, "Line Following Robot Using Image Processing," *Proc. 3rd Int. Conf. Comput. Methodol. Commun. ICCMC 2019*, no. Iccmc, 2019, pp. 1174–1179, doi: 10.1109/ICCMC.2019.8819826.

[13] R. Javanmard, A.H. Zabbah, M. Karimi, and K. Jeddisaravi, "Line Following Autonomous Driving Robot Using Deep Learning," *6th Iran. Conf. Signal Process. Intell. Syst. ICSPIS 2020*, 2020, doi: 10.1109/ICSPIS51611.2020.9349547.

[14] A. Moulay, F. Laoufi, T. Benslimane, and O. Abdelkhalek, "FPGA-Based Car-Like Robot Path Follower with Obstacle Avoidance," *Proc. 2020 Int. Conf. Math. Inf. Technol. ICMIT 2020*, 2020, pp. 125–131, doi: 10.1109/ICMIT47780.2020.9047008.

[15] A. Ghorbel, N. Ben Amor, and M. Jallouli, "Design Of A Flexible Reconfigurable Mobile Robot Localization System Using FPGA Technology," *SN Applied Science*, vol. 2, no. 7, 2020, doi: 10.1007/s42452-020-2960-4.

[16] M.A. Kader, M.Z. Islam, J. Al Rafi, M.R. Islam, and F.S. Hossain, "Line Following Autonomous Office Assistant Robot with PID Algorithm," *2018 Int. Conf. Innov. Sci. Eng. Technol. ICISET 2018*, no. October, 2018, pp. 109–114, doi: 10.1109/ICISET.2018.8745606.

[17] D. Nikolov, G. Zafirov, I. Stefanov, K. Nikov, and S. Stefanova, "Autonomous Navigation And Speed Control For Line Following Robot," *2018 IEEE 27th Int. Sci. Conf. Electron. 2018 - Proc.*, 2018, pp. 1–4, doi: 10.1109/ET.2018.8549580.

[18] X. Wu, P. Jin, T. Zou, Z. Qi, H. Xiao, and P. Lou, "Backstepping Trajectory Tracking Based on Fuzzy Sliding Mode Control for Differential Mobile Robots," *Journal of Intelligent Robotics*, vol. 96, no. 1, 2019, pp. 109–121, doi: 10.1007/s10846-019-00980-9.

[19] M. S. Gharajeh and H. B. Jond, "Speed Control For Leader-Follower Robot Formation Using Fuzzy System And Supervised Machine Learning," *Sensors*, vol. 21, no. 10, 2021, pp. 1–14, doi: 10.3390/s21103433.

[20] S. Tayal, H.P.G. Rao, S. Bhardwaj, and H. Aggarwal, "Line Follower Robot: Design and Hardware Application," *ICRITO 2020 - IEEE 8th Int. Conf. Reliab. Infocom Technol. Optim. (Trends Futur. Dir.)*, 2020, pp. 10–13, doi: 10.1109/ICRITO48877.2020.9197968.

[21] I.P. Latini, W.E. Barioni, M. Teixeira, F. Neves-Jr., and L.V.R. de Arruda, "Comparison between Line-Followers and Free Movement Robots in Tasks Execution in a Simulated Environment," in *2022 Latin American Robotics Symposium (LARS), 2022 Brazilian Symposium on Robotics (SBR), and 2022 Workshop on Robotics in Education (WRE)*, 2022, pp. 145–150. doi: 10.1109/LARS/SBR/WRE56824.2022.9995776.

[22] R. Farkh, K. Al Jaloud, S. Alhuwaimel, M. T. Quasim, and M. Ksouri, "A Deep Learning Approach For The Mobile-Robot Motion Control System," *Intelligent Automation & Soft Computing*, vol. 29, no. 2, 2021, pp. 423–435, doi: 10.32604/iasc.2021.016219.

[23] A. Roy and M.M. Noel, "Design Of A High-Speed Line Following Robot That Smoothly Follows Tight Curves," *Computer and Electrical Engineering*, vol. 56, 2016, pp. 732–747, doi: 0.1016/j.compeleceng.2015.06.014.

[24] B. G. Fernández et al., "Robotics vs. Game-Console-Based Platforms to Learn Computer Architecture," *IEEE Access*, vol. 8, 2020, pp. 95153–95169, doi: 10.1109/ACCESS.2020.2994196.

[25] M.H. Nushra, Q.A. Rahman, S.M.F. Mursalin, N.B. Asad, M.M. Asif Syeed, and M.M. Islam, "Smart Car Parking With The Assistance Of Line Following Robot," *2019 Int. Conf. Sustain. Technol. Ind. 4.0, STI 2019*, vol. 0, 2019, pp. 24–25, doi: 10.1109/STI47673.2019.9068046.

[26] J.W. Lok, W.M.W. Muda, and A.N. Woro, "Development Of Warehouse Robot With Advanced Line Following And Background Color Sensors," *Journal of Advanced Manufacturing Technology*, vol. 15, no. 2, 2021, pp. 23–34.

[27] H. Murcia, J.D. Valenciano, and Y. Tapiero, "Development of a Line-Follower Robot for Robotic Competition Purposes," *Applied Computer Sciences in Engineering*, 2018, pp. 464–474.

[28] L. Screw and D. Loads, "Motor Torque Calculation," *Wire*, no. 86, pp. 1–5.

[29] B. Zeng, J. Zhang, L. Chen, and Y. Wang, "Self-balancing car based on ARDUINO UNO R3," *2018 IEEE 3rd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, 2018, pp. 1939–1943. doi: 10.1109/IAEAC.2018.8577775.

[30] R. Chai, H. Niu, J. Carrasco, F. Arvin, H. Yin, and B. Lennox, "Design and Experimental Validation of Deep Reinforcement Learning-Based Fast Trajectory Planning and Control for Mobile Robot in Unknown Environment," *IEEE Trans. Neural Networks Learn. Syst.*, 2022, doi: 10.1109/TNNLS.2022.3209154.

[31] T. Guillod, P. Papamanolis, and J.W. Kolar, "Artificial Neural Network (ANN) Based Fast and Accurate Inductor Modeling and Design," *IEEE Open J. Power Electron.*, vol. 1, 2020, pp. 284–299, doi: 0.1109/OJPEL.2020.3012777.