# ADAPTIVE CRUISE CONTROL

**A PROJECT REPORT**

*submitted in partial fulfillment of the requirements*

*for the award of the degree of*

**BACHELOR OF TECHNOLOGY**

*in*

**ELECTRICAL AND ELECTRONICS ENGINEERING**

*by*

**ROHAN CHANDRASEKAR(17BEE1099)**

**MALLESH N (17BEE1181)**

*under the guidance of*

**DR.D. SUBBULEKSHMI**

**SCHOOL OF ELECTRICAL ENGINEERING**

**VIT CHENNAI**

***June 2021***

## CERTIFICATE

This is to certify that the project work titled **"ADAPTIVE CRUISE CONTROL"** submitted by ***ROHAN.C AND MALLESH.N*** is in partial fulfillment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY**, is a record of bona fide work done under my guidance. The contents of this project work, in full or in parts, have neither been taken from any other source nor have been submitted to any other Institute or University for award of any degree or diploma and the same is certified.

Dr.D.Subbulekshmi
**Project Supervisor**

**The thesis is satisfactory / unsatisfactory**

Approved by

**Head of the Department**
**B.Tech Electrical and Electronics Engineering**

**The thesis is satisfactory/unsatisfactory**

**(Internal Examiner)** **(External Examiner)**

## ACKNOWLEDGEMENT

First and foremost, we would like to thank the Academics management of Vellore Institute of Technology, Chennai Campus, for giving us the well-awaited opportunity to work on such an interesting and wholesome project.

We are grateful to the SELECT Department for the continued support and their belief in our competency which made us determined enough to see this project through to its end.

Next up, we would thank all the faculty and especially our guide Dr. D.Subbulekshmi, for her continued critics and supervision without which we would never have succeeded in what we did.

Last but not the least, we would pay our full gratitude and respect to our parents who have acted as our backbone throughout the years without which this feat would've never been possible in the first place.

**Rohan Chandrasekar**
**(17BEE1099)**

**Mallesh N**
**(17BEE1181)**

## ABSTRACT

Adaptive Cruise Control is a system in which the speed of a vehicle is controlled whilst maintaining a safe distance with the vehicle in front. Adaptive Cruise Control is being used extensively in many cars nowadays. The main focus of this project is to develop a multi-objective controller. The multi-objective controller employs three controllers in its setup, each for a specific operating range. The advantage of a multi-objective controller is that it can be used for a large operating range, while putting less strain on each of its controller which thereby ensures longevity of the controller's life. To select the controller that has to be used in the multi-objective controller setup, an adaptive cruise control system is developed and the responses of two different controllers : Proportional Integral Derivative controller(PID) and Fuzzy-Proportional Integral Derivative controller(F-PID) are compared. Since the PID controller gives us a better response, it is then used to design a multi-objective controller. The multi-objective controller is designed using MATLAB Simulink and Python. The output is then passed through a Kalman Filter to filter out the disturbances and estimate the velocity accurately. The hardware model of the adaptive cruise control setup is designed using an Arduino UNO. The speed of the motor is varied according to the distance measured by the ultrasonic sensor.

## TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# ABBREVIATIONS AND NOMENCLATURE

| | |
|---|---|
| CACC | Cooperative adaptive cruise control |
| ACC | Adaptive Cruise Control |
| P | Proportional |
| PI | Proportional-Integral |
| PID | Proportional-Integral-Derivative |
| MISO | Multi-input single-output |
| LCD | Liquid-Crystal Display |
| LED | Light Emitting Diode |

# CHAPTER I

## 1. INTRODUCTION

### 1.1 INTRODUCTION

The development of an efficient cruise control system has been in the forefront for automobile companies and researchers for some time now. The cruise control system accurately maintains the driver's desired set speed. Although this is an efficient way to drive on highways and other places where there is not much traffic, a normal cruise control system cannot be used to drive in roads which are densely packed. That is because a normal cruise control system is only responsible for controlling the speed of the vehicle and does not take into consideration the distance between the vehicle and the other vehicles on the road. This may lead to accidents. This is the reason why many automobile companies and researchers are shifting their focus to Adaptive Cruise Control. Adaptive Cruise Control can control the speed of a vehicle whilst maintaining a safe distance with the vehicle in front. Consequently, the Adaptive Cruise Control System eliminates the drawback of the conventional cruise control system. Most of the cruise control systems use a sensor to detect the vehicle in front. Sometimes these sensors are late at detecting the vehicle in front. This may lead to accidents. To nullify this problem, we are introducing a multi-objective controller. A multi-objective controller is a controller that varies the controller used according to the distance between the cars. Each controller is operated in a different operating range.

#### 1.1.1 Motivation

Adaptive Cruise Control has been in the forefront of research for many years now. It is a very useful technology for long drives. It helps in reducing driver fatigue across highways and sparsely populated roads, thereby ensuring driver comfort. It is also a very useful technology that drivers use to avoid violating speed limits as the car can be set to the speed that the driver desires. Adaptive cruise control also helps to improve fuel efficiency of the vehicle. Adaptive Cruise Control has a huge scope for development in the future. A concept known as Cooperative adaptive cruise control (CACC) is being studied right now in which apart from the feedback loop of the ACC, the acceleration of the lead vehicle is used in a feed-forward loop. The CACC concept improves stability of the system and reduces delays so that the system can react faster.

### 1.1.2 Objectives

The main objectives of the project are:

- To model a car system that emulates real world conditions.
- To design a PID controller and a Fuzzy-PID controller for the system and compare their responses.
- To design a multi-objective controller using the controller that previously gave the best response in MATLAB Simulink.
- To design the same multi-objective controller using Python.
- To design a Kalman Filter to filter out the measurement noises from the multi-objective controller's output and accurately measure its speed.
- To construct a hardware setup of adaptive cruise control using Arduino UNO.

### 1.1.3 Scope of the Work

The concepts that have been covered in this project are:

- Modelling a system based on real world scenarios.
- Implementing a Fuzzy-PID and a PID controller.
- Designing and Implementing a Multi-Objective Controller in MATLAB Simulink and Python.
- Developing a hardware model of adaptive cruise control.

## 1.2 ORGANIZATION OF THESIS

The organization of the thesis is as follows:

- In Chapter 2, we give the overview of the project followed by the literature review of the project. We then follow this by discussing the different modules of the project. The tasks and milestones of the project make up the final component of Chapter 2.
- In Chapter 3,we discuss the design of our car system thoroughly.
- Chapter 4 consists of the project demonstration. First Simulink models are discussed followed by their respective results. Then the hardware model is discussed followed by its results.
- Chapter 5, which is the last chapter of the report, consists of the conclusion, cost analysis and the future scope of the project.

# CHAPTER II

# 2. PROJECT DESCRIPTION

## 2.1 OVERVIEW OF PROJECT

The main objective of our project is to design and construct a multi-objective controller. In order to develop new concepts into prototypes and ultimately into products, physical system modeling is virtually a necessity. Initially we model our car system according to real world conditions. Then we implement adaptive cruise control to this system. The system is controlled using a PID controller and a Fuzzy-PID controller and their respective responses are compared. The PID controller gives us a better response and therefore it is then used to design the multi-objective controller. The multi-objective controller is designed in MATLAB Simulink and in Python. A Kalman filter is then used to filter the measurement noises and estimate the speed correctly. A hardware model of the adaptive cruise control system is constructed and programmed using an Arduino UNO.

Several papers have been published in these topics. [1] explores how an adaptive cruise control system is used on a car by using a PID controller to control the velocity and a fuzzy logic controller to control the distance. An adaptive cruise control system designed to work under the Stop and Go scenarios is designed and implemented in [2]. [3] and [12] explains what is an adaptive cruise control system and how it is incorporated in a vehicle. Fuel consumption and traffic congestion optimization has been considered while designing an adaptive cruise control strategy in [4]. PID controller with feed-forward control is proposed for a nonlinear model in [5]. [6] describes the different sensors and actuators that are used in an adaptive cruise control system and assesses their respective advantages and disadvantages. Adaptive Cruise Control on a battery electric vehicle is explored in [7], which also considers regenerative braking in the process. [8] explores implementing ACC using an online optimization controller. The distance between the cars is represented by a control barrier function and the velocity control is represented by a control Lyapunov function. This is then unified through a quadratic program, which balances safety and control objectives optimally.[9] implements adaptive cruise control using fuzzy logic and neural network and compares their respective results. [10] designs an adaptive Kalman filter that estimates the acceleration of the lead car, which is then implemented as a feed-forward signal to the car in which adaptive cruise control has been implemented. In [11], two control loops are being designed; one to control the speed of the vehicles and the second one to adjust the speed of the vehicle according to the distance between the vehicles.[13] explains how a Fuzzy-PID controller is implemented in an automatic cruise control system.[14] compares the performances of the P, PI and the PID controller in an adaptive cruise control system. The Kalman Filter and the extended Kalman filter are explained in [15] and [19].[16] explains the different industrial applications of the Kalman filter and its issues. [17] explores how components from various design phases can be integrated using models. This concept is applied to an adaptive cruise control system and the results are analysed.[18] reviews the control strategies currently present and then proposes a framework for the development of adaptive cruise control.[20] discusses how vehicle position is updated according to the Kalman filter prediction.

## 2.2. MODULES OF THE PROJECT

### 2.2.1 Module 1- Adaptive Cruise Control using PID Controller

The PID controller is used extensively in industrial control applications. It uses a control loop feedback mechanism to control the process variable. A PID controller is a combination of proportional, integral and derivative control on the system's error signal. The output equation of a PID controller is given by:

$$u = k_p e(t) + k_i \int e(t)dt + k_d \frac{de(t)}{dt} \qquad (1)$$

Where u is the control signal, $K_p$ is the proportional gain constant, $K_i$ is the integral gain constant, $K_d$ is the derivative gain constant and e(t) is the error signal which is the difference between the set-point and the measured process variable.

The PID controller was applied to the adaptive cruise control system. It was tuned using the Cohen Coon Process Reaction Curve Technique. The Cohen Coon method is a method of tuning in which a step change can be introduced to the input at steady state and the output can be measured based on the time delay and time constant. The Cohen-Coon tuning rules work well on virtually all self-regulating processes and is designed to give a very fast response. The values obtained for the PID controller gains after tuning are $K_p$=27.7593, $K_i$=3.9352 and $K_d$=24.0508.

### 2.2.2 Module 2- Adaptive Cruise Control using Fuzzy-PID Controller

A fuzzy logic controller is one of the major controllers that is extensively used in industries and industrial applications. Fuzzy logic uses linguistic variables to quantify vague terms. The fuzzy controller is a multi-input single output (MISO) system. Designing a fuzzy logic controller includes defining a set of membership functions for the inputs and outputs an creating a set of rules that relate the inputs and the outputs. The input usually is the error e and the change in error $\frac{de}{dt}$. The error is the difference between the set-point value and the measured process variable and the change in error is the difference between the errors at time t and t-1. Here three membership functions for each of the inputs, five membership functions for the output and nine rules to map the inputs to the outputs are being used. The Sugeno Fuzzy Inference System is being used here.

The error input varies from -1000 to 1000 and consists of three triangular membership functions as shown in Figure 1.

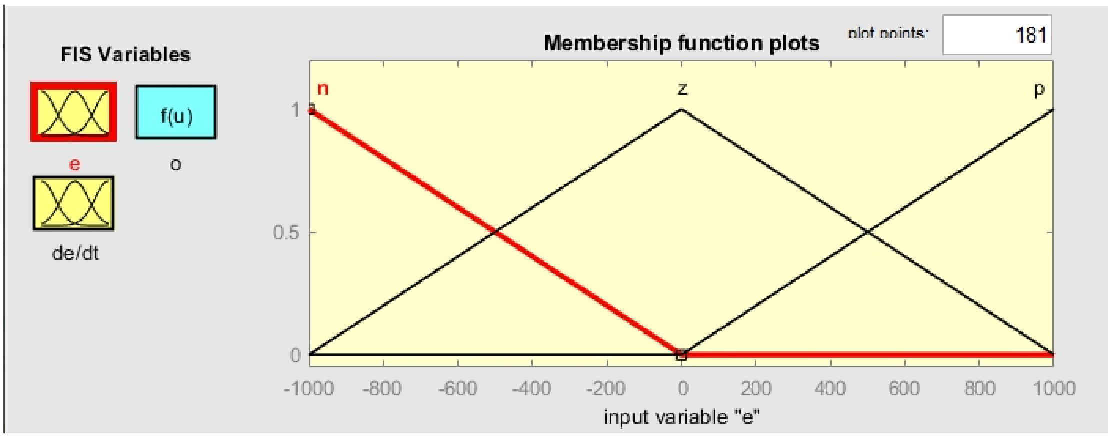


**Figure 1-Error membership function plots**

The change in error input also varies from -1000 to 1000 and consists of three triangular membership functions as shown in Figure 2.

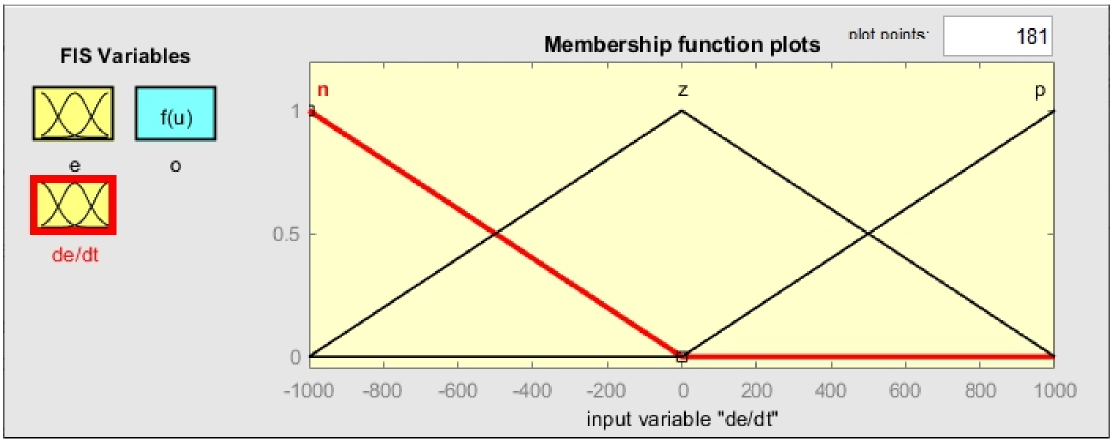


**Figure 2-Change in Error membership function plots**

The output linguistic variables are given in Table 1.

**Table 1: Linguistic variable and its value**

| Linguistic Variable | Value Assigned |
|---|---|
| LP | 10000 |
| P | 5000 |
| Z | 0 |
| N | -5000 |
| LN | -10000 |

Here LP stands for Large Positive, P stands for Positive, Z stands for Zero, N stands for Negative and LN stands for Large Negative.

The nine rules that have been formulated are displayed in Table 2.

**Table 2-Fuzzy Rules**

| e \ $\frac{de}{dt}$ | P | Z | N |
|---|---|---|---|
| P | LP | P | Z |
| Z | P | Z | N |
| N | Z | N | LN |

A fuzzy rule in a Sugeno Fuzzy model is of the form:

**If x is A and y is B, then z=f(x,y)**

Where A and B are fuzzy sets in the antecedent and z=f(x,y) is a crisp function in the consequent.

In the table above, the antecedent part (e and de/dt) is expressed by: If ….and…. while the consequent part (the outputs) is expressed by then…For example the first rule in the table is: If e is P and de/dt is P then output is LP.

As we can see in the Figure 3 below, the surface is linear which proves that the relationship between the inputs and the output is linear.

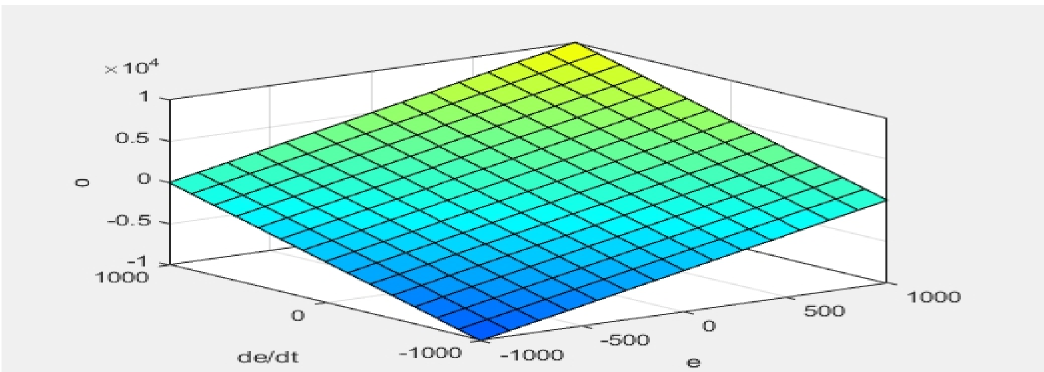


**Figure 3-Fuzzy Surface**

A Fuzzy-PID controller is a hybrid controller which consists of two controllers put together for better performance. The Fuzzy-PID controller was tuned using the

Cohen-Coon Process Reaction Curve Technique and the values of the PID controller gains are $K_p$=13.4671, $K_i$=1.2161, Kd=4.8089.

### 2.2.3 Module 3-Multi-Objective Controller

A single controller may not react as fast as its required to. This may lead to accidents. Incorporating various controllers and activating them according to the relative distance between the ego car and the leading car is more efficient.
Even though it is more complex than a single controller system, it ensures safety. It puts less burden on each controller thereby ensuring longevity of life of the controller. Another major advantage of the multi-objective controller is that each controller can be tuned according to the situation at hand which results in a better response.

We are incorporating three PID controllers in this setup. For a distance greater than 600m, the values of the first PID controller gains are: $K_p$=4.5,$K_i$=0.75 and $K_d$= 0.5. For a distance less than 200m, the values of the second PID controller gains are: Kp=1, Ki=1, Kd= 0.5. If the distance is in between these two ranges, the values are: Kp=5, Ki=1 and $K_d$= 0.7.

### 2.2.4 Module 4-Kalman Filter Implementation in the Multi-Objective Controller

Kalman Filter is an estimation algorithm that uses measurements over time which contain noises and other disturbances and gives an estimate of the unknown variables by estimating a joint probability distribution over the variables for each time frame.
The Kalman Filter is applied to the multi-objective controller which contains measurement noises. The Kalman Filter filters out the noise and estimates the speed of the vehicle.The sample time Ts for the Kalman filter is set as 0.1s. The value of Q, which is the covariance matrix associated with the noises in states, is set as 0.3 and the value of R, which is the covariance matrix of the measurement noise, is set as 0.8.

## 2.3 TASKS AND MILESTONES

**Table 3-Tasks and Milestones**

| S.No. | Tasks | Start date | End date |
|---|---|---|---|
| 1 | Literature review | 25-01-2021 | 02-02-2021 |
| 2 | Understanding the concepts of adaptive cruise control. | 03-02-2021 | 10-02-2021 |
| 3 | Implementing adaptive cruise control: - Designing and implementing adaptive cruise control in MATLAB Simulink. | 13-02-2021 | 20-02-2021 |
| 4 | Analyzing the response: - Understanding the response of the implemented adaptive cruise control in MATLAB Simulink. | 20-02-2021 | 22-02-2021 |
| 5 | Implementing a controller: - Designing a PID controller for our adaptive cruise control model in MATLAB Simulink. | 24-02-2021 | 07-03-2021 |
| 6 | Analyzing the response: - Understanding the response of the adaptive cruise control using a PID controller in MATLAB Simulink. | 07-03-2021 | 09-03-2021 |
| 7 | Implementing a different controller: - Designing a Fuzzy-PID Controller for our adaptive cruise control in MATLAB Simulink. | 11-03-2021 | 28-03-2021 |
| 8 | Analyzing the response: - Understanding the response of the adaptive cruise control using Fuzzy-PID controller in MATLAB Simulink. | 28-03-2021 | 30-03-2021 |
| 9 | Comparing the responses between both the controllers. | 02-04-2021 | 04-04-2021 |
| 10 | Implementing a new model: - Designing a multi-objective PID controller for our adaptive cruise control in MATLAB Simulink. | 09-04-2021 | 19-04-2021 |
| 11 | Analysing the response: - Understanding the response of the multi-objective controller in MATLAB Simulink. | 19-04-2021 | 21-04-2021 |
| 12 | Using a Kalman filter to improve the response of the multi-objective PID controller. | 21-04-2021 | 30-04-2021 |
| 13 | Implementing a hardware model of the cruise control system. | 25-03-2021 | 12-05-2021 |

# CHAPTER III

# 3. DESIGN OF A CAR SYSTEM

## 3.1 DESIGN APPROACH

We designed a system by emulating real world conditions. We considered all our parameters to almost match the real-world scenario. We start by drawing the free body diagram of the car and writing the force equation; using which we derive the transfer function of the system.

## 3.2 DESIGN SPECIFICATIONS

The adaptive cruise control system maintains a driver set speed and also maintains a safe distance with the vehicle in front. Figure 4 depicts the free body diagram of the car.

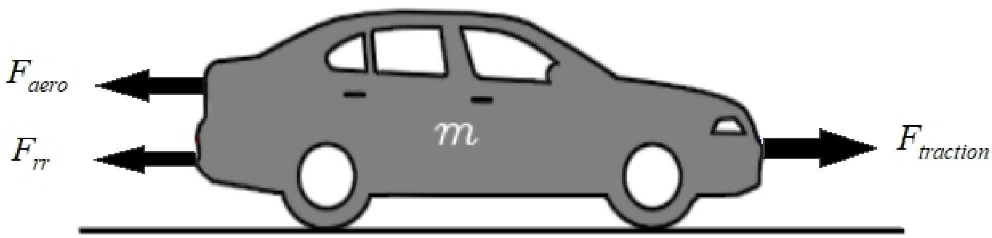


**Figure 4-Free Body Diagram of the car**

The longitudinal vehicle dynamics as governed by Newton's Law is:

$$m\frac{dv}{dt} = F_{traction} - F_{aero} - F_{rr} \qquad (2)$$

where $m\frac{dv}{dt}$ is the inertial force, $F_{aero}$ is the aerodynamic drag force on the car, which is the force on the car that resists its motion through air. $F_{rr}$ is the rolling resistance force on the car, which is the force resisting the motion of the wheels of the car when it rolls on the road.

The aerodynamic force on the car is given by:

$$F_{aero}=0.5\times P \times C_d \times A_f \times v^2 \qquad (3)$$

where P is the density of air which is equal to 1.2 $\frac{kg}{m^3}$, $C_d$ represents the coefficient of aerodynamic drag, which is equal to 0.3, $A_f$ is the frontal area of the car which is $2m^2$. Therefore, aerodynamic drag force on the car is:

$$F_{aero}= 0.5 \times 1.2\times 2\times 0.3\times v^2 = 0.36v^2 \quad (4)$$

The rolling resistance force on the car is given by:

$$F_{rr} = C_{rr} \times m \times g \qquad (5)$$

where $C_{rr}$ is the rolling resistance coefficient which is equal to 0.02, m is the mass of the car which is 1200kg and g is the acceleration due to gravity which is $9.81\frac{m}{s^2}$.Therefore, the rolling resistance force is given by:

$$F_{rr} = 0.02 \times 1200 \times 9.81 = 235\text{N} \qquad (6)$$

Therefore, the equation becomes:

$$F - 235 - 0.36v^2 = m\frac{dv}{dt} \qquad (7)$$

Since the equation is non-linear, we linearize the equation about F=1000.

$$0 = 1000 - 235 - 0.36v^2$$
$$v^2 \approx 2125$$
$$v \approx 46 \text{m/s}$$

So, the operating point about which we are going to linearize is (1000,46).

$$F(F, v, \frac{dv}{dt}) = F - 235 - 0.36v^2 - m\frac{dv}{dt} \qquad (8)$$

$$\frac{df}{dF} = 1$$
$$\frac{df}{dv} = -0.72v|_{v=46m/s} = -33.12$$
$$\frac{df}{d\dot{v}} = -m = -1200$$

Applying Taylor's Series and neglecting the higher order terms, we get

$$f(F, v, \dot{v}) \approx f(\overline{F}, \overline{v}, \overline{\dot{v}}) + \frac{df}{dF} \times (F - \overline{F}) + \frac{df}{dv} \times (v - \overline{v}) + \frac{df}{d\dot{v}} \times (\dot{v} - \overline{\dot{v}}) \qquad (9)$$
$$f(F, v, \dot{v}) \approx 0 + 1 * (F - \overline{F}) - 33.12 \times (v - \overline{v}) - 1200 \times (\dot{v} - \overline{\dot{v}})$$
$$F - \overline{F} = \hat{F}$$
$$v - \overline{v} = \hat{v}$$
$$\dot{v} - \overline{\dot{v}} = \widehat{\dot{v}}$$
$$\hat{F} - 33.12\hat{v} - 1200\hat{\dot{v}} = 0$$
$$\hat{F} - 33.12\hat{v} - 1200s\hat{v} = 0$$
$$\frac{\hat{v}(s)}{\hat{F}(s)} = \frac{1}{1200s + 33.12} \qquad (10)$$

Equation 5 is the transfer function of the system with the traction force of the car being the input and the speed being the output.

# CHAPTER IV

# 4. PROJECT DEMONSTRATION

## 4.1 INTRODUCTION

Initially we simulated adaptive cruise control with two different controllers(PID and Fuzzy-PID) and compared their responses. The one which gave a better response was the chosen to be incorporated in the multi-objective controller. The multi-objective controller was implemented in both MATLAB Simulink and Python. The output of the multi-objective controller in MATLAB Simulink was then passed through a Kalman filter to filter out the disturbances. The hardware model of an adaptive cruise control setup was also constructed.

## 4.2 SIMULATION RESULTS

**Simulation of Adaptive Cruise Control:**

The simulation setup for the adaptive cruise control technique is shown in Figure 5 below.

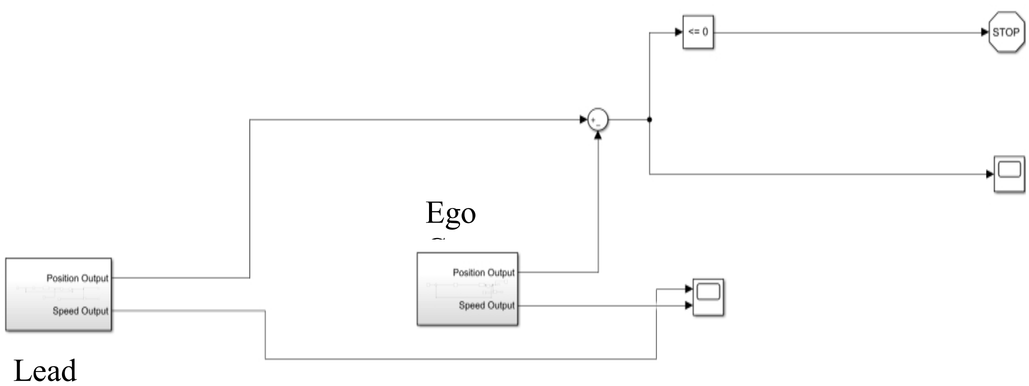


**Figure 5-Adaptive Cruise Control Setup**

The Ego car is the one in which the cruise control system is being deployed and it is following the Lead car.

The lead car starts at 25m/s and 1000m away from the ego car and slowly reduces its speed to 20m/s as shown below in Figure 6.A saturation block is added to the lead car so that the speed of the lead car does not cross 30m/s. The transfer function of the system (equation 5) is converted to the state space form here with A=0.0276, B=8.333e-4, C=1 and D=0.

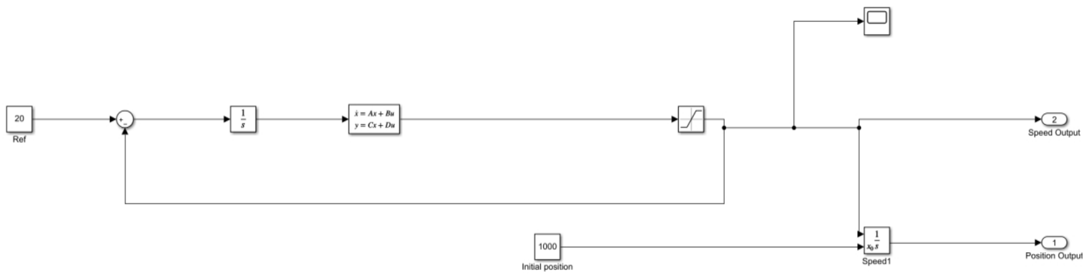


**Figure 6-Lead Car Setup**

The PID Controller and the Fuzzy-PID were applied on the Cruise Control system and their respective simulation diagrams and graphs are shown below.

**Simulation of PID Controller:**

The ego car setup is shown below in Figure 7. The ego car starts from origin and starts at 40m/s. A PID controller is tuned and deployed so that the car attains the set-speed of 20m/s. A saturation block is added to the ego car so that the speed of the lead car does not cross 40m/s. The speed response of the ego car is shown below in Figure 8.

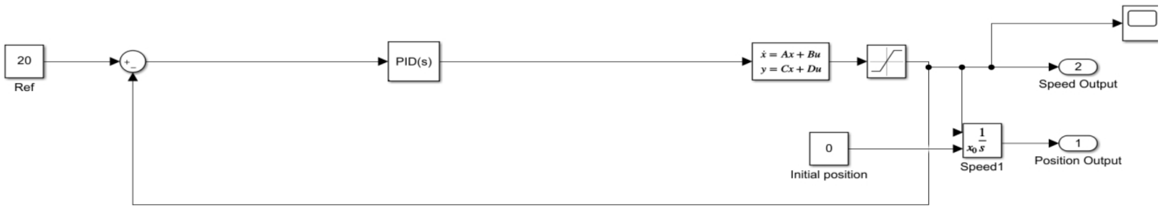


**Figure 7-Ego Car Setup**

**Results of PID Controller:**

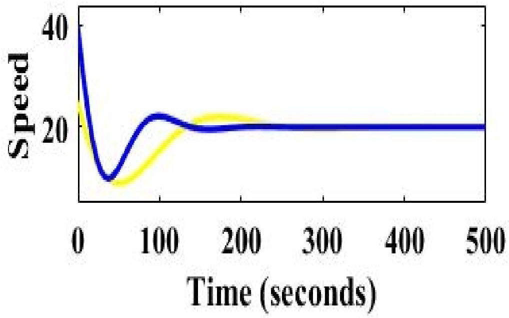


**Figure 8-Speed variation with time**

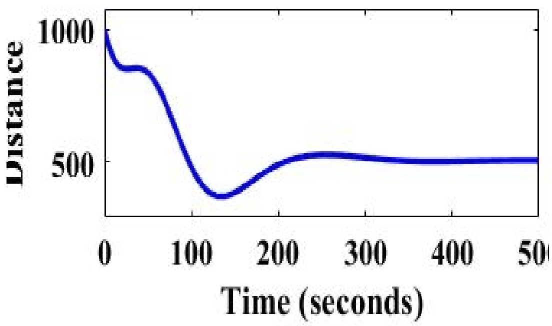


**Figure 9-Relative Distance variation with time**

We see in Figure 8 that the speed of the ego car starts at 40m/s and settles at 20m/s eventually because of the PID controller. The response has an undershoot and takes

171 seconds to settle to the set-point. It can be seen in Figure 9 the vehicles never come too close and at all times maintain a significant distance from each other.

**Simulation of Fuzzy-PID Controller:**

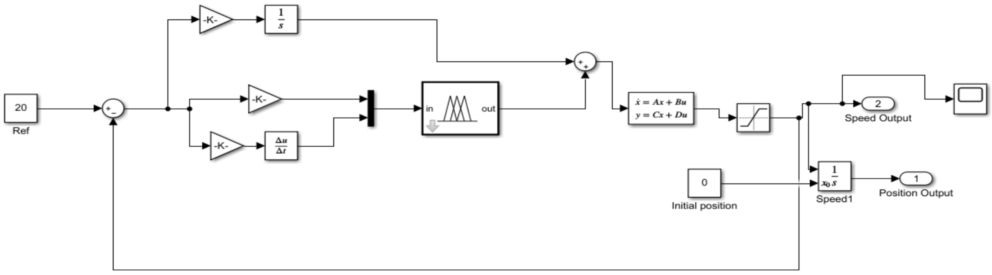


**Figure 10 -Ego Car Setup**

To the same ego car setup we replace the PID controller with a Fuzzy-PID controller, which is shown in Figure 10. The speed variations and the starting point of this setup is the same as the PID controller setup. The speed response of the ego car is shown below in Figure 11.

**Results of Fuzzy-PID Controller:**

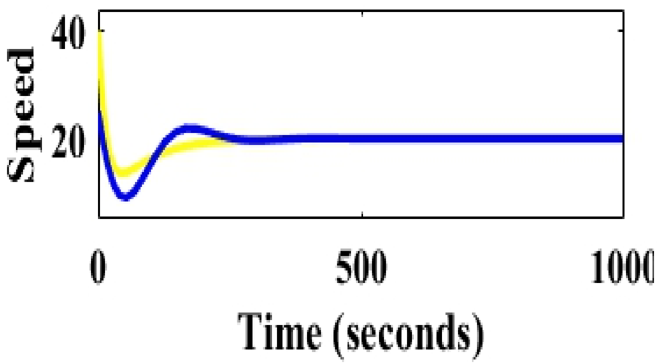


**Figure 11-Speed variation with time**

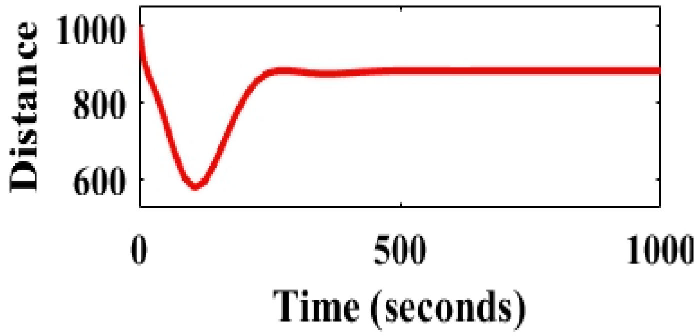


**Figure 12-Relative Distance Variation with time**

The speed variation graph of the Fuzzy-PID controller has an undershoot and takes 250s to settle as seen in Figure 11. Figure 12 confirms that the cars are maintaining a safe distance with each other. The responses of the controllers are compared in the table below.

**Comparison of PID and Fuzzy-PID:**

**Table 4-Comparison of PID and Fuzzy-PID**

| Controller | Max.Overshoot(%) | Settling time in s | Rise Time | Steady State Error |
|---|---|---|---|---|
| PID | 100% | 170.9027 | 12.1256 | 4.627e-05 |
| Fuzzy-PID | 100% | 249.5769 | 11.8595 | 0.0099 |

The comparison of PID controller and Fuzzy-PID controller is shown in Table 4. Even though Fuzzy-PID has a lower rise time than PID, the PID controller settles way quicker than the Fuzzy-PID and also has significantly less steady state error when compared to the Fuzzy-PID controller. Therefore, we conclude that the PID controller gives us a better response than the Fuzzy-PID controller.

Since the PID controller gives us a better input, we are going to use it to construct the multi-objective controller.

## Simulation of the Multi-Objective Controller:

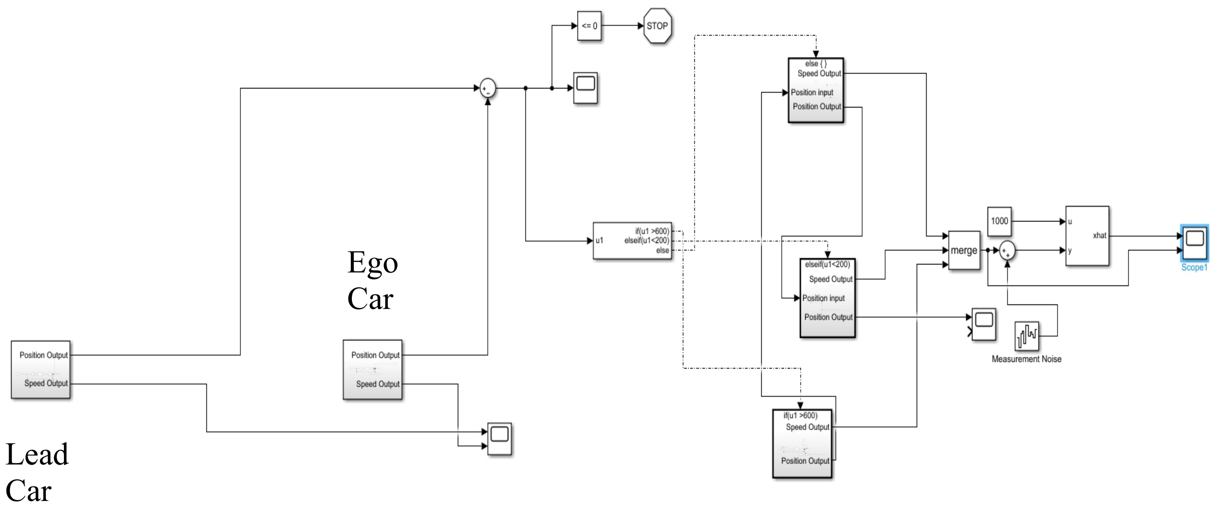


**Figure 13-Multi-Objective Controller Setup with a Kalman Filter**

Figure 13 depicts the multi-objective controller setup. The lead car starts at 800m with a speed of 40m/s and gradually slows down to 15m/s. The ego car starts at origin and at 50m/s for a relative distance greater than 600m.The controllers vary in activity according to the relative distance between the lead car and the ego car. If the relative distance between the cars is less than 600m but more than 200m, the speed of the ego car slows down to 25m/s. If the relative distance between the cars drops down below 200m, the speed of the ego car goes to 15m/s. Even though the system is more complex than a regular system, it ensures that the system is safe. We are also introducing a measurement noise to the system to emulate real-world conditions. The speed response of the ego car is depicted in Figure 14 below.

**Result of Multi-Objective Controller:**

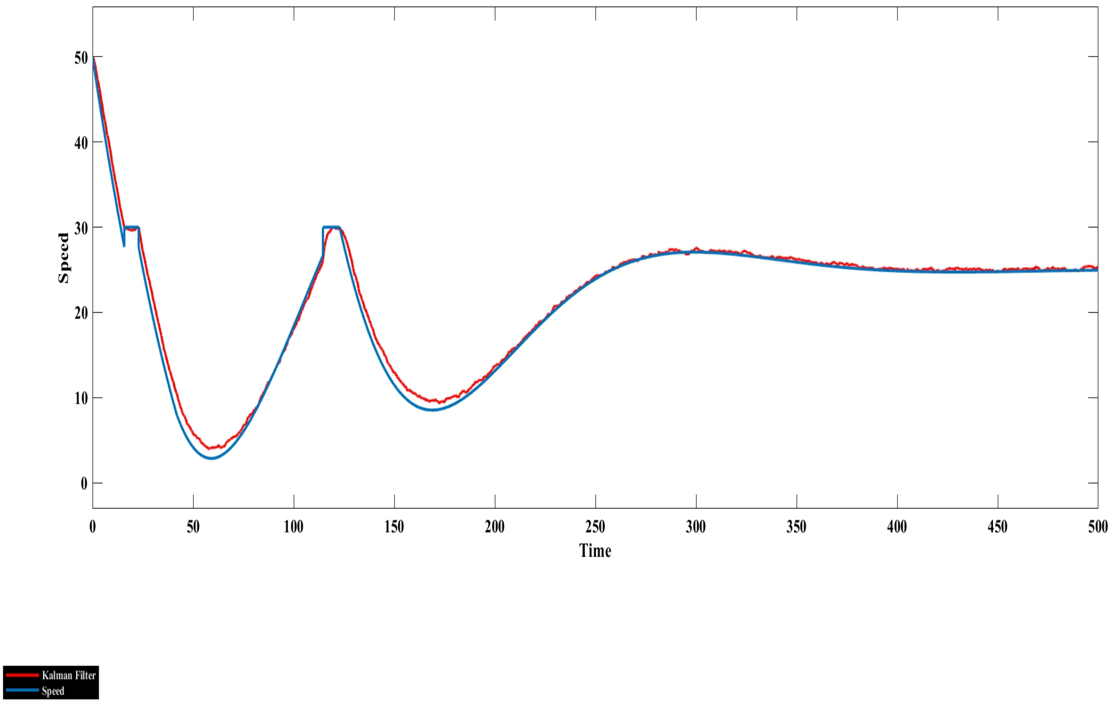


**Figure 14-Speed variation with time for a multi-objective controller**

In the graph above, we can see that the system is adjusting its speed according to the relative distance between the two cars. The response is smooth and more comfortable for the passengers as the speed variations are not drastic. We can also see that the Kalman Filter filters out the measurement noise and estimates the speed almost accurately.

## Simulation of the Multi-Objective Controller using Python:

Python is a high level and a robust programming language. The multi-objective controller was coded in python and the results were analysed.

**Flowchart:**

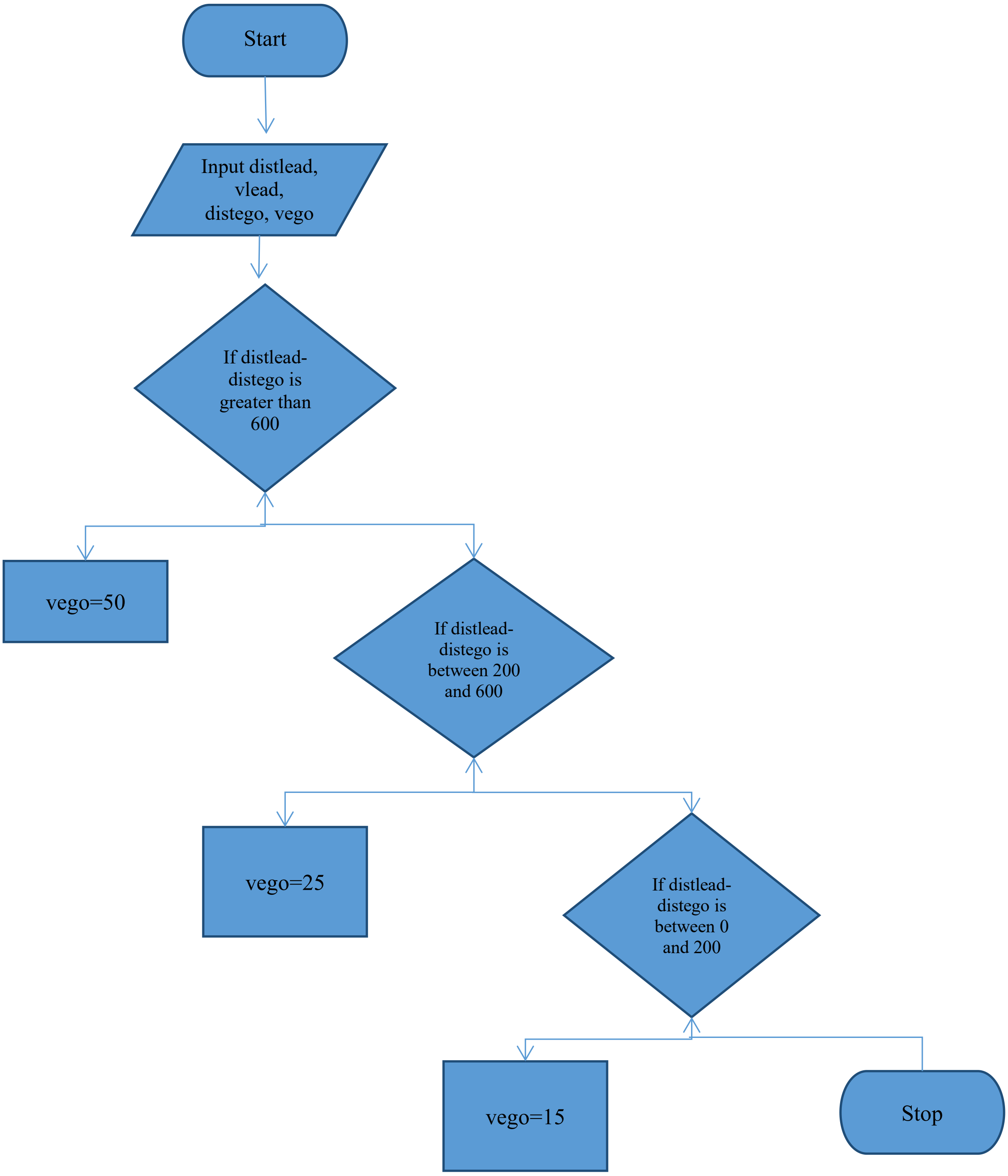


**Figure 15-Flowchart of a multi-objective controller in Python**

Figure 15 represents the flowchart of the multi-objective controller's code in python. Here, distlead denotes the lead car's position, distego denotes the ego car's position, vlead denotes the speed at which the lead car is going at and vego denotes the speed of the ego car.

The multi-objective controller was implemented in python and the following results were obtained.

**Results:**

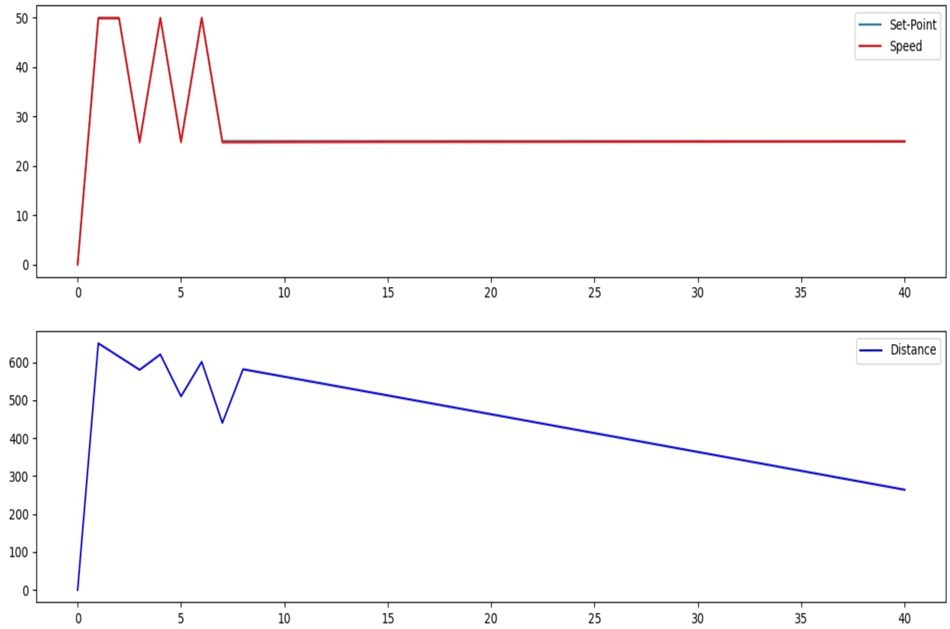


**Figure 16-Speed and Corresponding Distance vs Time**

In Figure 16, we can see the speed varying according to the distance. The speed variation conditions are the same as the one implemented in MATLAB Simulink. For a distance greater than 600m, the speed of the vehicle is 50m/s. When the distance drops below 600m, the speed drops to 25m/s.

## 4.3 HARDWARE RESULTS

The hardware setup is to develop a simple adaptive cruise control mechanism using the Arduino Uno, which is shown in Figure 17. The working of the setup is such that the Arduino is tuned as a PID controller to control the speed of the motor based on the set speed, which is set using the potentiometer. The distance between the device and the obstacle in front is sensed by the ultrasonic HC-SR04 sensor. The distance sensed is fed to the Arduino and based on the tuning of the PID controller the DC motor operates. During the acceleration of the speed one of the servo motors starts functioning and during the braking the of the speed the other servo motor starts functioning. The Kp, Ki and Kd values are the same values used in the simulation part. The LCD display displays the set value of speed and set distance to which the PID controller must start working and it also displays the distance between the device and the obstacle in front and also displays the speed of the motor. The potentiometer is used to vary the set speed and set distance in which the tuning of PID to function.

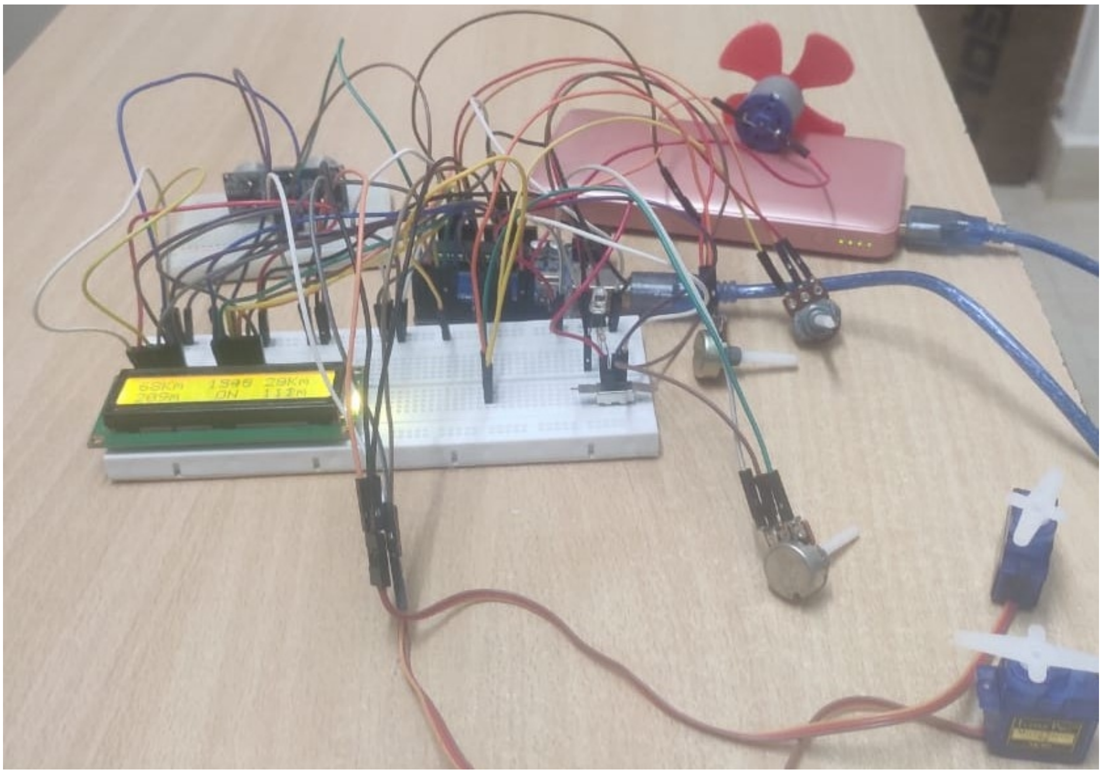

**Figure 17-Hardware Setup**

**Components Used:**

The components used in the hardware setup are:

1. Arduino UNO
2. Ultrasonic Sensor
3. Dc Servo Motor
4. Lcd Display
5. Rotary Potentiometer
6. Resistor
7. LED's

The function of each component is briefly explained below:

1.Arduino Uno:

The Arduino Uno is an open-source microcontroller board based on the Microchip ATmega328P microcontroller. The board is equipped with sets of digital and analog input/output pins that may be interfaced to various expansion boards and other circuits. This board includes digital I/O pins-14, a power jack, analog i/ps-6, ceramic resonator-A16 MHz, a USB connection, an RST button, and an ICSP header and is programmable with the Arduino IDE. The operating voltage of the Arduino Uno is 5V, the input voltage ranges from 6V to 20V and the DC current for each input/output pin is around 40mA.

2.Ultrasonic Sensor:

The Ultrasonic sensor used for this hardware setup is HC-SR04. An ultrasonic sensor is an electronic device that measures the distance of an object by emitting ultrasonic sound waves, and converts the reflected sound into an electrical signal. Ultrasonic sensors have two main components: the transmitter, which emits the sound using piezoelectric crystals and the receiver, which encounters the sound after it has travelled to and from the target. The HC-SR04 consists of 4 pins namely Vcc, Trigger, Echo and Ground. The Vcc pin is supplied 5V to power the ultrasonic sensor, the Trigger pin is the input, this pin has to be kept high for 10us to initialize measurement by sending US wave, the Echo pin is the output, this pin goes high for a period of time which will be equal to the time taken for the US wave to return back to the sensor, the Ground pin is connected to the ground of the system. The measuring distance range for the HC-SR04 is 5cm-450cm and the measuring angle is lesser than 15º.

3.Dc Servo Motor:

A DC motor along with servomechanism, closed-loop control system acts as a servo motor which is basically used as a mechanical transducer in the automation industry. Based on its accurate closed-loop control, it has versatile applications used in many industries. The DC servo motor definition is, a motor that is used in servo systems is known as a servo motor. A servo system is a closed-loop system where the feedback signal (position, velocity, acceleration, etc.) drives the motor. This signal acts as an error and based on controller, accurate position or velocity is achieved. The motors are coupled to an output shaft (load) through a gear train for power matching. Servo motor acts as a mechanical transducer as they convert an electrical signal to an angular velocity or position.

4.LCD Display:

The LCD display we use is LCD 1602a. An LCD is an electronic display module that uses liquid crystal to produce a visible image. The operating voltage of the LCD display is around 5V. The 1602a LCD display has two rows which can produce up to 16 characters. The utilization of current is 1mA with no backlight. A 16×2 LCD has two registers like data register and command register. The RS, register select, is mainly used to change from one register to another. When the register set is ‘0’, then it is known as command register, when the register set is ‘1’, then it is known as data register.

5.Rotary Potentiometer:

The potentiometer, commonly referred to as a “pot”, is a three-terminal mechanically operated rotary analogue device which can be found and used in a large variety of electrical and electronic circuits. Variable potentiometers are available in a variety of different mechanical variations allowing for easy adjustment to control a voltage, current, or the biasing and gain control of a circuit to obtain a zero condition. Rotary potentiometer varies their resistive value as a result of an angular movement. Rotating a knob or dial attached to the shaft causes the internal wiper to sweep around a curved resistive element. The most common use of a rotary potentiometer is the volume-control pot. Rotary potentiometers can produce a linear or logarithmic output with tolerances of typically 10 to 20 percent. As they are mechanically controlled, they can be used to the measure the rotation of a shaft, but a single-turn rotary potentiometer normally offers less than 300 degrees of angular movement from minimum to maximum resistance.

6.Resistor:

A resistor is a passive two-terminal electrical component that implements electrical resistance as a circuit element. In electronic circuits, resistors are used to reduce current flow, adjust signal levels, to divide voltages, bias active elements, and terminate transmission lines, among other uses. The behavior of an ideal resistor is dictated by the relationship specified by Ohm's law. Ohm's law states that the voltage across a resistor is proportional to the current, where the constant of proportionality is the resistance.

7.LED:

A light-emitting diode is a semiconductor light source that emits light when current flows through it. Electrons in the semiconductor recombine with electron holes, releasing energy in the form of photons. The color of the light determined by the energy required for electrons to cross the band gap of the semiconductor.

**Results:**

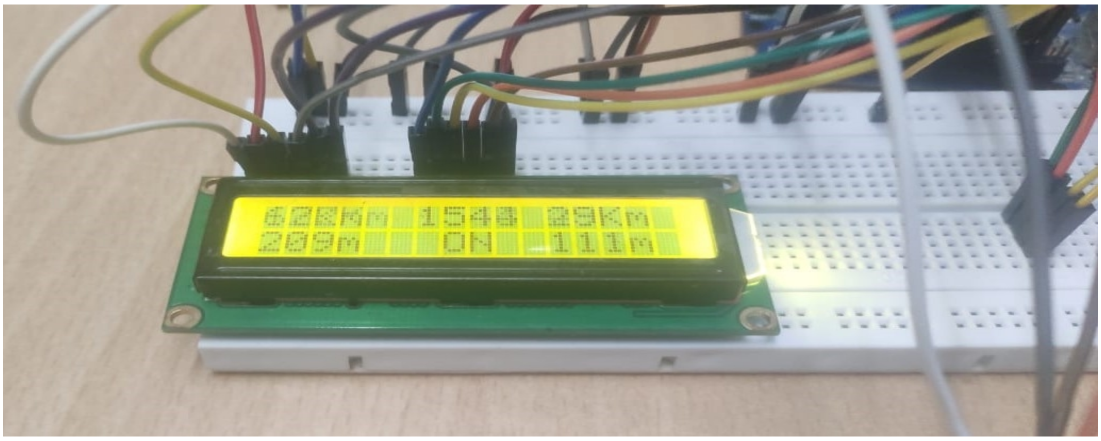


**Figure 18-Scenario 1**

In figure 18, the set speed is set to 62km/hr and the set distance to activate the cruise control is 209m, the distance between the setup and the obstacle identified is around 111m thus the speed of the motor is reducing from the set speed and being reduced(29km/hr).

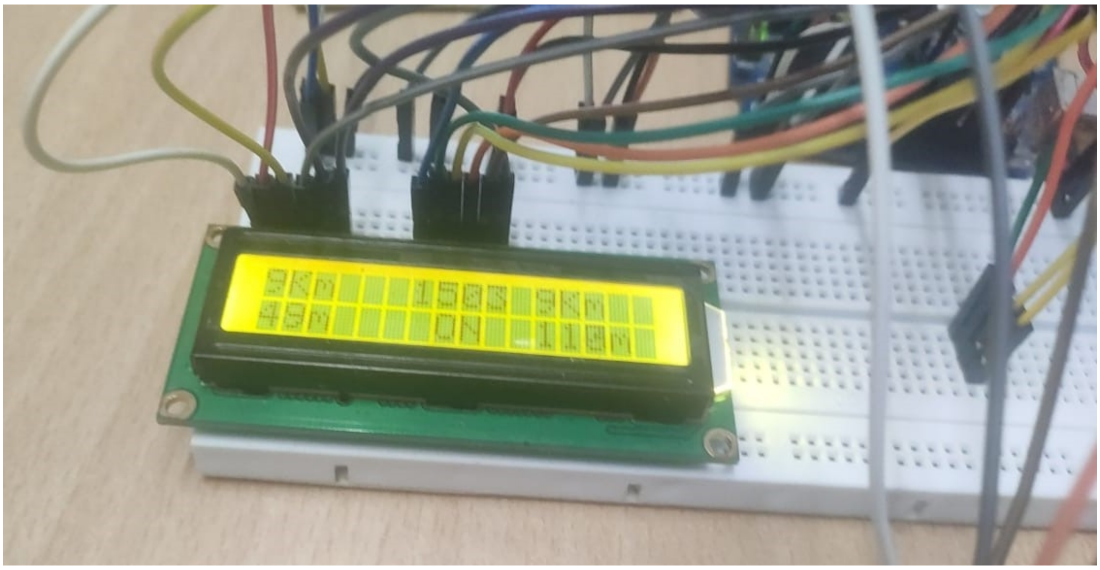

**Figure 19-Scenario 2**

In figure 19, above the set speed is set to 9km/hr and the set distance to activate the cruise control is 49m, the distance between the setup and the obstacle identified is around 111m thus the speed of the motor is maintained as the set speed(9km/hr).

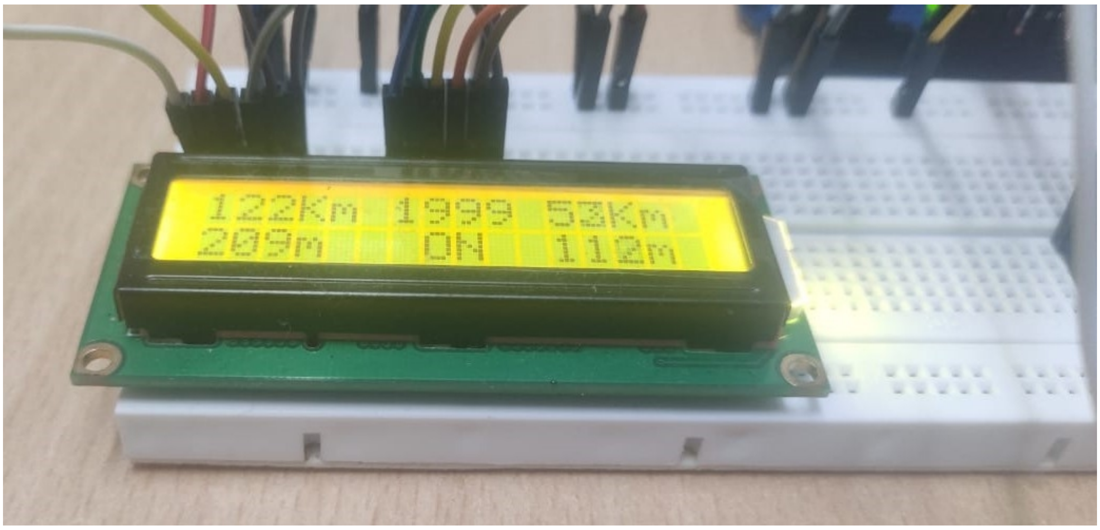


**Figure 20-Scenario 3**

In figure 20, above the set speed is set to 122km/hr and the set distance to activate the cruise control is 209m, the distance between the setup and the obstacle identified is around 111m thus the speed of the motor is reducing from the set speed and being reduced(53km/hr).

The inference of the hardware results is shown in Table 5 below.

**Hardware Inference:**

**Table 5-Hardware Inference**

| ADAPTIVE CRUISE CONTROL CONDITION | SET SPEED (km/hr) | SET DISTANCE(m) | MEASURED DISTANCE(m) | MAINTAINED SPEED (km/hr) |
|---|---|---|---|---|
| ON | 62 | 209 | 111 | 29 |
| ON | 9 | 49 | 110 | 9 |
| ON | 122 | 209 | 112 | 53 |

# CHAPTER V

# 5.CONCLUSION

## 5.1 COST ANALYSIS

**Table 6-Cost Analysis**

| S.No. | DESCRIPTION | QUANTITY | UNIT COST (₹) | COST (₹) |
|---|---|---|---|---|
| 1. | Arduino UNO | 1 | 550 | 550 |
| 2. | Ultrasonic Sensor HC-SR04 | 1 | 90 | 90 |
| 3. | LCD 1602a Display | 1 | 110 | 110 |
| 4. | 470kΩ Potentiometer | 4 | 15 | 60 |
| 5. | DC Servo motor | 2 | 120 | 240 |
| 6. | DC motor | 1 | 40 | 40 |
| 7. | 220kΩ Resistor | 1 | 15 | 15 |
| 8. | LEDs | 1 | 4 | 4 |
| 9. | Slider switch | 1 | 10 | 10 |
| 10. | Jumper wires | 75 | 3 | 225 |
| **TOTALCOST** | | | | **1344** |

## 5.2 SCOPE OF WORK

Adaptive Cruise Control has a huge scope for development in the future. A concept known as Cooperative adaptive cruise control (CACC) is being studied right now in which apart from the feedback loop of the ACC, the acceleration of the lead vehicle is used in a feed-forward loop. The CACC concept improves stability of the system and reduces delays so that the system can react faster. The multi-objective controller also has a huge scope in the future. It can be incorporated with different controllers according to the circumstances and the number of controllers also can be increased or decreased accordingly.

## 5.3 SUMMARY

In this project, we start off by modelling a system according to real-world conditions. To this system, we incorporate a PID and a Fuzzy-PID controller to find out which gives us the best response. Since the PID controller gives us the best response, it is then used to design a multi-objective controller. The output of the multi-objective controller is then passed through a Kalman Filter to filter out the disturbances and to accurately estimate the output. A hardware model of the adaptive cruise control technique is developed using an Arduino UNO.

# REFERENCES

[1] WorrawutPananurak and SomphongThanok ,"Adaptive cruise control for an intelligent vehicle", International Conference on Robotics and Biomimetics,2009

[2] Arvind Raj R, S. B. Sandhiya Kumar and T. C. Karthik ,"Cruise control operation from zero to preset speed", International Journal of Information and Education Technology, Vol .1, no.1, 9-14, 2011

[3] Rohan Kumar and RajanPathak ,"Adaptive cruise control-Towards a safer driving experience", International Journal of Scientific and Engineering Research, Vol 3, no. 8, 676-680, 2012

[4] Peter Gaspar and Balazs Nemeth, "Design of adaptive cruise control for road vehicles using topographic and traffic information", The International Federation of Automatic Control,Vol.47,no.3, 4184-4189, 2014

[5] Khaled Sailan and Klaus Dieter Kuhnert, "Modeling And Design of Cruise Control System With Feedforward For All Terrain Vehicles",Computer Science & Information Technology (CS &IT) , Vol.3, 339-349, 2013

[6] Hermann Winner, Stefan Witte, Werner Uhler, Bernd Lichtenberg, "Adaptive Cruise Control System Aspects and Development Trends", SAE International by University of Leeds, Vol.105, no.6, 1412-1421, 1996

[7] Sheng Zhang and XiangtaoZhuan, "Study on Adaptive Cruise Control Strategy for Battery Electric Vehicle", Hindawi-Mathematical Problems in Engineering , Vol.2019,no.7971514, 1-14, 2019

[8] AakarMehra, Wen-Loong Ma, Forrest Berg, Paulo Tabuada, Jessy W. Grizzle and Aaron D. Ames, "Adaptive Cruise Control: Experimental Validation of Advanced Controllers on Scale-Model Cars", Proceedings of the American Control Conference,Vol.17, 1411-1418, 2015

[9] Ankita Singh,C.S.Satsangi and Prashant Panse,"Adaptive Cruise Control using Fuzzy Logic",International Journal of Digital Application and Contemporary Research,Vol.3,no.8,146-152 ,2015

[10] Chaoxian. Wu,Yuan. Lin and Azim. Eskandarian,"Cooperative Adaptive Cruise Control with Adaptive Kalman Filter Subject to Temporary Communication Loss", ,Vol.xx,no.9, ,2017

[11]StefanChamraz and Richard Balogh,"Two Approaches to the Adaptive Cruise Control (ACC) Design", , , , ,2018

[12]A. Bjonberg,"Control design for autonomous intelligent cruise control",IFAC Proceedings Volumes,Vol.27,no.12,835-840,1994

[13] Pallab Maji, Sarat Kumar Patra, and Kamalakanta Mahapatra,"Design and Implementation of Fuzzy Approximation PI Controller for Automatic Cruise Control System",Hindawi-Advances in Artificial Intelligence,Vol.2015,no.62463,1-7,2015

[14] AjeethaApparna G and Aishwarya A,"Adaptive Cruise Control for Vechile modelling using MATLAB",IOSR Journal of Electrical and Electronics Engineering,Vol.12,no.2,82-88,2017

[15] Youngjoo Kim and Hyochoong Bang,"Introduction to Kalman Filter and Its Applications",

[16] François Auger, Mickael Hilairet, Josep M. Guerrero, Eric Monmasson, Teresa Orlowska-Kowalska and Seiichiro Katsura,"Industrial Applications of the Kalman Filter: A Review",IEEETransactions OnIndustrial Electronics,Vol.60,no.12,5458-5471,2013

[17] Emeka Eyisi, Zhenkai Zhang, XenofonKoutsoukos, Joseph Porter,GaborKarsai, And Janos Sztipanovits ,"Model-Based Control Design And Integration Of Cyberphysical Systems: An Adaptive Cruise Control Case Study ", Hindawi-Journal Of Control Science And Engineering,Vol.2013,No.678016,1-15,2013

[18] Yinglong He, BiagioCiuffo, Quan Zhou, MichailMakridis, Konstantinos Mattas, Ji Li, Ziyang Li, Fuwu Yan AndHongming Xu,“Adaptive Cruise Control Strategies Implemented On Experimental Vehicles: A Review”,IFAC Papersonline,Vol.52,No.5,21-27,2019

[19] Oanh Pham-Thi-Ngoc and Dung Nguyen-Thanh,”Introducing the Kalman filter and application”,International Journal of Scientific & Engineering Research,Vol.10,no.8,973-977,2019

[20] Yuanfu Mo, Dexin Yu, Jun Song, KunZheng,andYajuan Guo,”Vehicle Position Updating Strategy Based on Kalman Filter Prediction in VANETEnvironment”,Hindawi-Discrete Dynamics in Nature and Society,Vol.2016,no.1401396,1-9,2016

# APPENDICES

## Appendix 1: Multi-Objective Controller Python Code

Code:

```
from matplotlib import pyplot as plt
import numpy as np
from scipy.integrate import odeint
distego=0;
distlead=0;
vlead=15;
def vehicle(v,t,u):
   Cd=0.3;
   rho=1.2;
Af=2;
   F=1000;
   m=1200;
Crr=0.02;
   g=9.81;
dv_dt=(F*u-235-(0.36*v**2))/m;
   return dv_dt;
tf=20;
timesteps=21;
delta_t=0.1;
s=np.zeros(timesteps);
ts=np.linspace(0,tf,timesteps);
step=np.zeros(timesteps)
v0=50;
vs=np.zeros(timesteps);
Kc=11;
Ki=3;
Kd=0.1;
```

```
sum_int=0;
es=np.zeros(timesteps);
ies=np.zeros(timesteps);
des=np.zeros(timesteps);
sp_store=np.zeros(timesteps);
distlead1=650;
des1=0;
step=np.zeros(timesteps);
i=0;

for i in range(timesteps-1):
distego=v0*i;
distlead=distlead1+vlead*i;
   x=distlead-distego
   s[i+1]=x;
sp=50;
sp_store[i+1]=sp;
   error=sp-v0;
   es[i+1]=error;
sum_int=sum_int+error*delta_t;
   des=(error-des1)/delta_t;
   u=Kc*error+Ki*sum_int+Kd*des;
ies[i+1]=sum_int;
   step[i+1]=u;
   v=odeint(vehicle,v0,[0,delta_t],args=(u,))
   print(v)
   v0=v[-1];
   vs[i+1]=v0;
   if vs[i+1]>60:
      vs[i+1]=60;
   if vs[i+1]<0:
      vs[i+1]=0;

   if (distlead-distego<=600) and (distlead-distego>=200):
```

```
sp=25;
sp_store[i+1]=sp;
     error=sp-v0;
     es[i+1]=error;
     Kc1=10.7;
     Ki1=4;
     Kd1=0.1;
sum_int=sum_int+error*delta_t;
     des=(error-des1)/delta_t
     u=Kc1*error+Ki1*sum_int+Kd1*des
ies[i+1]=sum_int;
     step[i+1]=u;
     v=odeint(vehicle,v0,[0,delta_t],args=(u,))
     v0=v[1];
     vs[i+1]=v0;
     if vs[i+1] >60:
       vs[i+1]=60;
     if vs[i+1]<0:
       vs[i+1]=0;

elif (distlead-distego<200) and (distlead-distego>0):
sp=15;
sp_store[i+1]=sp;
     error=sp-v0;
     es[i+1]=error;
     Kc2=10.8;
     Ki2=3;
     Kd2=0.1;
sum_int=sum_int+error*delta_t;
     des=(error-des1)/delta_t
     u=Kc2*error+Ki2*sum_int+Kd2*des;
ies[i+1]=sum_int;
     step[i+1]=u;
     v=odeint(vehicle,v0,[0,delta_t],args=(u,))
```

```
        v0=v[-1];
        vs[i+1]=v0;
        if vs[i+1]>60:
          vs[i+1]=60;
        if vs[i+1]<0:
          vs[i+1]=0;
    if distlead-distego<=0:
          vs[i+1]=0;
sp_store[i+1]=0;
exit(0);

plt.figure()
plt.xlabel('Time')
plt.subplot(2,1,1)
plt.plot(ts, sp_store, label ='Set-Point')
plt.plot(ts, vs, label ='Speed',color='red')
plt.legend()
plt.subplot(2,1,2)
plt.plot(ts,s, label ='Distance',color='blue')
plt.legend()
plt.show()
```

**Appendix 2: Hardware Code**

```
#include <LiquidCrystal.h>
#include<Servo.h>
#define massSensor                A0
#define cruiseSpeedDial           A2
#define cruiseDistanceDial        A3
Servo myservo1,myservo2;
LiquidCrystallcd(12, 11, 5, 4, 3, 2);
const int echoPin=0;
const int trigPin=1;
const int servo1=9;
const int servo2=10;
const int start=8;
const int engineRPM=13;
int distance=0;
int distanceSet=0;
int speed=0;
int speedSet=0;
int tempSpeedSet=0;
int mass=0;
int servoValue=0;
long duration=0;
double error=0;
double lastError=0;
double cumError=0;
double rateError=0;
double outputPWM=0;
double kp=27.7593;
double ki=3.9352;
double kd=24.0508;
double force=0;
double friction=0;
double drag=0;
```

```
int brakeForce=0;
double acceleration=0;

void defaultPos(void);
void startACC(void);
void getDistance(void);
void getSpeed(void);
void checkRadar(void);
void getMass(void);
void getSimulatedSpeed(void);
void PID(void);
void simulateCar(void);
void cutSpeed(void);
void brake(void);

void setup()
{
lcd.begin(16, 2);
pinMode(cruiseSpeedDial,INPUT);
pinMode(cruiseDistanceDial,INPUT);
pinMode(speedSensor,INPUT);
pinMode(start,INPUT);
pinMode(engineRPM,OUTPUT);
  myservo1.attach(9);
  myservo2.attach(10);
  myservo1.write(90);
delay(15);
  myservo2.write(90);
delay(15);
lcd.write("Adaptive Cruise");
lcd.setCursor(0,1);
lcd.write(" Control System");
delay(1500);
  myservo1.write(0);
```

```
delay(15);
  myservo2.write(0);
delay(15);
delay(1500);
lcd.clear();
}

void defaultPos(void)
{
        lcd.setCursor(7,1);
        lcd.write("OFF ");
        lcd.setCursor(11,0);
        lcd.print(speed);
        lcd.write("Km ");
        getSpeed();
        getDistance();
        checkRadar();
        getMass();
}

void getDistance(void)
{
        distanceSet = (analogRead(cruiseDistanceDial)*0.1953125)+10;
        lcd.setCursor(0,1);
lcd.print(distanceSet);
lcd.print("m ");
}

void getSpeed(void)
{
        speedSet = (analogRead(cruiseSpeedDial)*0.1171875)+3;
        lcd.setCursor(0,0);
lcd.print(speedSet);
lcd.print("Km ");
```

```
}

void getMass(void)
{
        mass = (analogRead(massSensor)*0.48828125)+1500;
        lcd.setCursor(6,0);
lcd.print(mass);
}

void cutSpeed(void)
{
        tempSpeedSet=speedSet-((300-distance)*0.003*speedSet);
        if(speedSet!=tempSpeedSet)
speedSet=tempSpeedSet;
}

void brake(void)
{
        brakeForce=(mass*((speed*speed)+1))/(2*distance);
        outputPWM=0;
}

void PID(void)
{
        error=speedSet-speed;
        cumError+=error * 0.8;              //0.8 is the simulation time
        rateError=(error - lastError)/0.8;
        outputPWM=(kp*error) + (ki*cumError) + (kd*rateError);
        lastError=error;
analogWrite(engineRPM,outputPWM);
}

void simulateCar(void)
{
```

```
            friction=0.7*mass*9.8;
            drag=(0.3*1.225*(speed*speed)*4)/2;
            force=(outputPWM*78.125)-brakeForce-friction-drag;
            acceleration=force/mass;
            speed+=acceleration*0.8;
            if(speed<0)
      speed=0;
            lcd.setCursor(11,0);
lcd.print(speed);
lcd.print("Km ");
            servoValue=(outputPWM*0.703125);
            myservo1.write(servoValue);
            delay(15);
            servoValue=brakeForce;
            myservo2.write(servoValue);
}

void checkRadar(void)
{
            pinMode(trigPin, OUTPUT);
            pinMode(echoPin, INPUT);
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
    duration = pulseIn(echoPin, HIGH);

    // convert the time into a distance
    distance = duration/29/2;

lcd.setCursor(11,1);
            lcd.print(distance);
            lcd.print("m ");
```

```
}

void startACC(void)
{
        getSpeed();
        getDistance();
        getMass();
        lcd.setCursor(7,1);
        lcd.print("ON ");
        checkRadar();

        if(distance>=distanceSet)
  {
brakeForce=0;
PID();
        simulateCar();
  }
        else if((distance<distanceSet)&&(distance>=20))
  {
brakeForce=0;
cutSpeed();
PID();
        simulateCar();
  }
        else
  {
brake();
brakeForce+=10;
simulateCar();
  }

}
```

```
void loop()
{
	if(digitalRead(start)==HIGH)
		startACC();
	else
	defaultPos();
}
```

# CURRICULUM VITAE

| | | |
|---|---|---|
| 1. | Name | :Rohan Chandrasekar |
| 2. | Date of Birth | :03/11/1999 |
| 3. | Email | : rohan.chandrasekar2017@vitstudent.ac.in |
| 4. | Address for Communication | : 7/1,Seshadripuram Mn Rd,Velachery, Ch-42 |
| 5. | Education | : B.Tech, Electrical and Electronics Engineering |
| 6. | Interests | :Control Systems, Robotics, Internet of Things |

| | | |
|---|---|---|
| 1. | Name | : Mallesh N |
| 2. | Date of Birth | : 18/03/2000 |
| 3. | Email | : mallesh.n2017@vitstudent.ac.in |
| 4. | Address for Communication | : 39B, DhadeeshwaramNgr, Velachery, Ch-42 |
| 5. | Education | :B.Tech, Electrical and Electronics Engineering |
| 6. | Interests | :Control Systems, Instrumentation and Robotics |