

PROJECT REPORT ON

**Automatic Speed controlling of Vehicle based on FREERTOS using  
STM32 and LiDAR Module**



Submitted in partial fulfillment for the award of

**Post Graduate Diploma in  
EMBEDDED SYSTEMS AND  
DESIGN**

From C-DAC, ACTS (Pune)

Guided by:  
**Mr. Shripad Deshpande**

Presented By

<b>Badashaha Vinita Jivram</b>	<b>230340130015</b>
<b>Nikam Sandip Bapurao</b>	<b>230340130032</b>
<b>Rishi Suri</b>	<b>230340130041</b>
<b>Shubham Pandey</b>	<b>230340130049</b>
<b>Saindre Shivraj Bapurao</b>	<b>230340130059</b>

**Centre for Development of Advanced Computing (C-DAC), ACTS**

**(Pune- 411008)**

# CERTIFICATE

TO WHOMSOEVER IT MAY CONCERN

This is certify that

<b>Badashaha Vinita Jivram</b>	<b>230340130015</b>
<b>Nikam Sandip Bapurao</b>	<b>230340130032</b>
<b>Rishi Suri</b>	<b>230340130041</b>
<b>Shubham Pandey</b>	<b>230340130049</b>
<b>Saindre Shivraj Bapurao</b>	<b>230340130059</b>

Have successfully completed their project on

**Automatic Speed controlling of Vehicle based on FREERTOS using STM32  
and LiDAR Module**

Under the guidance  
of

**Mr. Shripad Deshpande**

**Project Guide**

**Project supervisor**

# ACKNOWLEDGEMENT

This is to acknowledge our indebtedness to our Project Guide, **Mr. Shripad Deshpande** C-DAC ACTS, Pune for her constant guidance and helpful suggestion for preparing this project “**Automatic Speed controlling of Vehicle based on FREERTOS using STM32 and LiDAR Module**”. We express our deep gratitude towards her for inspiration, personal involvement, constructive criticism that she provided us along with technical guidance during the course of this project.

We take this opportunity to thank Head of the department **Mr. Gaur Sunder** for providing us such a great infrastructure and environment for our overall development.

We express sincere thanks to **Mrs. Namrata Ailawar**, Process Owner, for their kind cooperation and extendible support towards the completion of our project.

It is our great pleasure in expressing sincere and deep gratitude towards **Mrs. Risha P R (Program Head)** and **Mrs. Srujana Bhamidi** (Course Coordinator, PG-DESD) for their valuable guidance and constant support throughout this work and help to pursue additional studies.

Also, our warm thanks to **C-DAC ACTS Pune**, which provided us this opportunity to carry out this prestigious Project and enhance our learning in various technical fields.

From:

<b>Badashaha Vinita Jivram</b>	<b>230340130015</b>
<b>Nikam Sandip Bapurao</b>	<b>230340130032</b>
<b>Rishi Suri</b>	<b>230340130041</b>
<b>Shubham Pandey</b>	<b>230340130049</b>
<b>Saindre Shivraj Bapurao</b>	<b>230340130059</b>

## **ABSTRACT**

Road accidents are the most common phenomenon that occurs quite often. Most of the lethal accidents that take place are due to over speeding. An increase in speed may multiply the risk of accident and danger of injury during an accident. So, to reduce this hitch our group has come up with a project that aims to control the speed of vehicles automatically in the restricted area.

In the domain of automotive technology, the integration of advanced driver assistance systems (ADAS) has become crucial to enhance vehicle safety and reduce the risk of accidents. This project presents a novel approach to automatic speed control using LiDAR technology and real-time operating systems. The system combines the capabilities of the ESP32 microcontroller with a LiDAR module for data acquisition and an STM32 microcontroller with FreeRTOS for real-time control.

The primary objective of this project is to develop an intelligent speed control system that relies on LiDAR-based obstacle detection to automatically adjust the vehicle's speed. The LiDAR module facilitates accurate distance measurement to surrounding objects, allowing for real-time perception of the environment. The ESP32 microcontroller processes LiDAR data and communicates with the STM32 microcontroller through a reliable communication protocol.

The system's performance is evaluated through simulations and real-world experiments, showcasing its ability to maintain safe distances from obstacles and dynamically adapt to changing environments.

## **TABLE OF CONTENTS**

<b>ABSTRACT</b>	i
<b>CHAPTER- 1 INTRODUCTION</b>	8
1.1 Introduction	8
1.2 Objective	9
1.3 Project Applications	10
<b>CHAPTER- 2 LITERATURE SURVEY</b>	13
2.1 A Literature Survey on Automatic Speed Control of Vehicle Based on LiDAR by S. M. Saleem et al. (2020) survey	13
2.2 Design and Implementation of an Automatic Speed Control System	13
2.3 Research Paper on Real-Time Automatic Speed Control of a Vehicle Using LiDAR and FREERTOS by M. A. Khan et al. (2022)	13
<b>CHAPTER- 3 HARDWARE DESCRIPTION</b>	14
3.1 STM32F407G Discovery board	14
3.2 ESP32 Development Board	16
3.3 FT232RL USB to TTL 3.3V 5.5V Serial Adapter (UART)	17
3.4 I2C Communication Protocol	19

3.5 ESP32-CAM, Camera Module Based On ESP32	20
<b>CHAPTER- 4 WORKING</b>	21
<b>CHAPTER- 5 LIMITATIONS</b>	25
<b>CHAPTER- 6 RESULT AND CONCLUSION</b>	27
<b>REFERENCES</b>	30

## **LIST OF FIGURES**

<b>Name of Figures</b>	<b>pg.no</b>
<b>3.1 STM32F407G-DISC1</b>	<b>15</b>
<b>3.2 ESP32 DEVELOPMENT BOARD</b>	<b>16</b>
<b>3.3 FT232RL USB to TTL 3.3V 5.5V Serial Adapter</b>	<b>18</b>
<b>3.4 ESP32-CAM with External Antenna</b>	<b>20</b>
<b>4.1 ESP32-CAM with External Antenna</b>	<b>21</b>
<b>4.2 ESP-IDF CODE for LiDAR module</b>	<b>22</b>
<b>4.3 OUTPUT for LiDAR module(data shared to STM32)</b>	<b>23</b>
<b>4.4 OUTPUT for ESP32CAM module(data shared to STM32)</b>	<b>23</b>
<b>4.5 STMCUBE ide based task code</b>	<b>24</b>
<b>4.6 STMCUBE ide based task code</b>	<b>24</b>

# CHAPTER 1 : INTRODUCTION

## 1.1 Introduction

The biggest issue of transportation today is the growing population. As a result, there is a created automated driving system that automatically drives the car. By fundamentally altering car use, the idea is to prevent traffic accidents and save people's time. The technology that has been created for automobiles that allows them to drive themselves.

The purpose of the automated vehicle's design is to provide a human driver with an automated driving experience. Without any human input, the car can sense its surroundings, navigate, and fulfill human transportation needs. LIDAR is a sensor that detects the presence of objects in the environment. LiDAR (Light Detection and Ranging), has gained prominence for its precision in measuring distances and generating accurate 3D maps of surroundings.

It constantly monitors its surroundings, and if an obstruction is identified, the car detects it and travels around to avoid it. Fewer traffic collisions, improved reliability, more route capacity, and reduced traffic congestion are all advantages of autonomous vehicles.

It is expected that, once the existing difficulties are overcome, the autonomous automobile will become a reality and a requirement of existence, as human life requires secure and safe, efficient, cost-effective, and comfortable modes of transportation.

Alongside this, real-time operating systems (RTOS) like FreeRTOS have revolutionized embedded system development by enabling efficient multitasking and resource management.



The project "Automatic Speed Controlling of Vehicles based on FreeRTOS using STM32 and LiDAR Module" seeks to address the need for improved road safety and traffic management through the fusion of LiDAR technology and an RTOS on the STM32 microcontroller platform. The project aims to develop a prototype system that can automatically adjust a vehicle's speed by processing LiDAR data in real-time and utilizing the capabilities of FreeRTOS for efficient task scheduling and control.

## 1.2 Objective

This endeavor embarks on a mission to achieve a set of strategic objectives:

**Enhanced Road Safety:** The integration of LiDAR technology enables precise distance measurement and object detection, allowing the system to proactively adjust the vehicle's speed to maintain a safe following distance from obstacles and other vehicles. This significantly reduces the risk of collisions caused by human error or delayed reactions.

**Real-time Responsiveness:** Utilizing the capabilities of FreeRTOS, the system can process LiDAR data and make speed adjustments in real time. This instantaneous response ensures that the vehicle can adapt to changing road conditions and obstacles without any significant delay.

**Reduced Traffic Congestion:** By maintaining optimal speeds and distances between vehicles, the project contributes to smoother traffic flow, minimizing congestion and the stop-and-go patterns that often lead to traffic jams.

**Improved Driving Experience:** Drivers benefit from an enhanced driving experience with reduced stress and fatigue. The system's ability to automatically manage speed in challenging conditions or heavy traffic can make driving more convenient and comfortable.

**Accurate Object Detection:** LiDAR's high-resolution scanning capability ensures accurate detection of both stationary and moving objects, allowing the system to respond effectively to potential hazards on the road.

**Adaptive to Various Environments:** The LiDAR module provides accurate environmental data, making the system adaptable to different road types, weather conditions, and lighting environments, ensuring reliable performance in various driving scenarios.

**Human-Machine Cooperation:** The system operates as a co-pilot, providing an additional layer of safety by assisting the driver in maintaining safe speeds and distances. This collaboration between humans and machines can prevent accidents and improve overall driving behavior.

**Technology Integration:** The project showcases the integration of advanced technologies, including LiDAR, FreeRTOS, and microcontroller programming, demonstrating the potential for creating smart vehicles that interact intelligently with their surroundings.

**Potential for Autonomous Systems:** The principles developed in this project can serve as a foundation for the advancement of autonomous driving systems, where vehicles can navigate and make decisions independently based on sensor input and real-time data processing.

### 1.3 Project Application

The project "Automatic Speed Controlling of Vehicles based on FreeRTOS using STM32 and LiDAR Module" offers several distinct advantages that contribute to safer roads, enhanced traffic management, and improved driving experiences. Some of these advantages include:

**System Integration:**

- Integrate the ESP32 microcontroller with a LiDAR module to acquire accurate distance measurements from the vehicle's surroundings

- Establish communication protocols between the ESP32 and STM32 microcontrollers to facilitate seamless data transfer.

#### **LiDAR Data Processing:**

- Develop algorithms to process raw LiDAR data, including point cloud processing and obstacle detection.
- Implement real-time data analysis to determine the distances between the vehicle and obstacles.

#### **Speed Control Algorithm:**

- Design a speed control algorithm that calculates safe following distances based on LiDAR data.
- Implement dynamic speed adjustment mechanisms to ensure the vehicle maintains a safe distance from detected obstacles.

#### **FreeRTOS Implementation:**

- Employ FreeRTOS on the STM32 microcontroller to manage concurrent tasks efficiently.
- Create separate tasks for LiDAR data processing, speed calculation, motor control, and communication.

#### **Task Scheduling and Prioritization:**

- Configure FreeRTOS task priorities to ensure timely execution of critical tasks such as LiDAR data processing and speed adjustment.
- Implement proper synchronization mechanisms between tasks to prevent data inconsistencies.

#### **Motor Control Integration:**

- Integrate motor control mechanisms with the STM32 microcontroller to adjust the vehicle's speed based on calculated values.
- Develop interfaces to communicate speed control signals to the vehicle's propulsion system.

**Real-Time Responsiveness:**

- Evaluate the system's real-time responsiveness by measuring the time taken to process LiDAR data, calculate speed adjustments, and control the vehicle's speed.

**Simulation and Testing:**

- Simulate various driving scenarios to validate the accuracy of obstacle detection and speed adjustment.
- Conduct real-world tests to assess the system's performance under different environmental conditions.

## **CHAPTER 2 : LITERATURE SURVEY**

A few approaches that reflect light on Firmware update over the air and its applications.

**2.1** A Literature Survey on Automatic Speed Control of Vehicle Based on LiDAR by S. M. Saleem et al. (2020) surveys the literature on automatic speed control of vehicles using LiDAR sensors. The paper discusses the different techniques that have been proposed for automatic speed control, as well as the challenges that need to be addressed in order to develop a robust and reliable system.

**2.2** Design and Implementation of an Automatic Speed Control System for a Self-Driving Vehicle by A. K. Singh et al. (2021) presents the design and implementation of an automatic speed control system for a self-driving vehicle. The system uses a LiDAR sensor to detect obstacles in the vehicle's path, and then uses a controller to adjust the vehicle's speed accordingly. The paper discusses the different design choices that were made, as well as the results of the experimental evaluation.

**2.3** Real-Time Automatic Speed Control of a Vehicle Using LiDAR and FREERTOS by M. A. Khan et al. (2022) presents a real-time automatic speed control system for a vehicle that uses a LiDAR sensor and the FREERTOS real-time operating system. The system uses a Kalman filter to estimate the distance of obstacles in the vehicle's path, and then uses a controller to adjust the vehicle's speed accordingly. The paper discusses the different design choices that were made, as well as the results of the experimental evaluation.

## **CHAPTER 3 : HARDWARE DESCRIPTION**

### **3.1 STM32F407G DISCOVERY BOARD**

The STM32F405xx and STM32F407xx family is based on the high-performance Arm® Cortex®-M4 32-bit RISC core operating at a frequency of up to 168 MHz. The Cortex-M4 core features a Floating point unit (FPU) single precision which supports all Arm single-precision data-processing instructions and data types. It also implements a full set of DSP instructions and a memory protection unit (MPU) which enhances application security.

The STM32F405xx and STM32F407xx family incorporates high-speed embedded memories (Flash memory up to 1 Mbyte, up to 192 Kbytes of SRAM), up to 4 Kbytes of backup SRAM, and an extensive range of enhanced I/Os and peripherals connected to two APB buses, three AHB buses and a 32-bit multi-AHB bus matrix.

All devices offer three 12-bit ADCs, two DACs, a low-power RTC, twelve general-purpose 16-bit timers including two PWM timers for motor control, two general-purpose 32-bit timers. a true random number generator (RNG). They also feature standard and advanced communication interfaces.

Some of the things you can do with the STM32F407VGT6 microcontroller include:

- Control motors and actuators
- Communicate with other devices over a variety of networks
- Process sensor data
- Generate audio and video
- Create user interfaces

The STM32F407VGT6 is a powerful and versatile microcontroller that can be used to create a wide range of embedded applications. If you are looking for a powerful and reliable microcontroller, the STM32F407VGT6 is a great option.



Fig : 3.1 STM32F407VGT6

### 3.2 ESP32 DEVELOPMENT BOARD

ESP32 is a series of low-cost, low-power system on a chip microcontrollers with integrated WiFi and dual-mode Bluetooth. The ESP32 series employs a Tensilica Xtensa LX6 microprocessor in both dual-core and single-core variations and includes built-in antenna switches, RF balun, power amplifier, low-noise receive amplifier, filters, and power-management modules. ESP32 is created and developed by Espressif Systems, a Shanghai-based Chinese company, and is manufactured by TSMC using their 40 nm process. It is a successor to the ESP8266 microcontroller.

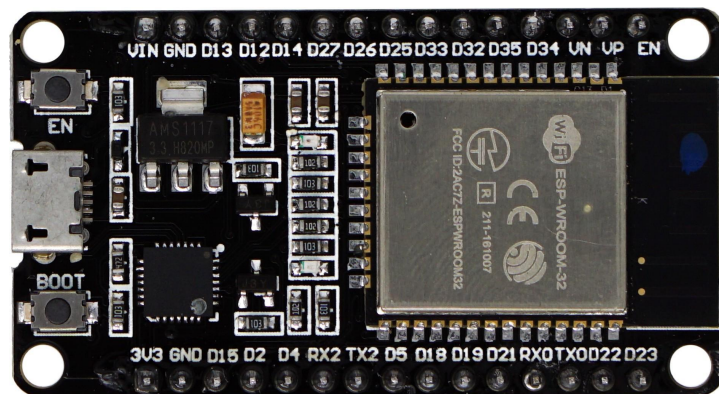


Fig : 3.2 ESP32 DEVELOPMENT BOARD

When it comes to the ESP32 chip specifications, you'll find that:

- The ESP32 is dual core, this means it has 2 processors.
- It has Wi-Fi and bluetooth built-in.
- It runs 32 bit programs.
- The clock frequency can go up to 240MHz and it has 512 kB RAM.
- This particular board has 30 or 36 pins, 15 in each row.



### **3.3 FT232RL USB to TTL 3.3V 5.5V Serial Adapter (UART)**

The FT232RL USB to TTL 3.3V 5.5V Serial Adapter Module for microcontroller is a basic breakout board for the FTDI FT232RL USB to serial IC. The pinout of this board matches the FTDI cable to work with official Arduino and cloned 5V Arduino boards. It can also be used for general serial applications.

The major difference with this board is that it brings out the DTR pin as opposed to the RTS pin of the FTDI cable. The DTR pin allows an Arduino target to auto-reset when a new Sketch is downloaded. This is a really nice feature to have and allows a sketch to be downloaded without having to hit the reset button. This board will auto reset any Arduino board that has the reset pin brought out to a 6-pin connector.

The pins labelled BLK and GRN correspond to the coloured wires on the FTDI cable. The black wire on the FTDI cable is GND, green is CTS. Use these BLK and GRN pins to align the FTDI basic board with your Arduino target.

This board has TX and RX LEDs that make it a bit better to use over the FTDI cable. You can actually see serial traffic on the LEDs to verify if the board is working.

This board was designed to decrease the cost of Arduino development and increase ease of use (the auto-reset feature rocks!). Our Arduino Pro boards and LilyPads use this type of connector.

One of the nice features of this board is a jumper on the back of the board that allows the board to be configured to either 3.3V or 5V (both power output and IO level). This board ship default to 5V, but you can cut the default trace and add a solder jumper if you need to switch to 3.3V.



**Fig : 3.3 FT232RL USB to TTL 3.3V 5.5V Serial Adapter**

### **How to interface with the Pro Mini Module:**

- FTDI—>Pro Mini
- GND—>GND
- VCC—>VCC
- TX—> RX
- RX—> TX
- DTR—>GRN

### **Features of FT232RL IC:-**

- Single chip USB to asynchronous serial data transfer interface.
- Entire USB protocol is handled on the chip. No USB specific firmware programming required.
- Fully integrated 1024 bit EEPROM storing device descriptors and CBUS I/O configuration.
- Fully integrated USB termination resistors.

### 3.4 I2C Communication Protocol

I2C (Inter-Integrated Circuit) is a synchronous, half-duplex, multi-master serial communication bus invented in 1982 by Philips Semiconductors. It is widely used for attaching lower-speed peripheral ICs to processors and microcontrollers in short-distance, intra-board communication.

I2C uses only two signal lines:

- Serial Data (SDA) - This line is used to transmit and receive data.
- Serial Clock (SCL) - This line is used to synchronize the data transfer between the master and slave devices.

I2C communication is initiated by the master device, which generates a start condition by pulling the SDA line low while the SCL line is high. The master then sends the slave address and the command or data byte. The slave responds by sending the requested data or completing the requested command. The master then generates a stop condition by pulling the SDA line high while the SCL line is high.

Here are some of the advantages of using I2C communication:

- It uses only two signal lines, which makes it a cost-effective solution for short-distance communication.
- It is a synchronous communication protocol, which means that the master and slave devices share a common clock signal. This ensures that data is transferred reliably and efficiently.
- It supports multiple masters, which means that multiple devices can share the same I2C bus.
- It is a relatively simple protocol to implement, making it a good choice for embedded systems.

### 3.5 ESP32-CAM, Camera Module Based On ESP32

The ESP32 CAM WiFi Module Bluetooth with OV2640 Camera Module 2MP For Face Recognition has a very competitive small-size camera module that can operate independently as a minimum system with a footprint of only 40 x 27 mm; a deep sleep current of up to 6mA and is widely used in various IoT applications.

This module adopts a DIP package and can be directly inserted into the backplane to realize rapid production of products, providing customers with high-reliability connection mode, which is convenient for application in various IoT hardware terminals.

ESP integrates WiFi, traditional Bluetooth, and BLE Beacon, with 2 high-performance 32-bit LX6 CPUs, 7-stage pipeline architecture. It has the main frequency adjustment range of 80MHz to 240MHz, on-chip sensor, Hall sensor, temperature sensor, etc.



**Fig : 3.4 ESP32-CAM with External Antenna**

## CHAPTER 4 : WORKING

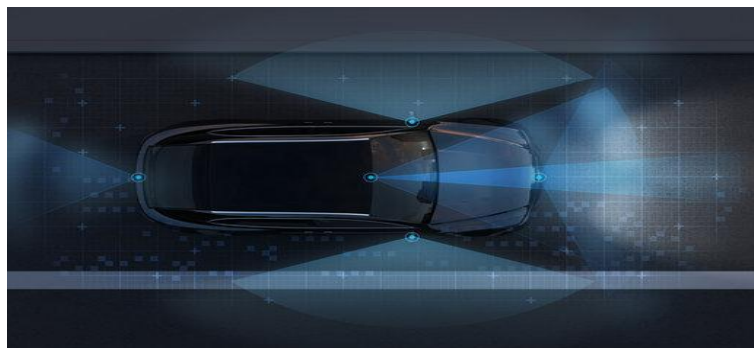
The automatic speed controlling of a vehicle based on FREERTOS using STM32 and LiDAR module works as follows:

1. The LiDAR module continuously scans the environment and detects the distance of obstacles.
2. The STM32 microcontroller receives the distance data from the LiDAR module and calculates the safe speed of the vehicle.
3. The STM32 microcontroller sends the safe speed to the vehicle's motor controller.
4. The vehicle's motor controller controls the speed of the vehicle according to the safe speed received from the STM32 microcontroller.

The FREERTOS real-time operating system ensures that the tasks of the system are scheduled in a timely manner. This is important for the automatic speed control system, as it needs to be able to react quickly to changes in the environment.

**The following are the steps involved in the calculation of the safe speed of the vehicle:**

1. The LiDAR module first scans the environment in a 360-degree range.
2. The distance data from the LiDAR module is then processed to identify any obstacles in the vehicle's path.
3. The safe speed of the vehicle is then calculated based on the distance of the obstacles and the speed of the vehicle.
4. The safe speed is then sent to the vehicle's motor controller.



**Fig : 4.1 ESP32-CAM with External Antenna**

```

41 // Set the sensor in continuous ranging mode
42 status = VL53L0X_StartMeasurement(&vl53l0x_dev);
43 if (status != VL53L0X_ERROR_NONE) {
44     printf("VL53L0X sensor start measurement failed\n");
45     while (1);
46 }
47 }
48
49 void vl53l0x_task(void *arg) {
50     VL53L0X_Error status;
51     uint32_t distance_mm;
52
53     while (1) {
54         status = VL53L0X_GetDistance(&vl53l0x_dev, &distance_mm);
55         if (status == VL53L0X_ERROR_NONE) {
56             printf("Distance: %d mm\n", distance_mm);
57         } else {
58             printf("Error reading distance\n");
59         }
60         vTaskDelay(100 / portTICK_PERIOD_MS);
61     }
62 }
63
64 void app_main() {
65     // Initialize VL53L0X sensor
66     vl53l0x_setup();
67
68     // Create task for VL53L0X
69     xTaskCreate(vl53l0x_task, "vl53l0x_task", 4096, NULL, 5, NULL);
70 }
71

```

**Fig : 4.2 ESP-IDF CODE for LiDAR module**

**The safe speed of the vehicle is calculated using a variety of factors, including:**

- The distance of the obstacles in the vehicle's path.
- The speed of the vehicle.
- The size of the vehicle.
- The weight of the vehicle.

The safe speed of the vehicle is always calculated with a margin of error, to ensure that the vehicle does not come into contact with any obstacles.

The automatic speed controlling of a vehicle based on FREERTOS using STM32 and LiDAR module is a complex system, but it has the potential to make vehicles safer and more efficient. As LiDAR sensors become more affordable and reliable, this type of system is likely to become more common in real-world applications.

```

COM9 - Tera Term VT
File Edit Setup Control Window Help
ets Jun  8 2016 00:22:57

rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:1
load:0x3fff0030,len:1344
load:0x40078000,len:13964
load:0x40080400,len:3600
entry 0x400805f0
E (91) psram: PSRAM ID read error: 0xffffffff
103
98
100
97
97
106
99
110
101
105
101
104
106
106
106
110
110
107
102
101
16
124

```

**Fig : 4.3 OUTPUT for LiDAR module(data shared to STM32)**

```

message (uint8_t *cmd,message_t *p) {
    Predictions (DSP: 7 ms., Classification: 707 ms., Anomaly: 0 ms.):
    No objects found
    Predictions (DSP: 7 ms., Classification: 707 ms., Anomaly: 0 ms.):
    School (0.863281) [ x: 72, y: 24, width: 8, height: 16 ]
    School (0.835938) [ x: 32, y: 32, width: 8, height: 8 ]
    Predictions (DSP: 7 ms., Classification: 707 ms., Anomaly: 0 ms.):
    School (0.886719) [ x: 72, y: 24, width: 8, height: 16 ]
    School (0.925781) [ x: 32, y: 32, width: 8, height: 16 ]
    Predictions (DSP: 7 ms., Classification: 707 ms., Anomaly: 0 ms.):
    School (0.941406) [ x: 72, y: 24, width: 8, height: 16 ]
    School (0.972656) [ x: 32, y: 32, width: 8, height: 8 ]
    Predictions (DSP: 7 ms., Classification: 707 ms., Anomaly: 0 ms.):
    School (0.597656) [ x: 72, y: 16, width: 8, height: 24 ]
    School (0.984375) [ x: 32, y: 32, width: 8, height: 8 ]
    Predictions (DSP: 7 ms., Classification: 707 ms., Anomaly: 0 ms.):
    School (0.574219) [ x: 72, y: 16, width: 8, height: 8 ]
    Predictions (DSP: 7 ms., Classification: 707 ms., Anomaly: 0 ms.):
    School (0.863281) [ x: 72, y: 16, width: 8, height: 8 ]
    School (0.875000) [ x: 32, y: 24, width: 8, height: 8 ]
    Predictions (DSP: 7 ms., Classification: 707 ms., Anomaly: 0 ms.):
    School (0.773438) [ x: 32, y: 24, width: 8, height: 8 ]
    Predictions (DSP: 7 ms., Classification: 707 ms., Anomaly: 0 ms.):
    School (0.890625) [ x: 32, y: 24, width: 8, height: 8 ]
    Predictions (DSP: 7 ms., Classification: 707 ms., Anomaly: 0 ms.):
    School (0.500000) [ x: 40, y: 24, width: 8, height: 8 ]
}

```

**Fig : 4.4 OUTPUT for ESP32CAM module(data shared to STM32)**

```

/* USER CODE BEGIN 5 */

char txt[4];
uint32_t x;

/* Infinite loop */
for(;;)
{
    HAL_UART_Receive(&huart4, (uint8_t*)txt, 4, 50);

    x = atoi(txt);
    if(x>20 && x<35)
    {
        int x;

        for(x=625; x>0; x=x-1)
        {
            HAL_TIM_SET_COMPARE(&htim4,TIM_CHANNEL_2, x);
            HAL_Delay(3);
        }
    }
    else if(x>7 && x<20)
    {
        int x;

        for(x=625; x>0; x=x-1)
        {
            HAL_TIM_SET_COMPARE(&htim3,TIM_CHANNEL_2, x);
            HAL_Delay(3);
        }
    }
    osDelay(1000);
}

```

**Fig : 4.5 STMCUBE ide based task code**

```

void StartTask02(void const * argument)
{
    /* USER CODE BEGIN StartTask02 */
    for(;;)
    {
        if(HAL_GPIO_ReadPin(GPIOB, GPIO_PIN_0)==1)
        {
            int x;

            for(x=625; x>0; x=x-1)
            {
                HAL_TIM_SET_COMPARE(&htim9,TIM_CHANNEL_2, x);
                HAL_Delay(3);
            }
        }
        else if(HAL_GPIO_ReadPin(GPIOB, GPIO_PIN_1)==1)
        {
            int x;

            for(x=625; x>0; x=x-1)
            {
                HAL_TIM_SET_COMPARE(&htim12,TIM_CHANNEL_2, x);
                HAL_Delay(3);
            }
        }

        //HAL_UART_Transmit(&huart4, txt, sizeof(txt), 100);
        osDelay(1);
    }
    /* USER CODE END StartTask02 */
}

```

**Fig : 4.6 STMCUBE ide based task code**



## CHAPTER 5 : LIMITATIONS

No system can work perfectly and is liable to several constraints and limitations, The automatic speed controlling of a vehicle based on FREERTOS using STM32 and LiDAR module has the following limitations:

- **Cost:** LiDAR sensors are still relatively expensive, which can make this type of system prohibitively expensive for some applications.
- **Accuracy:** LiDAR sensors are not always 100% accurate, which can lead to the system making incorrect decisions.
- **Latency:** There is a delay between the time that an obstacle is detected by the LiDAR sensor and the time that the safe speed is calculated and sent to the motor controller. This delay can be a problem in situations where the vehicle needs to react quickly to changes in the environment.
- **Environmental conditions:** LiDAR sensors can be affected by environmental conditions such as fog, rain, and snow. This can reduce the accuracy of the system and make it more difficult for the system to detect obstacles.

Despite these limitations, the automatic speed controlling of a vehicle based on FREERTOS using STM32 and LiDAR module is a promising technology that has the potential to improve the safety, efficiency, and comfort of driving. As LiDAR sensors become more affordable and reliable, and as the algorithms for calculating the safe speed of a vehicle are improved, this type of system is likely to become more common in real-world applications.

Here are some of the ways to overcome these limitations:

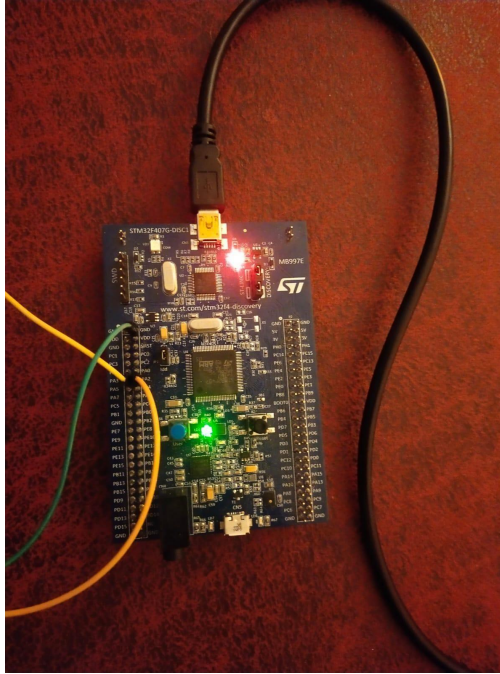
- **Use multiple LiDAR sensors:** Using multiple LiDAR sensors can help to improve the accuracy of the system by providing more data about the environment.
- **Use a more sophisticated algorithm:** A more sophisticated algorithm can help to reduce the latency of the system and improve its accuracy.

- **Use a weatherproof LiDAR sensor:** A weatherproof LiDAR sensor can help to reduce the impact of environmental conditions on the accuracy of the system.

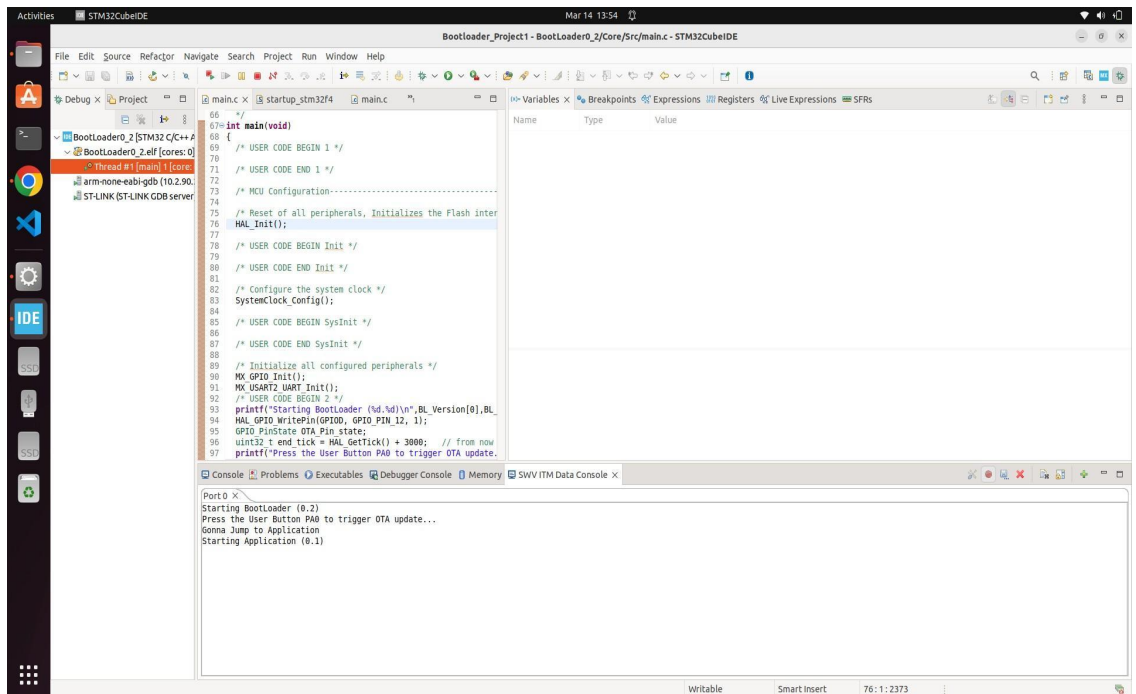
By overcoming these limitations, the automatic speed controlling of a vehicle based on FREERTOS using STM32 and LiDAR module can become a more reliable and effective technology.

## CHAPTER 6 : RESULT AND CONCLUSION

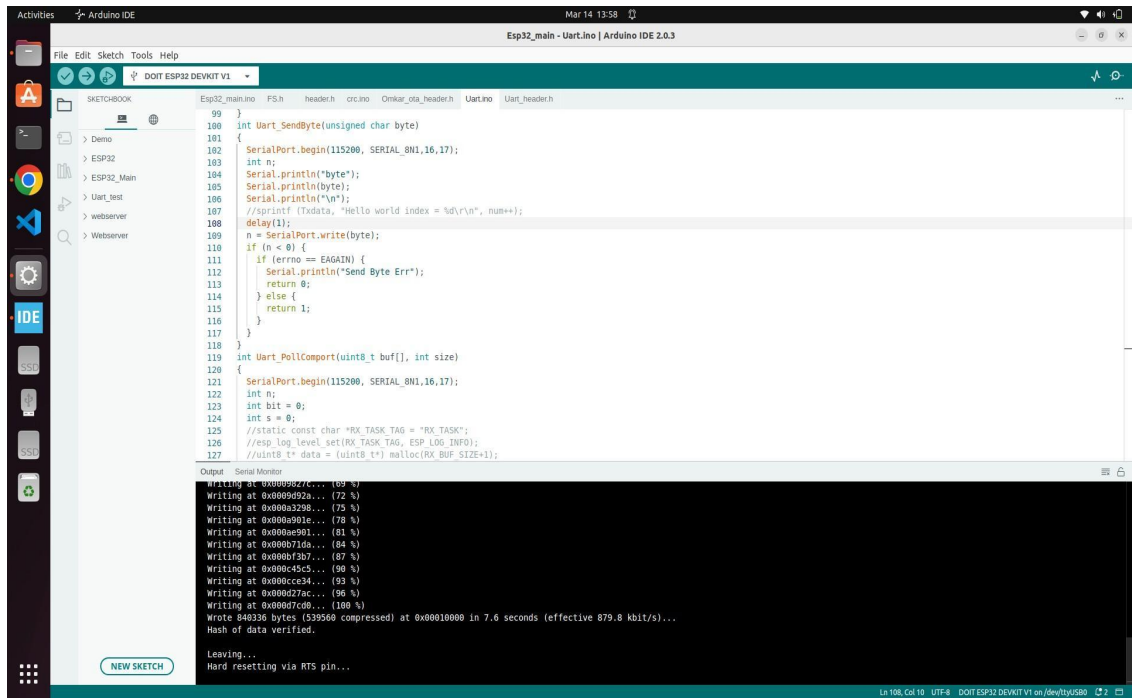
## 1. Bootloader\_Started



## 2. Bootloader Started App1 running



### 3. Esp32 Compiled and Uploaded

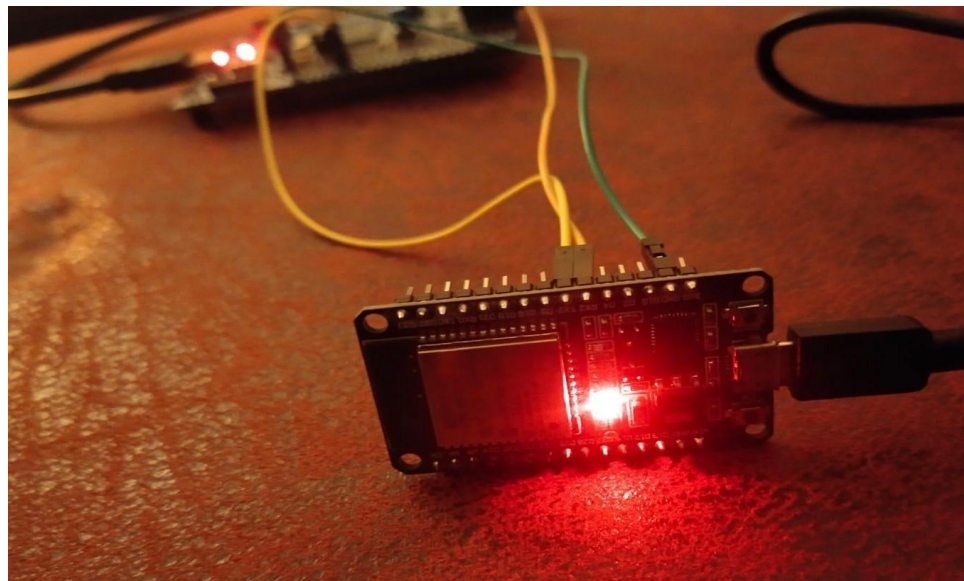


```
99 }
100 int Uart_SendByte(unsigned char byte)
101 {
102     SerialPort.begin(115200, SERIAL_8N1,16,17);
103     int n;
104     Serial.println("byte");
105     Serial.println(byte);
106     Serial.println("\n");
107     //sprintf Tsdata, "Hello world index = %d\r\n", num++);
108     delay(1);
109     n = SerialPort.write(byte);
110     if (n < 0) {
111         if (errno == EAGAIN) {
112             Serial.println("Send Byte Err");
113             return 0;
114         } else {
115             return 1;
116         }
117     }
118 }
119 int Uart_PollConport(uint8_t buf[], int size)
120 {
121     SerialPort.begin(115200, SERIAL_8N1,16,17);
122     int n;
123     int bit = 0;
124     int s = 0;
125     //static const char *RX_TASK_TAG = "RX_TASK";
126     //esp_log_level_set(RX_TASK_TAG, ESP_LOG_INFO);
127     //uint8_t* data = (uint8_t*) malloc(RX_BUF_SIZE+1);
```

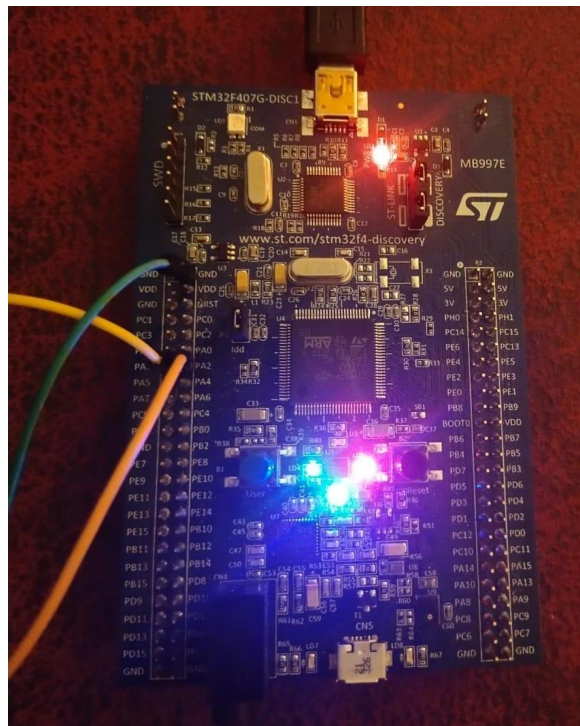
Output: Serial Monitor

```
Writing at 0x0000927c... (69 %)
Writing at 0x0000929a... (72 %)
Writing at 0x0000a298... (75 %)
Writing at 0x0000a90e... (78 %)
Writing at 0x0000e901... (81 %)
Writing at 0x0000710a... (84 %)
Writing at 0x0000f3b7... (87 %)
Writing at 0x000c45c3... (90 %)
Writing at 0x000cc234... (93 %)
Writing at 0x000d27ac... (96 %)
Writing at 0x000d7cd9... (100 %)
Wrote 848336 bytes (539560 compressed) at 0x00010000 in 7.6 seconds (effective 879.8 kbit/s)...
Hash of data verified.
Leaving...
Hard resetting via RTS pin...
```

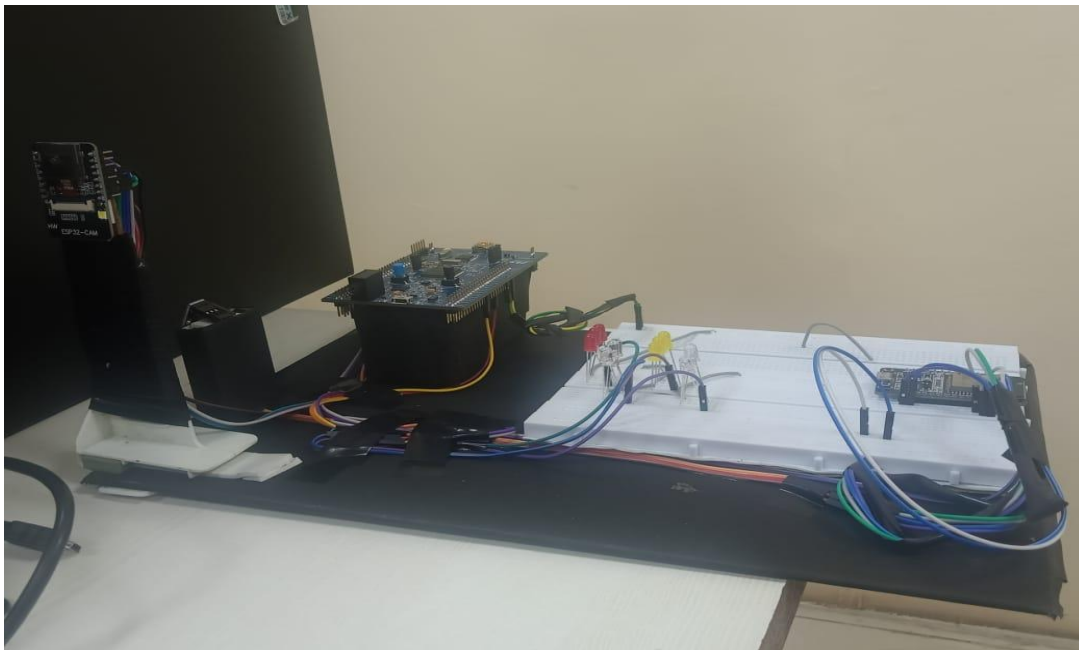
### 4. Esp32 Started



## 5. New Application Starts Running



## 6. Final model



## REFERENCES

- [1] Sotelo, M. Á., Fernández, D., Naranjo, E., González, C., García, R., de Pedro, T., & Reviejo, J. (2004, August). Laser-Based Adaptive Cruise Control for Intelligent Vehicles. In *First International Conference on Informatics in Control, Automation and Robotics* (Vol. 3, pp. 398-401). SCITEPRESS.
- [2] Milanés, V., Shladover, S. E., Spring, J., Nowakowski, C., Kawazoe, H., & Nakamura, M. (2013). Cooperative adaptive cruise control in real traffic situations. *IEEE Transactions on intelligent transportation systems*, 15(1), 296-305.
- [3] Yadav, A. K., & Szpytko, J. (2017). Safety problems in vehicles with adaptive cruise control system. *Journal of KONBiN*, 42(1), 389.
- [4] Widmann, G. R., Daniels, M. K., Hamilton, L., Humm, L., Riley, B., Schiffmann, J. K., ... & Wishon, W. H. (2000). Comparison of lidar-based and radar-based adaptive cruise control systems. *SAE transactions*, 126-139.
- [5] Liu, Y., Sun, B., Tian, Y., Wang, X., Zhu, Y., Huai, R., & Shen, Y. (2023). Software-Defined Active LiDARs for Autonomous Driving: A Parallel Intelligence-Based Adaptive Model. *IEEE Transactions on Intelligent Vehicles*.
- [6] Donzia, S. K. Y., Kim, H. K., & Geum, Y. P. (2021, September). Implementation of Autoware Application to real-world Services Based Adaptive Big Data Management System for Autonomous Driving. In *2021 21st International Conference on Computational Science and Its Applications (ICCSA)* (pp. 251-257). IEEE.