List of Figures

List of Tables

Table of Contents

I. Introduction

Adaptive Cruise Control (ACC) is an exciting advancement in automotive technology that aims to enhance both convenience and safety on the roads. Unlike traditional cruise control systems, ACC utilizes sensors and intelligent algorithms to automatically adjust a vehicle's speed and maintain a safe distance from the vehicle in front, even in changing traffic conditions [1].

In the past, cruise control allowed drivers to set a specific speed for their vehicles, but it lacked the ability to adapt to traffic situations. ACC changes the game by incorporating radar, lidar, and camera sensors that continuously scan the road ahead [2]. These sensors provide real-time information about the distance and speed of other vehicles, enabling the system to make informed decisions about accelerating or decelerating to match the leading vehicle's pace.

The primary goal of ACC is to improve safety on the roads. By automatically adjusting the speed and maintaining a safe following distance, ACC reduces the risk of rear-end collisions, which are a common type of accident [3]. This feature also helps prevent driver fatigue by reducing the need for constant monitoring of traffic conditions.

The benefits of ACC extend beyond safety. It offers a significant convenience factor, allowing drivers to enjoy the advantages of cruise control while still relying on the system to handle the changing traffic environment. ACC optimizes driving efficiency and provides a more comfortable experience, particularly in congested or variable-speed situations.

ACC serves as a crucial step as the automotive industry moves towards the development of autonomous vehicles. It is part of a broader concept known as Advanced Driver-Assistance Systems (ADAS) and contributes to the ongoing progress of creating self-driving cars [4].

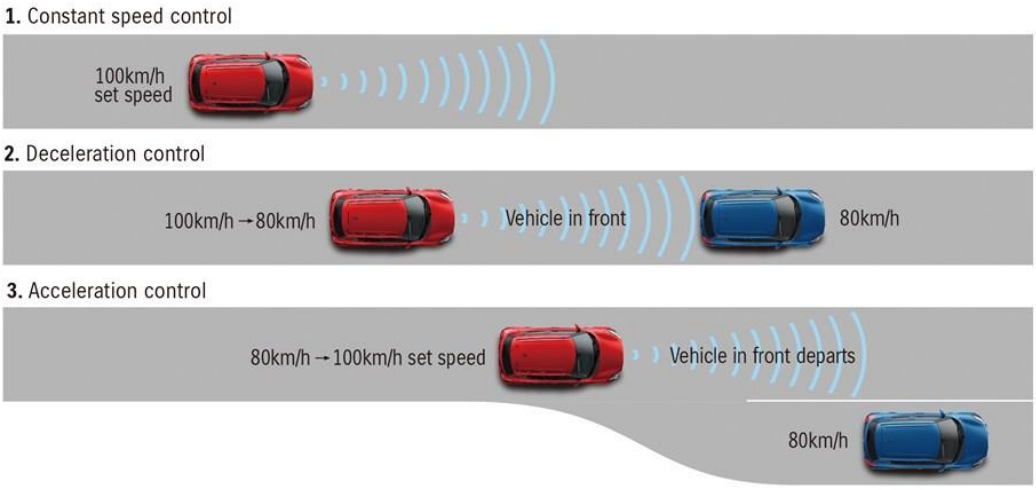

Figure 1: Adaptive Cruise Control [5]

II. Project Objectives

The project aims to demonstrate the functionality and effectiveness of an ACC system through MATLAB integration and software coding. The project seeks to accomplish three following goals.

1. Develop a control system that automatically adjusts the host vehicle's speed based on the movements of the preceding vehicle. This system will maintain a safe and comfortable following distance between the two vehicles.
2. Apply a sensor that continuously monitors the distance between the vehicles and adjust the host vehicle's speed to avoid collisions or unsafe situations.
3. Analyze and showcase the ACC system's response to changes in the preceding vehicle's speed. The system will promptly and smoothly adapt to changes in the preceding vehicle's speed while maintaining a safe distance and adapting to traffic conditions.

By achieving these objectives, the project will provide a practical demonstration of the benefits and effectiveness of ACC in enhancing driving safety, convenience, and comfort [6].

III. Hardware Components and Tools

A. Hardware Components

1) LCD Display: Visual displays are used to provide real-time information and feedback to the driver, such as current speed and distance measurements [7].

Figure 2: Arduino's LCD Interface [8]

2) *Arduino Uno:* A microcontroller board serves as the central control unit for the ACC system [7]. It receives input from sensors, processes data, and generates output signals for speed control.

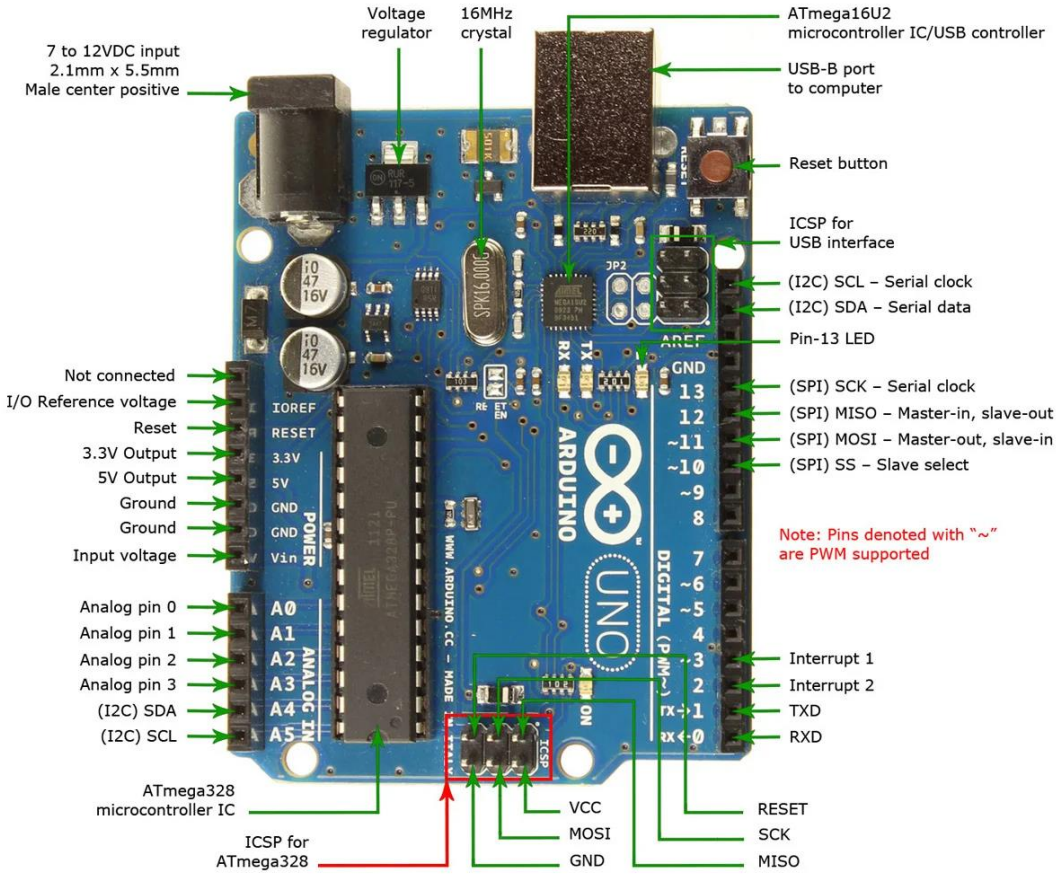

Figure 3: Arduino Uno [9]

3) *Push Buttons:* Five buttons are used for various functionalities, including setting the desired speed, enabling/disabling the cruise control, and adjusting system parameters.

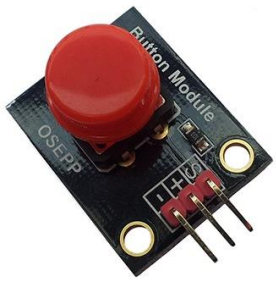

Figure 4: Push Button [10]

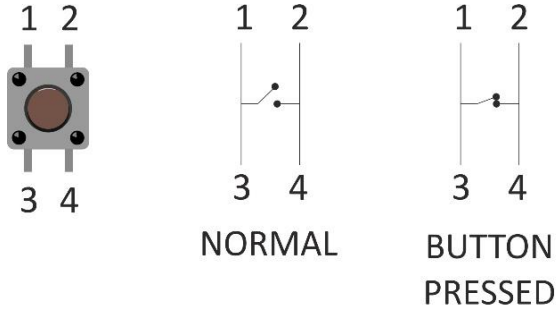

Figure 5: Digital Input [11]

4) *Ultrasonic Sensor:* The ultrasonic sensor is an essential component for accurately measuring the distance between the host vehicle and the preceding vehicle [7]. It utilizes ultrasonic waves to determine the distance and provides input to the control system.

Figure 6: Ultrasonic Sensor [12]

5) *Resistors:* Resistors are used to limit current flow and protect components from damage [7]. They are employed in various parts of the circuitry, such as voltage dividers and pull-up/pull-down resistors.

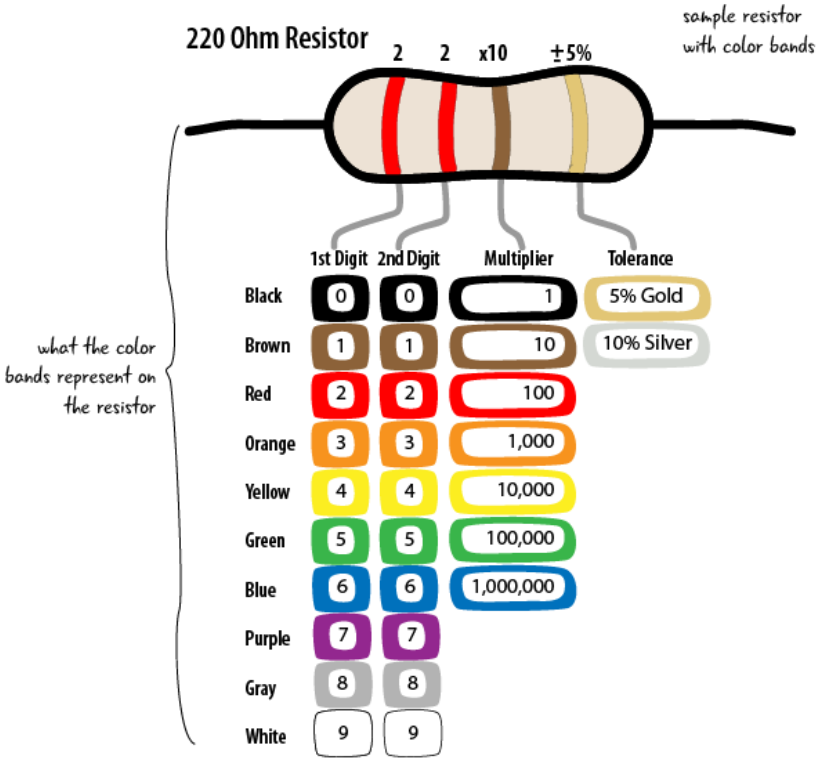

Figure 7: Resistor [13]

6) *Battery:* The battery is a suitable power source that provides the necessary electrical energy to run the ACC system.
7) *Battery Connector Cable:* The battery connector cable is used to connect the battery to the Arduino board and other components, ensuring a reliable power supply.
8) *Jumper Wires:* Wires are used to establish connections between different components on the breadboard or PCB, thereby enabling the flow of signals and power [7].

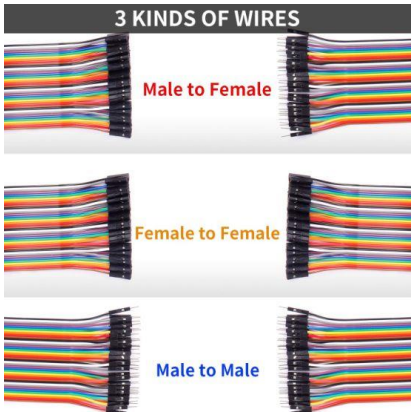

Figure 8: Jumper Wires [14]

9) *PCB or Breadboard:* PCB is a prototyping platform that allows for the interconnection of various hardware components and circuits, which facilitate the development and testing of the ACC system [7].

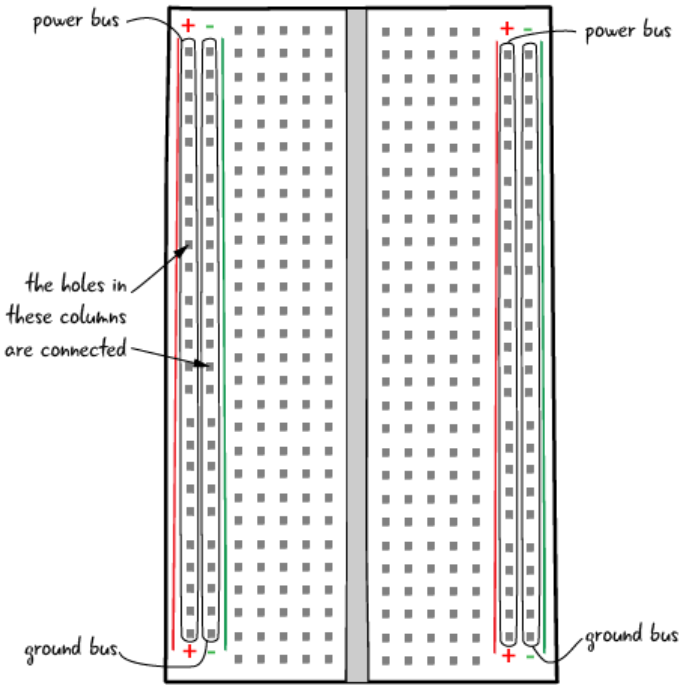

Figure 9: Breadboard [15]

10) Potentiometer: Potentiometers are variable resistors that are used to adjust system parameters, such as sensitivity or the desired following distance [7].

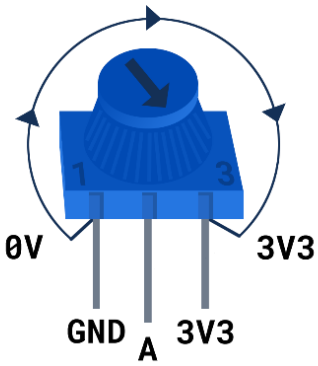

Figure 10: Potentiometer [16]

B. Software Tools

Table 1: Software Tools

Software Tools	Description
1) Simulink	• A powerful graphical programming environment provided by MATLAB. • It enables the modeling, simulation, and implementation of control systems using intuitive block diagrams [17].
2) MATLAB	• A high-level programming language and environment widely used for mathematical modeling, algorithm development, and data analysis. • It offers extensive functionality and toolboxes for system simulation, control design, and signal processing [18].

IV. Methodology

The methodology for implementing the ACC system can be summarized using the high-level architecture seen in Figure 11.

A. Logic Diagram

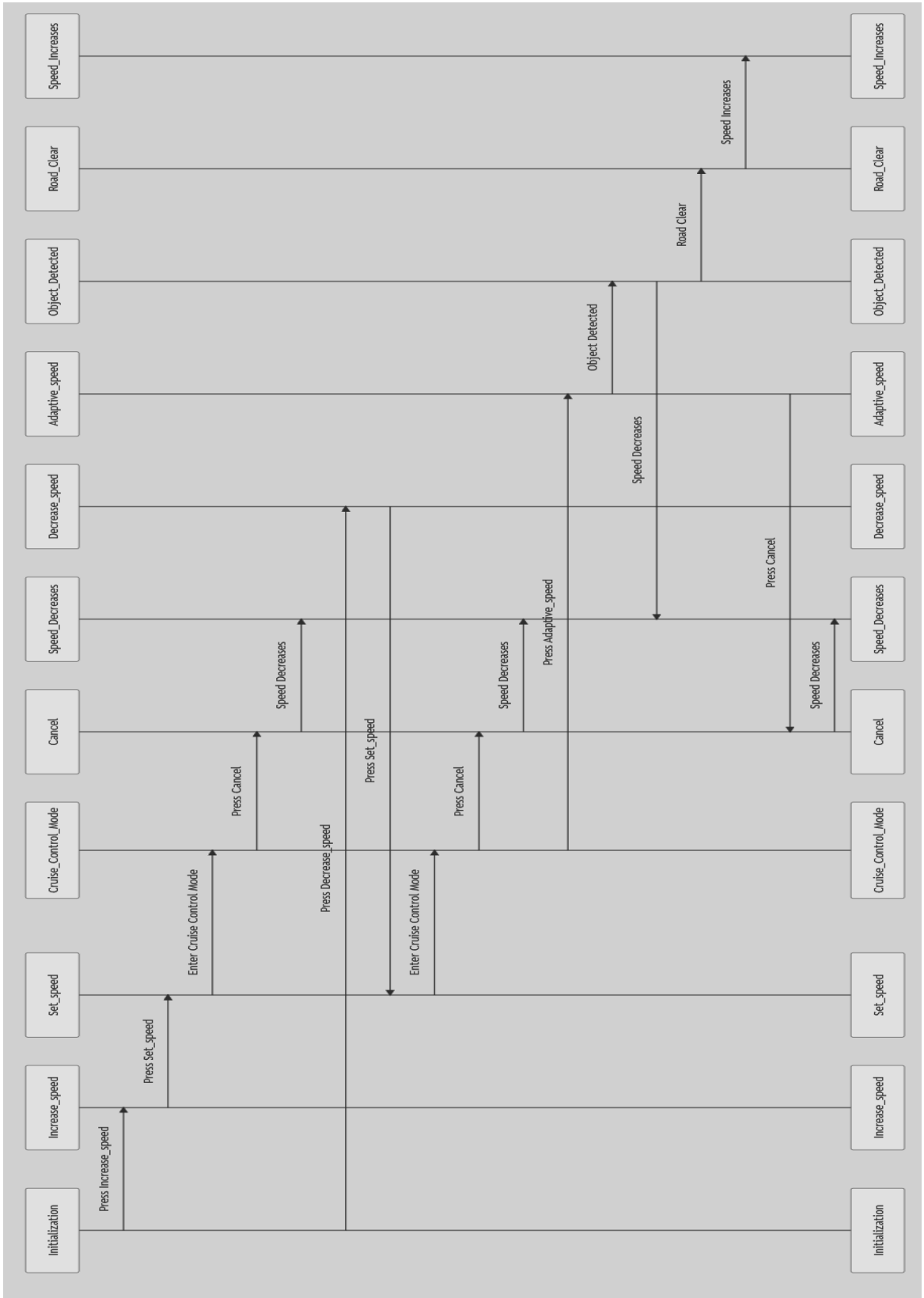

Figure 11: Logic Diagram

B. Flowchart

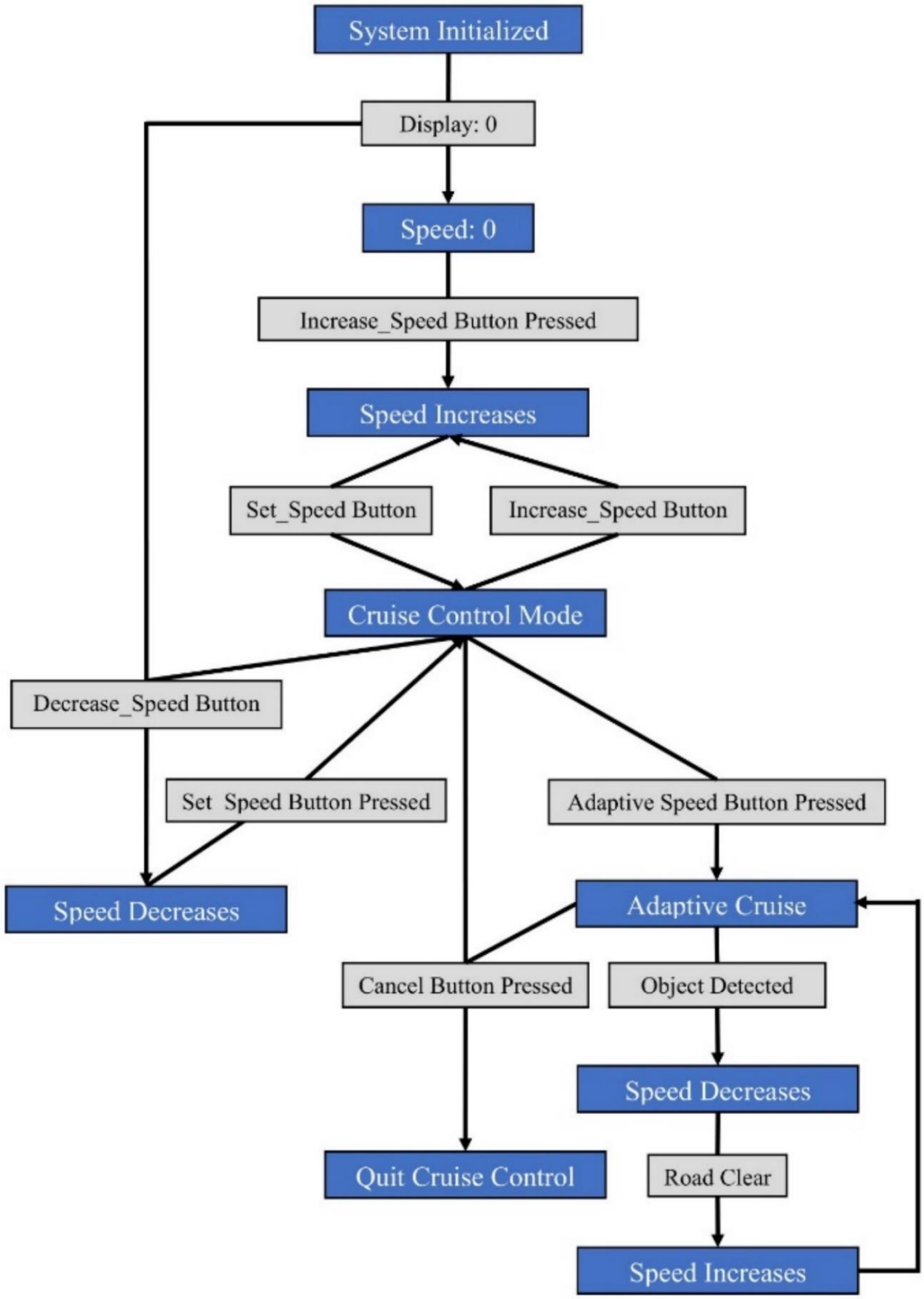

Figure 12: Flowchart of ACC

V. Preliminary Pseudocode

The following pseudocode outlines the basic structure and steps involved in the ACC system:

A. Algorithm: ACC

1. Initialize system variables and parameters.
2. Initialize communication with radar sensor.
3. While vehicle is running,
    a. read distance and velocity data from radar sensor,
    b. preprocess data to remove noise and extract relevant features,
    c. calculate desired speed based on control algorithm,
    d. adjust speed command using speed controller,
    e. apply control signals to the vehicle's throttle or brake system, and
    f. repeat steps a-e continuously.
4. End.

B. Pseudocode

```
% Initialize the system
speed = 0; % Initial speed
display(speed); % Display the initial speed

% Define the function for the Increase_speed button
function Increase_speed()
  % Check if the button is pressed
  if Increase_speed_button_is_pressed
    % Increase the speed
    speed = speed + 1;
    display(speed); % Display the updated speed
  end
end

% Define the function for the Decrease_speed button
function Decrease_speed()
```

```
    % Check if the button is pressed
    if Decrease_speed_button_is_pressed
        % Decrease the speed
        speed = speed - 1;
        display(speed); % Display the updated speed
    end
end

% Define the function for the Set_speed button
function Set_speed()
    % Check if the button is pressed
    if Set_speed_button_is_pressed
        % Enter the cruise control mode
        cruise_control_mode = true;
        % In this mode, the speed is held constant
        constant_speed = speed;
        display(constant_speed); % Display the constant speed
    end
end

% Define the function for the Cancel button
function Cancel()
    % Check if the button is pressed
    if Cancel_button_is_pressed
        % Quit the cruise control mode
        cruise_control_mode = false;
        % The speed decreases slowly
        speed = speed - 0.1;
        display(speed); % Display the updated speed
    end
end
```

```
% Define the function for the Adaptive_speed button
function Adaptive_speed()
    % Check if the button is pressed
    if Adaptive_speed_button_is_pressed
        % Enter the ACC mode
        adaptive_cruise_control_mode = true;
        % In this mode, the speed is held constant until a vehicle shows up at the front or an object is detected
        constant_speed = speed;
        display(constant_speed); % Display the constant speed
    end
end

% Define the function for detecting an object
function Object_Detected()
    % If an object is detected, the speed automatically decreases
    if object_is_detected
        speed = speed - 1;
        display(speed); % Display the updated speed
    end
end

% Define the function for when the road becomes clear
function Road_Clear()
    % When the road becomes clear, the speed increases to reach the set speed again
    if road_is_clear
        speed = speed + 1;
        display(speed); % Display the updated speed
    end
end
```

```
% Main function to run the system
function main()
    % Initialize the system
    Initialize_system();
    % Loop to keep the system running
    while true
        % Check the status of each button and perform the corresponding action
        Increase_speed();
        Decrease_speed();
        Set_speed();
        Cancel();
        Adaptive_speed();
        % Check if an object is detected or if the road is clear
        Object_Detected();
        Road_Clear();
    end
end

% Call the main function
main();
```

VI. Timeline

A. Milestones

The project timeline is outlined below:

Table 2: Milestones Completed

Semester Weeks	Milestones Completed
Week 1-2	Project planning and research
Week 3	System modeling and simulation
Week 4	Hardware component acquisition
Week 5	Preliminary pseudocode
Week 6	Radar sensor integration

Table 3: Milestones to Complete

Semester Weeks	Milestones to Complete
Week 7	Control algorithm development
Week 8	System integration and testing
Week 8-9	Fine-tuning and optimization
Week 9	System integration and testing
Week 10-11	Fine-tuning and optimization

B. Gantt Chart

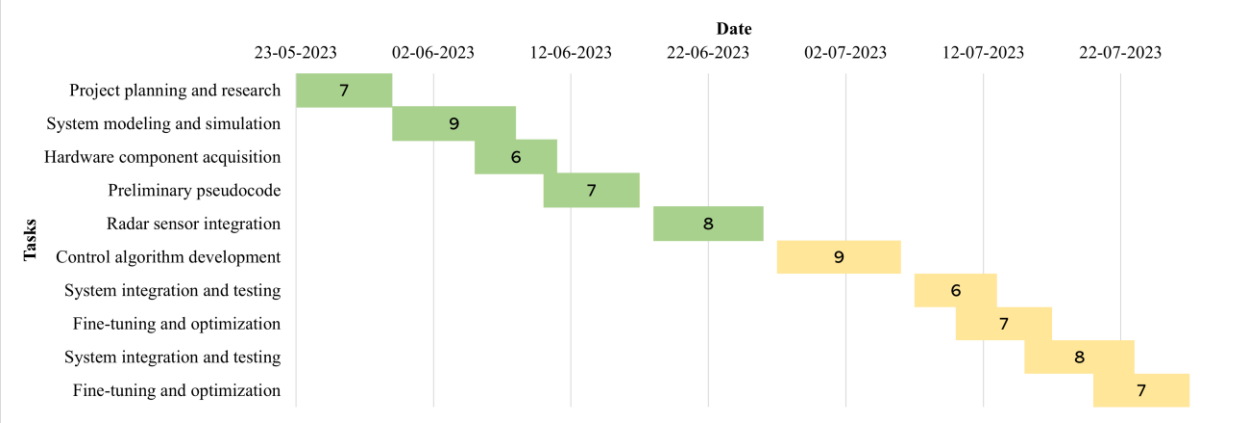

Figure 13: Gantt Chart

C. Limitations and Risks

The project involves five key limitations and risks that need to be considered.

1) *Hardware Availability:* Acquiring the necessary hardware components, such as the radar sensor or microcontroller, may pose challenges and potentially delay project progress. Efforts will be made to identify suitable alternatives or workarounds in such situations.
2) *Real-world Constraints:* The performance of the control system can be influenced by real-world factors, such as variations in weather conditions, road conditions, or vehicle dynamics [19]. Simulating these constraints accurately in MATLAB may present challenges and require additional calibration.
3) *System Complexity:* Developing an ACC system involves dealing with complex algorithms and the integration of multiple components. The inherent complexity of the system may lead to unexpected issues or difficulties during the implementation phase.

4) *Safety Considerations:* Ensuring the safety of the ACC system is of utmost importance. By testing and validation of the control algorithm and hardware integration are necessary to mitigate any potential risks associated with incorrect speed adjustments or unreliable distance measurements [20].
5) *Time Management:* Effective time management is critical for meeting project milestones. Delays in any phase of the project could impact subsequent tasks and the overall project timeline. Maintaining open communication and coordination among team members will be essential to mitigate this risk.

It is essential to proactively address these limitations and risks to ensure the successful completion of the project.

D. Roles and Responsibilities

Table 4: Roles and Responsibilities

Group Member	Roles and Responsibilities
1) Amey	• Research and analysis, • MATLAB code development, • documentation, • project coordination.
2) Nandeshwar	• Hardware acquisition and integration, • MATLAB code development, • system testing, • troubleshooting.
3) Brano	• Control algorithm development, • MATLAB simulation, • system optimization, • report writing.

VII. Conclusion

The MATLAB project on ACC is summarized in this report. Its objective is to create a control system capable of automatically modifying a vehicle's speed according to the distance from the preceding vehicle. The report details the hardware components and software tools utilized, presents an overview of the methodology using a block diagram, and introduces initial pseudocode for the control system. Moreover, a timeline is provided along with a discussion of potential limitations or risks to ensure a comprehensive comprehension of the project's scope.

References

[1] G. Marsden, M. McDonald, and M. Brackstone, "Towards an understanding of adaptive cruise control," *Transportation Research Part C: Emerging Technologies*, vol. 9, no. 1, pp. 33–51, Feb. 2001, https://doi.org/10.1016/S0968-090X(00)00022-X [accessed Jun. 2, 2023].

[2] F. Rosique, P. J. Navarro, C. Fernández, and A. Padilla, "A systematic review of perception system and simulators for Autonomous Vehicles Research," *Sensors*, vol. 19, no. 3, p. 648, 2019, https://doi.org/10.3390/s19030648 [accessed Jun. 2, 2023].

[3] Y. Li *et al.*, "Evaluation of the impacts of cooperative adaptive cruise control on reducing rear-end collision risks on freeways," *Accident Analysis & Prevention*, vol. 98, pp. 87–95, Jan. 2017, https://doi.org/10.1016/j.aap.2016.09.015 [accessed Jun. 2, 2023].

[4] Synopsys, "What is ADAS (advanced driver assistance systems)? – overview of Adas Applications," https://www.synopsys.com/automotive/what-is-adas.html [accessed Jun. 2, 2023].

[5] S. Naylor, "What is ACC (Adaptive Cruise Control)?," *Parkers*, https://www.parkers.co.uk/what-is/acc-adaptive-cruise-control [accessed Jun. 2, 2023].

[6] J. Lee, D. McGehee, T. Brown, and D. Marshall, "Effects of adaptive cruise control and alert modality on driver performance," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 1980, no. 1, pp. 49–56, Jan. 2006, https://doi.org/10.1177/0361198106198000108 [accessed Jun. 2, 2023].

[7] Arduino, "Arduino documentation," https://docs.arduino.cc [accessed Jun. 5, 2023].

[8] D. Kumar, et al., "Interfacing LCD with 8051," *Embedded and Robotics*, https://yadavdharm.wordpress.com/2019/03/08/interfacing-lcd-with-8051 [accessed Jun. 5, 2023]

[9] Arduino France, "Arduino Uno: Advantages, Disadvantages, Use and Operation," https://www.arduino-france.com/review/arduino-uno [accessed Jun. 5, 2023].

[10] OSEPP, "Push button module," https://www.osepp.com/electronic-modules/sensor-modules/76-push-button-module [accessed Jun. 5, 2023].

[11] Robo India, "Digital input -how to use the button with Arduino," https://roboindia.com/tutorials/digital-input-how-to-use-the-button-with-arduino [accessed Jun. 5, 2023].

[12] R. L. Pendergast et al., "Complete Guide for Ultrasonic sensor HC-SR04 with Arduino," *Random Nerd Tutorials*, https://randomnerdtutorials.com/complete-guide-for-ultrasonic-sensor-hc-sr04 [accessed Jun. 5, 2023].

[13] arduino2go, "Appendix A: Reading resistor codes," *Arduino to Go*, https://arduinotogo.com/2017/03/10/appendix-a-reading-resistor-codes [accessed Jun. 5, 2023].

[14] Electronics Infra, "Jumper wire 0.5mm - electronics infra," https://electronicsinfra.com/product/jumper-wire-0-5mm [accessed Jun. 5, 2023].

[15] arduino2go, "Chapter 2: Building a circuit step by step," *Arduino to Go*, https://arduinotogo.com/2016/08/22/chapter-2-building-a-circuit-step-by-step [accessed Jun. 5, 2023].

[16] Raspberry Pi, "Getting started with raspberry pi," *Raspberry Pi Foundation*, https://projects.raspberrypi.org/en/projects/raspberry-pi-getting-started [accessed Jun. 5, 2023].

[17] Simulink, "Simulink Documentation," https://www.mathworks.com/help/simulink [accessed Jun. 12, 2023].

[18] Matlab, "MATLAB Documentation," https://www.mathworks.com/help/matlab [accessed Jun. 12, 2023].

[19] Team-BHP "Adaptive Cruise Control Limitations,", https://www.team-bhp.com/forum/indian-car-scene [accessed Jun. 15, 2023].

[20] B. D. Seppelt and J. D. Lee, "Making Adaptive Cruise Control (ACC) limits visible," *International Journal of Human-Computer Studies*, vol. 65, no. 3, pp. 192–205, Mar. 2007, https://doi.org/10.1016/j.ijhcs.2006.10.001 [accessed Jun. 15, 2023].