INDEX

ABSTRACT

This project aims to automatically detect the location of a collided or accidental vehicle using a Node MCU module, GSM, and GPS modules. The Node MCU is programmed to track the vehicle's location, and in the event of an accident, it transmits an alert message with the vehicle's position, i.e., latitude and longitude, to a designated mobile phone. This ensures that rescue teams can quickly reach the accident spot. Vehicle tracking involves monitoring the vehicle's location using GPS coordinates, which can also help detect stolen vehicles. The GPS data is fed to the Node MCU and displayed on an LCD. The vehicle is continuously tracked, and in the event of a crash, location information is automatically sent to the concerned mobile. The hit sensor circuit, designed with a limit switch and lever, activates when the vehicle crashes, triggering the controller to collect and transmit the GPS data via the GSM module. Two limit switches identify if the crash occurred at the front or rear, and a push button sends a "safe" message if the occupants are unharmed.

The GPS provides parameters like longitude and latitude, making it easy to identify the crashed vehicle's position. The GSM modem ensures that data can be transmitted globally without range restrictions. When a crash is detected, the microcontroller uses the GPS modem to determine the vehicle's location and sends this information to a designated mobile phone via the GSM modem. The system is portable and battery-powered, allowing it to collect coordinates from various points. The Node MCU has limited I/O lines, so an I2C LCD interfacing module is used, which converts I2C data from the Node MCU into the parallel data required for the LCD. This setup requires only two lines (SDA and SCL) for interfacing, and includes a trim pot for display contrast adjustments and a jumper for backlight power.

The major components of this project include the main processing unit built with Node MCU, GSM module, GPS module, limit switches, push button, LCD, rechargeable battery and charger, suitable PCB, and I2C LCD interfacing module. This setup ensures efficient communication between the vehicle and the mobile phone, providing accurate location data in case of an accident, and enabling timely rescue operations.

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

S. No	Abbreviations	Full form
1.	ESP8266	Espressif Systems 8266
2.	SDA	Serial Data Line
3.	SCL	Serial Clock Line
4.	PCB	Printed Circuit Board
5.	GPIO	General Purpose Input/Output
6.	GSM	Global System for Mobile Communications
7.	GPS	Global Positioning System
8.	I2C	Inter Integrated Circuit
9.	LCD	Liquid Crystal Display

1. INTRODUCTION

1.1 About the project

This project develops an automated vehicle accident detection and location reporting system using Node MCU, GPS, and GSM modules to enhance rescue operations by providing timely and precise accident location data. The system detects crashes via limit switches and uses a push button for safety messages if occupants are unharmed. An I2C LCD displays real-time location data, ensuring user-friendly access to critical information. By integrating GPS accuracy with GSM communication, the system effectively covers large geographical areas, contributing to vehicle safety and accident response technology with a practical and scalable solution.

1.2 Background of the project

Vehicle tracking systems have become increasingly popular for monitoring the position of vehicles globally. Equipped with GPS and GSM systems, modern vehicles can transmit their location to the concerned authorities via mobile phones. However, these systems typically do not provide information about accidents. This project aims to fill this gap by designing a system that automatically sends the location of an accident to the concerned mobile phone, allowing rescue teams to reach the spot immediately. Utilizing the Global Positioning System (GPS) for accurate location tracking and the Global System for Mobile Communications (GSM) for data transmission, this system ensures that the exact location of a crashed vehicle can be identified and communicated efficiently.

1.3 Objectives of this project

- Implement GSM technology to transmit accident location data to the concerned mobile phone in real-time.
- Identify whether the crash occurred at the front or rear of the vehicle using limit switches.
- Display location data on an LCD screen and ensure the system is easy to operate and understand.
- Enable quick response from rescue teams by providing accurate accident location information promptly.

1.4 Methodology

The development of an automated vehicle accident detection and location reporting system involves a structured process, starting with system design, where the Node MCU is selected as the central unit for its efficiency. Limit switches detect crashes, distinguishing between front and rear impacts, while a GPS module provides precise location coordinates and a GSM module transmits these details via SMS. An I2C LCD module displays real-time data. During hardware setup, components like GPS, GSM, and limit switches are assembled and connected to the Node MCU, powered by a rechargeable battery for portability, and mounted on a PCB for durability. In software development, the Node MCU is programmed to read GPS data, detect crashes, and transmit location information, with a user-friendly interface on the LCD.

Testing ensures accuracy and reliability, validating GPS precision, GSM transmission, and crash detection. The system is then deployed in a vehicle for real-world testing, with adjustments made based on feedback to enhance performance, ultimately creating a robust vehicle tracking and accident detection system to improve rescue response times and provide accurate location data in emergencies.

1.5 Hardware and Software Used

Hardware

1.Node MCU

NodeMCU is an open-source IoT platform with the ESP8266/ESP32 microcontroller, offering Wi-Fi, Lua support, and is popular for DIY projects and prototyping.

2.Limit Switches

Limit switches are mechanical devices that detect objects to control movement and ensure safety in machinery by preventing over-travel or collisions.

3.GSM

GSM (Global System for Mobile Communications) is a standard for mobile networks that enables voice, SMS, and data services globally.

4.GPS

GPS (Global Positioning System) is a satellite-based navigation system that provides precise location and time information anywhere on Earth.

4.LCD

LCD (Liquid Crystal Display) is a flat-panel display technology that uses liquid crystals to produce images, widely used in screens for TVs, computers, and smartphones.

Software

- Arduino IDE

Arduino IDE is an open-source software used to write, compile, and upload code to Arduino microcontrollers. It supports multiple programming languages, with a simple interface ideal for beginners and hobbyists.

\

2. HARDWARE AND SOFTWARE DESCRIPTION

2.1 Node MCU

The main processing unit is constructed with Node MCU and it is an open source IoT platform. It includes both firmware which runs on the ESP8266 Wi-Fi SoC, and hardware which is based on the ESP-12 module. The applications in these samples that are running on Node MCU are written using Lua scripting language which is quite simple and easy to understand. In other words, the Node MCU (Micro Controller Unit) is an open-source software and hardware development environment built around an inexpensive System-on-a-Chip (SoC) called the ESP8266. The ESP8266, designed and manufactured by Espressif Systems, contains the crucial elements of a computer: CPU, RAM, networking (WiFi), and even a modern operating system and SDK. That makes it an excellent choice for Internet of Things (IoT) projects of all kinds.

However, as a chip, the ESP8266 is also hard to access and use. It is supposed to program it in low-level machine instructions that can be interpreted by the chip hardware. This level of integration is not a problem using the ESP8266 as an embedded controller chip in mass-produced electronics. It is a huge burden for hobbyists, or students who want to experiment with it in their own IoT projects.

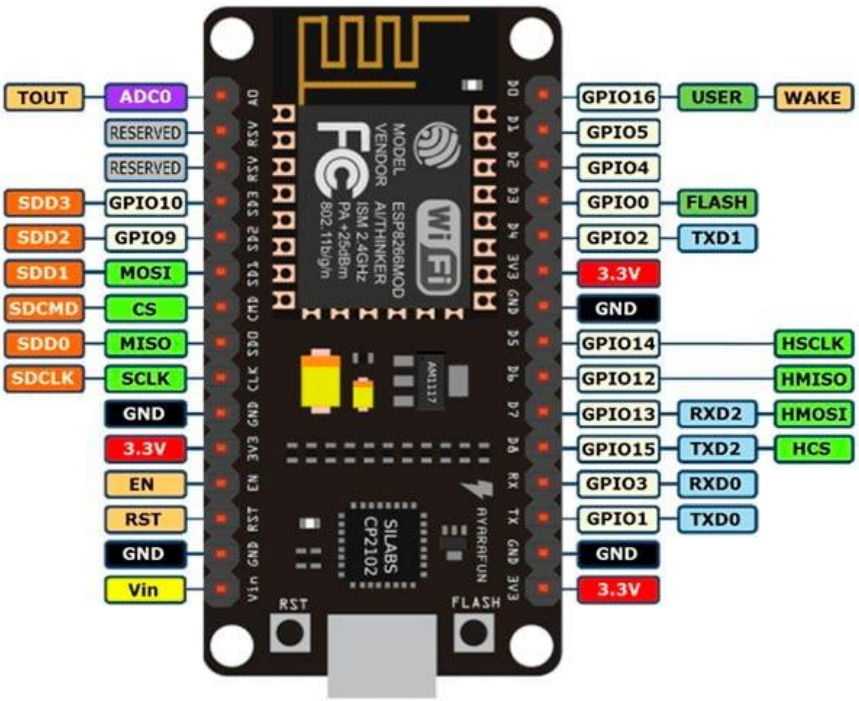

Fig. 2.1 Node MCU ESP8266

Specifications and Construction

- The onboard ESP8266 chip manufactured by Espressif Systems has an ESP8266EX core Wi-Fi SoC and a Tensilica L106 32-bit processor in the same chip.
- Node MCU ESP8266 has 2.4GHz 802.11b/g/n Wi-fi with on-board antenna which supports WPA/WPA2.
- It has 17 GPIO, 11 are usable(6 are used for communication with the onboard flash memory chip) and some of them support PWM. This GPIO can be used to interface ESP8266 NodeMCU with the external environment through sensors and actuators.
- It also has UART, SDIO, SPI, I2C, I2S, and IR remote control peripheral interface.
- It operates on 2.5-3.6V and average 80mA current with temperatures ranging from -40 to 125 C.
- It comes in a QFN 32-pin (5mm * 5mm) package.
- It supports firmware upgrades via UART download/ OTA (via network).
- It also has SDK for customized development/cloud server development.

Let us look at the Node MCU ESP8266 pin diagram.

Pins from D0-D8 can be used as GPIO pins.

Let's discuss these ESP8266 NodeMCU pin configuration in detail.

UART pins

GPIO1 and GPIO3 are tx and rx pins respectively used for UART communication.

I2C pins

D1 and D2 can be used SCL and SDA for I2C communication.

Note that NodeMCU doesn't has hardware I2C pins. We can implement I2C in software on any GPIO pins.

SPI pins

D5-D8 (SCLK,MISO,MOSI,AND CS) can be used for SPI communication.

You can see that in the above pinout diagram we have not labelled pins from GPIO6 to GPIO11. The reason they are not shown is they cannot be used for interfacing because they are connected ot the flash chip.

Analog Pin

The pin marked as A0 on the silkscreen layer of PCB is the only pin which can analog read.

ESP8266 On-board LED:

On-board LED is connected to GPIO2.

Reset pin

You can reset the ESP8266 NodeMCU board in 2 ways one is by pressing the reset button and other is by pulling the reset pin LOW.

I2C(Inter-Integrated Circuit) Interface of ESP32

I2C interface of ESP8266 is used for communications between ESP8266 (Master) and sensors (Slaves). For such communications, the ESP8266's GPIO-2 and GPIO-3 pins are allocated. All the I2C pins are listed with PINs in the following table:

Table 2.1 List of I2C pins used in corresponding ESP8266 pins

I2C Communications	ESP8266 Pins
SCL	D2
SDA	D3

2.2 Limit Switches

Limit switch is one kind of force sensor. In this project, it is used to detect the accident when the vehicle collides with obstacles and this switch is interfaced with Arduino as input signal. This limit switch is having long lever & when little pressure is applied to the lever, switch will be activated automatically. Limit switch is one of the

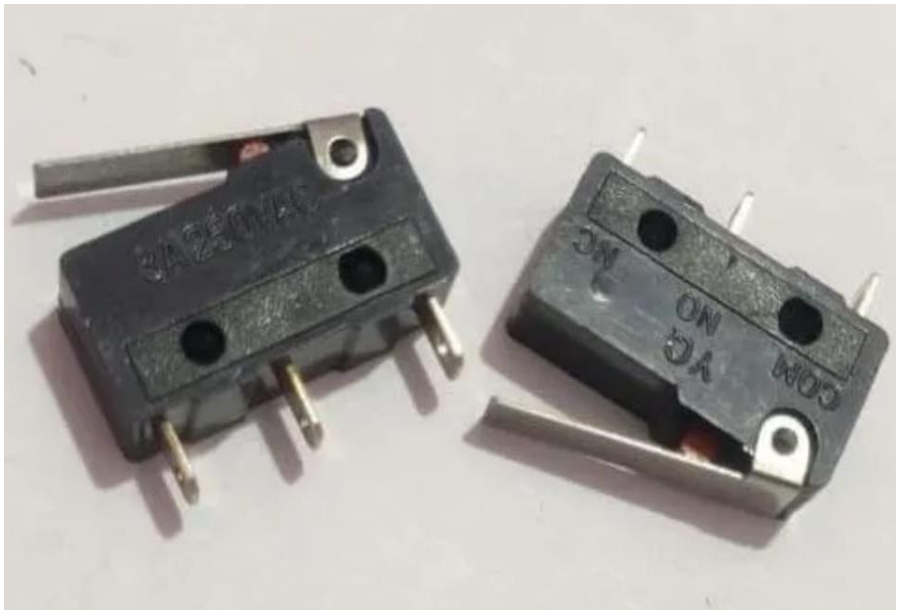

Fig. 2.2 Limit Switches

Limit switch is one kind of force sensor. In this project, it is used to detect the accident when the vehicle collides with obstacles and this switch is interfaced with Arduino as input signal. This limit switch is having long lever & when little pressure is applied to the lever, switch will be activated automatically. Limit switch is one of the most basic of all sensors available which is reliable, exhibit the lowest noise and most easily interpreted signal. Limit switches are small and normally used as bumper detection in robot application. A thin aluminium plate and a spring are placed together with the limit switch, which act like a long push button. There are two types of limit switch which are normally open and normally closed. Commonly used is the normally open, when the accident is occurred, logic low, '0' will be sent to micro controller. In contrast, when the vehicle did not detect any obstacles (no collision or contact), micro controller will detect logic 'high' 1 but it more depends on how the connection of limit switches. There are difference types and design of limit switches available in the market. However all limit switches have the same operating concept.

2.3 GSM

GSM (Global System for Mobile Communications) is a digital cellular communications system. It was developed in order to create a common European mobile telephone standard but it has been rapidly accepted worldwide. GSM is designed to provide a comprehensive range of services and features to the users not available on analogue cellular networks and in many cases very much in advance of the old public switched telephone network (PSTN). In addition to digital transmission, GSM incorporates many advanced services and features like worldwide roaming in other GSM networks During the early 1980s, analog cellular telephone systems were experiencing rapid growth in Europe, particularly in Scandinavia and the United Kingdom, but also in France and Germany. Each country developed its own system, which was incompatible with everyone else in equipment and operation. This was an undesirable situation, because not only was the mobile equipment limited to operation with national boundaries, which in a unified Europe were increasingly unimportant, but there was also a very limited market for each type of equipment, so economies of scale and the subsequent savings could not be realized. The Europeans realized this early on, and in 1982 the

conference of European Posts and Telegraphs (CEPT) formed a study group called the Group Special Mobile (GSM) to study and develop a pan-European public land mobile system. The proposed system has to meet certain criteria

Fig. 2.3 GSM

The proposed system has to meet certain criteria

- Good subjective speech quality
- Low terminal and service cost
- Support for international roaming
- Ability to support handheld terminals
- Support for range of new services and facilities
- Spectral efficiency

And interaction with integrated service digital network (ISDN) which offers the capability to extend the single-subscriber – line system with the various to a multi-service system. The first commercial GSM system, called D2, was implemented in Germany in 1982.This valuable channel of communication can equip us with a powerful tool for controlling desired device or process parameter from distant location, through

electromagnetic waves. With a little effort logic can be setup to even receive a feedback on the status of the device or the process being controlled.

SEND MESSAGE +CMGS

Description

The <address> field is the address of the terminal to which the message is sent. To send the message, simply type, <ctrl-z> character (ACII 26). The text can contain all existing characters except <ctrl-z> and <ESC> (ASCII 27). This command can be aborted using the <ESC> character when entering text. In PDU mode, only hexadecimal characters are used ('0'…'9','A'…'F').

Syntax

Command syntax in text mode:

AT+CMGS= <da> [,<toda>] <CR> text is entered <ctrl-z / ESC >

Command syntax in PDU mode:

AT+CMGS= <length><CR> PDU is entered <ctrl-z / ESC >

Table 2.3 List of Commands used for GSM

Command	Possible response
AT+CMGS="+33146290800" <CR> Please call me soon fred. <ctrl.z> Note: send a message in text mode.	+CMGS;<mr> OK Note: successful transmission
AT+CMGS+<length><CR><pdu><ctrlz> Note: Send a message ij pdu mode	+CMGS;<mr> OK Note: successful transmission

Example for CMGS Commands

The message reference, <mr>, which is returned to the application, is allocated by the product. This number begins with 0 and is incremented by one for each outgoing message (successful and failure cases); it is cyclic on one byte (0 follows 255).

Note: this number is not a storage number – outgoing message are not stored.

2.4 LCD

The liquid crystal display uses the property of light monitoring of liquid crystal and they do not emit the light directly. The Liquid crystal display is a flat panel display or the electronic visual display. With low information, content the LCD' s are obtained in the fixed image or the arbitrary image which are displayed or hidden like present words, digits, or 7 segment display. The arbitrary images are made up of large no of small pixels and the element has larger elements.

Liquid Crystal Display of 16×2

The 16×2 liquid crystal display contains two horizontal lines and they are used for compressing the space of 16 display characters. In inbuilt, the LCD has two registers which are described below.

Fig 2.4 LCD

Command Register: This register is used to insert a special command in the LCD. The command is a special set of data and it is used to give the internal command to the liquid crystal display like clear screen, move to line 1 character 1, setting the curser and etc.

Data Register: The data registers are used to enter the line in the LCD

Table 2.4(a) List of Pins for GCD

Pin No	Pin Name	Pin Description
Pin 1	GND	This pin is a ground pin and the LCD is connected to the Ground
Pin 2	VCC	The VCC pin is used to supply the power to the LCD
Pin 3	VEE	This pin is used for adjusting the contrast of the LCD by connecting the variable resistor in between the VCC & Ground.
Pin 4	RS	The RS is known as register select and it selects the Command/Data register. To select the command register the RS should be equal to zero. To select the Data register the RS should be equal to one.
Pin 5	R/W	This pin is used to select the operations of Read/Write. To perform the write operations the R/W should be equal to zero. To perform the read operations the R/W should be equal to one.
Pin 6	EN	This is a enable signal pin if the positive pulses are passing through a pin, then the pin function as a read/write pin.
Pin 7	DB0 to DB7	The pin 7 contains total 8 pins which are used as a Data pin of LCD.
Pin 15	LED +	This pin is connected to VCC and it is used for the pin 16 to set up the glow of backlight of LCD.
Pin 16	LED –	This pin is connected to Ground and it is used for the pin 15 to set up the glow of backlight of the LCD.

LCD Interfacing

.In this project, the RS pin of the LCD is connected to a digital pin of the Arduino, with the program configured accordingly. The R/W pin of the LCD is grounded, and the LCD is interfaced in 4-bit mode, using Arduino pins 2-5 for the data lines DB4-DB7. A 1K potentiometer is used to adjust the display contrast, and a 100-ohm resistor is connected to the backlight LED. The LCD is powered by a +5V supply, while the Arduino is powered through a 9V regulator.

LCD's are widely used due to their ability to display numbers, characters, and graphics, surpassing the limited functionality of seven-segment displays. The data on the LCD remains until an erase signal is received. The RS pin selects between the command and data registers; RS = 0 selects the command register, while RS = 1 selects the data register. The busy flag (D7) should be checked before writing data to ensure the LCD is ready. If D7 is high, the LCD is busy; if low, it's ready to receive new data.

Table 2.4(b) List of Instruction command codes for LCD

Code	Command to LCD Instruction
1	Clear display screen
2	Return home
4	Decrement cursor (shift cursor to left)
6	Increment cursor (shift cursor to right)
5	Shift display right
7	Shift display left
8	Display off, cursor off
A	Display off, cursor on
C	Display on, cursor off
E	Display on, cursor blinking
F	Display on, cursor blinking
10	Shift cursor position to left
14	Shift cursor position to right
18	Shift the entire display to the left
IC	Shift the entire display to the right

80	Force cursor to beginning of first line
CO	Force cursor to beginning of second line
38	2 Lines and 5x7 Matrix

I2C (Inter-Integrated Circuit) Bus

I2C (Inter-Integrated Circuit) is a synchronous, multi-master/multi-slave serial communication bus invented by Philips in 1982. It's widely used for short-distance communication between microcontrollers and peripheral devices, prioritizing simplicity and low cost. I2C employs just two bidirectional lines, SDA (serial data) and SCL (serial clock), making it efficient for controlling multiple devices with minimal I/O pins. It supports various speeds, including standard mode (100 kbit/s), fast mode (400 kbit/s), and higher speeds up to 5 Mbit/s. I2C's key features include clock stretching and arbitration, allowing for reliable communication between multiple controllers and targets. It is commonly used in applications like sensors, real-time clocks, and small displays. SMB (System Management Bus) is a subset of I2C with stricter rules for robustness, while PMB (Power Management Bus) extends SMB with additional protocols.

2.5 GPS

The Global Positioning System (GPS) is a space-based global navigation satellite system that provides reliable location and time information in all weather and at all times and anywhere on or near the Earth when and where there is an unobstructed line of sight to four or more GPS satellites. It is maintained by the United States government and is freely accessible by anyone with a GPS receiver.

GPS was created and realized by the U.S. Department of Defense (DOD) and was originally run with 24 satellites. It was established in 1973 to overcome the limitations of previous navigation systems.

GPS consists of three parts: the space segment, the control segment, and the user segment. The U.S. Air Force develops, maintains, and operates the space and control segments. GPS satellites broadcast signals from space, which each GPS receiver uses to

calculate its three-dimensional location (latitude, longitude, and altitude) plus the current time.

The space segment is composed of 24 to 32 satellites in medium Earth orbit and also includes the boosters required to launch them into orbit. The control segment is composed of a master control station, an alternate master control station, and a host of dedicated and shared ground antennas and monitor stations. The user segment is composed of hundreds of thousands of U.S. and allied military users of the secure GPS Precise Positioning Service and tens of millions of civil, commercial, and scientific users of the Standard Positioning Service (see GPS navigation devices).

A GPS tracking unit is a device that uses the Global Positioning System to determine the precise location of a vehicle, person, or other asset to which it is attached and to record the position of the asset at regular intervals. The recorded location data can be stored within the tracking unit, or it may be transmitted to a central location data base, or internet-connected computer, using a cellular , radio, or satellite modem embedded in the unit. This allows the asset's location to be displayed against a map backdrop either in real-time or when analysing the track later, using customized software.

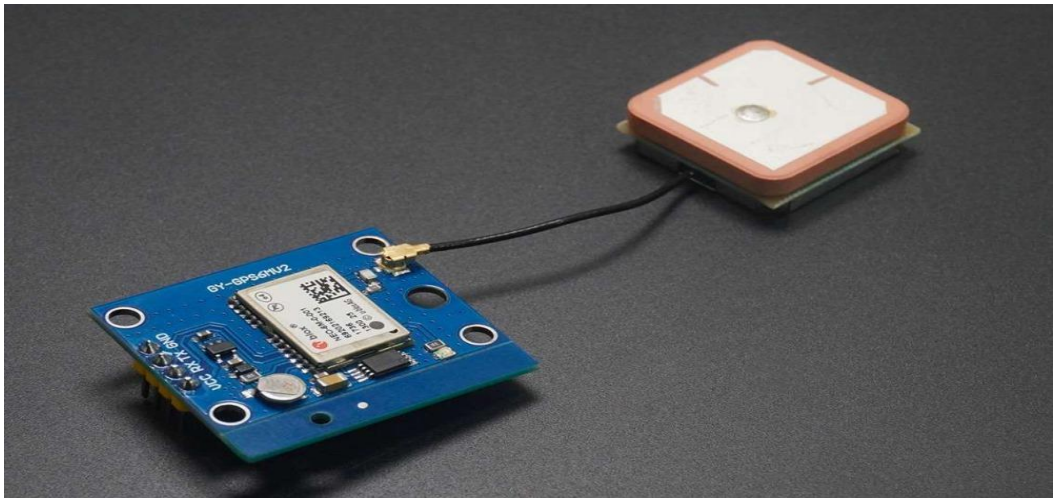

Fig. 2.5 GPS

A GPS tracking system uses the GNSS (Global Navigation Satellite System) network. This network incorporates a range of satellites that use microwave signals, which are transmitted to GPS devices to give information on location, vehicle speed, time

and direction. So, a GPS tracking system can potentially give both real-time and historic navigation data on any kind of journey.

A GPS tracking system can work in various ways. From a commercial perspective, GPS devices are generally used to record the position of vehicles as they make their journeys. Some systems will store the data within the GPS tracking system itself (known as passive tracking) and some send the information to a centralized database or system via a modem within the GPS system unit on a regular basis (known as active tracking).

2.6 Arduino IDE

The Arduino Integrated Development Environment (IDE) is an essential software platform for developing the code necessary for the Vehicle Accident Detection System using NodeMCU, GSM, and GPS. The Arduino IDE is an open-source platform that provides a comprehensive environment for writing, compiling, and uploading code to microcontroller boards, including the NodeMCU, which is based on the ESP8266 Wi-Fi module. It is widely used due to its simplicity and ease of use, making it suitable for both beginners and experienced developers alike.

One of the key features of the Arduino IDE is its intuitive code editor, which offers syntax highlighting, automatic indentation, and bracket matching, facilitating the coding process. Additionally, the IDE includes a robust library manager that allows users to easily add and manage libraries. These libraries are essential for extending the functionality of the project by providing pre-written code for various sensors and modules, such as the GPS and GSM modules used in this system.

The library manager simplifies the process of integrating these components into the project.

Wire.h: Library for I2C communication used with LCDs.

LiquidCrystal_I2C.h: Library for controlling I2C LCD displays.

Adafruit_GPS.h: Library for interfacing with GPS module

3. BLOCK DIAGRAM AND WORKING

The transmitter section consists of Node MCU, GPS receiver, GSM modem, etc and the whole system is designed to energize through rechargeable battery. The idea of using battery is to make the system as portable so that it can be carried in to the field to acquire different co-ordinate values for demo purpose.

3.1 Block diagram

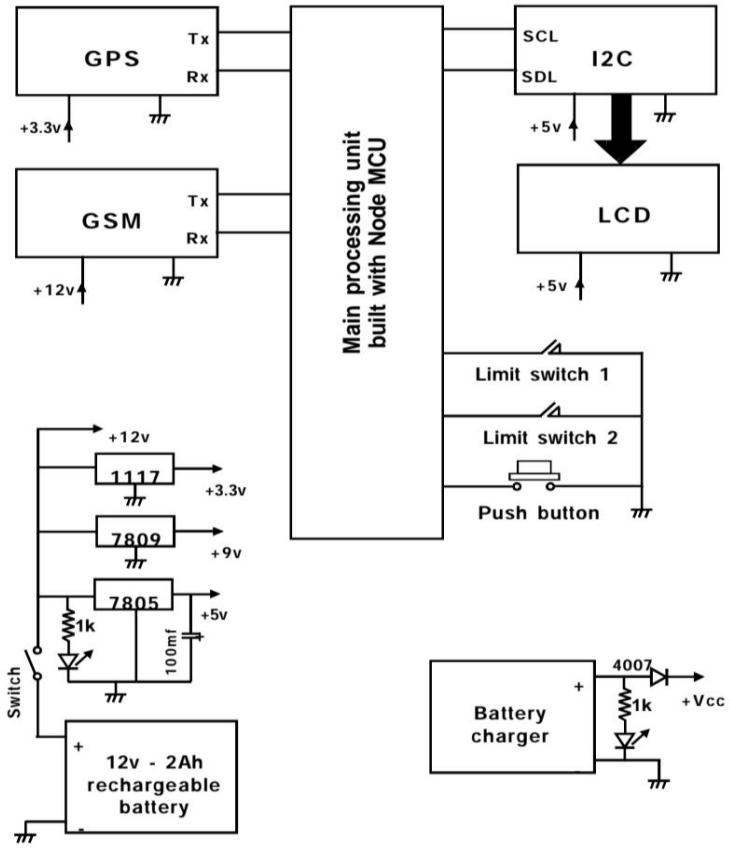

Fig 3.1 Block Diagram

GPS receives the data and GSM transmits and received data. So the GPS system will receive the Longitude and Latitude values corresponding collided vehicle position through the satellites. GPS TX pin is connected to the Rx pin of Arduino board. Similarly GSM module TX and RX pins are also connected to the same board. Here the processor is programmed to acquire data from GPS, display the data through LCD and transmit the same data to the concern mobile number through GSM module. A Program has been developed which is used to locate the exact position of the collided vehicle. Here to detect

the collided vehicle, limit switch is used and this switch is having long lever by which if the vehicle is damaged, switch will be activated automatically. This switch can be arranged inside the bonnet of the car such that lever will be pressed when accident took place. Whenever the switch is activated, the position of the vehicle will be transmitted. Since it is a prototype module, only 2 limit switches are used, but for real time applications, many switches can be used and they can place at different locations inside the car.

Now coming to the GSM, it is a Global system for mobile communications, in this project it acts as a SMS transmitter with a baud rate of 9600 bits/sec. In the battery mode, power is supplied to components like GSM, GPS and Arduino circuitry using a12v /2Ah rechargeable battery. GSM module is powered by 12v dc as this module circuit board is equipped with the regulators internally and derives the required voltages for the components to be operating in the board. The GPS requires a stable supply source of +3.3v DC and for this purpose LM1117 voltage regulator is used. Similarly for LCD and I2C devices, +5v regulated supply source is required and hence 7805 three pin regulator is used.

Programming the ESP32

Write the code for Esp32 using the Arduino IDE framework

• Include all the necessary libraries for Bluetooth communication. In the Arduino IDE, you might use libraries like "BluetoothSerial.h".

•And initialize the Bluetooth serial connection using the appropriate functions. This usually involves setting up the baud rate and configuring the serial pin.

The overview of the algorithm:

1. An instance of Bluetooth Serial named SerialBT is created for Bluetooth communication.

2. In the `setup()` function:

- Bluetooth communication is initiated with the device name "esp32BOT".

- GPIO pins 26, 25, 14, 27, and 33 are configured as outputs.

- The `Stop()` function is called to set all motor control pins to LOW.

3. The `Stop()` function turns off all motor control pins, effectively stopping the robot.

4. Motor control functions are defined:

- `Forward()`: Moves the robot forward.

- `Backward()`: Moves the robot backward.

- `Left()`: Rotates the robot to the left.

- `Right()`: Rotates the robot to the right.

5. In the `loop()` function:

- It checks if there is data available from the Bluetooth connection.

- If data is available, it reads a character (command) from the Bluetooth connection.

- Depending on the received command:

- If '1' is received, the left motor is turned on for 5 seconds.

- If '2' is received, the robot moves backward.

- If '3' is received, the robot turns left.

- If '4' is received, the robot turns right.

- If '5' is received, the `Stop()` function is called.

- If '6' is received, GPIO pin 33 is set to HIGH.

- If '7' is received, GPIO pin 33 is set to LOW.

- A delay of 50 milliseconds is introduced to prevent constant checking and execution of commands.

4. RESULTS AND DISCUSSIONS

The Project titled **"Vehicle Accident detection system**" is a model for Vehicle Tracking system which can be used for practical utilization because it is a real working system. Since the system acquires global position data in the form of latitude and longitude, these co-ordinates values must be entered in the Google maps by exact location will be displayed.

The positioning is done in the form of latitude and longitude along with the exact location of the place, by making use of Google maps. The system tracks the location of particular vehicle and sends to users mobile in the form of SMS. The received data can bedisplayed through LCD interfaced with Arduino board and this is the main processing unit. The information displayed through lcd is also transmitted to the concern mobile phone through GSM module. Since the GPS device used in the project work belongs to commercial model by which the location accuracy will be around 15 square metres.

1)Accident at front side of vehicle

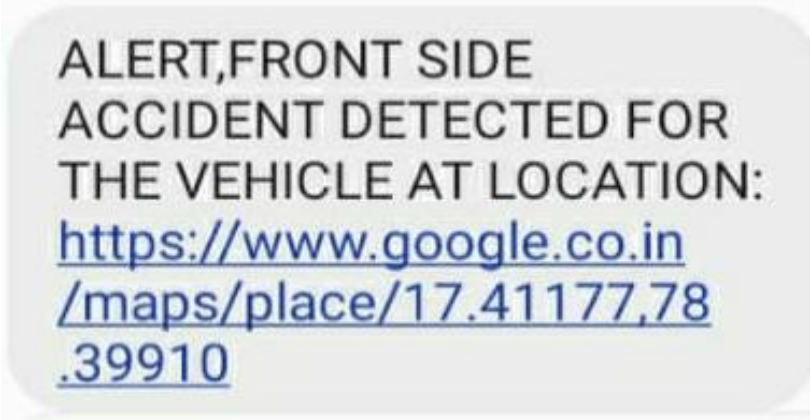

Fig. 4(a) Accident at front side of vehicle

2) Accident at Back side of vehicle

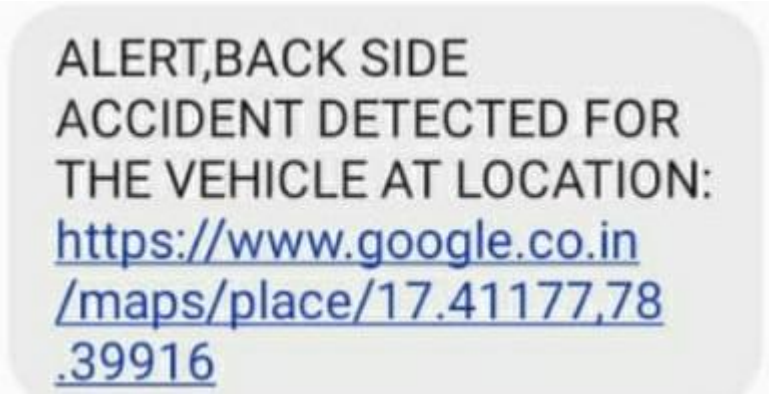

Fig. 4(b) Accident at back side of vehicle

3) Person is Safe

Fig. 4(c) Person is Safe

5. ADVANTAGES, DISADVANTAGES AND APPLICATIONS

5.1 Advantages

1. Rapid Emergency Response: The primary advantage of this system is its ability to quickly detect accidents and send immediate alerts to emergency services and predefined contacts. This rapid notification can significantly reduce the time taken for emergency responder's to reach the accident site, potentially saving lives by providing timely medical assistance.

2.Automated Alerts: The use of GSM technology enables the system to send automated SMS alerts containing critical information such as the accident location, time, and severity. This automation ensures that help is summoned even if the occupants are unconscious or unable to make a call for help themselves.

3.Enhanced Safety and Security: Having an accident detection system installed in vehicles increases the overall safety and security of passengers. It acts as a safeguard, providing peace of mind to drivers and their families, knowing that they have an additional layer of protection in case of emergencies.

4.Cost-Effective Solution: Using affordable and widely available components like NodeMCU, GSM, and GPS modules makes the system cost-effective. This affordability ensures that it can be implemented in a wide range of vehicles, from personal cars to commercial fleets, enhancing safety on a broader scale.

5.2 Disadvantages

1. Accuracy and Reliability: The system may produce false positives or miss accidents due to the limitations of limit switches and dependency on GSM network coverage.

2. Power and Maintenance: The setup relies on a consistent power source and requires regular maintenance, which can be challenging in remote areas and harsh environments.

3. Delayed Response: Delays in GPS lock and GSM transmission can result in slower accident detection and notification, reducing the system's effectiveness.

5.3 Applications

1.Public Transportation: Vehicle accident detection systems can be integrated into public transportation networks to enhance passenger safety. By providing real-time monitoring and instant alerts in case of accidents, these systems ensure that emergency services are promptly notified. This can significantly reduce response times, minimize the impact of accidents, and improve overall safety for passengers. Additionally, the data collected can help optimize routes and improve the reliability of public transportation services.

2.Vehicle Tracking and Recovery: Accident detection systems equipped with GPS tracking can assist in locating and recovering vehicles after an incident. In the event of an accident, the system can relay the vehicle's exact location to authorities or recovery services. This capability is crucial for quickly finding and securing vehicles, especially in remote or hard-to-reach areas. It also aids in theft prevention and recovery, as stolen vehicles can be tracked and retrieved more efficiently.

3. Emergency Response: The primary benefit of vehicle accident detection systems is their ability to alert emergency responders immediately when an accident occurs. These systems can automatically send detailed information, including the location, severity of the crash, and potential injuries, to emergency services. This rapid communication ensures that help is dispatched quickly, improving the chances of survival and reducing the severity of injuries for those involved.

6. CONCLUSION AND FUTURE SCOPE

6. 1 Conclusion

The "Vehicle Accident Detecting System" integrates GPS and GSM technologies to accurately detect vehicle accident locations. It leverages latitude and longitude data for precise tracking, viewable on Google Maps. A Node MCU board handles processing, and an LCD ensures user-friendly display. Critical updates are sent via SMS, offering a reliable and practical solution for enhancing vehicle safety and monitoring.

6.2 Future Scope

- Cloud-Based Data Storage and Analysis: Implementing cloud-based platforms to store and analyze accident data can help in identifying accident-prone areas and improving overall road safety. This data can also be used for insurance purposes and traffic management

- Raspberry Pi Integration for Image Transmission: Utilizing Raspberry Pi to capture and send real-time images or video of accident scenes to police, providing valuable visual information for accurate assessment and quicker response.

7. REFERENCES

1.Kumar, R.; Kumar, H., "Availability and handling of data received through GPS device: In tracking a vehicle," Advance Computing Conference (IACC), 2014 IEEE International, vol., no., pp.245, 249, 21- 22 Feb. 2014.

2. 9S. Roy, A. Kumari, P. Roy and R. Banerjee, "An Arduino Based Automatic Accident Detection and Location Communication System," 2020 IEEE 1st International Conference for Convergence in Engineering (ICCE), Kolkata, India

3.Y.Chenetal.,"Areal-time vehicle safety system," 2012 IEEE/SICE International Symposium on System Integration (SII), Fukuoka, Japan.

APPENDIX

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27,16,2);  // If your LCD has a PCF8574 chip from Texas Instruments, its default I2C address is 0x27Hex.
                                   //If your LCD has a PCF8574 chip from NXP semiconductors, its default I2C address is 0x3FHex.
//D2 Pin  ( SDA ) SDA Pin
//D1 Pin  ( SCL ) SCL Pin
#include <SoftwareSerial.h>
SoftwareSerial mySerial(16,14); //D0,D5
#include <Adafruit_GPS.h>
#define GPSSerial Serial
#define GPSECHO false
uint32_t timer = millis();
Adafruit_GPS GPS(&GPSSerial);
#define BUZ    D3
#define LS1    D4
#define LS2    D7
#define KEY    D6
int cnt=0;
int i=0;
void setup()
{
  lcd.init();
  lcd.clear();
  lcd.backlight();      // Make sure backlight is on
  lcd.setCursor(0,0);
  lcd.print("   WELCOME      ");
  lcd.setCursor(0,1);
  lcd.print("                ");
```

```
  Serial.begin(9600);
  mySerial.begin(9600);

  Serial.println("WELCOME");
  pinMode(LS1,INPUT);
  pinMode(LS2,INPUT);
  pinMode(KEY,INPUT);
  pinMode(BUZ,INPUT);
  digitalWrite(BUZ,LOW);
  digitalWrite(LS1,HIGH);
  digitalWrite(LS2,HIGH);
  digitalWrite(KEY,HIGH);
  delay(500);
  GPS.begin(9600);
  GPS.sendCommand(PMTK_SET_NMEA_OUTPUT_RMCGGA);
  GPS.sendCommand(PMTK_SET_NMEA_UPDATE_1HZ); // 1 Hz update rate
  GPS.sendCommand(PGCMD_ANTENNA);
  delay(1000);
  GPSSerial.println(PMTK_Q_RELEASE);
}
void loop()
{

home:
  gpss();
  for(i=0;i<100;i++)
  {
  if(digitalRead(LS1)==LOW)
  {
  lcd.setCursor(12,0);
  lcd.print("FRNT");
    digitalWrite(BUZ,HIGH);
```

```
  Serial.println("FRNT");
  mySerial.begin(9600);
  delay(500);
  mySerial.print("AT+CMGF=1\n\r");
  delay(1000);
  mySerial.println("AT"); //Handshaking with SIM900
  delay(1000);
  mySerial.println("AT+CMGF=1"); // Configuring TEXT mode
  delay(1000);
  mySerial.println("AT+CMGS=\"+917569216175\"");
 //  mySerial.println("AT+CMGS=\"+919885197565\"");
  delay(1000);
       mySerial.print("ALERT,FRONT SIDE ACCIDENT DETECTED FOR THE
VEHICLE AT LOCATION: https://www.google.co.in/maps/place/");
  mySerial.print(GPS.latitudeDegrees,5);
  mySerial.print(",");
  mySerial.print(GPS.longitudeDegrees,5);
  delay(1000);
  mySerial.write(26);
  delay(1000);
 digitalWrite(LS1,HIGH);
 digitalWrite(LS2,HIGH);
 digitalWrite(KEY,HIGH);
 }
 if(digitalRead(LS2)==LOW)
 {
 lcd.setCursor(12,0);
 lcd.print("BACK");
  digitalWrite(BUZ,HIGH);
  Serial.println("BACK");
  mySerial.begin(9600);
  delay(500);
```

```
   mySerial.print("AT+CMGF=1\n\r");
   delay(1000);
   mySerial.println("AT"); //Handshaking with SIM900
   delay(1000);
   mySerial.println("AT+CMGF=1"); // Configuring TEXT mode
   delay(1000);
     mySerial.println("AT+CMGS=\"+917569216175\"");//change ZZ with country code and xxxxxxxxxx with phone number to sms
//   mySerial.println("AT+CMGS=\"+919885197565\"");
   delay(1000);
    mySerial.print("ALERT,BACK SIDE ACCIDENT DETECTED FOR THE VEHICLE AT LOCATION: https://www.google.co.in/maps/place/");
   mySerial.print(GPS.latitudeDegrees,5);
   mySerial.print(",");
   mySerial.print(GPS.longitudeDegrees,5);
   delay(1000);
   mySerial.write(26);
   delay(1000);
  digitalWrite(LS1,HIGH);
  digitalWrite(LS2,HIGH);
  digitalWrite(KEY,HIGH);
  }
  if(digitalRead(KEY)==LOW)
  {
  lcd.setCursor(12,0);
  lcd.print("SAFE");
   digitalWrite(BUZ,LOW);
   Serial.println("SAFE");
   mySerial.begin(9600);
   delay(500);
   mySerial.print("AT+CMGF=1\n\r");
   delay(1000);
```

```
  mySerial.println("AT"); //Handshaking with SIM900
  delay(1000);
  mySerial.println("AT+CMGF=1"); // Configuring TEXT mode
  delay(1000);
    mySerial.println("AT+CMGS=\"+917569216175\"");//change ZZ with country code and xxxxxxxxxxx with phone number to sms
 // mySerial.println("AT+CMGS=\"+919885197565\"");
  delay(1000);
                              mySerial.print("ALERT,WE ARE SAFE AT LOCATION: https://www.google.co.in/maps/place/");
  mySerial.print(GPS.latitudeDegrees,5);
  mySerial.print(",");
  mySerial.print(GPS.longitudeDegrees,5);
  delay(1000);
  mySerial.write(26);
  delay(1000);
 digitalWrite(LS1,HIGH);
 digitalWrite(LS2,HIGH);
 digitalWrite(KEY,HIGH);
 }
 delay(10);
 }
 cnt=cnt+1;
 if(cnt==20)
 {
 cnt=0;
 lcd.setCursor(13,0);
 lcd.print("SMS");
  mySerial.begin(9600);
  delay(500);
  mySerial.print("AT+CMGF=1\n\r");
  delay(1000);
```

```
   mySerial.println("AT"); //Handshaking with SIM900
   delay(1000);
   mySerial.println("AT+CMGF=1"); // Configuring TEXT mode
   delay(1000);
     mySerial.println("AT+CMGS=\"+917569216175\"");//change ZZ with country code and xxxxxxxxxx with phone number to sms
//    mySerial.println("AT+CMGS=\"+919885197565\"");
   delay(1000);
mySerial.print("VEHICLE IS AT LOCATION: https://www.google.co.in/maps/place/");
   mySerial.print(GPS.latitudeDegrees,5);
   mySerial.print(",");
   mySerial.print(GPS.longitudeDegrees,5);
   delay(1000);
   mySerial.write(26);
   delay(1000);
  }
}

void gpss()
 {
  GPS.begin(9600);
  GPS.sendCommand(PMTK_SET_NMEA_OUTPUT_RMCGGA);
  GPS.sendCommand(PMTK_SET_NMEA_UPDATE_1HZ); // 1 Hz update rate
  GPS.sendCommand(PGCMD_ANTENNA);
  delay(1000);
  GPSSerial.println(PMTK_Q_RELEASE);
 again:

  char c = GPS.read();
  if (GPSECHO)
  if (c) Serial.print(c);
  if (GPS.newNMEAreceived()) {
```

```
  //   Serial.println(GPS.lastNMEA()); // this also sets the newNMEAreceived() flag to false
     if (!GPS.parse(GPS.lastNMEA())) // this also sets the newNMEAreceived() flag to false
      goto again; // we can fail to parse a sentence in which case we should just wait for another
  }
  else
  {
   goto again;
  }
  if (timer > millis()) timer = millis();
  if (millis() - timer > 2000) {
   timer = millis(); // reset the timer

   if (GPS.fix) {
    Serial.print("Location: ");
    Serial.print(GPS.latitude, 4); Serial.print(GPS.lat);
    Serial.print(", ");
    Serial.print(GPS.longitude, 4); Serial.println(GPS.lon);
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print(GPS.latitude, 4); lcd.print(GPS.lat);
    lcd.setCursor(0,1);
    lcd.print(GPS.longitude, 4); lcd.print(GPS.lon);
    delay(500);
    }
   else
   {
   Serial.print("gps not initilized ");
   lcd.setCursor(0,0);
   lcd.print("   WELCOME      ");
```

```
    lcd.setCursor(0,1);
    lcd.print(" GPS NOT READY  ");
    goto again;
    }
  }
}
```