Line Follower Robot with PID Adaptive Cruise Control

Ahmet Serdar EKER
Department of Electrical and Electronics Engineering
Abdullah Gül University

Kayseri, Türkiye
ahmetserdar.eker@agu.edu.tr

Muhammed Enes ÜNAL
Department of Electrical and Electronics Engineering
Abdullah Gül University

Kayseri, Türkiye
muhammedenes.unal@agu.edu.tr

Osman Samet ŞENOL
Department of Electrical and Electronics Engineering
Abdullah Gül University

Kayseri, Türkiye
osmansamet.senol@agu.edu.tr

Süleyman ŞAHİN
Department of Electrical and Electronics Engineering
Abdullah Gül University

Kayseri, Türkiye
suleyman.sahin@agu.edu.tr

***Abstract*— This paper aims to represent the project of building a Line Follower Robot with PID Adaptive Cruise Control. Line tracking is the most convenient and reliable navigation technique by which autonomous mobile robots navigate in a controlled environment. Furthermore, adaptive cruise control techniques are commonly used in autonomous driving technologies nowadays. This project includes Proportional-Integral-Derivative (PID) control algorithm and register-level ARM C code. The primary objectives are following the track and following a board at specified distance. To build robots, low-cost components are used including a microcontroller, motor driver and reflectance sensor.**

Keywords - Line follower robot, PID, Adaptive cruise control, microcontroller

I. INTRODUCTION

The goal of this study is to design and build a line-following robot with adaptive cruise control with implementing Proportional-Integral-Derivative (PID) control algorithm and register-level ARM C code. The design of line follower robot is directly related to PID control which is one of the most important techniques is used in autonomous navigation. This project aims to innovate mobile robotics and solving cost problems in this area. Line tracking is important behaviour in mobile robotics where robots use to identify specific tracks like reflective tape. They are related to reflactance sensors which are using technology of IR to detect and follow the specific tracks. Moreover, ultrasonic distance sensor is the another important aspect of this project. The sensor uses ultrasonic sound waves to detect the distance of the object so that it can be detected the distance of the object in front of the mobile robot. Mostly, line tracking is made digitally. Digtal sensors send receving signals to microcontroller, that adjusts the robot's motor speeds related to robot's position on center of the line. Finding the exact motor speed includes trial and error to make sure pure line tracking. Furthermore, detecting distance made by the ultrasonic distance sensor with the code implementation. It should follow given instruction which is 20 cm to in front of the robot.

This project improves upon line tracking by implementing PID control, an advanced mechanism for tuning motor behavior. PID is a control algorithm that operates on three fundamental parameters, namely Proportional, Integral, and Derivative, and serves as the heart of the robot, helping to achieve smoother and accurate line tracking. A PID controller is a control loop feedback mechanism most commonly implemented in industrial control systems and used in many other applications requiring continuously modulated control. This helps the system reduce disturbances, but increasing the gain can cause oscillations and instability. The Proportional and Integral components correct steady-state errors by calculating the sum of past errors; however, a high integral gain may result in overshooting and instability. The derivative term aids in preventing overshooting channels and allows faster transient responses by predicting future errors. In noisy environments, though, derivative action can contribute more to high-frequency noise than to the system output, which may lead to destabilization. To navigate more complex paths efficiently, the robot dynamically tunes PID parameters using feedback from onboard sensors. PID control is a significant part of autonomous robotics, so combining traditional line tracking and adaptive cruise control with PID control enhances our understanding of closed-loop methods in a more robust way. The project has two objectives—ensuring the robot completes the entire track and minimizing the total run time of task completion. This paper provides the design of a high-speed and high-precision PID controller for a line tracking and adaptive cruise control system. The robot uses DC motors adjusted through a microcontroller programmed in ARM C language up to the register level. The PID controller is utilized to determine errors based on a reflectance sensor array and ultrasonic distance sensor. Integrating PID control aimed to minimize overshooting, eliminate steady-state error, and prevent oscillations, delivering an effective navigation system.

II. COMPONENTS

The mechanical and electronic components are represented in line following robot with PID Adaptive Cruise control in this chapter.

A. Mechanical Components

The line follower robot with PID Adaptive Cruise Control serve as the test platform for this project. These cost-effective DC gear motors, equipped with built-in gearboxes, are designed for quick responses with adequate torque. These motors are inexpensive and can operate with higher input voltages for increased speeds over short

durations, if necessary. Each motor is securely mounted to the frame using two plastic components and two screws.
The center of mass significantly affects the robot's stability. Since the battery is the heaviest component, its placement on the chassis allows for precise control over the robot's center of mass. When positioned toward the front, there is an increased risk of slipping, while a rearward placement results in noticeable pitching of the front end during sharp turns. To ensure stability, the center of mass is aligned with the center of the frame. An optimal distance of 7 mm between the ground and the QTR-8RC reflectance sensor is also critical for stability. This distance is essential, especially during sharp turns, as deviations—whether too far or too close—will hinder the robot's ability to accurately follow the line.

B. Electronic Components

1. Microcontroller (STM32F103C8T6A)

Sensor values were read and the DC motors were controlled using a microcontroller. A microcontroller is a small computer on a single integrated circuit containing a processor core, memory, and programmable input/output peripherals. The STM32F103C8T6A, part of the STM32 family made by STMicroelectronics, is based on the ARM Cortex-M3 core.It features a 32-bit ARM Cortex-M3 processor running at up to 72 MHz and offers various I/O peripherals, including 12-bit ADCs, UARTs, and GPIO pins. Additionally, it has a built-in Real-Time Clock (RTC) and supports low-power operation. The STM32F103C8 also supports the ARM Cortex-M3's Memory Protection Unit (MPU), which enhances security and operational reliability.
This microcontroller is widely used in consumer electronics, industrial control systems, and automotive applications. Its low cost, availability, and extensive manufacturer support and resources make it popular among hobbyists and for educational projects. Using an authentic STM32F103C8 board is essential for ensuring proper system functionality.

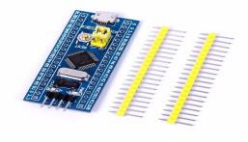

Figure 1. Microcontroller

2. *Motor Driver(L298N)*

An L298N dual full-bridge motor driver module was used to control the two DC motors. This module operates on a 12V DC power supply (provided by three batteries) and includes a 5V voltage regulator to power the STM32 microcontroller. The module has two input pins for speed control of each motor (IN1 and IN2 for Motor A, IN3 and IN4 for Motor B) and two output pins for direction control (OUT1 and OUT2 for Motor A, OUT3 and OUT4 for Motor Additionally, the module features four input pins to receive PWM signals, which control both the speed and direction of the motors. The output pins are specifically designed for DC motors, delivering a maximum current of 2A per motor, making it suitable for efficient motor operation.

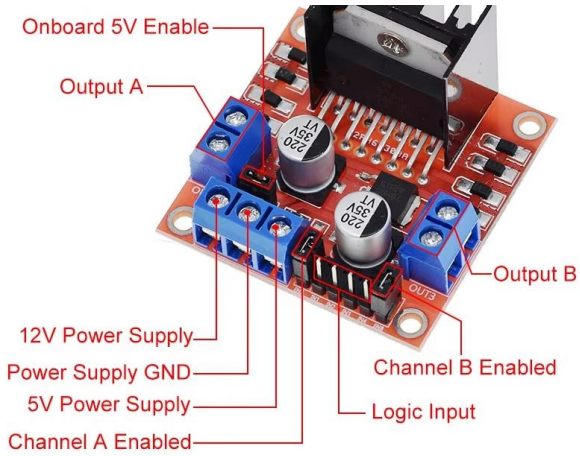

Figure 2. Motor Driver(L298N)

3. *Batteries*

Three rechargeable 3.7V Li-ion batteries were used to power the robot system, providing a combined 11.1V DC input voltage. The positive terminal of the battery is connected to the 12V input pin of the motor driver. A minimum input voltage of 7V is required to maintain a stable 5V output from the motor driver's internal regulator, which is essential for the efficient operation of both the STM32F103C86TA microcontroller and the QTR-8RC reflectance sensor.

4. *Ultrasonic Sensor*

As such, the ultrasonic sensor is a very important part of this project. The robot relies on this sensor to keep a constant distance from a moving or stationary board—a critical requirement for its adaptive cruise control. Normally, an HC-SR04 is used for the same purpose. This sensor works by emitting ultrasonic sound waves and calculating the time it takes for these waves to travel to the target, which is the board, and then return as an echo. With the speed of sound, the sensor calculates the distance. The ultrasonic sensor is placed at the front of the robot and continuously monitors the proximity of the board, hence supplying real-time data that will be important in the decision-making process of the robot. The data in regard to the distance recorded from the ultrasonic sensor will feed into the PID control. It allows for the dynamic tuning of speed by a robot and for keeping a steady, 20-cm position in respect to the board. It is also highly essential that this board is sometimes speeding and at times remains in a constant state. Correspondingly, it either accelerates or decelerates in such a way to allow for the set distance. Ultrasonic sensors have the added advantage of operating independently of reflective surfaces or interference from ambient light, conditions that would render infrared and

similar light-based sensors unreliable. Using an ultrasonic sensor, this system ensures the reliable and accurate distance measurements that will be required for the adaptive cruise control to function smoothly. The integration enhances both the performance of the robot and points toward the very important role of accurate, real-time sensor data in control applications, whereby it enables the robot to fulfill its objectives successfully on the test track.

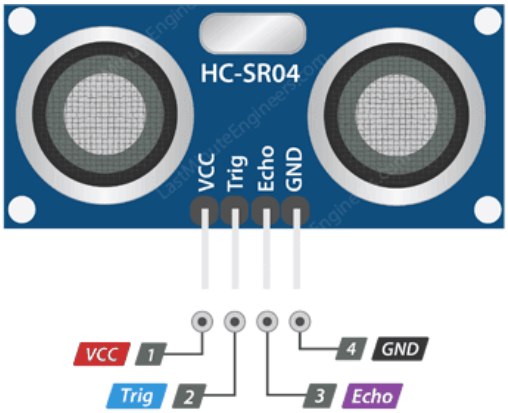

Figure 3. Ultrasonic Sensor (HC-SR04)

5. *Reflectance Sensor (QTR-8RC)*

The QTR-8 sensor module allows the robot to follow the line. This module contains eight reflectance sensors, as illustrated in Fig. 2.4, each of which consists of infrared sign emitters, MOSFETs based on infrared light, and capacitors. The MOSFETs regulate the grounding path for the capacitors, and the output pin of each sensor is respectively connected with the GPIO pins of the STM32 microcontroller shown on Fig. 2.5 for read and write purposes. When the infrared light hits the MOSFETs, the capacitors are discharged to ground. After some delay of a few milliseconds, the GPIO pins become inputs. A logic 0 reading from a sensor means it is above a reflective surface, while a logic 1 means a non-reflective or, more commonly, dark surface. The setup allows for accuracy in the working of the microcontroller in detecting the position of the line to be followed by the robot. The gap between the sensor and the track should not exceed 7 mm for best function. Additionally, the voltage regulator steps down the 5V supply coming from the motor driver to 3.3V for the microcontroller. This regulation protects the microcontroller from heating up by keeping it in its safe working voltage of 3.3V, not 5V.

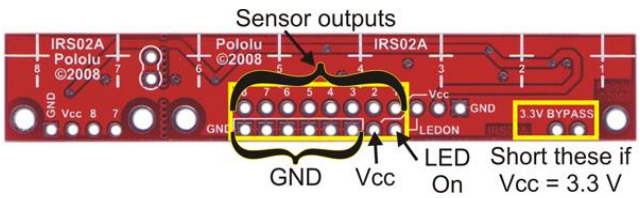

Figure 4. QTR-RC Reflactance Sensor

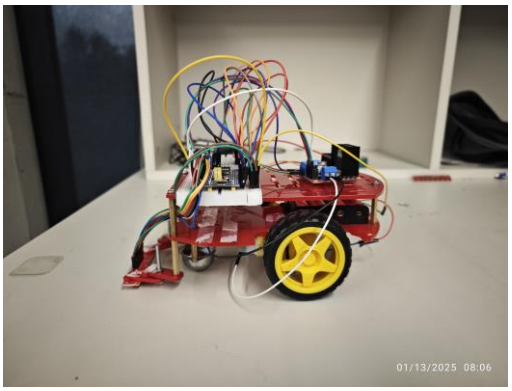

Figure 5. Left side of the car.

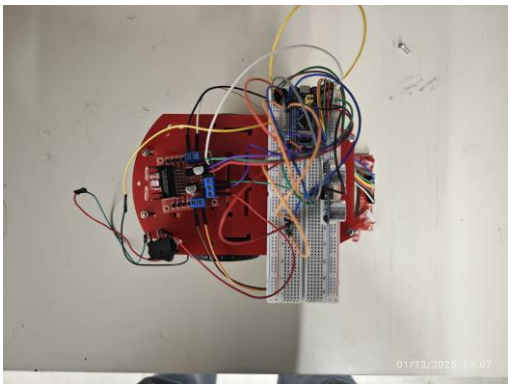

Figure 6. Upper view of the car.

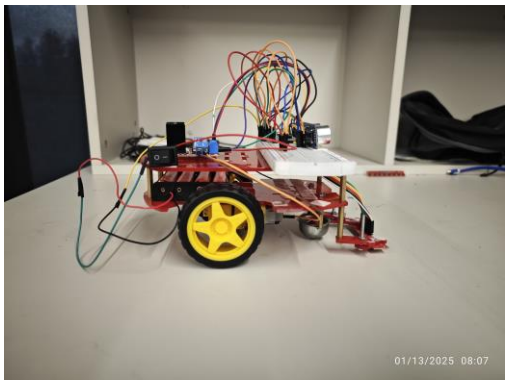

Figure 7. Right side of the car.

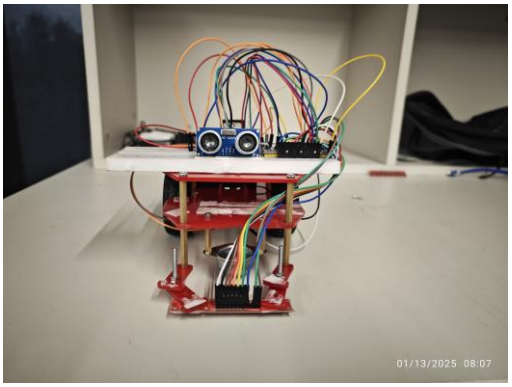

Figure 8. Front view of the car.

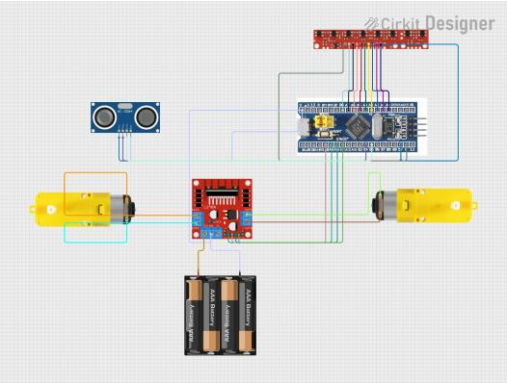

Figure 9. Combinations of Components.

III. PID CONTROLLER DESIGN

The PID controller is one of the general feedback mechanisms applied to most control systems, and its application domains span where control and correction of errors require precision. A PID controller enables a line follower robot to track along a predefined track through continuous adjustment of motor speeds based on an error detected by its sensors. Error is defined as the difference between the desired position of the robot on the track and its actual position measured using a reflectance sensor array. The main objective of the PID controller, in this case, is to minimize the error signal to keep the robot aligned with the line without oscillations or overshooting during turns.

The action of a PID controller results from the combination of three control actions: proportional, integral, and derivative. In the P term, the response is directly proportional to the current error; this helps the robot fast to go back to the track once it moves away from the track. However, with a proportional term alone, there is one problem of a steady-state error-a small, sustained deviation from the desired position. The remedy to this is the integral, I, which sums the past errors through time and drives the output to reject this steady-state error. While doing this, it gives better accuracy, but at the same time, if not tuned, it might also result in overshooting. The derivative term, D, predicts future errors from the rate of change of the current error. By doing so, it reduces overshoot and stabilizes the system by dampening oscillations.

As previously noted, PID performance greatly depends on proper tuning of the parameters: Kp, or proportional gain; Ki, or integral gain; and Kd, or derivative gain. A critical balance between responsiveness and stability must be achieved. Numerous tuning methods range from the original Ziegler-Nichols and Cohen-Coon rules to modern optimization techniques like genetic algorithms and particle swarm optimization. Manual tuning is one of the common methods used in conjunction with trial and error to try and get the right balance between speed and precision for a line follower robot. The aim will be for it to track along a line without oscillations or loss of line on sharp turns or changes in track.

In the context of this project, the PID controller processes data from the reflectance sensor array and calculates necessary adjustments to the motor speeds. When the robot drifts off the center of the line, the PID controller increases the speed of one motor and decreases the speed of the other to steer the robot back on track. For example, if the robot is turning to the left, it speeds up the right motor, and then the robot turns right until it is on the desired path. Such dynamic adjustment of the motor's speed allows it to make smooth and accurate navigation.

The PID controller gives an overall better control performance to the line follower robot with higher accuracy and response time. Another application of the flexibility of the PID approach is adaptive cruise control using an ultrasonic sensor that maintains a set distance with any moving or stationary object. It would continuously monitor the distance and hence regulate the speed to make safe and efficient navigation even in dynamic environments.

IV. SOFTWARE

Software is the prime ingredient of the line follower robot to interpret sensor data, apply the control algorithm, and drive motors. It will be composed of three major tasks: sensor interfacing, logic control, and motor actuation. The capabilities of the robot in following a line and maintaining its set distance from an object will depend on how efficiently the software is carrying out these tasks in real time.

This module of the software is responsible for data acquisition from the sensors interfaced to the robot. The reflectance sensor array detects the position of the line, while the ultrasonic distance sensor measures the distance from a target object. Reflectance sensor data is processed to find the error, showing how much the robot has strayed from the line. That means the ultrasonic sensor continuously provides the distance measurement for the robot to maintain a safe following distance from any obstacle in front. The collected data from these sensors acts as input for the PID controller. Software contains the control logic, which is implemented through a PID algorithm. It accepts the error calculated from reflectance sensor data as an input and calculates adjustments in motor speeds. The proportional term acts on the immediate error, the integral term on past errors to remove steady-state deviations, while the derivative term predicts future errors to avoid overshooting. The software will dynamically adjust Kp, Ki, and Kd values based on performance input from the robot to enable the robot to cruise smoothly and precisely. That creates a self-regulating feed forward cycle that keeps the robot aligned to the track. The second portion translates the control output, resulting from the output of the PID controller into movement-the actuation of the motor. The software will generate PWM signals to control the DC motors' speed. The software changes the duty cycle of the PWM signals to vary the motor speeds independently, enabling the robot to make left or right turns as required. If, for

example, it detects that it has veered to the left, then the software will increase the speed on the right motor and decrease the speed on the left one until the robot will steer itself towards the center of the line. Since such adjustments take place in less than a second, the robot can handle both straight paths and sharp turns without substantial delays or errors.

Along with the usual line tracking, the program realizes an ACC feature with the use of an ultrasonic sensor: the robot is always watching the distance from the target object and correcting the speed of movement depending on the target's moving direction. When a target moves closer, the robot slows down; if the target moves farther away, it speeds up to catch up. This is especially useful in dynamic environments where the robot needs to react to changes in its environment.

For this to be reliable, a number of considerations have to be taken into account at the design of the software. First, it needs to be real-time-that is, the control loop should execute fast enough to process sensor data and adjust motor speeds without noticeable delay. Filtering noise will be required to get rid of undesirable fluctuations in sensor readings that may otherwise cause the robot to make unwanted adjustments. Lastly, the software should contain failsafe mechanisms for any unforeseen eventualities-for instance, malfunction of sensors or obstacles on the track.

In the end, this is software that bridges hardware components with the control logic of the robot. The software allows, through an efficient and powerful control algorithm, the robot to track a line with much more accuracy or to react in a dynamic environment. Sensor data processing, PID control, and actuation of motors are the very basics of software for a line follower robot that make it possible for the robot to navigate in an accurate and reliable manner.

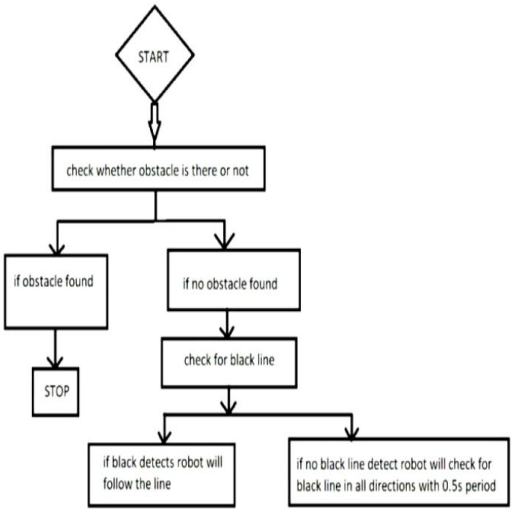

Figure 10. Flow chart of adaptive cruise control from Suryawan.

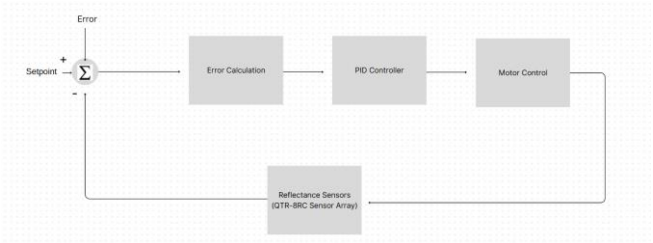

Figure 11. Flow chart of general system.

V. RESULTS

This PID adaptive cruise control-equipped line follower robot was able to successfully perform the main tasks of correctly tracking the line and maintaining a specified distance from obstacles. Its performance was tested on various track layouts: straight paths, curves, and sharp turns. It would not be possible without the realized PID controller, where the robot dynamically changes the speed of its motors while receiving sensor feedback, with an objective to keep reduction and make the car more stable.

In a test, this allowed the robot to reveal very smooth and accurate behavior in line-tracking. On one hand, proportional control made the robot sensitive to any kind of deviation from a track; this created oscillations in sharp turns. The integral control eliminated the steady-state error by considering the previous deviations so that the robot could keep to the center of the line. The addition of D control reduced overshoot and made the system much more stable, especially when the robot went through sudden changes in track. After some rounds of tuning, the optimized PID parameters came to Kp = 0.0148, Ki = 0.00225, Kd = 10, which provided a good balance between responsiveness and stability.

Adaptive cruise control puts the test right in the track, with an ultrasonic distance sensor that is maintaining a 20 cm steady distance in front of the robot. Based on this sensor, the speed of the robot is regulated upon how close a target would approach in a dynamic fashion. The velocity decreased when the object was closer to it, not to bump into it, and it picked up again when the object moved further away to maintain the set distance. This was made possible with the help of the ultrasonic sensor that gave an accurate measurement of the distance.

This way, smooth acceleration could be achieved in the robot without any abrupt jerks.

This can be considered representative of real-life scenarios of autonomous driving. During testing, the stability and response time of the robot were quite remarkable. It was able to make error corrections in less than 500 milliseconds, which is critical for real-time line tracking. During sharp turns, the robot performed well without losing track of the line, and after being manually pushed off the track, it managed to recover smoothly. These include noise filtering algorithms that minimize the effect of sensor noise, especially on reflective surfaces,

further enhancing the reliability of the sensor readings. It consumed very low power; its battery lasted for an average of two hours of continuous running. Energy dissipation is avoided by applying PWM to regulate the motor's speed and therefore avoiding every probable means of dissipating energy. In all, three 3.7 V Li-ion batteries were applied to provide 11.1V input to the motor driver-which value is adequate to make sure that stability will be ensured in the performance of the robot. During testing and development, there were a number of issues encountered. Of these, sensor noise-on reflective surfaces especially-was one of the most pervasive problems which gave inconsistent reading from the reflectance sensors. For this purpose, threshold values of sensors were changed and noise filtering was performed. Sharp turns also proved to be a problem where initial PID tuning caused oscillations. These gains were tuned iteratively until the PID smoothly navigated through straight paths and sharp turns. In setting the ultrasonic sensor very accurately with the same energy for the distance correct measurement, adjustments were done on that point again to maintain uniformity in its reliability.

Table 1. Measuremnts of the line follower robot.

Metric	Value	Remarks
Line Tracking Accuracy	≈ 95%	Minor deviations on sharp turns
Response Time	< 500 ms	Fast corrections for deviations
Distance Maintenance	± 2 cm	Maintained a stable 20 cm distance
Battery Life	≈ 2 hours	Continuous operation
Maximum Speed	100% duty cycle (PWM)	Achieved during straight paths
Sharp Turn Handling	Successful (90-degree turns)	Stable recovery without overshooting

VI. CONCLUSION

The following work is based on the design and building of a line-following robot with adaptive cruise control using a Proportional-Integral-Derivative algorithm. Its onboard navigation effectively couples line tracking and distance control to ensure smooth, precise, and adaptive motion. Equipped with both reflectance sensors for line detection and an ultrasonic sensor for maintaining specified distances from any object, the robot dynamically adjusts the speed to meet high real-time performance with high accuracy.

The PID algorithm played a great role in optimizing the behavior of the robot by means of error minimization, prevention of oscillations, and stable navigation on turns or off the track. This was improved in performance due to continuous feedback from sensors by changing the motor speeds w.r.t. the PID controller to keep the robot steadily on course. This allowed the robot to adapt automatically to changes in distance from a moving object by adjusting speed with respect to maintaining a safe and consistent following distance. This added much sophistication to the project and insight into how multiple control techniques can be combined into autonomous robotics. Efficiency in real-time right from processing the sensor data to setting motor speed has been performed at the register level using ARM C programming on the microcontroller STM32F103C8T6A. The low-level cost components used in the system, such as the microcontroller STM32, motor driver L298N, and Li-ion batteries, made the system efficient and effective to be used for practical applications and educational purposes.

Finally, the project has been able to successfully implement PID control and adaptive cruise control in a line-following robot with very promising results on performance and adaptability.

Future improvements would be in developing obstacle detection with more sensors and using advanced methods of PID tuning for finer responsiveness and stability of the robot. The result of this work is expected to contribute to autonomous mobile robots and thus will act as the basis for further research works in the area of navigation and control.

Video Link: EmbeddedSystemsProject

REFERENCE

[1] H. O. Bansal, R. Sharma, and P. R. Shreeraman, "PID Controller Tuning Techniques: A Review," *Journal of Control Engineering and Technology (JCET)*, vol. 2, no. 4, pp. 168–176, 2012.

[2] S. S. Suryawan, S. M. Musamwar, S. R. Kolhe, S. A. Thengane, S. S. Hanumante, and P. H. Sahare, "Line Follower & Obstacle Avoider Robot," *International Research Journal of Engineering and Technology (IRJET)*, vol. 6, no. 12, pp. 460–468, 2019.