Table of Contents

List of Figures

List of Tables

1. Introduction

Adaptive Cruise Control (ACC), the modern driver assistance technology, automatically adjusts the speed of the car to maintain a safe distance from the car in front of it, enhancing both vehicle and driver safety.The system uses radar and sensors to measure the speed and distance of the cars ahead of it. This data is processed by an Electronic Control Unit (ECU), which then modifies the throttle and braking systems to change speed as needed. To minimize accident risk and the necessity for manual speed adjustments, the system maintains the vehicle at a predetermined speed until it detects a slower vehicle. At this point, it modifies the speed to hold a safe distance. The process of implementing ACC includes programming the ECU, creating control algorithms, integrating sensors, and conducting rigorous testing in a range of driving scenarios. By reducing abrupt speed changes, this technology increases safety, boosts driver comfort, and facilitates smoother traffic flow.

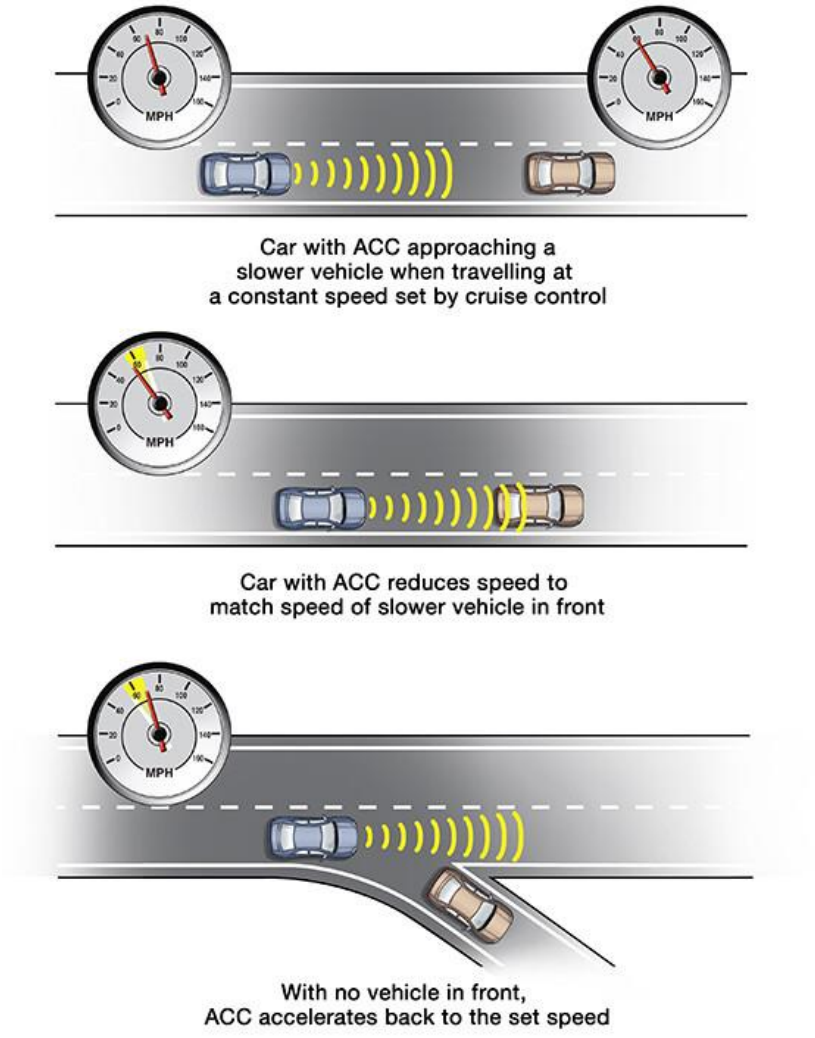

Fig.1 Working of Adaptive Cruise Control

The outcomes of this project using MATLAB typically involve developing a functional ACC system that enhances road safety and traffic efficiency. This includes creating a robust control algorithm, integrating sensors effectively, designing a user-friendly interface, and thoroughly validating the system through real-world testing.

The project aims to deliver a reliable ACC system that not only improves driving safety but also contributes to smoother traffic flow.

1.1 Project Objective

The objective of this project is to utilize MATLAB for testing and developing an Adaptive Cruise Control (ACC) system. This system aims to enhance vehicle comfort and safety by automatically adjusting the vehicle's speed to maintain a safe distance from the car ahead. The implementation involves MATLAB programming, an LCD display, an ultrasonic sensor, push buttons, and an Arduino Uno microcontroller.

2. Hardware Components & Software

A. LCD Display x1

Fig.2 LCD Display

B. Arduino Uno x1

Fig.3 Arduino Uno

C. Push Buttons x5

Fig.4 Push Buttons

D. Ultrasonic Sensor x1

Fig.5 Ultrasonic Sensor

E. PCB or breadboard x1

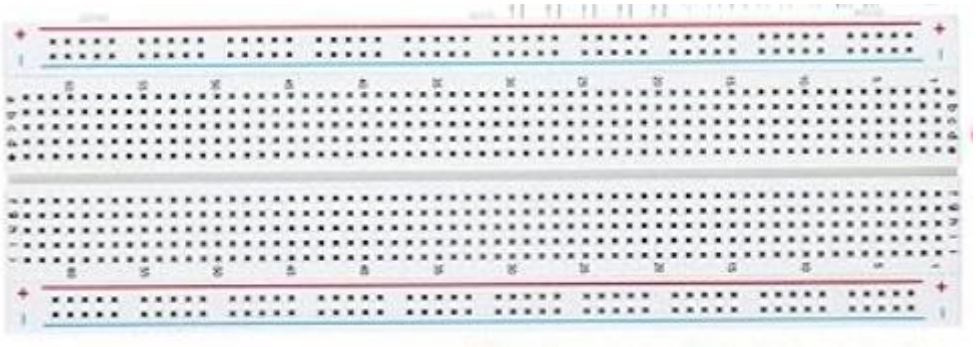

Fig.6 PCB or Breadboard

F. Software – MATLAB

3. Methodology

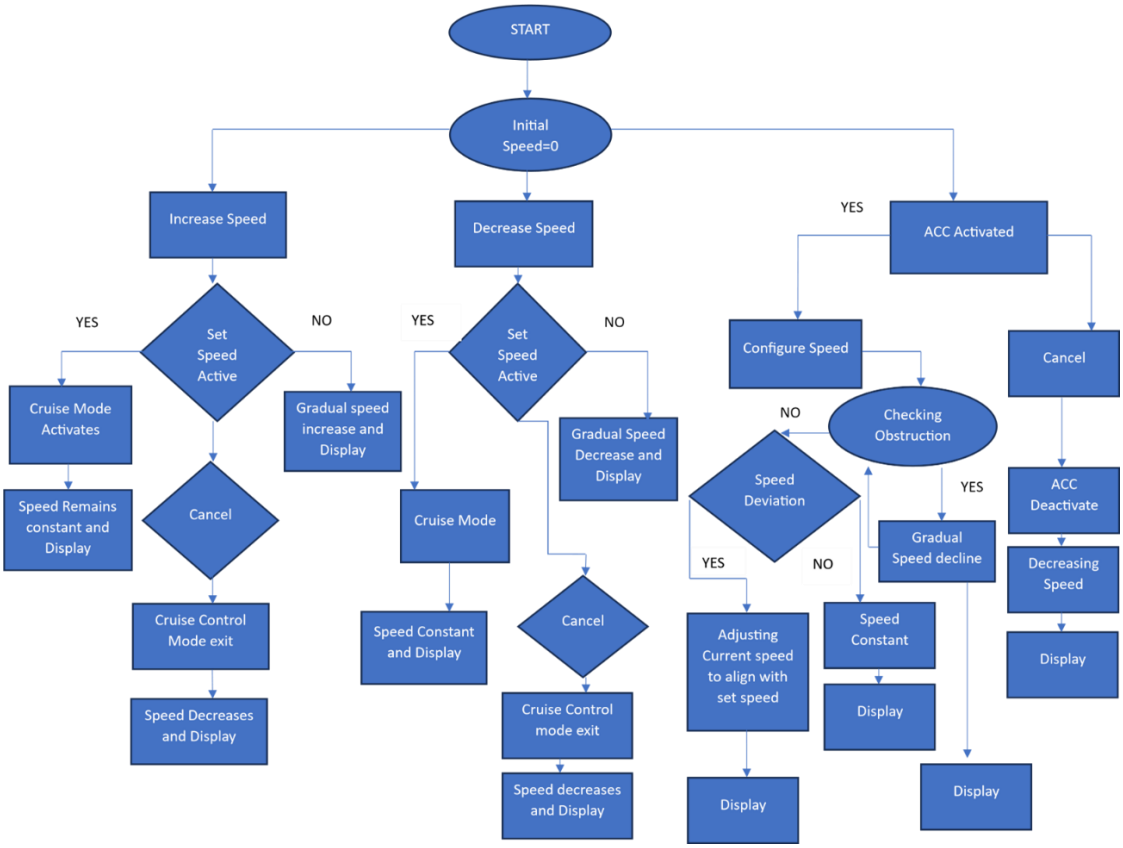

Fig.7 Flowchart of ACC

This MATLAB code serves as the foundation for developing an ACC system with an Arduino board, an ultrasonic sensor, and an LCD display. It includes various buttons to control conditions such as speed and control modes. The increase speed control and decrease speed control buttons change the system's speed manually. The code allows selection between "Set Cruise Speed" mode, "Adaptive Cruise Control" mode, and "Normal" Mode. It also ensures that the speed does not drop below zero.

In "Normal" mode, the system's speed can be increased or decreased using their respective buttons and gradually returns to zero.
While "Set Cruise Speed" mode is activated, the current speed is maintained and does not decrease gradually. It is possible to increase or decrease the current holding speed using the buttons.
In "Adaptive Cruise Control" mode, after a speed is set, the speed cannot be manually increased or reduced. However, it should take obstacle distance measurements from the ultrasonic sensor to automatically adjust the speed to the assigned speed as required, preventing accidents in a real-life scenario. The LCD should provide a visual feedback based on modes and different speed selection.

4. Preliminary Pseudocode

```
% Set the Initial settings

car_speed = 0;

collision_distance = 40; % Example distance to safe stop


while car_is_on
  % While pressing increase speed button
  if increase_speed && ~cruise_control_mode && ~adaptive_cruise_control_mode
     car_speed = car_speed + 1;
   % While pressing decrease speed button
   elseif decrease_speed && ~cruise_control_mode && ~adaptive_cruise_control_mode
      car_speed = car_speed - 1;
```

```
    if car_speed < 0
        car_speed = 0;
    End
%Automatically decrease speed
else
    car_speed = car_speed – 1;
end

% Setting cruise control mode
when setting cruise control mode
    if increase_speed && ~adaptive_cruise_control_mode
        car_speed = car_speed + 1;
    end
    if decrease_speed && ~adaptive_cruise_control_mode
        car_speed = car_speed - 1;
        if car_speed < 0
            car_speed = 0;
        end
    end
    if cancel
        % Exit cruise control mode and return to normal mode
        cruise_control_mode = 0;
    end

% Setting adaptive cruise control mode
```

```
when setting to adaptive_cruise_control_mode
    set_speed = car_speed;
    % Get reading from the Ultrasonic distance sensor
    distance_to_obstacle = get Ultrasonic Distance;

    % Manual speed adjustments should not be taken
    if distance_to_obstacle < collision_distance
        % When the obstacle approaches
        car_speed = car_speed - 1;
        if car_speed < 0
            car_speed = 0;
        end
    elseif distance_to_obstacle > collision_distance
        car_speed = car_speed + 1;
        if car_speed > set_speed
            car_speed = set_speed;
        end
    end
    if cancel
        % Exit adaptive cruise control mode and return to normal mode
        adaptive_cruise_control_mode = 0;
    end
End
```

5. Timeline & Limitations

Weeks	**Milestones completed**
Week 1	Understood project goals and requirements.
Weeks	**Milestones pending**
Week 1	Identify all the required components and set up the initial design
Week 2	Assemble and test components and check basic working of each component
Week 3	Synthesis the code for each of the required test cases so it works in an Arduino
Week 4	Test all component work together as required
Week 5	Document and finalize the data

Table 1: Project timeline

There are a few limitations that comes into play, such as:

Sensor Accuracy: Low-cost sensors might not provide very precise or accurate readings, which could affect how the system works.

Real-World Testing: Trying out the system in real-life driving scenario and making sure it works well in different can be difficult.

Scalability: While an Arduino-based system can be a good prototype, making it work for a full-sized vehicle may require more advanced hardware and software.

6. Conclusion

This project utilizes the advantages of vehicle safety and efficiency by creating an Adaptive Cruise Control (ACC) system. Through the integration of MATLAB software with an Arduino Uno kit, a dual-mode system can be developed maintain a set speed and adapt to changing traffic conditions. The ability of the ACC system to adjust speed automatically based on nearby vehicles reduces driver tiredness and decreases the chances of accidents, ultimately improving individual driving safety and promoting smoother traffic flow. The initial pseudocode describes the basic logic of the ACC system, covering speed changes and mode shifts to ensure the car reacts correctly to different driving situations. This basic code will be improved and built upon as the project continues, enhancing the overall efficiency and dependability of the ACC system.

The project schedule guarantees organized advancement: the team has grasped the project

objectives and needs in the first week and is planning to establish initial design elements in the second week, putting together and examining individual components in the third week, merging code for Arduino and evaluating integration in the fourth week, and recording and completing data in the fifth week.

Several constraints need to be tackled. Cheap sensors may lack accuracy, impacting the efficiency of the system. It can be difficult to make sure the system operates well in a variety of real-world driving situations. Moreover, although the prototype built using Arduino can show promise, enhancing it for use in full-sized vehicles will necessitate more sophisticated hardware and software. However, the team is aiming to make collaborative efforts in design, coding, and testing of ACC technology.

7. References

[1] "Adaptive Cruise Control," *My Car Does What?*, https://mycardoeswhat.org/deeper-learning/adaptive-cruise-control/ (accessed Jun. 5, 2024).