Design and Implementation of PID-Based Longitudinal and Lateral Control for Small-Scale Vehicle

João Vitor Cavalcanti Duarte[1], Pedro Henrique Lourenço Figueiredo[2], Rafael Rodrigues da Silva[3], Evandro Leonardo Silva Teixeira[4], André Murilo Pinto[5], James Duvan Garcia Montoya[6]

[1]*Dept. of Automotive Engineering, University of Brasília (UnB)*
Brasília, Brasil
jvcduarte82010@gmail.com
[2]*Dept. of Automotive Engineering, University of Brasília (UnB)*
Brasília, Brasil
pedrohlfigueiredo@outlook.com
[3]*Dept. of Automotive Engineering, University of Brasília (UnB)*
Brasília, Brasil
rafael.rodrigues@unb.br
[4]*Dept. of Automotive Engineering, University of Brasília (UnB)*
Brasília, Brasil
evandro.leonardo@gmail.com
[5]*Dept. of Automotive Engineering, University of Brasília (UnB)*
Brasília, Brasil
andre.murilo@gmail.com
[6]*Dept. of Automotive Engineering, University of Brasília (UnB)*
Brasília, Brasil
james.montoya@aluno.unb.br

Abstract. The rapid advancement of technology has brought about significant transformations in various domains. Electronics and embedded software reached a high-efficiency level, highlighting the need for their implementation in vehicles. The automotive industry is increasingly recognizing the importance of integrating high-tech control systems and driver assistance features to ensure safety and comfort. This study will present the development of an electronic architecture for a small vehicle, comprising an integrated acceleration and braking system and a steering system which will be controlled by two Electronic Control Units (ECUs). These ECUs will include control algorithms based on PID (Proportional Integrative Derivative) controllers and the communication between these units will be performed through the CAN (Controller Area Network) protocol. Virtual tools will be used for accurate modeling and simulation of control systems, allowing the evaluation of subsystem behavior under various input conditions. Furthermore, validation tests will be carried out on a small-scale vehicle equipped with embedded systems. In this way, the use of PID controllers should present satisfactory results, reaching a harmonious balance between performance and stability.

Keywords: Control, PID, CAN, ECU, Software, Electronics

1 Introduction

The widespread adoption of Advanced Driver Assistance Systems (ADAS) in modern vehicles can largely be attributed to the discerning demands of consumers seeking heightened safety and comfort. [1]. Regarding safety, studies mentioned that some ADAS features can have a crashing avoidance rate greater than 30% [2]. ADAS can be subdivided into two main groups: connected-vehicle based where there is an exchange of information among vehicles and sensor-based where the perception of the surrounding is provided by sensors such as cameras, radars and Lidars [3]. There are many different types of ADAS technologies such as Forward Collision Warning (FCW), Autonomous Emergency Braking (AEB), Auto brake, Blind Spot Warning (BSW), Lane Departure Warning (LDW), Side View Assist (SVA), Forward Collision Warning (FCW), Lane Departure Warning (LDW) [3, 4].

An ADAS feature that has achieved significant relevance in recent years consists of the Lane Keeping Assist System (LKAS). This system contains a driver-assistance function designed to ensure the vehicle remains within the boundaries of the road lane, thereby preventing lateral displacements that exceed the road limits. [5]. An integrated vision system within the vehicle employs specific vision algorithms for lane recognition. This system then follows a control strategy that interacts with the steering mechanism to ensure the vehicle stays within the lane, preserving its intended trajectory. This functionality can also be complemented by the Lane Departure Warning System (LDWS), which employs audiovisual and/or vibratory signals to promptly notify the driver of any unintended departure from the intended route.[6].

While ADAS features have seen widespread implementation in vehicles, challenges persist in the ongoing development of this function, primarily centered around the establishment of a robust development framework for such applications [7]. Moreover, a universal and standardized method for acceptance testing has not yet been established. Instead, various companies employ a variety of testing and validation techniques in accordance with their own standards. [8]. Adopting a strategy that focuses on the development and integration of ADAS and automotive functions, small-scale vehicles emerge as a potential solution for enabling test validation and requirements analysis. The study of small-scale prototypes offers interesting advantages, particularly concerning cost and time efficiency.

Several authors developed and integrated ADAS functions in small scale vehicles in order to testing and validate control functions. In [9] an autonomous small-scale vehicle capable of navigating predetermined routes without human intervention, developing the vehicle's ability to follow pre-established routes and detect potentially risky situations. In [10], the focus was on identifying and controlling a small vehicle using the N4SID technique (Numerical algorithms for Subspace State Space System Identification). In addition, a LQR (Linear Quadratic Regulator) control algorithm for both open and closed-loop scenarios. In the research described by [11], a data logger was developed specifically for a scaled-down car model equipped with a CAN bus. The car incorporated various sensors and microcontrollers in order to acquire data from the sensors, such as acceleration, braking, potential collision indications, GPS coordinates, and vehicle speed, and store it all in a .txt format for subsequent analysis. The study conducted by [12] addresses the concept of predefined trajectories for an autonomous model. Their research proposes the generation algorithms that enable the vehicle to navigate along predefined paths, including specific areas. The [13] project aimed to transform a small-scale vehicle into a semi-autonomous platform, enabling predictable behavior within an experimental environment and facilitating data collection and processing for training purposes.

In this paper, the development of an automatic speed control system and steering system for implementation in a small-scale vehicle is proposed. The dynamic models that represent the systems to be controlled were identified and experimentally validated to assist the process of control tuning. The complete hardware and software architecture was also developed and integrated into the Controller Area Network (CAN) communication network. In this way, a microprocessor-based system using an ESP32 was employed to work as an Electronic Control Unit (ECU) responsible for reading the CAN network and transmitting commands to the steering system's actuation motors and wheel speed. In this context, the main contribution of this study relies on the use of widely adopted tools from the automotive industry to enhance small-scale prototype control systems for ADAS. This approach enables the validation of solutions on a reduced scale, providing comprehensive analysis and testing prior to their application in real-world vehicles.

2 System Modeling

For the steering system, the Matlab software systemIdentification Toolbox was used to estimate transfer functions that could mathematically model the behavior of the system. The data used for the estimation were obtained by measuring the angle in the steering column, through a potentiometer, when the system was submitted to a known PWM input, built based on the structure of a PRBS signal (Pseudorandom Binary Sequence). The approach adopted for modeling the steering system proved to be ineffective because the transfer functions obtained resulted in a maximum fit of 76%, which is not enough to be used as a model for the controller design.

In order to obtain a model that best describes the behavior of the acceleration and braking system, the first step was to collect, in an open loop, the rotation speed of the wheels, coupled to DC motors, through a speed encoder sensor. At this stage, two sets of data were generated. The first set was obtained by moving the small-scale vehicle forward and the second, by moving it backward, in both cases a step input is used in the system, with a magnitude of 255 PWM. With those results, a transfer function of a first-order system was estimated for each test scenario, based on the time constant found through the analysis of the values obtained experimentally.

3 Control System Design

For both systems, a discrete PID controller is implemented due to its simplicity and efficiency. Obtaining the Kp, Ki, and Kd gains for the steering system was based on a trial-and-error methodology, To obtain the Kp, Ki, and Kd gains for the steering system, a trial-and-error method was utilized, due to uncertainties and non-linearities intrinsic to the operation of a DC motor. The procedure involved assuming initial values for the parameter set and iteratively fine-tuning them based on the system's observed behavior. Through the application of control theory principles, the optimal set of gains for the given application could be determined. It is important to highlight that a gain scheduling approach was used in this application, due to the benefits highlighted in [14]. The steering control system required the application of this method, owing to the different motor responses at different angle setpoint values.

Assuming that the obtained transfer functions represent correctly and mathematically the behavior of the acceleration and braking system of the small-scale vehicle, through the use of an optimization script, which defines the PID controller gains necessary for the controlled system response, in front of a step input, to become as close as possible to a target behavior, it was possible to choose a time constant, aiming for behavior similar to that of a first-order system.

Figure 1 presents the block diagrams of the control systems implemented.

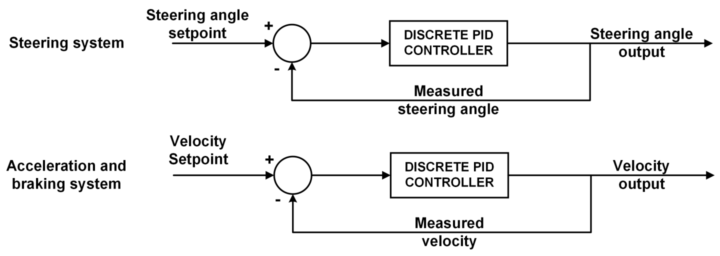

Figure 1. Control systems diagrams

4 Hardware and Software Architecture

In terms of hardware, the small-scale vehicle contains three DC motors, two for the acceleration and braking system and one for the steering mechanism. The two rear wheels rotate independently, while the front wheels steer according to the steering column's movements.

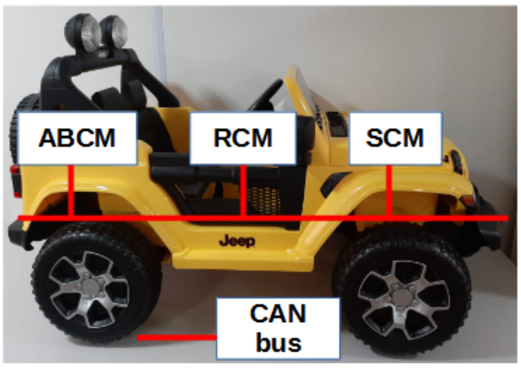

Figure 2. Electronic architecture of small-scale vehicle

As shown in 2, the following ECUs were designed: ABCM (Acceleration-Braking Control Module) responsible for the longitudinal movement, SCM (Steering Control Module) which controls the steering angle and RCM (Remote Control Module) responsible for the radio communication. The ECUs are connected with each other via CAN bus in order to exchange information. (see Fig. 3). To measure the rotational speed, a velocity sensor encoder is implemented and to obtain the angle of the steering column, a potentiometer was coupled to a gear gear system which is directly connected to the column. Figure 4 shows the above mentioned sensors.

The ECUs are powered by 12 V battery, once the other devices work in a different voltage level, a voltage regulator is implemented. The MCP 2515 is a CAN bus interface module that allows communication through this protocol between the ECUs. It has a high-speed SPI interface (10 MHz), ESP32 compatibility and the capacity to implement CAN bus up to 1 Mb/s.

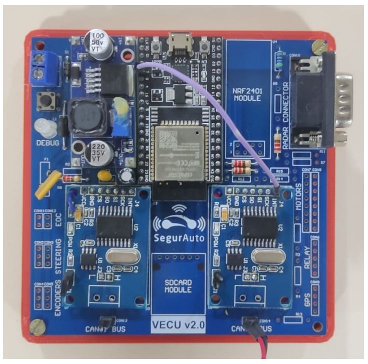

Figure 3. ECU standard model

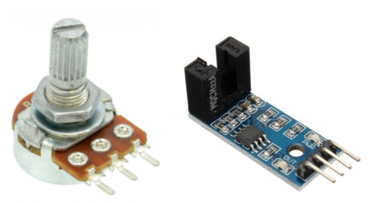

Figure 4. Angle and velocity sensors

The ESP32 receives the information from the steering and velocity sensors and sends commands to the motors to execute and reach the desired setpoint. In order to control the DC motors control, BTS7960 H-bridges were used to control and enable them rotation in both senses. Figure 5 shows the flowchart of the eletroelectronic architecture of the small-scale vehicle.

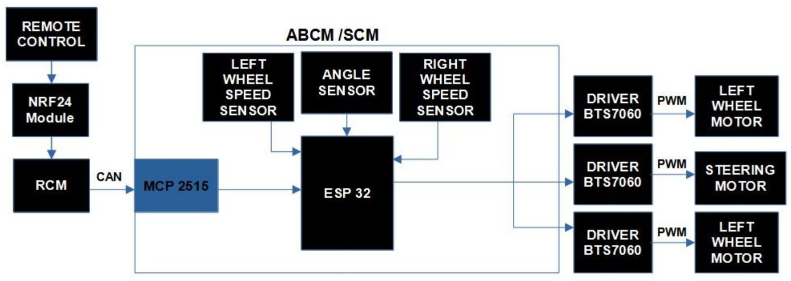

Figure 5. Flowchart of Hardware Components

In broad terms, the depicted flowchart illustrates the process whereby the remote control transmits commands via radio frequency to the RCM. Given the interconnection of the ECUs through the CAN bus, the setpoints are sent to the ABCM and SCM where ESP32 executes the control algorithm, coordinating the transmission of a PWM signal to the drivers. This step regulates the motors behavior, culminating in the desired control outcome.

The built software structure to control the acceleration and braking of the low-scale vehicle can be divided into four main parts, CAN handling, sensors monitoring, discrete PID controller, and motors control. Using MCP2515 CAN modules associated with the used microcontroller, to interpret, send, and receive CAN messages, was used the MCP CAN Library. Through this library could be sent and received CAN messages dynamically, sending information just by changing each byte in the CAN frame data field, and for that, was used the standard frame CAN (CAN 2.0A). In this case, the velocity setpoint for the acceleration and braking system comes from a message sent by the CAN bus.

The principal magnitude to control the acceleration and braking performance of the vehicle, in this case, is the velocity. To measure the velocity, the interruptions resource of ESP 32 was used to count the time between each pulse generated by the perforate disk rotation. Therefore, every pulse means an interruption, and whether saved the last pulse time, the current angular velocity can be calculated knowing how many holes the disc has.

That output is based on the error behavior and the input variation also. The integral and derivative terms are implemented through the discrete method of calculating integrals and derivatives. To set the motors were used PWM signals were sent from ESP 32 to the BTS7960 driver. For that, was considered the value of controller output as the value of PWM, i.e., a value between 0 and 255. Thus, through the AnalogWrite function, different pins could be used for it, adjusting the motor sense accordingly with the driver operating logic and the required sense also.

For the steering system, the structure of the built software can also be divided in the same way as the structure of the acceleration and braking system was split, with minor differences in the way of implementation. Using the MCP2515 module, the steering system ECU was also created to be able to receive CAN messages over the bus.

Therefore, using the MCP CAN library, the first part of the software deals with receiving and translating messages coming over the CAN bus. In this case, the information of interest received by this system is the desired setpoint of the steering angle of the wheels.

The difference between the setpoint value and the value read by the sensor results in the error, quantity used as input in the discrete PID controller, generating a command variable as the output. The final part of the software adjusts the command variable to PWM values within the range of 0 to 255, for powering the steering DC motor, through a linear function. Therefore, the motors are controlled and the loop is closed with the current angle reading.

5 Results and Vehicle model validation

The obtained results can be segmented into the steering system and the acceleration and braking system. Firstly, for the steering system, we have the analysis of four main tests for the controller, which can be observed in the figures below. The set of parameters chosen was $k_p = 0.01$, $k_i = 0.00001$, and $k_d = 0.00002$ for the range between 0° and 10°, and $k_p = 0.015$, $k_i = 0.00002$, and $k_d = 0.00002$ for the range between 11° and 25°. Different inputs were applied to the system to validate the gains, and the responses can be observed in figures 6, 7, 8, and 9.

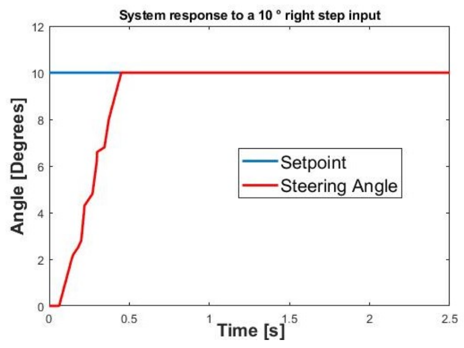

Figure 6. System Response to a 10° Right Step Input

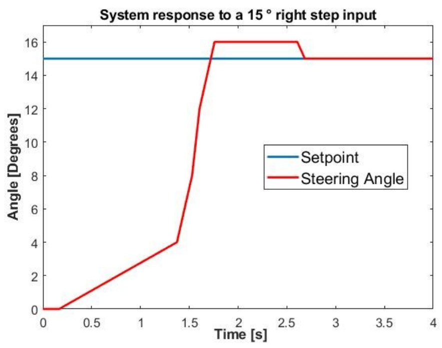

Figure 7. System Response to a 15° Right Step Input.

Figure 6 shows that the total time until the system reaches the desired setpoint was approximately 2.6 seconds. The rise time of a response is the time it takes for the system to go from 10% (1.5°) to 90% (13.5°) of the response. Analyzing the graph, it can be observed that this time was about 1.4 seconds. Another parameter that can be identified in the figure is the peak time, which is the time elapsed until the first peak of the response is reached. In this case, this value was close to 1.7 seconds and the overshoot for this response was approximately 7%.

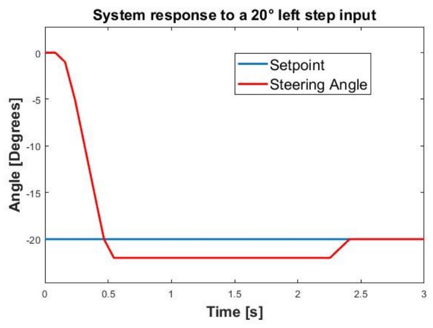

Figure 8. System Response to a 20° Left Step Input

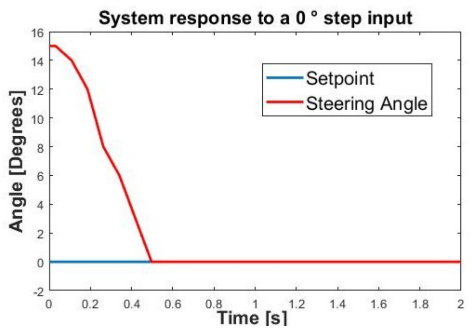

Figure 9. System Response to a 0° Step Input

Figure 7 shows that the total time until the system reaches the desired setpoint was approximately 0.4 seconds. The rise time, analyzed from the graph, was about 0.25 seconds. In this case, there was no overshoot, what means that the controller had a satisfactory performance.

The observed response in Figure 8 shows that the total time until the system reaches the desired setpoint was approximately 2.3 seconds. The rise time was about 0.5 seconds. The peak time was 0.55 seconds. In this case, the angle reached at the peak was 22°, therefore, the overshoot was 10%. The observed response in Figure 9 shows that the total time until the system reaches the desired setpoint was approximately 0.5 seconds. The rise time was close to 0.45 seconds. There was no peak.

The presence of the overshoot in some of the test cases is due to some factors, such as, for example: the mechanical clearances of the system, the type of floor on which the data acquisition was carried out, the dead zones of the motor, load losses of the motor, the voltage level of the battery, and other factors associated.

For the acceleration and braking system, six tests were carried out to evaluate the performance of the implemented methodology. Data acquisition in tests 1 and 2 was done with the vehicle moving forward and for tests 3,4,5,6 the collection was done with the vehicle moving backwards.

Figure 10 presents a graph with two curves, the curve in blue represents the response of a first order system, with a time constant equal to 1 second, when subjected to a step input. And the curve in red represents the response of the acceleration and braking system, to the same step input, with a PID controller implemented, with the gains obtained through the use of the optimization script, whose target behavior was precisely that of a first order system with a time constant equal to 1 second. For each test, the fit between the curves was calculated using the NRMSE (Normalized Root Mean Squared Error). With this approach, a fit of 88.2% was obtained for the test 1.

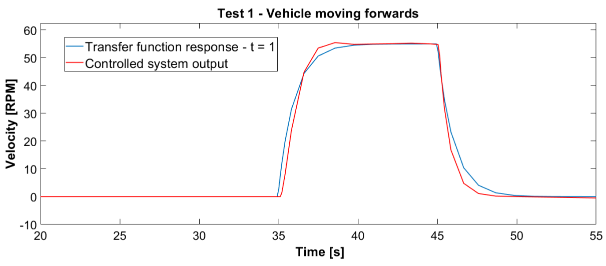

Figure 10. Test 1 - Vehicle moving forwards - T = 1

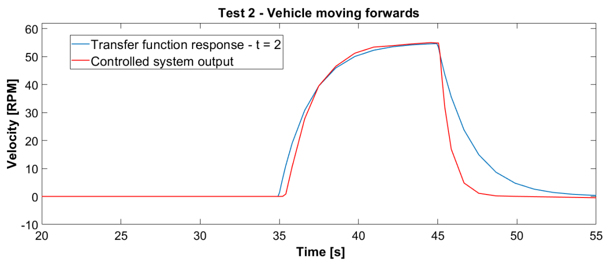

Figure 11. Test 2 - Vehicle moving forwards - T = 2

The other tests were carried out in a similar way to test 1, changing, in addition to the sense of movement of the vehicle, the time constant of the first order transfer function that is desired to be achieved with the controlled system. For tests 2, 4 and 6, the time constant considered was equal to 2 seconds and for tests 3 and 5 equal to 1 second, as well as test 1. Through test 2, the graph of Figure 11 was constructed. And for this test, a fit of 67.5% was obtained.

Now with the vehicle moving backwards, test 3 generated the results shown in Figure 12. The fit obtained with test 3 was 80%. Using the time constant equal to 2, the result of test 4 can be seen in Figure 13. The result of test 4 generated a fit of 65.2%.

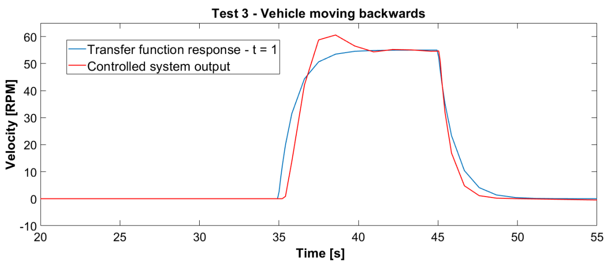

Figure 12. Test 3 - Vehicle moving backwards - T = 1

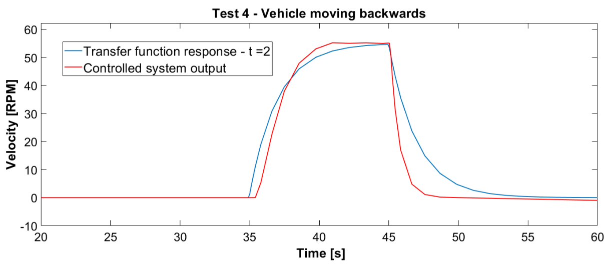

Figure 13. Test 4 - Vehicle moving backwards - T = 2

As the fit results obtained with the vehicle moving forward were better, tests 5 and 6 were carried out moving the vehicle backwards, but the gains of the PID controller implemented were the gains generated in the optimization that used the model of function of transfer obtained with open-loop data when the vehicle was moving forward. Thus, Figures 14 and 15 present the results of tests 5 and 6 respectively.

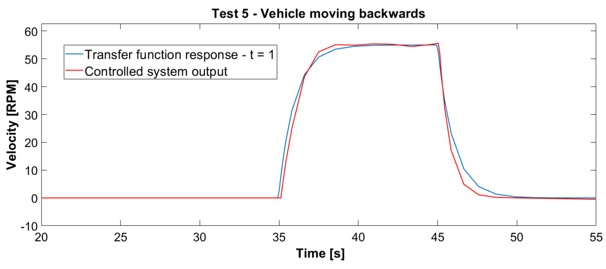

Figure 14. Test 5 - Vehicle moving backwards - T = 1

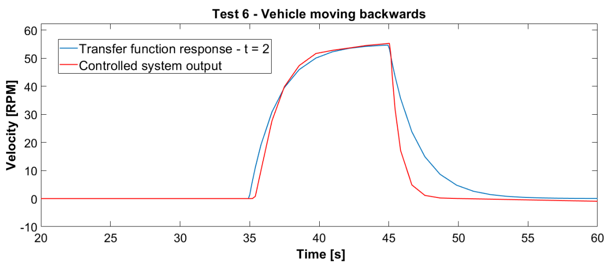

Figure 15. Test 6 - Vehicle moving backwards - T = 2

Test 5 generated a fit of 89.8%. And, with test 6 a fit of 67.4% was obtained.

It is concluded that the way in which the model of the acceleration and braking system, in open loop, was estimated is valid, because through its use, it was possible to achieve results considerably close to the mathematically expected.

6 Conclusions and Future Work

The development of the electronic architecture for the small-scale vehicle is of immense importance for the consolidation of studies directed towards autonomous vehicles, as it allows for the evaluation of control system behavior in different subsystems of a vehicle, although on a smaller scale. In the steering system, choosing a proportional controller alone resulted in a high overshoot. Adding an integral term, i.e., a PI controller, did not improve the response performance as the overshoot remained unchanged. By using a derivative term, i.e., a PD controller, the overshoot was reduced but remained relatively high, which is not ideal for a steering system. However, with the implementation of a PID controller, the issues encountered with other controllers were resolved. It exhibited a significantly lower overshoot and a suitable settling time for this application.

In the acceleration and braking system, considering it as a first-order system proved to be valid. Constructing a discrete PID controller associated with a plant represented by a transfer function obtained through the identified time constant, using an optimization script, provided results that validate the approach. Therefore, the obtained results demonstrate that the system operates efficiently. If needed, the system's response can be adjusted according to a new reference.

As a result, the applied control systems theory made the modeling and simulation faithful to the actual behavior of the system. Thus, this work has demonstrated that the development of an electronic architecture in a small-scale vehicle represents a viable, cost-effective, and less complex alternative for implementing autonomous technology in real vehicles. Furthermore, this work can be subject to further improvements and enhancements, such as the addition of other sensors like radars and cameras, which ensures greater reliability in environment data acquisition and provides the implementation of functions present in autonomous vehicles, such as AEBS (Advanced Emergency Braking System), and LKA (Lane Keep Assistant).

References

[1] S. P. J. Tunnell. V. K. Kukkala and T. Bradley. Advanced driver-assistance systems: A path toward autonomous vehicles. *IEEE Consumer Electronics Magazine*, vol. 7, n. 5, pp. 18–25, 2018.

[2] A. Farid. *The Practical Effectiveness of Advanced Driver Assistance Systems at Different Roadway Facilities: System Limitation, Adoption, and Usage*. Publisher, 2018.

[3] J. B. Cicchino. *Effects of lane departure warning on police-reported crash rates*. Publisher, 2018.

[4] K. M. Kockelman and T. Li. Implications of connected and automated vehicles on the safety and operations of roadway networks: A final report. *Transportation Research Board*, 2016.

[5] X. Ye. Adaptive lane keeping assistance system design based on driver's behavior, 2019.

[6] Y. Jeong. Interactive lane keeping system for autonomous vehicles using lstm-rnn considering driving environments. *Sensors*, vol. 22, n. 24, 2022.

[7] Y. An, X. Tang, J. Zheng, F. He, L. Ma, and M. Li. An evaluation framework to measure the performance of adas. pp. 3712–3723, 2019.

[8] S. Wei, P. E. Pfeffer, and J. Edelmann. State of the art: Ongoing research in assessment methods for lane keeping assistance systems. *IEEE Transactions on Intelligent Vehicles*, vol. , pp. 1–28, 2023.

[9] C. e. a. CIUCIU. Autonomous scale model car with ultrasonic sensors and arduino board., 2019.

[10] D. M. D.; GUIDA. SIMONE. Identification and control of an unmanned ground vehicle by using arduino. *UPB Scientific Bulletin*, 2018.

[11] S. SREEVATSAN. Data logger for a miniature model car using can bus. *Journal of Biotechnology*, 2021.

[12] M. P. PETTERMAN. Projeto de um veículo autônomo capaz de cobrir uma área poligonal sem passar mais de uma vez pela mesma região., 2015.

[13] D. R.-R. P. S. de; RICCOBONI. ROSIER. Small scale intelligent vehicle final design repor., 2018.

[14] D. J. L. . W. E. Leithead. Survey of gain-scheduling analysis and design. *International Journal of Control*, vol. , 2000.