ROADTRACK

A

major project report

submitted in the partial fulfillment of the

academic requirements

for the award of the degree of

Bachelor of Technology

in

Electronics and Communication Engineering

by

A.SRIKANTH REDDY	**13AG1A0401**
DHEERAJ ALLAMANENI	**13AG1A0408**
Ch.VAISHNAVI	**13AG1A0428**

Under the esteemed guidance of

Dr.P.SUMITHABHASHINI

PROFESSOR

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEEERING

ACE Engineering College

(Sponsored by Yadala Satyanarayana Memorial Educational Society, Hyderabad)

Approved by AICTE & Affiliated to JNTUH

NBA Accredited B.Tech courses: EEE,ECE &CSE

Ankushapur(V), Ghatkesar(M), R.R.Dist-501301

2016-17

ACE Engineering College

Ankushapur (V), Ghatkesar (M), R.R.Dist – 501 301

(Approved by AICTE, New Delhi and Affiliated to JNTUH)

NBA Accredited B.Tech Courses: EEE, ECE & CSE

CERTIFICATE

This is to certify that the major project entitled **'' ROADTRACK''** done by

A.SRIKANTH REDDY	**13AG1A0401**
DHEERAJ ALLAMANENI	**13AG1A0408**
Ch.VAISHNAVI	**13AG1A0428**

of **Department of Electronics and Communication Engineering**, is a record of bonafide work carried out by them. This major project is done as a partial fulfillment of obtaining **Bachelor of Technology** Degree to be awarded by **Jawaharlal Nehru Technological University, Hyderabad**, during the academic year 2016-17.

Dr. P.Sumithabhashini

(Professor)

(MAJOR PROJECT SUPERVISOR)

Dr. V. S. S. N. Srinivasa Baba

Professor & HOD of ECE

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

Hard work, commitment and team spirit are the basic ingredients for the success of any group task. Be it in education, sports, arena, battlefield or even in our lives, similarly this project is the result of contribution from the students and staff. While doing this project we have come across many difficulties but finally we could complete it with the guidance and suggestions from the following intelligentsia who gave generously of their time and expertise.

We take the opportunity to thank one and all who have helped in making this project possible. We are thankful to **JNTUH** for giving us this opportunity to work on a project as a part of our curriculum.

We would like to convey our heart full thanks to **Dr. B. L. RAJU, Principal of the college**, for the wonderful encouragement given to us to move ahead the execution of this project.

We are grateful to thank **Dr. V. S. S. N. SRINIVASA BABA, Head of the Dept. of ECE,** for this thought, provoking suggestions and constant encouragement that led us to this success.

We are very happy for being guided by **Dr. P.SUMITHABHASHINI**, for her able guidance given to us to complete our technical work successfully.

We wish to express our deep sense of gratitude to our **project coordinator, Prof. B. GIRIRAJU** of ECE Department, **Mr. S. KARUNAKAR REDDY, Associate Professor ECE Dept**, for their valuable guidance throughout this project.

Above all, we are very much thankful to the management of **ACE Engineering College** which was established by the high profiled intellectuals for the cause of technical education in modern era. We wish that **ACE** sooner should become a deemed university and produce uncountable young engineers and present them to the modern technical world.

With regards

A.SRIKANTH REDDY (13AG1A0401)

DHEERAJ ALLAMANENI (13AG1A0408)

Ch.VAISHNAVI (13AG1A0428)

CONTENTS

4. BREADBOARD AND USB

5. GSM AND GPS MODULE

6. SERVERS

7. PRODRAM OF ROADTRACK

List of Figures

List of Tables

CHAPTER-1

INTRODUCTION TO ROADTRACK

1.1. Introduction

This project solves the problem that is unattended by any (Google maps, Apple maps, Yandex.).One of the most important factors for a safe journey or safe/careful transportation is the road quality. This major attribute is never been mapped. Using various sensors attached to a bike frame to measure road quality (cobblestone, gravel, bumps, and holes) and logs it by combining it with GPS and odometry data. The data can be analyzed to extract road features, classify different types of roads and detect damaged areas.

The aim is to provide cyclists (and eventually: skateboarders, long boarders and others) with smarter route planning by prioritizing smooth roads. Additionally, it can serve local governments to focus repair work on the streets that need it most.

The system can also be used to measure the intensity of downhill and off-road tracks. This allows the user to measure their own performance over time, compare different runs (also against other users), and choose new paths based on the desired difficulty/intensity.

We initially developed hardware which detects the vibrations and send all the data (speed, vibration signal, Geo Coordinates, UTC time) to wolfram database.

The Vibration signal(Waveform) collected in our database is analyzed and classified among 12 pre-defined classifiers ,which determines the quality of road (eg: smooth, Patterned, Potholes, Rough road, etc) using Mathematica and these are plotted on a map in real time. We also developed a web application to plot the road quality recorded on map.The problems that we are working on.

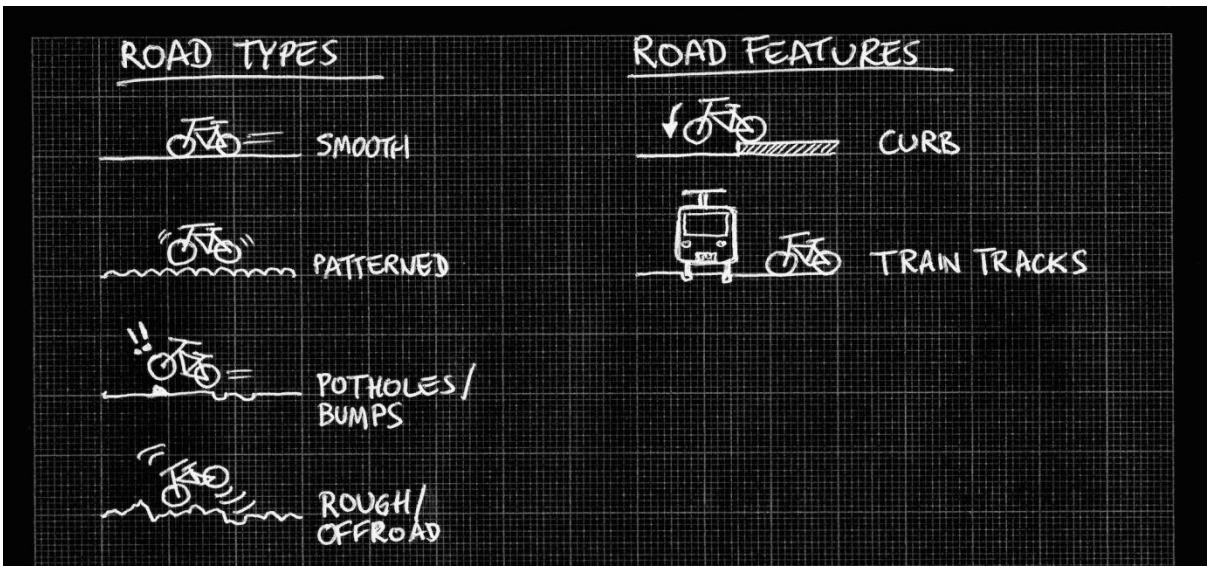

Fig.1.1. Target areas

As displayed in the image above, the application that we are developing has to be generalized so as, it shall have no problems in collection of data. In case of smooth roads, the data collection is done seamlessly, but whereas in case of patterned roads or off terrain roads, the data collection has to be accurate, so as the visualization of the different segments of the project are clearly depicted.

We have implemented a pattern recognizing algorithm which can detect and classify the different types of roads based on their features. This algorithm can identify train tracks and warn the user to maintain caution while traversing on them.

1.2. Flow diagram

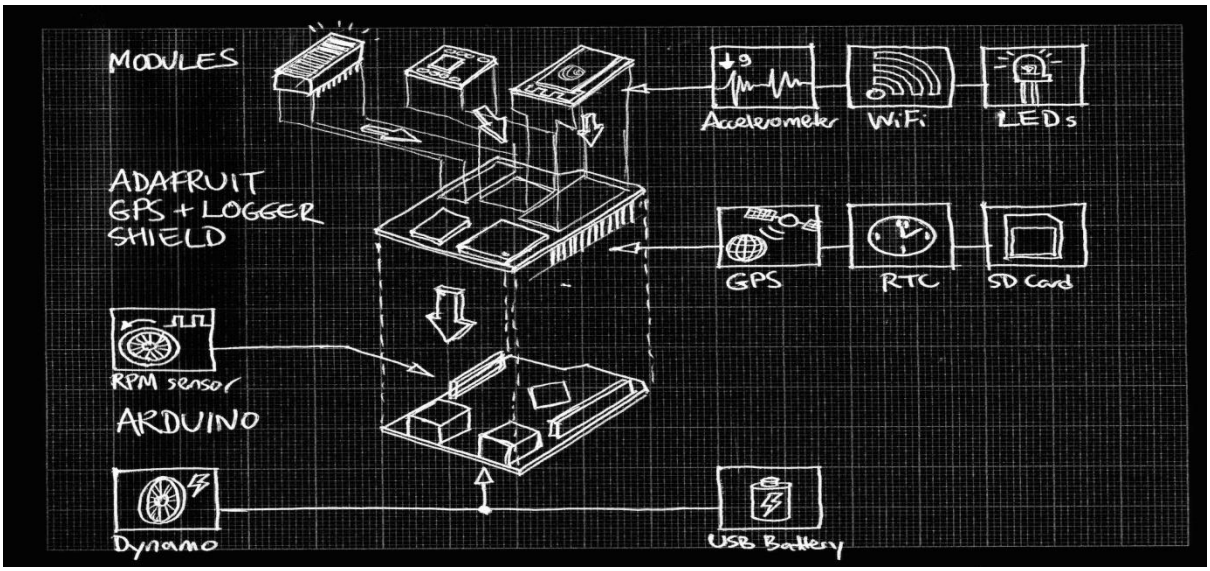

Fig.1.2. Flow diagram of RoadTrack

As depicted in the image above, the mechanism consists of various modules, such as Adafruit GPS module and the logger shield. Each module works independently and the responses of the modules are collected and stitched together by the arduino.

The accelerometer module identifies the bumps or patterns in the roads and sends these values to the cloud using Wi-Fi technology. Indication of certain types of roads is done using LED's.

The Accelerometer and the GPS values are sent to the arduino, which in turn sends the combination of these values to the cloud where the data is analyzed and plotted on the map.

1.3. Hardware used

We use ArduinoMega2560, GSM and GPS Module with respective antenna, Accelerometer sensor (3 axis), jumpers, Servers to achieve best results in RoadTrack.

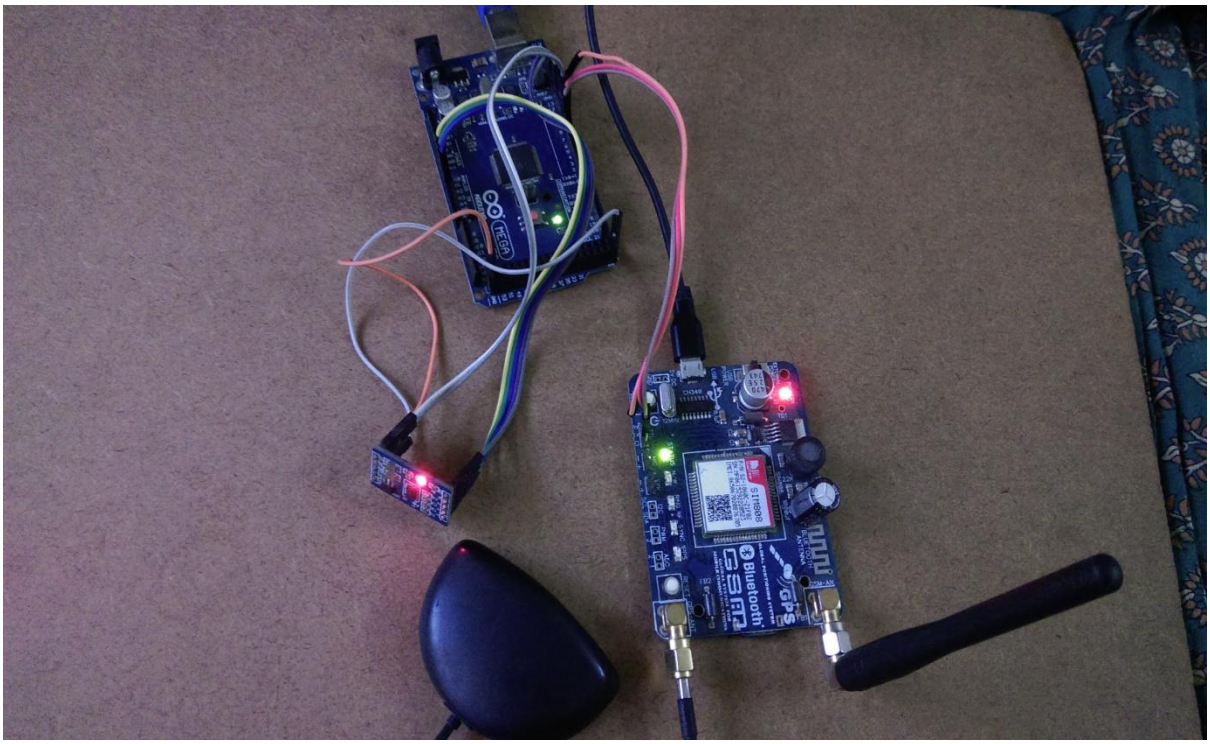

Fig.1.3. Hardware used

The Arduino Mega is considered as the heart of the system, it is responsible for collecting the data from various sensors and organizing it in an understandable form and then uploads it to the cloud.

The GSM module in the system is used to facilitate internet. The GPS module in the system is used to access the Geo-position of the user.

The accelerometer module identifies the bumps or patterns in the roads and sends these values to the cloud using Wi-Fi technology. Indication of certain types of roads is done using LED's.

The jumpers are used as interconnections between different modules. The servers are used to store the recorded data and complex neural network techniques to analyze and classify the obtained values to a user-readable form.

1.4. Implementation steps

The system can be explained in four simple steps:-

1. Measure
2. Upload
3. Classify
4. Plan route

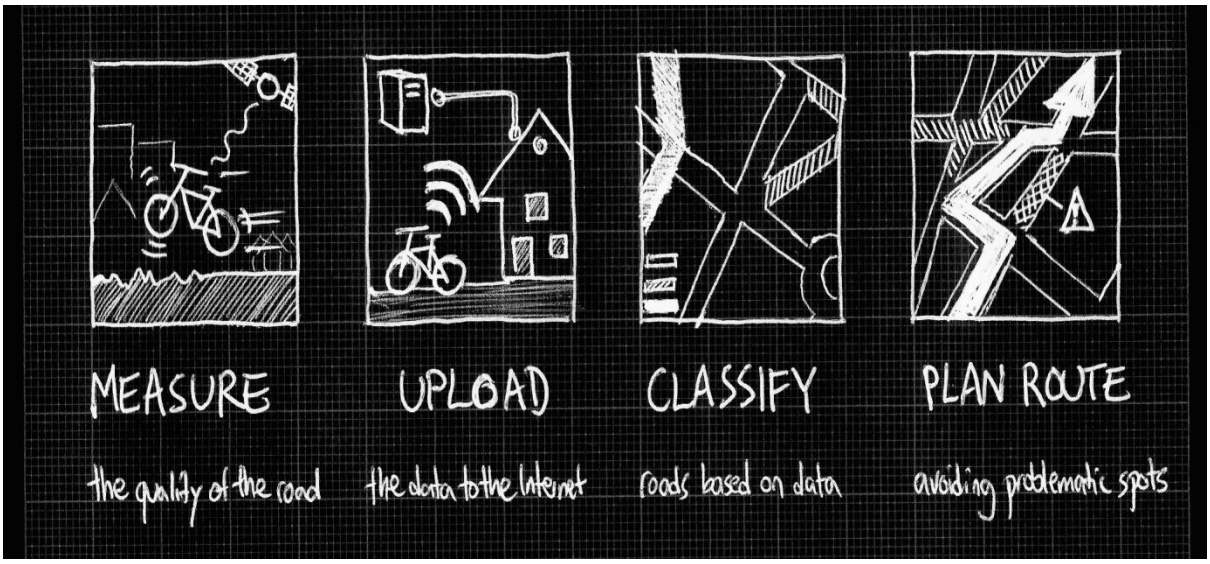

Fig.1.4.Implementation steps

1) **Measure:** The values of the condition of the road are collected on-the-go with the user. The data is classified based on predefined parameters. These values are sent to the cloud for further analysis.

2) **Upload:** The values are pre-processed by the arduino and then are sent reliable cloud storage. The GSM module is used for accessing internet in terms of sending the data.

3) **Classify:** Over the cloud, the data undergoes a series of complex neural networks to classify and identify different classes of roads over the places travelled by the users.

4) **Plan Route:** The analyzed data is used to further suggest the user to follow predicted routes which are smooth/ideal condition for the user to travel. This can help in avoiding unnecessary routes which may cause delays or accidents.

1.5. Expected Result

As the above picture indicates, the collected data is analyzed using complex neural network algorithms and is represented over the map.

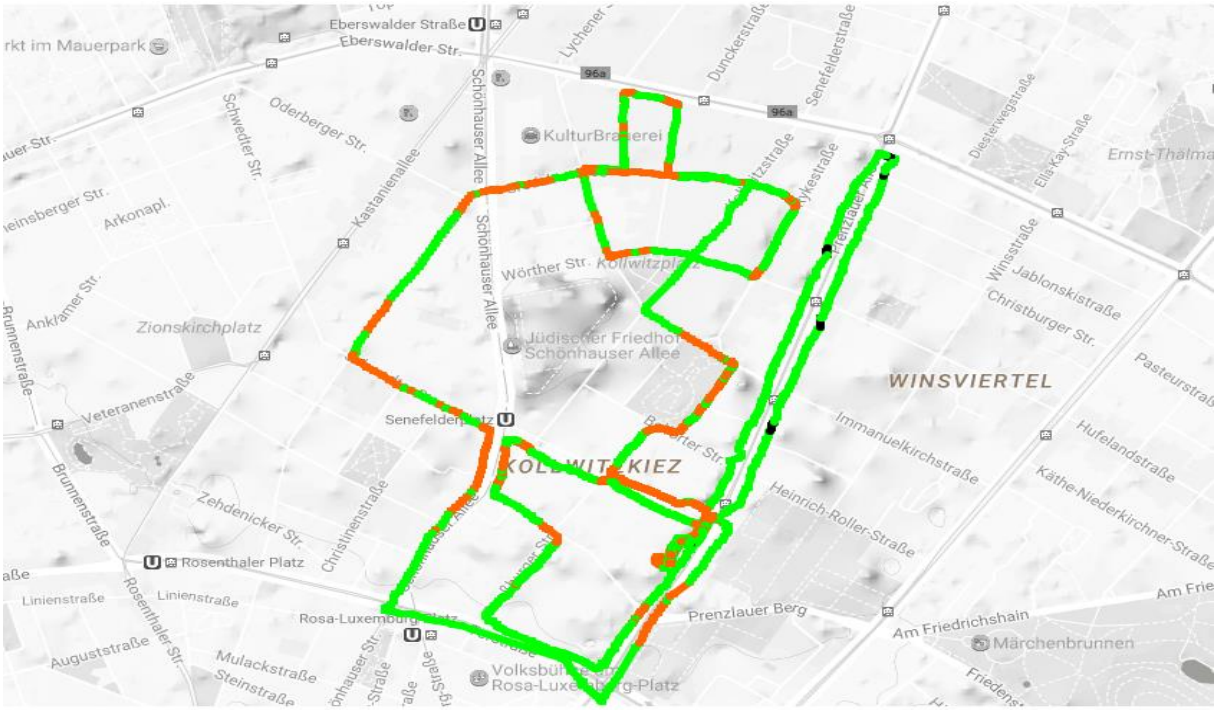

Fig.1.5. Expected Result

All the data is processed and classified in accordance to the pre-defined parameters. All the data that is collected over the period of travel is analyzed in the cloud and is represented over the map in terms of easy-to-understand depiction.

The green path indicates that the roads are smooth and are ideal for the user to travel.

The orange path indicates that the roads are not ideal and may cause discomfort to the user. All the processed data is later used as reference for further received data to compare and update the previously stored data. This helps in creating and maintaining a real-time system that constantly improves with use and gets better in prediction over time.

CHAPTER-2

ARDUINO MEGA

2.1. Introduction

The Mega 2560 is a microcontroller board based on the ATmega2560. It has 54 digital input/output pins (of which 15 can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Mega 2560 board is compatible with most shields designed for the Uno and the former boards Duemilanove or Diecimila.

The Mega 2560 is an update to the Arduino Mega, which it replaces.

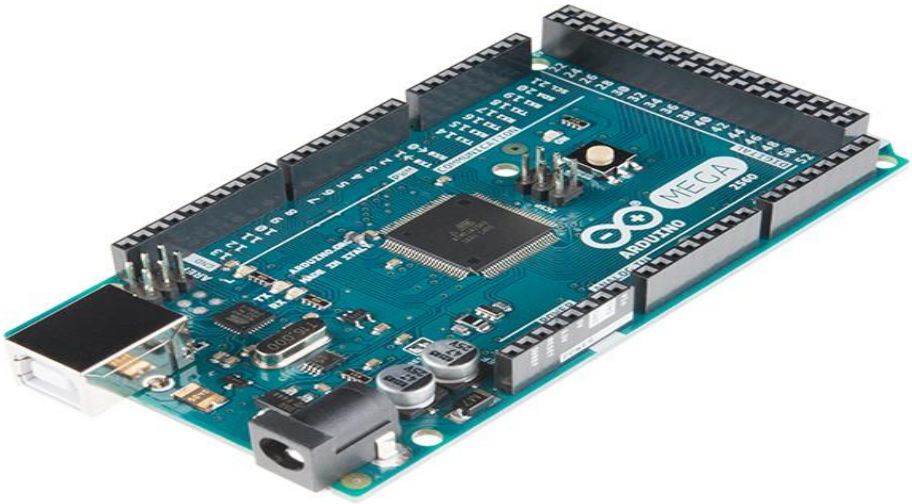

Fig.2.1. Arduino mega 2560

2.2. Technical specifications

Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	54 (of which 15 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz
LED_BUILTIN	13
Length	101.52 mm
Width	53.3 mm
Weight	37 g

2.3. Programming

The Mega 2560 board can be programmed with the Arduino Software (IDE).

The ATmega2560 on the Mega 2560 comes preprogrammed with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol (reference, C header files).

You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header using Arduino ISP or similar; see these instructions for details.

The ATmega16U2 (or 8U2 in the rev1 and rev2 boards) firmware source code is available in the Arduino repository. The ATmega16U2/8U2 is loaded with a DFU bootloader, which can be activated by:

- On Rev1 boards: connecting the solder jumper on the back of the board (near the map of Italy) and then resetting the 8U2.

- On Rev2 or later boards: there is a resistor that pulling the 8U2/16U2 HWB line to ground, making it easier to put into DFU mode. You can then use Atmel's FLIP software (Windows) or the DFU programmer (Mac OS X and Linux) to load a new firmware. Or you can use the ISP header with an external programmer (overwriting the DFU bootloader).

Warnings:

The Mega 2560 has a resettable poly fuse that protects your computer's USB ports from shorts and over current. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

Power:

The Mega 2560 can be powered via the USB connection or with an external power supply. The power source is selected automatically.

External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the GND and Vin pin headers of the POWER connector.

The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may become unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The power pins are as follows:

- **Vin**: The input voltage to the board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- **5V**: This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 12V), the USB connector (5V), or the VIN pin of the board (7-12V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage your board. We don't advise it.
- **3V3**: A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- **GND**: Ground pins.
- **IOREF**: This pin on the board provides the voltage reference with which the microcontroller operates. A properly configured shield can read the IOREF pin voltage and select the appropriate power source or enable voltage translators on the outputs for working with the 5V or 3.3V.

Memory

The ATmega2560 has 256 KB of flash memory for storing code (of which 8 KB is used for the bootloader), 8 KB of SRAM and 4 KB of EEPROM (which can be read and written with the EEPROM library.

Input and Output

See the mapping between Arduino pins and Atmega2560 ports:

Each of the 54 digital pins on the Mega can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive 20 mA as recommended operating condition and has an internal pull-up resistor (disconnected by default) of 20-50 k ohm. A maximum of 40mA is the value that must not be exceeded to avoid permanent damage to the microcontroller.

In addition, some pins have specialized functions:

- **Serial**: 0 (RX) and 1 (TX); Serial 1: 19 (RX) and 18 (TX); Serial 2: 17 (RX) and 16 (TX); Serial 3: 15 (RX) and 14 (TX). Used to receive (RX) and transmit (TX) TTL serial data. Pins 0 and 1 are also connected to the corresponding pins of the ATmega16U2 USB-to-TTL Serial chip.
- **External Interrupts**: 2 (interrupt 0), 3 (interrupt 1), 18 (interrupt 5), 19 (interrupt 4), 20 (interrupt 3), and 21 (interrupt 2). These pins can be configured to trigger an interrupt on a low level, a rising or falling edge, or a change in level. See the attachInterrupt () function for details.
- **PWM**: 2 to 13 and 44 to 46. Provide 8-bit PWM output with the analogWrite () function.

- **SPI**: 50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS). These pins support SPI communication using the SPI library. The SPI pins are also broken out on the ICSP header, which is physically compatible with the Arduino/Genuino Uno and the old Duemilanove and Diecimila Arduino boards.
- **LED**: 13. There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.
- **TWI**: 20 (SDA) and 21 (SCL). Support TWI communication using the Wire library. Note that these pins are not in the same location as the TWI pins on the old Duemilanove or Diecimila Arduino boards.

See also the mapping Arduino Mega 2560 PIN diagram.

The Mega 2560 has 16 analog inputs, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and analogReference() function. There are a couple of other pins on the board:

- AREF. Reference voltage for the analog inputs. Used with analogReference().
- Reset. Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

Communication:

The Mega 2560 board has a number of facilities for communicating with a computer, another board, or other microcontrollers. The ATmega2560 provides four hardware UARTs for TTL (5V) serial communication. An ATmega16U2 (ATmega 8U2 on the revision 1 and revision 2 boards) on the board channels one of these over USB and provides a virtual com port to software on the computer (Windows machines will need a .inf file, but OSX and Linux machines will recognize the board as a COM port automatically. The Arduino Software (IDE) includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the ATmega8U2/ATmega16U2 chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A SoftwareSerial library allows for serial communication on any of the Mega 2560's digital pins.

The Mega 2560 also supports TWI and SPI communication. The Arduino Software (IDE) includes a Wire library to simplify use of the TWI bus; see the documentation for details. For SPI communication, use the SPI library.

2.4. Pin Configuration

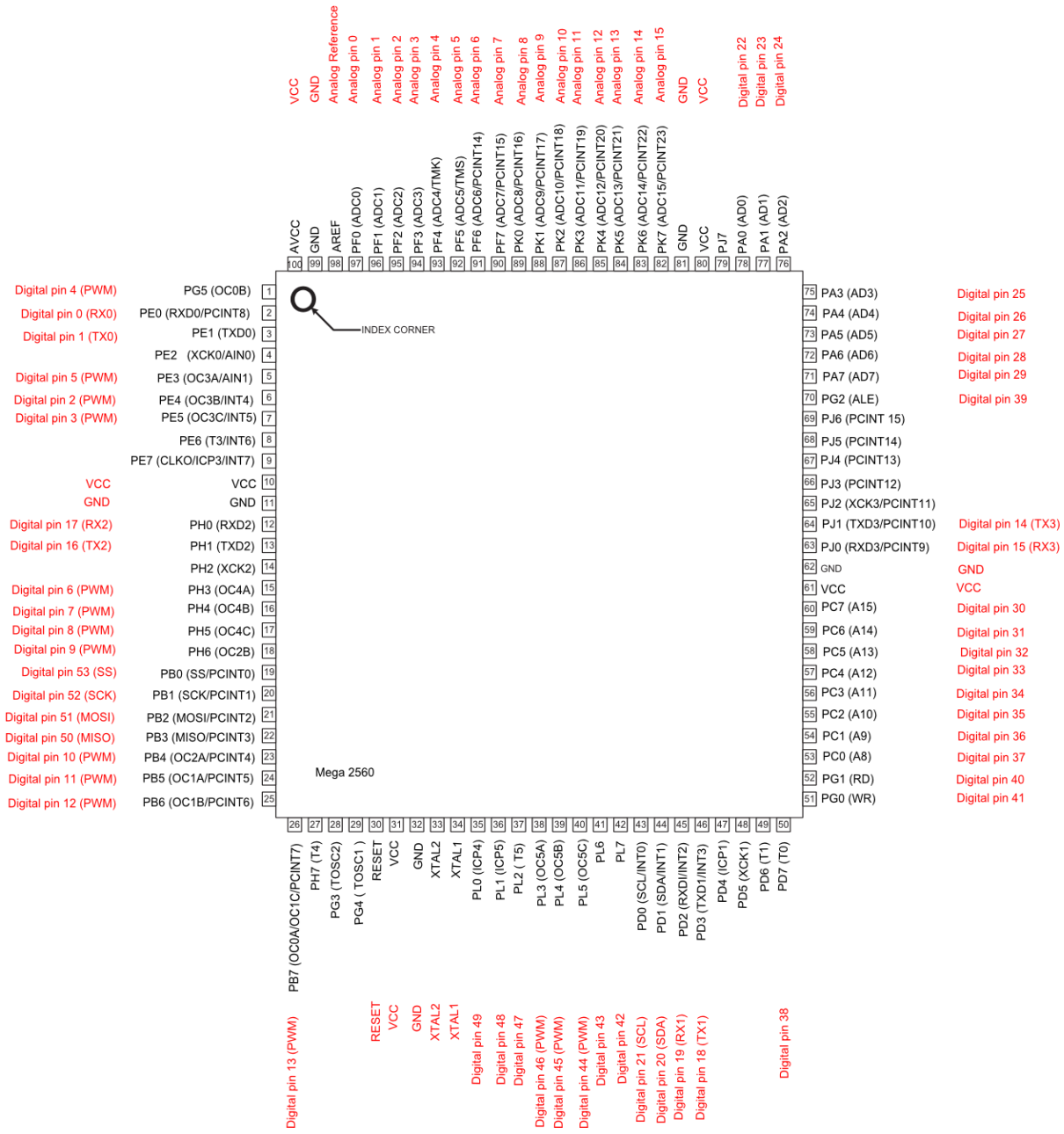

Fig 2.2.Arduino Mega 2560 Pin Configuration

2.5. Physical Characteristics and Shield Compatibility

The maximum length and width of the Mega 2560 PCB are 4 and 2.1 inches respectively, with the USB connector and power jack extending beyond the former dimension. Three screw holes allow the board to be attached to a surface or case. Note that the distance between digital pins 7 and 8 is 160 mil (0.16"), is not an even multiple of the 100 mil spacing of the other pins.

The Mega 2560 is designed to be compatible with most shields designed for the Uno and the older Diecimila or Duemilanove Arduino boards. Digital pins 0 to 13 (and the adjacent AREF and GND pins), analog inputs 0 to 5, the power header, and ICSP header are all in equivalent locations. Furthermore, the main UART (serial port) is located on the same pins (0 and 1), as are external interrupts 0 and 1 (pins 2 and 3 respectively). SPI is available through the ICSP header on both the Mega 2560 and Duemilanove / Diecimila boards. Please note that I2C is not located on the same pins on the Mega 2560 board (20 and 21) as the Duemilanove / Diecimila boards (analog inputs 4 and 5).

Automatic (Software) Reset

Rather than requiring a physical press of the reset button before an upload, the Mega 2560 is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2 is connected to the reset line of the ATmega2560 via a 100 nano farad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The arduino Software (IDE) uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment. This means that the boot loader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload.

This setup has other implications. When the Mega 2560 board is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the boot loader is running on the ATMega2560. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data.

The Mega 2560 board contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESET-EN". You may also be able to disable the auto-reset by connecting a 110 ohm resistor from 5V to the reset line; see this forum thread for details.

2.6. Revisions

The Mega 2560 does not use the FTDI USB-to-serial driver chip used in past designs. Instead, it features the ATmega16U2 (ATmega8U2 in the revision 1 and revision 2 Arduino boards) programmed as a USB-to-serial converter.

Revision 2 of the Mega 2560 board has a resistor pulling the 8U2 HWB line to ground, making it easier to put into DFU mode.

Revision 3 of the Arduino Mega 2560 has the following improved features:

- 1.0 pinout: SDA and SCL pins - near to the AREF pin - and two other new pins placed near to the RESET pin, the IOREF that allow the shields to adapt to the voltage provided from the board. In future, shields will be compatible both with the board that use the AVR, which operate with 5V and with the board that uses ATSAM3X8E, that operate with 3.3V. The second one is a not connected pin that is reserved for future purposes.
- Stronger RESET circuit and ATmega 16U2 replace the 8U2.

CHAPTER-3
ACCELEROMETER SENSOR

3.1. Introduction

One of the most common inertial sensors is the accelerometer, a dynamic sensor capable of a vast range of sensing. Accelerometers are available that can measure acceleration in one, two, or three orthogonal axes. They are typically used in one of three modes:

- As an inertial measurement of velocity and position;
- As a sensor of inclination, tilt, or orientation in 2 or 3 dimensions, as referenced from the acceleration of gravity (1 g = 9.8m/s^2);
- As a vibration or impact (shock) sensor.

An accelerometer is a device that measures proper acceleration; proper acceleration is not the same as coordinate acceleration (rate of change of velocity). For example, an accelerometer at rest on the surface of the Earth will measure acceleration due to Earth's gravity, straight upwards (by definition) of g ≈ 9.81 m/s^2. By contrast, accelerometers in free fall (falling toward the center of the Earth at a rate of about 9.81 m/s^2) will measure zero.

Accelerometers have multiple applications in industry and science. Highly sensitive accelerometers are components of inertial navigation systems for aircraft and missiles. Accelerometers are used to detect and monitor vibration in rotating machinery. Accelerometers are used in tablet computers and digital cameras so that images on screens are always displayed upright. Accelerometers are used in drones for flight stabilization. Coordinated accelerometers can be used to measure differences in proper acceleration, particularly gravity, over their separation in space; i.e., gradient of the gravitational field. This gravity gradiometry is useful because absolute gravity is a weak effect and depends on local density of the Earth which is quite variable.

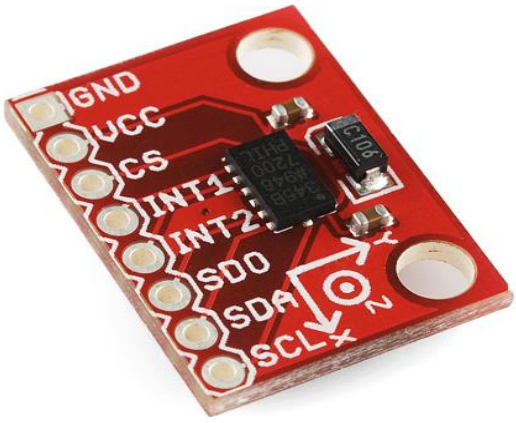

Fig.3.1. Accelerometer (ADXL345)

Single- and multi-axis models of accelerometer are available to detect magnitude and direction of the proper acceleration, as a vector quantity, and can be used to sense orientation (because direction of weight changes), coordinate acceleration, vibration, shock, and falling in a resistive medium (a case where the proper acceleration changes, since it starts at zero, then increases). Micro machined accelerometers are increasingly present in portable electronic devices and video game controllers, to detect the position of the device or provide for game input.

There are considerable advantages to using an analog accelerometer as opposed to an inclinometer such as a liquid tilt sensor – inclinometers tend to output binary information (indicating a state of on or off), thus it is only possible to detect when the tilt has exceeded some thresholding angle.

3.2. Physical principle

An accelerometer measures proper acceleration, which is the acceleration it experiences relative to freefall and is the acceleration felt by people and objects. Put another way, at any point in space time the equivalence principle guarantees the existence of a local inertial frame, and an accelerometer measures the acceleration relative to that frame. Such accelerations are popularly denoted g-force; i.e., in comparison to standard gravity.

An accelerometer at rest relative to the Earth's surface will indicate approximately 1 g upwards, because any point on the Earth's surface is accelerating upwards relative to the local inertial frame (the frame of a freely falling object near the surface). To obtain the acceleration due to motion with respect to the Earth, this "gravity offset" must be subtracted and corrections made for effects caused by the Earth's rotation relative to the inertial frame.

The reason for the appearance of a gravitational offset is Einstein's equivalence principle, which states that the effects of gravity on an object are indistinguishable from acceleration. When held fixed in a gravitational field by, for example, applying a ground reaction force or an equivalent upward thrust, the reference frame for an accelerometer (its own casing) accelerates upwards with respect to a free-falling reference frame. The effects of this acceleration are indistinguishable from any other acceleration experienced by the instrument, so that an accelerometer cannot detect the difference between sitting in a rocket on the launch pad, and being in the same rocket in deep space while it uses its engines to accelerate at 1 g. For similar reasons, an accelerometer will read *zero* during any type of free fall. This includes use in a coasting spaceship in deep space far from any mass, a spaceship orbiting the Earth, an airplane in a parabolic "zero-g" arc, or any free-fall in vacuum. Another example is free-fall at a sufficiently high altitude that atmospheric effects can be neglected.

However this does not include a (non-free) fall in which air resistance produces drag forces that reduce the acceleration, until constant terminal velocity is reached. At terminal velocity the accelerometer will indicate 1 g acceleration upwards. For the same reason a skydiver, upon reaching terminal velocity, does not feel as though he or she were in "free-fall", but rather experiences a feeling similar to being supported (at 1 g) on a "bed" of up rushing air.

Acceleration is quantified in the SI unit metres per second per second (m/s^2), in the cgs unit gal (Gal), or popularly in terms of standard gravity (*g*).

For the practical purpose of finding the acceleration of objects with respect to the Earth, such as for use in an inertial navigation system, knowledge of local gravity is required. This can be obtained either by calibrating the device at rest, or from a known model of gravity at the approximate current position.

3.3. Structure

Conceptually, an accelerometer behaves as a damped mass on a spring. When the accelerometer experiences acceleration, the mass is displaced to the point that the spring is able to accelerate the mass at the same rate as the casing. The displacement is then measured to give the acceleration.

In commercial devices, piezoelectric, piezoresistive and capacitive components are commonly used to convert the mechanical motion into an electrical signal. Piezoelectric accelerometers rely on piezoceramics (e.g. lead zirconate titanate) or single crystals (e.g. quartz, tourmaline). They are unmatched in terms of their upper frequency range, low packaged weight and high temperature range. Piezoresistive accelerometers are preferred in high shock applications. Capacitive accelerometers typically use a silicon micro-machined sensing element. Their performance is superior in the low frequency range and they can be operated in servo mode to achieve high stability and linearity.

Modern accelerometers are often small micro electro-mechanical systems (MEMS), and are indeed the simplest MEMS devices possible, consisting of little more than a cantilever beam with a proof mass (also known as seismic mass). Damping results from the residual gas sealed in the device. As long as the Q-factor is not too low, damping does not result in a lower sensitivity.

Under the influence of external accelerations the proof mass deflects from its neutral position. This deflection is measured in an analog or digital manner. Most commonly, the capacitance between a set of fixed beams and a set of beams attached to the proof mass is measured. This method is simple, reliable, and inexpensive. Integrating piezoresistors in the springs to detect spring deformation, and thus deflection, is a good alternative, although a few more process steps are needed during the fabrication sequence. For very high sensitivities quantum tunneling is also used; this requires a dedicated process making it very expensive. Optical measurement has been demonstrated on laboratory scale.

Another, far less common, type of MEMS-based accelerometer contains a small heater at the bottom of a very small dome, which heats the air inside the dome to cause it to rise. A thermocouple on the dome determines where the heated air reaches the dome and the deflection off the center is a measure of the acceleration applied to the sensor.

Most micromechanical accelerometers operate in-plane, that is, they are designed to be sensitive only to a direction in the plane of the die. By integrating two devices perpendicularly on a single die a two-axis accelerometer can be made. By adding another out-of-plane device three axes can be measured. Such a combination may have much lower misalignment error than three discrete models combined after packaging.

Micromechanical accelerometers are available in a wide variety of measuring ranges, reaching up to thousands of *g*'s. The designer must make a compromise between sensitivity and the maximum acceleration that can be measured.

3.4. Principles of Operation

Most accelerometers are Micro-Electro-Mechanical Sensors (MEMS). The basic principle of operation behind the MEMS accelerometer is the displacement of a small proof mass etched into the silicon surface of the integrated circuit and suspended by small beams. Consistent with Newton's second law of motion (F = ma), as an acceleration is applied to the device, a force develops which displaces the mass. The support beams act as a spring, and the fluid (usually air) trapped inside the IC acts as a damper, resulting in a second order lumped physical system. This is the source of the limited operational bandwidth and non-uniform frequency response of accelerometers. For more information, see reference to Elwenspoek, 1993.

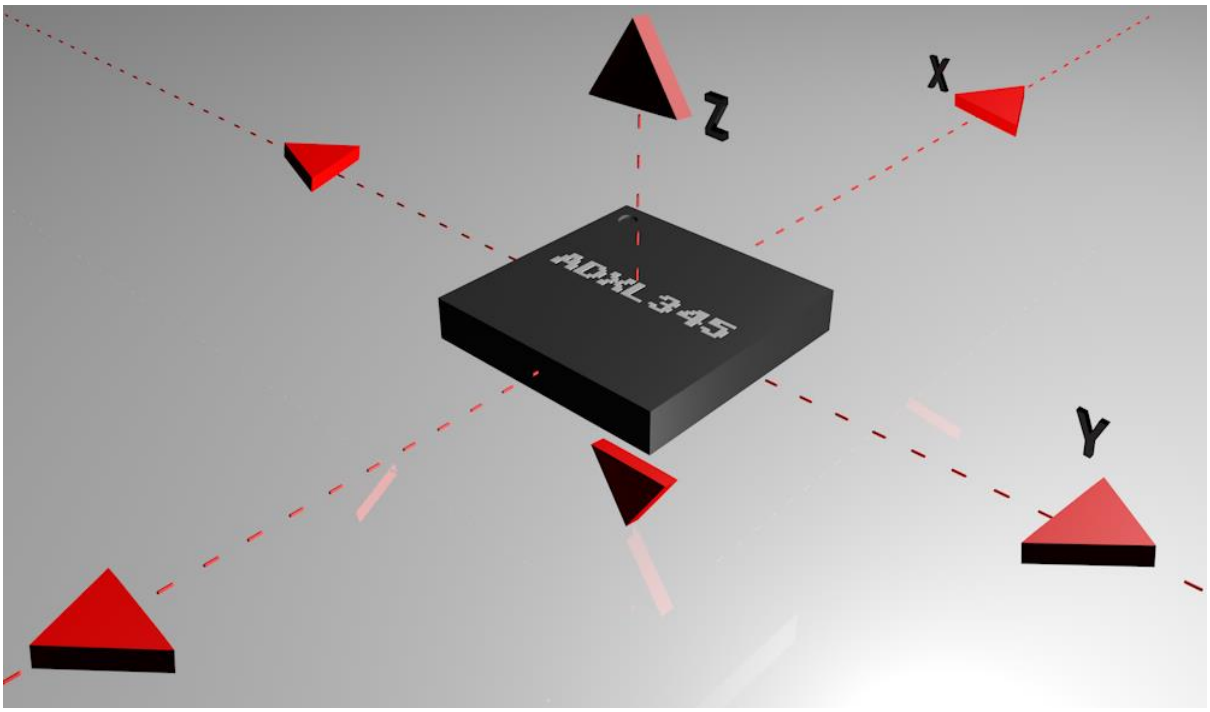

Fig.3.2. Axes of measurement for a 3-axis accelerometer

3.5. Types of Accelerometers

There are several different principles upon which an analog accelerometer can be built. Two very common types utilize capacitive sensing and the piezoelectric effect to sense the displacement of the proof mass proportional to the applied acceleration.

They are:

- Capacitive accelerometer
- Piezoelectric accelerometer

3.5.1. Capacitive:

Accelerometers that implement capacitive sensing output a voltage dependent on the distance between two planar surfaces. One or both of these “plates” are charged with an electrical current. Changing the gap between the plates changes the electrical capacity of the system, which can be measured as a voltage output. This method of sensing is known for its high accuracy and stability. Capacitive accelerometers are also less prone to noise and variation with temperature, typically dissipates less power, and can have larger bandwidths due to internal feedback circuitry. (Elwenspoek 1993).

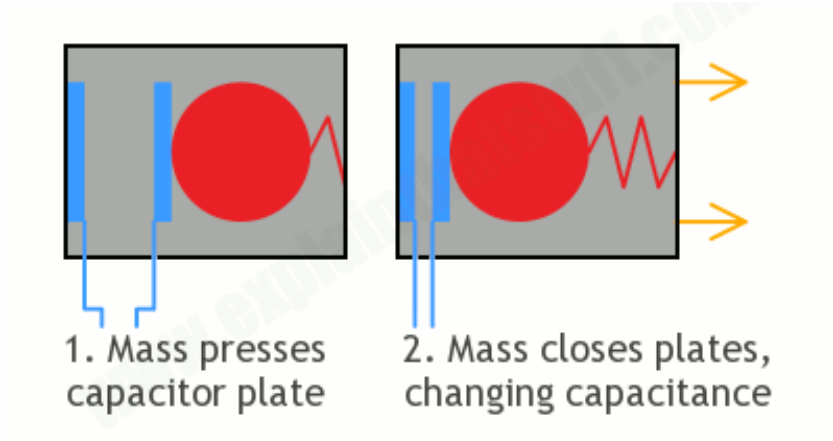

Fig.3.3. Working of Capacitive accelerometer

3.5.2. Piezoelectric:

Piezoelectric sensing of acceleration is natural, as acceleration is directly proportional to force. When certain types of crystal are compressed, charges of opposite polarity accumulate on opposite sides of the crystal. This is known as the piezoelectric effect. In a piezoelectric accelerometer, charge accumulates on the crystal and is translated and amplified into either an output current or voltage.

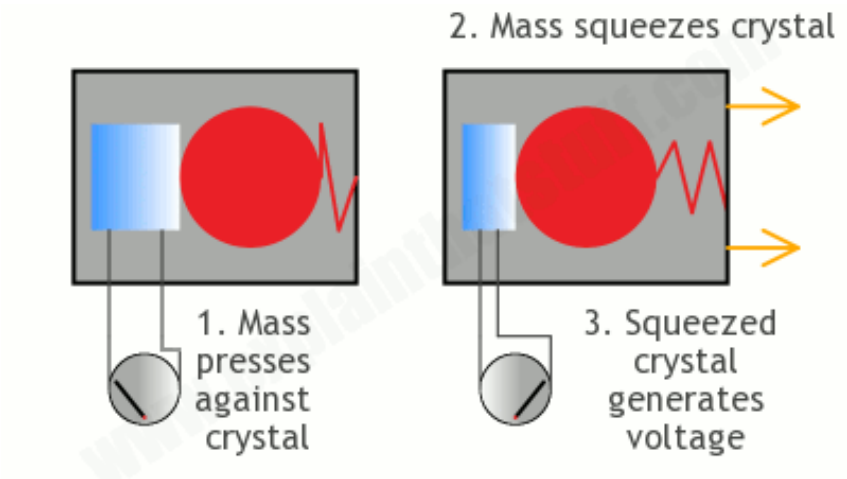

Fig.3.4. Working of Piezoelectric accelerometer

Piezoelectric accelerometers only respond to AC phenomenon such as vibration or shock. They have a wide dynamic range, but can be expensive depending on their quality (Doscher 2005)

Piezo-film based accelerometers are best used to measure AC phenomenon such as vibration or shock, rather than DC phenomenon such as the acceleration of gravity. They are inexpensive, and respond to other phenomenon such as temperature, sound, and pressure (Doscher 2005)

Other Types:

There are many other types of accelerometer, including:

- Null-balance
- Servo force balance
- Strain gauge
- Resonance
- Optical
- Surface acoustic wave (SAW)

3.6. Specifications

A typical accelerometer has the following basic specifications:

- Analog/digital
- Number of axes
- Output range (maximum swing)
- Sensitivity (voltage output per g)
- Dynamic range
- Bandwidth
- Amplitude stability
- Mass

Analog vs. digital:

The most important specification of an accelerometer for a given application is its type of output. Analog accelerometers output a constant variable voltage depending on the amount of acceleration applied. Older digital accelerometers output a variable frequency square wave, a method known as pulse-width modulation. A pulse width modulated accelerometer takes readings at a fixed rate, typically 1000 Hz (though this may be user-configurable based on the IC selected). The value of the acceleration is proportional to the pulse width (or duty cycle) of the PWM signal. Newer digital accelerometers are more likely to output their value using multi-wire digital protocols such as I^2C or SPI.

For use with ADCs commonly used for music interaction systems, analog accelerometers are usually preferred.

Number of axes:

Accelerometers are available that measure in one, two, or three dimensions. The most familiar type of accelerometer measures across two axes. However, three-axis accelerometers are increasingly common and inexpensive.

Output range:

To measure the acceleration of gravity for use as a tilt sensor, an output range of ±1.5 g is sufficient. For use as an impact sensor, one of the most common musical applications, ±5 g or more is desired.

Sensitivity:

An indicator of the amount of change in output signal for a given change in acceleration. A sensitive accelerometer will be more precise and probably more accurate.

Dynamic range:

The range between the smallest acceleration detectable by the accelerometer to the largest before distorting or clipping the output signal.

Bandwidth:

The bandwidth of a sensor is usually measured in Hertz and indicates the limit of the near-unity frequency response of the sensor, or how often a reliable reading can be taken. Humans cannot create body motion much beyond the range of 10-12 Hz. For this reason, a bandwidth of 40-60 Hz is adequate for tilt or human motion sensing.

For vibration measurement or accurate reading of impact forces, bandwidth should be in the range of hundreds of Hertz. It should also be noted that for some older microcontrollers, the bandwidth of an accelerometer may extend beyond the Nyquist frequency of the A/D converters on the MCU, so for higher bandwidth sensing, the digital signal may be aliased. This can be remedied with simple passive low-pass filtering prior to sampling, or by simply choosing a better microcontroller. It is worth noting that the bandwidth may change by the way the accelerometer is mounted. A stiffer mounting (ex: using studs) will help to keep a higher usable frequency range and the opposite (ex: using a magnet) will reduce it.

Amplitude stability:

This is not a specification in itself, but a description of several. Amplitude stability describes a sensor's change in sensitivity depending on its application, for instance over varying temperature or time (see below).

Mass:

The mass of the accelerometer should be significantly smaller than the mass of the system to be monitored so that it does not change the characteristic of the object being tested.

Other specifications include:

- Zero g offset (voltage output at 0 g)
- Noise (sensor minimum resolution)
- Temperature range

- Bias drift with temperature (effect of temperature on voltage output at 0 g)
- Sensitivity drift with temperature (effect of temperature on voltage output per g)
- Power consumption

3.7. Output

An accelerometer output value is a scalar corresponding to the magnitude of the acceleration vector. The most common acceleration, and one that we are constantly exposed to, is the acceleration that is a result of the earth's gravitational pull. This is a common reference value from which all other accelerations are measured (known as g, which is ~9.8m/s^2).

3.7.1. Digital Output:

Accelerometers with PWM output can be used in two different ways. For most accurate results, the PWM signal can be input directly to a microcontroller where the duty cycle is read in firmware and translated into a scaled acceleration value. (Check with the datasheet to obtain the scaling factor and required output impedance.) When a microcontroller with PWM input is not available, or when other means of digitizing the signal are being used, a simple RC reconstruction filter can be used to obtain an analog voltage proportional to the acceleration. At rest (50% duty-cycle) the output voltage will represent no acceleration, higher voltage values (resulting from a higher duty cycle) will represent positive acceleration, and lower values (<50% duty cycle) indicate negative acceleration. These voltages can then be scaled and used as one might the output voltage of an analog output accelerometer. One disadvantage of a digital output is that it takes a little more timing resources of the microcontroller to measure the duty cycle of the PWM signal. Communication protocols could use I2C or SPI.

3.7.2. Analog Output:

When compared to most other industrial sensors, analog accelerometers require little conditioning and the communication is simple by only using an Analog to Digital Converter (ADC) on the microcontroller. Typically, an accelerometer output signal will need an offset, amplification, and filtration. For analog voltage output accelerometers, the signal can be a positive or negative voltage, depending on the direction of the acceleration. Also, the signal is continuous and proportional to the acceleration force. As with any sensor destined for an analog to digital converter, the value must be scaled and/or amplified to maximally span the range of acquisition. Most analog to digital converters used in musical applications acquire signals in the 0-5 V range.

The image at right depicts an amplification and offset circuit, including the on-board operational amplifier in the adxl 105, minimizing the need for additional IC components. The gain applied to the output is set by the ratio R2/R1. The offset is controlled by biasing the voltage with variable resistor R4. Accelerometers output bias will drift according to ambient temperature. The sensors are calibrated for operation at a specific temperature, typically room temperature. However, in most short duration indoor applications the offset is relatively constant and stable, and thus does not need adjustment. If the sensor is intended to be used in multiple environments with differing ambient temperatures, the bias function should be sufficient for analog calibration of the device. If the ambient temperature is subject to drastic

changes over the course of a single usage, the temperature output should be summed into the bias circuit. Smart sensors may even take this into consideration.

The resolution of the data acquired is ultimately determined by the analog to digital converter. It is possible, however, that the noise floor is above the minimum resolution of the converter, reducing the resolution of your system. Assuming that the noise is equally distributed across all frequencies, it is possible to filter the signal to only include frequencies within the range of operation. The filter required depends upon both the type of acquisition as well as the location of the sensor. The bandwidth is primarily influenced by the three different modes of operation of the sensor.

3.8. Uses

The acceleration measurement has a variety of uses. The sensor can be implemented in a system that detects velocity, position, shock, vibration, or the acceleration of gravity to determine orientation (Doscher 2005)

A system consisting of two orthogonal sensors is capable of sensing pitch and roll. This is useful in capturing head movements. A third orthogonal sensor can be added to the network to obtain orientation in three dimensional space. This is appropriate for the detection of pen angles, etc. The sensing capabilities of this network can be furthered to six degrees of spatial measurement freedom by the addition of three orthogonal gyroscopes.

As a shock detector, an accelerometer is looking for changes in acceleration. This jerk is sensed as an over damped vibration.

Verplaetse has outlined the bandwidths associated with various implementations of accelerometers as an input device. These are:

Table.3.1. various implementations of accelerometers

Location	Usage	Frequency	Acceleration
Head	Tilt	0-8 Hz	xx
Hand , Wrist, Finger	Cont.	8-12 Hz	0.04-1.0 g
Hand, Arm, Upper Body	Cont.	0-12 Hz	0.5-9.0 g
Foot, Leg	Cont.	0-12 Hz	0.2-6.6 g

Depending on the sensitivity and dynamic range required, the cost of an accelerometer can grow to thousands of dollars. Nonetheless, highly accurate inexpensive sensors are available.

CHAPTER-4

BREADBOARD AND USB

4.1. Breadboard

A breadboard is a construction base for prototyping of electronics. The solderless breadboard does not require soldering, it is reusable. This makes it easy to use for creating temporary prototypes and experimenting with circuit design. For this reason, solderless breadboards are also extremely popular with students and in technological education. Older breadboard types did not have this property. A strip board (Veroboard) and similar prototyping printed circuit boards, which are used to build semi-permanent soldered prototypes or one-offs, cannot easily be reused. A variety of electronic systems may be prototyped by using breadboards, from small analog and digital circuits to complete central processing units (CPUs).

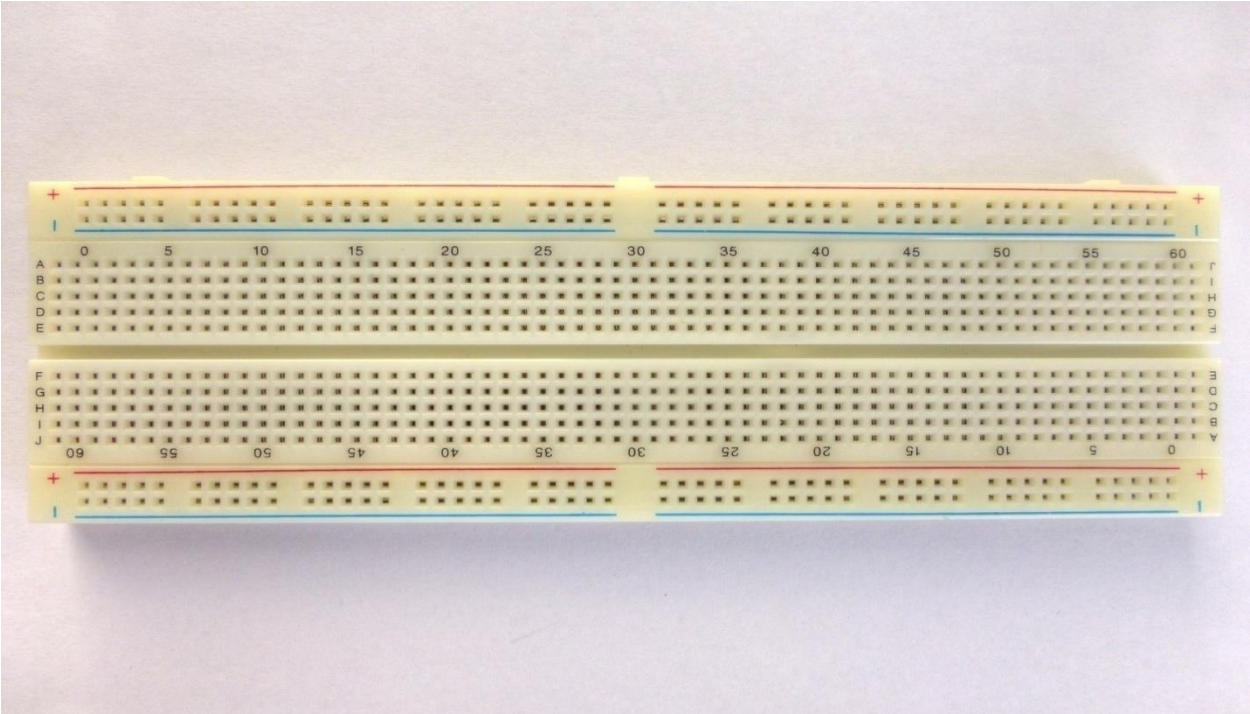

Fig.4.1 – Image of Breadboard

4.2. UNIVERSAL SERIAL BUS (USB)

USB, short for Universal Serial Bus, is an industry standard initially developed in the mid-1990s that defines the cables, connectors and communications protocols used in a bus for connection, communication, and power supply between computers and electronic devices. It is currently developed by the USB Implementers Forum (USB IF).

USB was designed to standardize the connection of computer peripherals (including keyboards, pointing devices, digital cameras, printers, portable media players, disk drives and network adapters) to personal computers, both to communicate and to supply electric power. It has become commonplace on other devices, such as smart phones, PDAs and video game consoles. USB has effectively replaced a variety of earlier interfaces, such as parallel ports, as well as separate power chargers for portable devices.

In general, there are three basic formats of USB connectors: the default or standard format intended for desktop or portable equipment (for example, on USB flash drives), the mini intended for mobile equipment (now deprecated except the Mini-B, which is used on many cameras), and the thinner micro size, for low-profile mobile equipment (most modern mobile phones). Also, there are 5 modes of USB data transfer, in order of increasing bandwidth: Low Speed (from 1.0), Full Speed (from 1.0), High Speed (from 2.0), SuperSpeed (from 3.0), and SuperSpeed+ (from 3.1); modes have differing hardware and cabling requirements. USB devices have some choice of implemented modes, and USB version is not a reliable statement of implemented modes. Modes are identified by their names and icons, and the specifications suggests that plugs and receptacles be colour-coded (SuperSpeed is identified by blue).

Unlike other data buses (e.g., Ethernet, HDMI), USB connections are directed, with both upstream and downstream ports emanating from a single host. This applies to electrical power, with only downstream facing ports providing power; this topology was chosen to easily prevent electrical overloads and damaged equipment. Thus, USB cables have different ends: A and B, with different physical connectors for each.

Therefore, in general, each different format requires four different connectors: a plug and receptacle for each of the A and B ends. USB cables have the plugs, and the corresponding receptacles are on the computers or electronic devices. In common practice, the A end is usually the standard format, and the B side varies over standard, mini, and micro. The mini and micro formats also provide for USB On-The-Go with a hermaphroditic AB receptacle, which accepts either an A or a B plug. On-the-Go allows USB between peers without discarding the directed topology by choosing the host at connection time; it also allows one receptacle to perform double duty in space-constrained applications.

There are cables with A plugs on both ends, which may be valid if the cable includes, for example, a USB host-to-host transfer device with 2 ports, but they could also be non-standard and erroneous and should be used carefully.

The micro format is the most durable from the point of designed insertion lifetime. The standard and mini connectors have a design lifetime of 1,500 insertion-removal cycles,he improved Mini-B connectors increased this to 5,000. The micro connectors were designed with frequent charging of portable devices in mind, so have a design life of 10,000 cycles and also place the flexible contacts, which wear out sooner, on the easily replaced cable, while the more durable rigid contacts are located in the receptacles. Likewise, the springy components of the retention mechanism, parts that provide required gripping force, were also moved into plugs on the cable side.

CHAPTER-5

GSM AND GPS MODULE

Tracking of vehicle is a process in which we track the vehicle location in form of Latitude and Longitude (GPS coordinates). GPS Coordinates are the value of a location. This system is very efficient for outdoor application purpose.

This kind of Vehicle Tracking System Project is widely in tracking Cabs/Taxis, stolen vehicles, school/colleges buses etc.

Fig.5.1. GSM and GPS Module

GPS stands for Global Positioning System and used to detect the Latitude and Longitude of any location on the Earth, with exact UTC time (Universal Time Coordinated). GPS module is the main component in our vehicle tracking system project. This device receives the coordinates from the satellite for each and every second, with time and date.

GPS module sends the data related to tracking position in real time, and it sends so many data in NMEA format (see the screenshot below). NMEA format consist several sentences, in which we only need one sentence. This sentence starts from **$GPGGA** and contains the coordinates, time and other useful information. This GPGGA is referred to Global Positioning System Fix Data**.**

We can extract coordinate from $GPGGA string by counting the commas in the string. Suppose you find $GPGGA string and stores it in an array, then Latitude can be

found after two commas and Longitude can be found after four commas. Now these latitude and longitude can be put in other arrays.

5.1. Global Positioning System (GPS)

The Global Positioning System (GPS) is a space-based radionavigation system owned by the United States government and operated by the United States Air Force. It is a global navigation satellite system that provides geolocation and time information to a GPS receiver anywhere on or near the Earth where there is an unobstructed line of sight to four or more GPS satellites.

The GPS system does not require the user to transmit any data, and it operates independently of any telephonic or internet reception, though these technologies can enhance the usefulness of the GPS positioning information. The GPS system provides critical positioning capabilities to military, civil, and commercial users around the world. The United States government created the system, maintains it, and makes it freely accessible to anyone with a GPS receiver. However, the US government can selectively deny access to the system, as happened to the Indian military in 1999 during the Kargil War.

The GPS project was launched in the United States in 1973 to overcome the limitations of previous navigation systems, integrating ideas from several predecessors, including a number of classified engineering design studies from the 1960s. The U.S. Department of Defense developed the system, which originally used 24 satellites. It became fully operational in 1995. Roger L. Easton of the Naval Research Laboratory, Ivan A. Getting of The Aerospace Corporation, and Bradford Parkinson of the Applied Physics Laboratory are credited with inventing it.

Advances in technology and new demands on the existing system have now led to efforts to modernize the GPS and implement the next generation of GPS Block IIIA satellites and Next Generation Operational Control System (OCX). Announcements from Vice President Al Gore and the White House in 1998 initiated these changes. In 2000, the U.S. Congress authorized the modernization effort, GPS III.

In addition to GPS, other systems are in use or under development, mainly because of a potential denial of access and potential monitoring by the US government. The Russian Global Navigation Satellite System (GLONASS) was developed contemporaneously with GPS, but suffered from incomplete coverage of the globe until the mid-2000s. GLONASS can be added to GPS devices, making more satellites available and enabling positions to be fixed more quickly and accurately, to within two meters. There are also the European Union Galileo positioning system and China's BeiDou Navigation Satellite System.

5.1.1. Fundamentals:

The GPS concept is based on time and the known position of specialized satellites. The satellites carry very stable atomic clocks that are synchronized with one another and to ground clocks. Any drift from true time maintained on the ground is corrected daily. Likewise, the satellite locations are known with great precision. GPS receivers have clocks as well; however, they are usually not synchronized with true time, and are less stable. GPS satellites continuously transmit their current time and position.

A GPS receiver monitors multiple satellites and solves equations to determine the precise position of the receiver and its deviation from true time. At a minimum, four satellites

must be in view of the receiver for it to compute four unknown quantities (three position coordinates and clock deviation from satellite time).

Each GPS satellite continually broadcasts a signal (carrier wave with modulation) that includes:

- A pseudorandom code (sequence of ones and zeros) that is known to the receiver. By time-aligning a receiver-generated version and the receiver-measured version of the code, the time of arrival (TOA) of a defined point in the code sequence, called an epoch, can be found in the receiver clock time scale
- A message that includes the time of transmission (TOT) of the code epoch (in GPS system time scale) and the satellite position at that time

Conceptually, the receiver measures the TOAs (according to its own clock) of four satellite signals. From the TOAs and the TOTs, the receiver forms four time of flight (TOF) values, which are (given the speed of light) approximately equivalent to receiver-satellite range differences. The receiver then computes its three-dimensional position and clock deviation from the four TOFs.

In practice the receiver position (in three dimensional Cartesian coordinates with origin at the Earth's center) and the offset of the receiver clock relative to the GPS time are computed simultaneously, using the navigation equations to process the TOFs.

The receiver's Earth-centered solution location is usually converted to latitude, longitude and height relative to an ellipsoidal Earth model. The height may then be further converted to height relative to the geoid (e.g., EGM96) (essentially, mean sea level). These coordinates may be displayed, e.g., on a moving map display, and/or recorded and/or used by some other system (e.g., a vehicle guidance system).

User-satellite geometry:

Although usually not formed explicitly in the receiver processing, the conceptual time differences of arrival (TDOAs) define the measurement geometry. Each TDOA corresponds to a hyperboloid of revolution (see Multilateration). The line connecting the two satellites involved (and its extensions) forms the axis of the hyperboloid. The receiver is located at the point where three hyperboloids intersect.

It is sometimes incorrectly said that the user location is at the intersection of three spheres. While simpler to visualize, this is only the case if the receiver has a clock synchronized with the satellite clocks (i.e., the receiver measures true ranges to the satellites rather than range differences). There is significant performance benefits to the user carrying a clock synchronized with the satellites. Foremost is that only three satellites are needed to compute a position solution. If this were part of the GPS system concept so that all users needed to carry a synchronized clock, then a smaller number of satellites could be deployed. However, the cost and complexity of the user equipment would increase significantly.

Receiver in continuous operation:

The description above is representative of a receiver start-up situation. Most receivers have a track algorithm, sometimes called a tracker that combines sets of satellite measurements collected at different times—in effect, taking advantage of the fact that successive receiver positions are usually close to each other. After a set of measurements are

processed, the tracker predicts the receiver location corresponding to the next set of satellite measurements. When the new measurements are collected, the receiver uses a weighting scheme to combine the new measurements with the tracker prediction. In general, a tracker can

(a) Improve receiver position and time accuracy

(b) Reject bad measurements, and

(c) Estimate receiver speed and direction.

The disadvantage of a tracker is that changes in speed or direction can only be computed with a delay, and that derived direction becomes inaccurate when the distance traveled between two position measurements drops below or near the random error of position measurement. GPS units can use measurements of the Doppler shift of the signals received to compute velocity accurately. More advanced navigation systems use additional sensors like a compass or an inertial navigation system to complement GPS.

5.1.2. Non-navigation Applications:

In typical GPS operation as a navigator, four or more satellites must be visible to obtain an accurate result. The solution of the navigation equations gives the position of the receiver along with the difference between the time kept by the receiver's on-board clock and the true time-of-day, thereby eliminating the need for a more precise and possibly impractical receiver based clock. Applications for GPS such as time transfer, traffic signal timing, and synchronization of cell phone base stations, make use of this cheap and highly accurate timing. Some GPS applications use this time for display, or, other than for the basic position calculations, do not use it at all.

Although four satellites are required for normal operation, fewer apply in special cases. If one variable is already known, a receiver can determine its position using only three satellites. For example, a ship or aircraft may have known elevation. Some GPS receivers may use additional clues or assumptions such as reusing the last known altitude, dead reckoning, inertial navigation, or including information from the vehicle computer, to give a (possibly degraded) position when fewer than four satellites are visible.

Communication:

The navigational signals transmitted by GPS satellites encode a variety of information including satellite positions, the state of the internal clocks, and the health of the network. These signals are transmitted on two separate carrier frequencies that are common to all satellites in the network. Two different encodings are used: a public encoding that enables lower resolution navigation, and an encrypted encoding used by the U.S. military.

5.1.3. Message format:

Table.5.1. GPS message format

Subframes	Description
1	Satellite clock, GPS time relationship
2–3	Ephemeris (precise satellite orbit)
4–5	Almanac component (satellite network synopsis, error correction)

Each GPS satellite continuously broadcasts a navigation message on L1 (C/A and P/Y) and L2 (P/Y) frequencies at a rate of 50 bits per second (see bit rate). Each complete message takes 750 seconds (12 1/2 minutes) to complete. The message structure has a basic format of a 1500-bit-long frame made up of five Subframes, each sub frame being 300 bits (6 seconds) long. Subframes 4 and 5 are sub commutated 25 times each, so that a complete data message requires the transmission of 25 full frames. Each sub frame consists of ten words, each 30 bits long. Thus, with 300 bits in a subframe times 5 Subframes in a frame times 25 frames in a message, each message is 37,500 bits long. At a transmission rate of 50-bit/s, this gives 750 seconds to transmit an entire almanac message (GPS). Each 30-second frame begins precisely on the minute or half-minute as indicated by the atomic clock on each satellite.

The first subframe of each frame encodes the week number and the time within the week, as well as the data about the health of the satellite. The second and the third subframes contain the ephemeris– the precise orbit for the satellite. The fourth and fifth subframes contain the almanac, which contains coarse orbit and status information for up to 32 satellites in the constellation as well as data related to error correction. Thus, to obtain an accurate satellite location from this transmitted message, the receiver must demodulate the message from each satellite it includes in its solution for 18 to 30 seconds. To collect all transmitted almanacs, the receiver must demodulate the message for 732 to 750 seconds or 12 1/2 minutes.

All satellites broadcast at the same frequencies, encoding signals using unique code division multiple access (CDMA) so receivers can distinguish individual satellites from each other. The system uses two distinct CDMA encoding types: the coarse/acquisition (C/A) code, which is accessible by the general public, and the precise (P(Y)) code, which is encrypted so that only the U.S. military and other NATO nations who have been given access to the encryption code can access it.

The ephemeris is updated every 2 hours and is generally valid for 4 hours, with provisions for updates every 6 hours or longer in non-nominal conditions. The almanac is updated typically every 24 hours. Additionally, data for a few weeks following is uploaded in case of transmission updates that delay data upload.

5.1.4. Satellite frequencies:

Table.5.2. GPS frequency overview

Band	Frequency	Description
L1	1575.42 MHz	Coarse-acquisition (C/A) and encrypted precision (P(Y)) codes, plus the L1 civilian (L1C) and military (M) codes on future Block III satellites.
L2	1227.60 MHz	P(Y) code, plus the L2C and military codes on the Block IIR-M and newer satellites.
L3	1381.05 MHz	Used for nuclear detonation (NUDET) detection.
L4	1379.913 MHz	Being studied for additional ionospheric correction.
L5	1176.45 MHz	Proposed for use as a civilian safety-of-life (SoL) signal.

All satellites broadcast at the same two frequencies, 1.57542 GHz (L1 signal) and 1.2276 GHz (L2 signal). The satellite network uses a CDMA spread-spectrum technique where the low-bit rate message data is encoded with a high-rate pseudo-random (PRN) sequence that is different for each satellite. The receiver must be aware of the PRN codes for each satellite to reconstruct the actual message data. The C/A code, for civilian use, transmits data at 1.023 million chips per second, whereas the P code, for U.S. military use, transmits at 10.23 million chips per second. The actual internal reference of the satellites is 10.22999999543 MHz to compensate for relativistic effects that make observers on the Earth perceive a different time reference with respect to the transmitters in orbit. The L1 carrier is modulated by both the C/A and P codes, while the L2 carrier is only modulated by the P code. The P code can be encrypted as a so-called P(Y) code that is only available to military equipment with a proper decryption key. Both the C/A and P(Y) codes impart the precise time-of-day to the user.

The L3 signal at a frequency of 1.38105 GHz is used to transmit data from the satellites to ground stations. This data is used by the United States Nuclear Detonation (NUDET) Detection System (USNDS) to detect, locate, and report nuclear detonations (NUDETs) in the Earth's atmosphere and near space. One usage is the enforcement of nuclear test ban treaties.

The L4 band at 1.379913 GHz is being studied for additional ionospheric correction.

The L5 frequency band at 1.17645 GHz was added in the process of GPS modernization. This frequency falls into an internationally protected range for aeronautical

navigation, promising little or no interference under all circumstances. The first Block IIF satellite that provides this signal was launched in 2010. The L5 consists of two carrier components that are in phase quadrature with each other. Each carrier component is bi-phase shift key (BPSK) modulated by a separate bit train. "L5, the third civil GPS signal, will eventually support safety-of-life applications for aviation and provide improved availability and accuracy."

A conditional waiver has recently (2011-01-26) been granted to LightSquared to operate a terrestrial broadband service near the L1 band. Although LightSquared had applied for a license to operate in the 1525 to 1559 band as early as 2003 and it was put out for public comment, the FCC asked LightSquared to form a study group with the GPS community to test GPS receivers and identify issue that might arise due to the larger signal power from the LightSquared terrestrial network. The GPS community had not objected to the LightSquared (formerly MSV and SkyTerra) applications until November 2010, when LightSquared applied for a modification to its Ancillary Terrestrial Component (ATC) authorization. This filing (SAT-MOD-20101118-00239) amounted to a request to run several orders of magnitude more power in the same frequency band for terrestrial base stations, essentially repurposing what was supposed to be a "quiet neighborhood" for signals from space as the equivalent of a cellular network. Testing in the first half of 2011 has demonstrated that the impact of the lower 10 MHz of spectrum is minimal to GPS devices (less than 1% of the total GPS devices are affected). The upper 10 MHz intended for use by LightSquared may have some impact on GPS devices. There is some concern that this may seriously degrade the GPS signal for many consumer uses. Aviation Week magazine reports that the latest testing (June 2011) confirms "significant jamming" of GPS by LightSquared's system.

5.1.5. Demodulation and decoding:

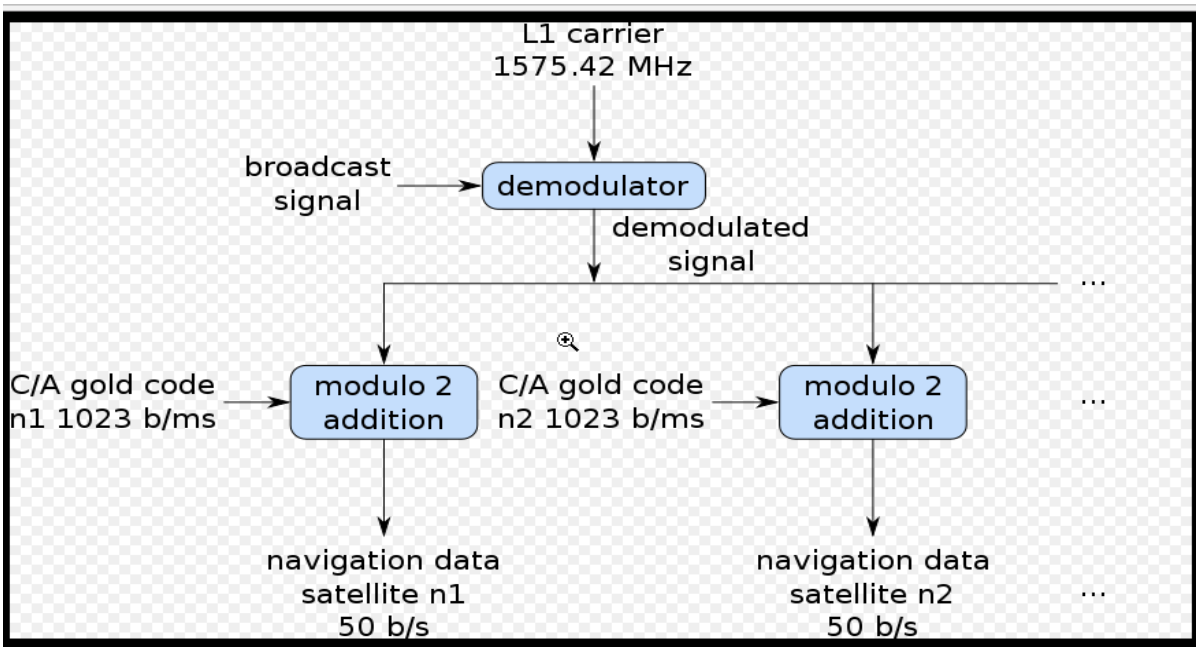

Fig.5.2. Modulation and Demodulation in GPS

Because all of the satellite signals are modulated onto the same L1 carrier frequency, the signals must be separated after demodulation. This is done by assigning each satellite a unique binary sequence known as a Gold code. The signals are decoded after demodulation using addition of the Gold codes corresponding to the satellites monitored by the receiver.

If the almanac information has previously been acquired, the receiver picks the satellites to listen for by their PRNs, unique numbers in the range 1 through 32. If the almanac information is not in memory, the receiver enters a search mode until a lock is obtained on one of the satellites. To obtain a lock, it is necessary that there be an unobstructed line of sight from the receiver to the satellite. The receiver can then acquire the almanac and determine the satellites it should listen for. As it detects each satellite's signal, it identifies it by its distinct C/A code pattern. There can be a delay of up to 30 seconds before the first estimate of position because of the need to read the ephemeris data.

Processing of the navigation message enables the determination of the time of transmission and the satellite position at this time. For more information see Demodulation and Decoding, Advanced.

5.2. General Packet Radio Service (GPRS)

General Packet Radio Service (GPRS) is a packet oriented mobile data service on the 2G and 3G cellular communication system's global system for mobile communications (GSM). GPRS was originally standardized by European Telecommunications Standards Institute (ETSI) in response to the earlier CDPD and i-mode packet-switched cellular technologies. It is now maintained by the 3rd Generation Partnership Project (3GPP).

GPRS usage is typically charged based on volume of data transferred, contrasting with circuit switched data, which is usually billed per minute of connection time. Sometimes billing time is broken down to every third of a minute. Usage above the bundle cap is charged per megabyte, speed limited, or disallowed.

GPRS is a best-effort service, implying variable throughput and latency that depend on the number of other users sharing the service concurrently, as opposed to circuit switching, where a certain quality of service (QoS) is guaranteed during the connection. In 2G systems, GPRS provides data rates of 56–114 kbit/second. 2G cellular technology combined with GPRS is sometimes described as 2.5G, that is, a technology between the second (2G) and third (3G) generations of mobile telephony. It provides moderate-speed data transfer, by using unused time division multiple access (TDMA) channels in, for example, the GSM system. GPRS is integrated into GSM Release 97 and newer releases.

5.2.1. Technical Overview:

The GPRS core network allows 2G, 3G and WCDMA mobile networks to transmit IP packets to external networks such as the Internet. The GPRS system is an integrated part of the GSM network switching subsystem.

Services offered:

GPRS extends the GSM Packet circuit switched data capabilities and makes the following services possible:

- SMS messaging and broadcasting
- "Always on" internet access
- Multimedia messaging service (MMS)
- Push-to-talk over cellular (PoC)
- Instant messaging and presence-wireless village
- Internet applications for smart devices through wireless application protocol (WAP)
- **Point-to-point (P2P) service:** inter-networking with the Internet (IP)
- **Point-to-multipoint (P2M) service:** point-to-multipoint multicast and point-to-multipoint group calls

If SMS over GPRS is used, an SMS transmission speed of about 30 SMS messages per minute may be achieved. This is much faster than using the ordinary SMS over GSM, whose SMS transmission speed is about 6 to 10 SMS messages per minute.

Protocols supported:

GPRS supports the following protocols:

- **Internet Protocol (IP):** In practice, built-in mobile browsers use IPv4 since IPv6 was not yet popular.
- **Point-to-Point Protocol (PPP):** In this mode PPP is often not supported by the mobile phone operator but if the mobile is used as a modem to the connected computer, PPP is used to tunnel IP to the phone. This allows an IP address to be assigned dynamically (IPCP not DHCP) to the mobile equipment.
- **X.25 connections:** This is typically used for applications like wireless payment terminals, although it has been removed from the standard. X.25 can still be supported over PPP, or even over IP, but doing this requires either a network-based router to perform encapsulation or intelligence built into the end-device/terminal; e.g., user equipment (UE).

When TCP/IP is used, each phone can have one or more IP addresses allocated. GPRS will store and forward the IP packets to the phone even during handover. The TCP handles any packet loss (e.g. due to a radio noise induced pause).

5.2.2. Hardware:

Devices supporting GPRS are divided into three classes:

Class A:

Can be connected to GPRS service and GSM service (voice, SMS), using both at the same time. Such devices are known to be available today.

Class B:

Can be connected to GPRS service and GSM service (voice, SMS), but using only one or the other at a given time. During GSM service (voice call or SMS), GPRS service is suspended, and then resumed automatically after the GSM service (voice call or SMS) has concluded. Most GPRS mobile devices are Class B.

Class C:

Are connected to either GPRS service or GSM service (voice, SMS). Must be switched manually between one or the other service.

A true Class A device may be required to transmit on two different frequencies at the same time, and thus will need two radios. To get around this expensive requirement, a GPRS mobile may implement the dual transfer mode (DTM) feature. A DTM-capable mobile may use simultaneous voice and packet data, with the network coordinating to ensure that it is not required to transmit on two different frequencies at the same time. Such mobiles are considered pseudo-Class A, sometimes referred to as "simple class A". Some networks support DTM since 2007.

USB 3G/GPRS modems use a terminal-like interface over USB 1.1, 2.0 and later, data formats V.42bis, and RFC 1144 and some models have connector for external antenna. Modems can be added as cards (for laptops) or external USB devices which are similar in shape and size to a computer mouse, or nowadays more like a pendrive.

5.2.3. Addressing:

A GPRS connection is established by reference to its access point name (APN). The APN defines the services such as wireless application protocol (WAP) access, short message service (SMS), multimedia messaging service (MMS), and for Internet communication services such as email and World Wide Web access.

In order to set up a GPRS connection for a wireless modem, a user must specify an APN, optionally a user name and password, and very rarely an IP address, provided by the network operator.

5.2.4. Coding schemes and speeds:

The upload and download speeds that can be achieved in GPRS depend on a number of factors such as:

- The number of BTS TDMA time slots assigned by the operator.
- The channel encoding is used.
- The maximum capability of the mobile device expressed as a GPRS multislot class.

Multiple access schemes:

The multiple access methods used in GSM with GPRS are based on frequency division duplex (FDD) and TDMA. During a session, a user is assigned to one pair of up-link and down-link frequency channels. This is combined with time domain statistical multiplexing which makes it possible for several users to share the same frequency channel. The packets have constant length, corresponding to a GSM time slot. The down-link uses first-come first-served packet scheduling, while the up-link uses a scheme very similar to reservation ALOHA (R-ALOHA). This means that slotted ALOHA (S-ALOHA) is used for reservation inquiries during a contention phase, and then the actual data is transferred using dynamic TDMA with first-come first-served.

Channel Encoding:

The channel encoding process in GPRS consists of two steps: first, a cyclic code is used to add parity bits, which are also referred to as the Block Check Sequence, followed by coding with a possibly punctured convolutional code. The Coding Schemes CS-1 to CS-4 specifies the number of parity bits generated by the cyclic code and the puncturing rate of the convolutional code. In Coding Schemes CS-1 through CS-3, the convolutional code is of rate 1/2, i.e. each input bit is converted into two coded bits. In Coding Schemes CS-2 and CS-3, the output of the convolutional code is punctured to achieve the desired code rate. In Coding Scheme CS-4, no convolutional coding is applied. The following table summarizes the options.

Table.5.3. GPRS coding schemes

GPRS Coding scheme	Bitrate including RLC/MAC overhead (kbit/s/slot)	Bitrate excluding RLC/MAC overhead (kbit/s/slot)	Modulation	Code rate
CS-1	9.20	8.00	GMSK	½
CS-2	13.55	12.00	GMSK	≈2/3
CS-3	15.75	14.40	GMSK	≈3/4
CS-4	21.55	20.00	GMSK	1

a. **Jump up^**: This is rate at which the RLC/MAC layer protocol data unit (PDU) (called a radio block) is transmitted. As shown in TS 44.060 section 10.0a.1, a radio block consists of MAC header, RLC header, RLC data unit and spare bits. The RLC data unit represents the payload, the rest is overhead. The radio block is coded by the convolutional code specified for a particular Coding Scheme, which yields the same PHY layer data rate for all Coding Schemes.
b. **Jump up^**: Cited in various sources, e.g. in TS 45.001 table 1. is the bit rate including the RLC/MAC headers, but excluding the uplink state flag (USF), which is part of the MAC header, yielding a bit rate that is 0.15 kbit/s lower.
c. **Jump up^**: The net bit rate here is the rate at which the RLC/MAC layer payload (the RLC data unit) is transmitted. As such, this bit rate excludes the header overhead from the RLC/MAC layers.

The least robust, but fastest, coding scheme (CS-4) is available near a base transceiver station (BTS), while the most robust coding scheme (CS-1) is used when the mobile station (MS) is further away from a BTS.

Using the CS-4 it is possible to achieve a user speed of 20.0 kbit/s per time slot. However, using this scheme the cell coverage is 25% of normal. CS-1 can achieve a user speed of only 8.0 kbit/s per time slot, but has 98% of normal coverage. Newer network equipment can adapt the transfer speed automatically depending on the mobile location.

In addition to GPRS, there are two other GSM technologies which deliver data services: circuit-switched data (CSD) and high-speed circuit-switched data (HSCSD). In contrast to the shared nature of GPRS, these instead establish a dedicated circuit (usually

billed per minute). Some applications such as video calling may prefer HSCSD, especially when there is a continuous flow of data between the endpoints.

The following table summarizes some possible configurations of GPRS and circuit switched data services.

Table.5.4. Configurations of GPRS

Technology	Download (kbit/s)	Upload (kbit/s)	TDMA timeslots allocated (DL+UL)
CSD	9.6	9.6	1+1
HSCSD	28.8	14.4	2+1
HSCSD	43.2	14.4	3+1
GPRS	85.6	21.4 (Class 8 & 10 and CS-4)	4+1
GPRS	64.2	42.8 (Class 10 and CS-4)	3+2
EGPRS (EDGE)	236.8	59.2 (Class 8, 10 and MCS-9)	4+1
EGPRS (EDGE)	177.6	118.4 (Class 10 and MCS-9)	3+2

5.2.5. Multislot Class:

The multislot class determines the speed of data transfer available in the Uplink and Downlink directions. It is a value between 1 and 45 which the network uses to allocate radio channels in the uplink and downlink direction. Multislot class with values greater than 31 are referred to as high multislot classes.

A multislot allocation is represented as, for example, 5+2. The first number is the number of downlink timeslots and the second is the number of uplink timeslots allocated for use by the mobile station. A commonly used value is class 10 for many GPRS/EGPRS mobiles which uses a maximum of 4 timeslots in downlink direction and 2 timeslots in uplink direction. However simultaneously a maximum number of 5 simultaneous timeslots can be used in both uplink and downlink. The network will automatically configure for either 3+2 or 4+1 operation depending on the nature of data transfer.

Some high end mobiles, usually also supporting UMTS, also support GPRS/EDGE multislot class 32. According to 3GPP TS 45.002 (Release 12), Table B.1, mobile stations of this class support 5 timeslots in downlink and 3 timeslots in uplink with a maximum number of 6 simultaneously used timeslots. If data traffic is concentrated in downlink direction the network will configure the connection for 5+1 operation. When more

data is transferred in the uplink the network can at any time change the constellation to 4+2 or 3+3. Under the best reception conditions, i.e. when the best EDGE modulation and coding scheme can be used, 5 timeslots can carry a bandwidth of 5*59.2 kbit/s = 296 kbit/s. In uplink direction, 3 timeslots can carry a bandwidth of 3*59.2 kbit/s = 177.6 kbit/s.

Table.5.5. Multislot Classes for GPRS/EGPRS

Multislot Class	Downlink TS	Uplink TS	Active TS
1	1	1	2
2	2	1	3
3	2	2	3
4	3	1	4
5	2	2	4
6	3	2	4
7	3	3	4
8	4	1	5
9	3	2	5
10	4	2	5
11	4	3	5
12	4	4	5
30	5	1	6

31	5	2	6
32	5	3	6
33	5	4	6
34	5	5	6

Attributes of a Multislot Class

Each multislot class identifies the following:

- The maximum number of Timeslots that can be allocated on uplink
- The maximum number of Timeslots that can be allocated on downlink
- The total number of timeslots which can be allocated by the network to the mobile
- The time needed for the MS to perform adjacent cell signal level measurement and get ready to transmit
- The time needed for the MS to get ready to transmit
- The time needed for the MS to perform adjacent cell signal level measurement and get ready to receive
- The time needed for the MS to get ready to receive.

The different multislot class specification is detailed in the Annex B of the 3GPP Technical Specification 45.002 (Multiplexing and multiple access on the radio path)

5.3. Working Principle

The arduino mega is programmed such that it communicates with GPS+GSM module by AT commands through serial communication. When the power is supplied first it gives tough time for all the devices to turn-on and then by using AT commands the individual modules are turned-on then the GSM module for internet connectivity and the values from GPS are read and stored in arduino memory and by the data drop API technology the values are appended to a URL and then by sending a request to the wolfram cloud, it saves the coordinate the values in its data bin. And by using the powerful wolframs mathematica engine we can simulate the coordinate values by writing a code so that we get the target pointed on maps.

In this project, Arduino is used for controlling whole the process with a GPS Receiver and GSM module. GPS Receiver is used for detecting coordinates of the vehicle, GSM module is used for sending the coordinates to user by SMS. And an optional 16x2 LCD is also used for displaying status messages or coordinates. We have used GPS Module SKG13BL and GSM Module SIM900A.

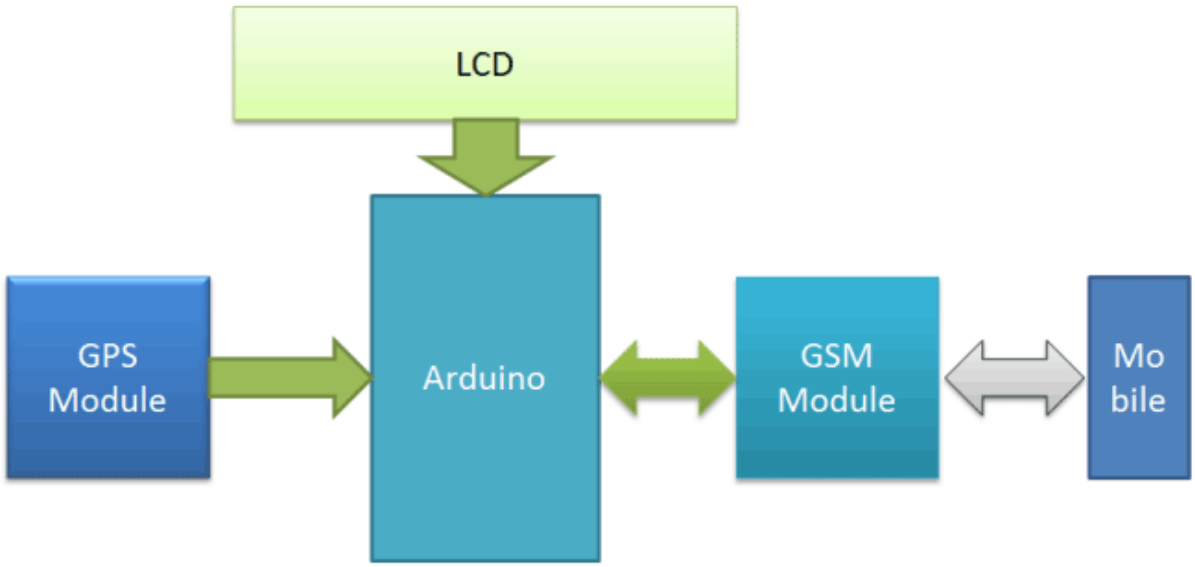

Fig.5.3. Flow diagram of the GSM and GPS Module

When we ready with our hardware after programming, we can install it in our vehicle and power it up. Then we just need to send a SMS, "Track Vehicle", to the system that is placed in our vehicle. We can also use some prefix (#) or suffix (*) like #Track Vehicle*, to properly identify the starting and ending of the string.

Sent message is received by GSM module which is connected to the system and sends message data to Arduino. Arduino reads it and extract main message from the whole message. And then compare it with predefined message in Arduino. If any match occurs then Arduino reads coordinates by extracting $GPGGA String from GPS module data (GPS working explained above) and send it to user by using GSM module. This message contains the coordinates of vehicle location.

CHAPTER-6

SERVERS

In computing, a server is a computer program or a device that provides functionality for other programs or devices, called "clients". This architecture is called the client–server model, and a single overall computation is distributed across multiple processes or devices. Servers can provide various functionalities, often called "services", such as sharing data or resources among multiple clients, or performing computation for a client. A single server can serve multiple clients, and a single client can use multiple servers. A client process may run on the same device or may connect over a network to a server on a different device. Typical servers are database servers, file servers, mail servers, print servers, web servers, game servers, and application servers.

Client–server systems are today most frequently implemented by (and often identified with) the request–response model: a client sends a request to the server, which performs some action and sends a response back to the client, typically with a result or acknowledgement. Designating a computer as "server-class hardware" implies that it is specialized for running servers on it. This often implies that it is more powerful and reliable than standard personal computers, but alternatively, large computing clusters may be composed of many relatively simple, replaceable server components.

6.1. Operation

Strictly speaking, the term server refers to a computer program or process (running program). Through metonymy, it refers to a device used to (or a device dedicated to) running one or several server programs. On a network, such a device is called a host. In addition to server, the words serve and service (as noun and as verb) are frequently used, though servicer and servant are not. The word service (noun) may refer to either the abstract form of functionality, e.g. Web service. Alternatively, it may refer to a computer program that turns a computer into a server, e.g. Windows service. Originally used as "servers serve users" (and "users use servers"), in the sense of "obey", today one often says that "servers serve data", in the same sense as "give". For instance, web servers "serve [up] web pages to users" or "service their requests".

The server is part of the client–server model; in this model, a server serves data for clients. The nature of communication between a client and server is request and response. This is in contrast with peer-to-peer model in which the relationship is on-demand reciprocation. In principle, any computerized process that can be used or called by another process (particularly remotely, particularly to share a resource) is a server, and the calling process or processes is a client. Thus any general purpose computer connected to a network can host servers. For example, if files on a device are shared by some process, that process is a file server. Similarly, web server software can run on any capable computer, and so a laptop or a personal computer can host a web server.

While request–response is the most common client–server design, there are others, such as the publish–subscribe pattern. In the publish–subscribe pattern, clients register with a pub–sub server, subscribing to specified types of messages; this initial registration may be done by request–response. Thereafter, the pub–sub server forwards matching messages to the

clients without any further requests: the server pushes messages to the client, rather than the client pulling messages from the server as in request–response.

When referring to hardware, the word server typically designates computer models specialized for their role. In general, a server performs its role better than a generic personal computer.

6.2. Purpose

The purpose of a server is to share data as well as to share resources and distribute work. A server computer can serve its own computer programs as well; depending on the scenario, this could be part of a quid pro quo transaction, or simply a technical possibility. The following table shows several scenarios in which a server is used.

Table.6.1. purpose of servers

Server type	Purpose	Clients
Application server	Hosts web apps (computer programs that run inside a web browser) allowing users in the network to run and use them, without having to install a copy on their own computers. Unlike what the name might imply, these servers need not be part of the world wide web; any local network would do.	Computers with a web browser
Catalog server	Maintains an index or table of contents of information that can be found across a large distributed network, such as computers, users, files shared on file servers, and web apps. Directory servers and name servers are examples of catalog servers.	Any computer program that needs to find something on the network, such a Domain member attempting to log in, an email client looking for an email address, or a user looking for a file
Communications server	Maintains an environment needed for one communication endpoint (user or devices) to find other endpoints and communicate with them. It may or may not include a directory of communication endpoints and a presence detection service, depending on the openness and security parameters of the network	Communication endpoints (users or devices)

Computing server	Shares vast amounts of computing resources, especially CPU and random-access memory, over a network.	Any computer program that needs more CPU power and RAM than a personal computer can probably afford. The client must be a networked computer; otherwise, there would be no client–server model.
Database server	Maintains and shares any form of database (organized collections of data with predefined properties that may be displayed in a table) over a network.	Spreadsheets, accounting software, asset management software or virtually any computer program that consumes well-organized data, especially in large volumes
Fax server	Shares one or more fax machines over a network, thus eliminating the hassle of physical access	Any fax sender or recipient
File server	Shares files and folder, storage space to hold files and folders, or both, over a network	Networked computers are the intended clients, even though local programs can be clients
Game server	Enables several computers or gaming devices to play multiplayer games	Personal computers or gaming consoles
Mail server	Makes email communication possible in the same way that a post office makes snail mail communication possible	Senders and recipients of email
Media server	Shares digital video or digital audio over a network through media streaming (transmitting content in a way that portions received can be watched or listened as they arrive)	User-attended personal computers equipped with a monitor and a speaker

Print server	Shares one or more printers over a network, thus eliminating the hassle of physical access	Computers in need of printing something
Sound server	Enables computer programs of a computer to play sound and record sound, individually or cooperatively	Computer programs of the same computer
Proxy server	Acts as an intermediary between a client and a server, accepting incoming traffic from the client and sending it to the server. Reasons for doing so includes content control and filtering, improving traffic performance, preventing unauthorized network access or simply routing the traffic over a large and complex network.	Any networked computer
Web server	Hosts web pages. A web server is what makes world wide web possible. Each website has one or more web servers.	Computers with a web browser

Almost the entire structure of the Internet is based upon a client–server model. High-level root name servers, DNS, and routers direct the traffic on the internet. There are millions of servers connected to the Internet, running continuously throughout the world and virtually every action taken by an ordinary Internet user requires one or more interactions with one or more server. There are exceptions that do not use dedicated servers; for example peer-to-peer file sharing, some implementations of telephony (e.g. pre-Microsoft Skype).

6.3. Backend/ Server side Development

54Layered Neural network (Artificial Intelligence) is developed for the classification of DATA. The Data collected is being analysed by AI and it is classified and mapped on the maps. This AI learns with each and every new data from multiple sources and the Decisions are taken and classify the croudsourced data.

6.3.1. Artificial Intelligence (AI):

Artificial intelligence (AI) is intelligence exhibited by machines. In computer science, the field of AI research defines itself as the study of "intelligent agents": any device that perceives its environment and takes actions that maximize its chance of success at some goal. Colloquially, the term "artificial intelligence" is applied when a machine mimics "cognitive" functions that humans associate with other human minds, such as "learning" and "problem solving" (known as Machine Learning). As machines become increasingly capable, mental facilities once thought to require intelligence are removed from the definition. For instance, optical character recognition is no longer perceived as an example of "artificial intelligence", having become a routine technology. Capabilities currently classified as AI include successfully understanding human speech, competing at a high level in strategic game systems (such as Chess and Go), self-driving cars, intelligent routing in content delivery networks, and interpreting complex data.

AI research is divided into subfields that focus on specific problems or on specific approaches or on the use of a particular tool or towards satisfying particular applications.

AI research include reasoning, knowledge, planning, learning, natural language processing (communication), perception and the ability to move and manipulate objects. General intelligence is among the field's long-term goals. Approaches include statistical methods, computational intelligence, and traditional symbolic AI. Many tools are used in AI, including versions of search and mathematical optimization, logic, methods based on probability and economics. The AI field draws upon computer science, mathematics, psychology, linguistics, philosophy, neuroscience and artificial psychology.

The field was founded on the claim that human intelligence "can be so precisely described that a machine can be made to simulate it". This raises philosophical arguments about the nature of the mind and the ethics of creating artificial beings endowed with human-like intelligence, issues which have been explored by myth, fiction and philosophy since antiquity. Some people also consider AI a danger to humanity if it progresses unabatedly. Attempts to create artificial intelligence have experienced many setbacks, including the ALPAC report of 1966, the abandonment of perceptrons in 1970, the Lighthill Report of 1973, the second AI winter 1987–1993 and the collapse of the Lisp machine market in 1987.

In the twenty-first century, AI techniques, both "hard" and "soft", have experienced a resurgence following concurrent advances in computer power, sizes of training sets, and theoretical understanding, and AI techniques have become an essential part of the technology industry, helping to solve many challenging problems in computer science. Recent advancements in AI, and specifically in machine learning, have contributed to the growth of Autonomous Things such as drones and self-driving cars, becoming the main driver of innovation in the automotive industry.

6.3.2. Artificial Neural Networks (ANN):

Artificial neural networks (ANNs) or connectionist systems are a computational model used in computer science and other research disciplines, which is based on a large collection of simple neural units (artificial neurons), loosely analogous to the observed behavior of a biological brain's axons. Each neural unit is connected with many others, and links can enhance or inhibit the activation state of adjoining neural units. Each individual neural unit computes using summation function. There may be a threshold function or limiting function on each connection and on the unit itself, such that the signal must surpass the limit before propagating to other neurons. These systems are self-learning and trained, rather than explicitly programmed, and excel in areas where the solution or feature detection is difficult to express in a traditional computer program.

Neural networks typically consist of multiple layers or a cube design, and the signal path traverses from the first (input), to the last (output) layer of neural units. Back propagation is the use of forward stimulation to reset weights on the "front" neural units and this is sometimes done in combination with training where the correct result is known. More modern networks are a bit more free flowing in terms of stimulation and inhibition with connections interacting in a much more chaotic and complex fashion. Dynamic neural networks are the most advanced, in that they dynamically can, based on rules, form new connections and even new neural units while disabling others

The goal of the neural network is to solve problems in the same way that the human brain would, although several neural networks are more abstract. Modern neural network projects typically work with a few thousand to a few million neural units and millions of connections, which is still several orders of magnitude less complex than the human brain and closer to the computing power of a worm.

New brain research often stimulates new patterns in neural networks. One new approach is using connections which span much further and link processing layers rather than always being localized to adjacent neurons. Other research being explored with the different types of signal over time that axons propagate, such as Deep Learning, interpolates greater complexity than a set of boolean variables being simply on or off.

Neural networks are based on real numbers, with the value of the core and of the axon typically being a representation between 0.0 and 1.

An interesting facet of these systems is that they are unpredictable in their success with self-learning. After training, some become great problem solvers and others don't perform as well. In order to train them, several thousand cycles of interaction typically occur.

Like other machine learning methods – systems that learn from data – neural networks have been used to solve a wide variety of tasks, like computer vision and speech recognition, that are hard to solve using ordinary rule-based programming.

Historically, the use of neural network models marked a directional shift in the late eighties from high-level (symbolic) artificial intelligence, characterized by expert systems with knowledge embodied in if-then rules, to low-level (sub-symbolic) machine learning, characterized by knowledge embodied in the parameters of a cognitive model with some dynamical system.

Network Function

The word network in the term 'artificial neural network' refers to the interconnections between the neurons in the different layers of each system. An example system has three

layers. The first layer has input neurons which send data via synapses to the second layer of neurons, and then via more synapses to the third layer of output neurons. More complex systems will have more layers of neurons, some having increased layers of input neurons and output neurons. The synapses store parameters called "weights" that manipulate the data in the calculations.

An ANN is typically defined by three types of parameters:

1. The interconnection pattern between the different layers of neurons
2. The weights of the interconnections, which are updated in the learning process.
3. The activation function that converts a neuron's weighted input to its output activation.

Multilayer Neural Networks

Consider a supervised learning problem where we have access to labeled training examples (x(i),y(i))(x(i),y(i)). Neural networks give a way of defining a complex, non-linear - form of hypotheses hW,b(x)hW,b(x), with parameters W,bW,b that we can fit to our data.
To describe neural networks, we will begin by describing the simplest possible neural network, one which comprises a single “neuron.” We will use the following diagram to denote a single neuron:

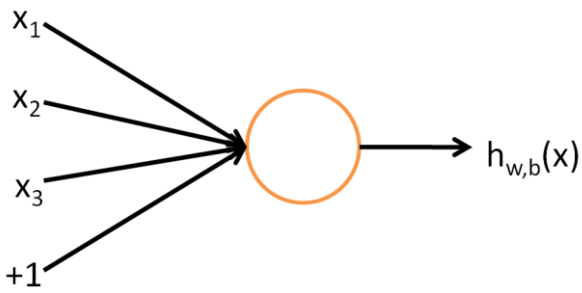

Fig.6.1. Single Neuron

This “neuron” is a computational unit that takes as input x1,x2,x3x1,x2,x3 (and a +1 intercept term).

and outputs hW,b(x)=f(WTx)=f(∑3i=1Wixi+b)hW,b(x)=f(WTx)=f(∑i=13Wixi+b), where f:R↦Rf:ℜ↦ℜ is called the activation function. In these notes, we will choose f(·)f(·) to be the sigmoid function:

f(z)=11+exp(−z).f(z)=11+exp(−z).

Thus, our single neuron corresponds exactly to the input-output mapping defined by logistic regression. Although these notes will use the sigmoid function, it is worth noting that another common choice for ffis the hyperbolic tangent, or tanh, function:

f(z)=tanh(z)=ez−e−zez+e−z.f(z)=tanh(z)=ez−e−zez+e−z.

Recent research has found a different activation function, the rectified linear function, often works better in practice for deep neural networks. This activation function is different from sigmoid and tanhtanh because it is not bounded or continuously differentiable. The rectified linear activation function is given by,

f(z)=max(0,x).f(z)=max(0,x).

Here are plots of the sigmoid, tanhtanh and rectified linear functions:

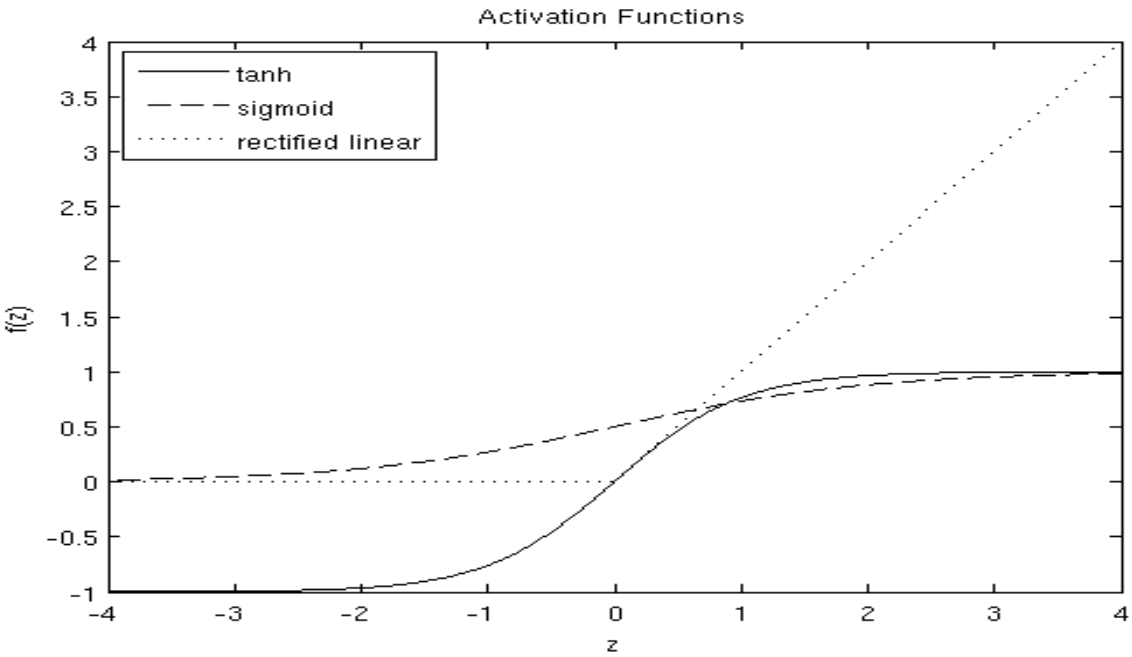

Fig.6.2. Activation Functions of Neural Network

The tanh(z)tanh(z) function is a rescaled version of the sigmoid, and its output range is [−1,1][−1,1] instead of [0,1][0,1]. The rectified linear function is piece-wise linear and saturates at exactly 0 whenever the input zz is less than 0. Finally, one identity that'll be useful later:

If f(z)=1/(1+exp(−z))f(z)=1/(1+exp(−z)) is the sigmoid function, then its derivative is given by f′(z)=f(z)(1−f(z))f′(z)=f(z)(1−f(z)).

If ff is the tanh function, then its derivative is given by f′(z)=1−(f(z))2f′(z)=1−(f(z))2.

You can derive this yourself using the definition of the sigmoid (or tanh) function. The rectified linear function has gradient 0 when z≤0z≤0 and 1 otherwise. The gradient is undefined at z=0z=0, though this doesn't cause problems in practice because we average the gradient over many training examples during optimization.

Neural Network model

A neural network is put together by hooking together many of our simple "neurons," so that the output of a neuron can be the input of another. For example, here is a small neural network:

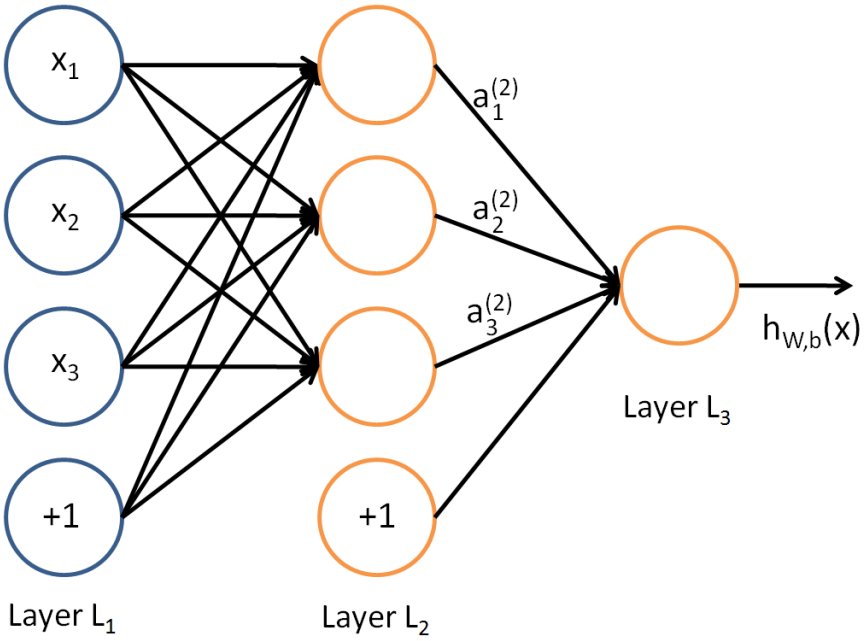

Fig.6.3. Neural Network

In this figure, we have used circles to also denote the inputs to the network. The circles labeled "+1" are called bias units, and correspond to the intercept term. The leftmost layer of the network is called the input layer, and the rightmost layer the output layer (which, in this example, has only one node). The middle layer of nodes is called the hidden layer, because its values are not observed in the training set. We also say that our example neural network has 3 input units (not counting the bias unit), 3 hidden units, and 1 output unit.

We will let nlnl denote the number of layers in our network; thus nl=3nl=3 in our example. We label layer llas LlLl, so layer L1L1 is the input layer, and layer LnlLnl the output layer.

Our neural network has parameters (W,b)=(W(1),b(1),W(2),b(2)) (W,b)=(W(1),b(1),W(2),b(2)), where we write W(l)ijWij(l) to denote the parameter (or weight) associated with the connection between unit jj in layer ll, and unit ii in layer l+1l+1. (Note the order of the indices.) Also, b(l)ibi(l) is the bias associated with unit ii in layer l+1l+1. Thus, in our example, we have W(1)∈R3×3W(1)∈ℜ3×3, and W(2)∈R1×3W(2)∈ℜ1×3. Note that bias units don't have inputs or connections going into them, since they always output the value +1. We also let slsl denote the number of nodes in layer ll (not counting the bias unit).

We will write $a(l)_i$ai(l) to denote the activation (meaning output value) of unit ii in layer ll. For l=1l=1, we also use $a(1)_i=x_i$ai(1)=xi to denote the ii-th input. Given a fixed setting of the parameters W,bW,b, our neural network defines a hypothesis hW,b(x)hW,b(x) that outputs a real number. Specifically, the computation that this neural network represents is given by:

a(2)1a(2)2a(2)3hW,b(x)=f(W(1)11x1+W(1)12x2+W(1)13x3+b(1)1)=f(W(1)21x1+W(1)22x2+W(1)23x3+b(1)2)=f(W(1)31x1+W(1)32x2+W(1)33x3+b(1)3)=a(3)1=f(W(2)11a(2)1+W(2)12a(2)2+W(2)13a(2)3+b(2)1)a1(2)=f(W11(1)x1+W12(1)x2+W13(1)x3+b1(1))a2(2)=f(W21(1)x1+W22(1)x2+W23(1)x3+b2(1))a3(2)=f(W31(1)x1+W32(1)x2+W33(1)x3+b3(1))hW,b(x)=a1(3)=f(W11(2)a1(2)+W12(2)a2(2)+W13(2)a3(2)+b1(2))

In the sequel, we also let $z(l)_i$zi(l) denote the total weighted sum of inputs to unit ii in layer ll, including the bias term (e.g., $z(2)_i=\sum_{j=1}^{n}W(1)_{ij}x_j+b(1)_i$zi(2)=∑j=1nWij(1)xj+bi(1)), so that $a(l)_i=f(z(l)_i)$ai(l)=f(zi(l)).

Note that this easily lends itself to a more compact notation. Specifically, if we extend the activation function f(·)f(·) to apply to vectors in an element-wise fashion (i.e., f([z1,z2,z3])=[f(z1),f(z2),f(z3)]f([z1,z2,z3])=[f(z1),f(z2),f(z3)]), then we can write the equations above more compactly as:

z(2)a(2)z(3)hW,b(x)=W(1)x+b(1)=f(z(2))=W(2)a(2)+b(2)=a(3)=f(z(3))z(2)=W(1)x+b(1)a(2)=f(z(2))z(3)=W(2)a(2)+b(2)hW,b(x)=a(3)=f(z(3))

We call this step forward propagation**.** More generally, recalling that we also use a(1)=xa(1)=x to also denote the values from the input layer, then given layer ll's activations a(l)a(l), we can compute layer l+1l+1's activations a(l+1)a(l+1) as:

z(l+1)a(l+1)=W(l)a(l)+b(l)=f(z(l+1))z(l+1)=W(l)a(l)+b(l)a(l+1)=f(z(l+1))

By organizing our parameters in matrices and using matrix-vector operations, we can take advantage of fast linear algebra routines to quickly perform calculations in our network.

We have so far focused on one example neural network, but one can also build neural networks with other architectures (meaning patterns of connectivity between neurons), including ones with multiple hidden layers. The most common choice is a nlnl-layered network where layer 11 is the input layer, layer nlnlis the output layer, and each layer ll is densely connected to layer l+1l+1. In this setting, to compute the output of the network, we can successively compute all the activations in layer L2L2, then layer L3L3, and so on, up to layer LnlLnl, using the equations above that describe the forward propagation step. This is one example of a feed forward neural network, since the connectivity graph does not have any directed loops or cycles.

Neural networks can also have multiple output units. For example, here is a network with two hidden layers layers L2L2 and L3L3 and two output units in layer L4L4:

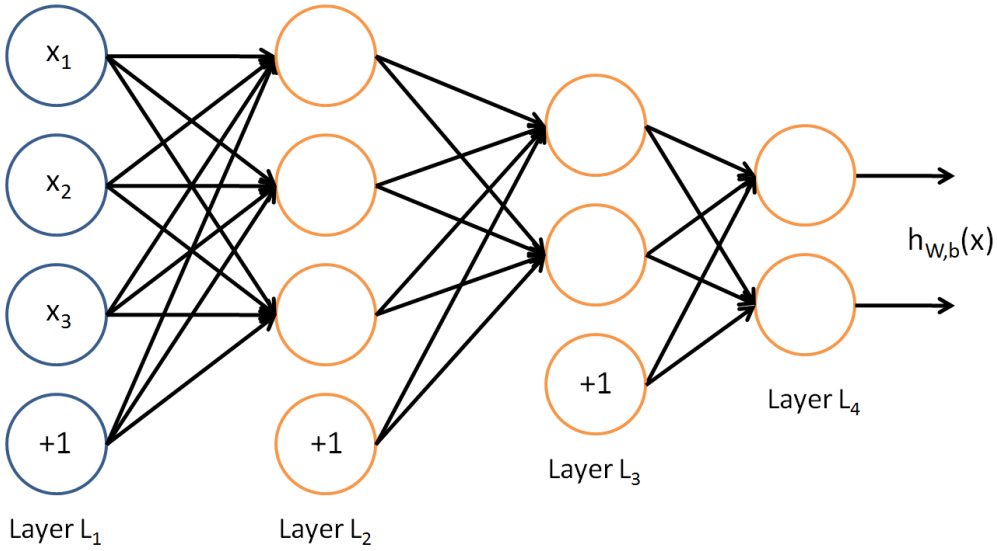

Fig.6.4. Two layer Neural Network

To train this network, we would need training examples (x(i),y(i))(x(i),y(i)) where y(i)∈R2y(i)∈ℜ2. This sort of network is useful if there're multiple outputs that you're interested in predicting. (For example, in a medical diagnosis application, the vector xx might give the input features of a patient, and the different outputs yiyi's might indicate presence or absence of different diseases.)

6.3.3. Back propagation Algorithm:

Suppose we have a fixed training set {(x(1),y(1)),…,(x(m),y(m))}{(x(1),y(1)),…,(x(m),y(m))} of mm training examples. We can train our neural network using batch gradient descent. In detail, for a single training example (x,y)(x,y), we define the cost function with respect to that single example to be:

J(W,b;x,y)=12‖hW,b(x)−y‖2.J(W,b;x,y)=12‖hW,b(x)−y‖2.

This is a (one-half) squared-error cost function. Given a training set of mm examples, we then define the overall cost function to be:

J(W,b)=[1m∑i=1mJ(W,b;x(i),y(i))]+λ2∑l=1nl−1∑i=1sl∑j=1sl+1(W(l)ji)2=[1m∑i=1m(12‖‖hW,b(x(i))−y(i)‖‖2)]+λ2∑l=1nl−1∑i=1sl∑j=1sl+1(W(l)ji)2J(W,b)=[1m∑i=1mJ(W,b;x(i),y(i))]+λ2∑l=1nl−1∑i=1sl∑j=1sl+1(Wji(l))2=[1m∑i=1m(12‖hW,b(x(i))−y(i)‖2)]+λ2∑l=1nl−1∑i=1sl∑j=1sl+1(Wji(l))2

The first term in the definition of J(W,b)J(W,b) is an average sum-of-squares error term. The second term is a regularization term (also called a weight decay term) that tends to decrease the magnitude of the weights, and helps prevent over fitting.

(**Note:** Usually weight decay is not applied to the bias terms b(l)ibi(l), as reflected in our definition for J(W,b)J(W,b). Applying weight decay to the bias units usually makes only a small difference to the final network, however. If you've taken CS229 (Machine Learning) at Stanford or watched the course's videos on YouTube, you may also recognize this weight decay as essentially a variant of the Bayesian regularization method you saw there, where we placed a Gaussian prior on the parameters and did MAP (instead of maximum likelihood) estimation.)

The weight decay parameter λλ controls the relative importance of the two terms. Note also the slightly overloaded notation: J(W,b;x,y)J(W,b;x,y) is the squared error cost with respect to a single example; J(W,b)J(W,b) is the overall cost function, which includes the weight decay term.

This cost function above is often used both for classification and for regression problems. For classification, we let y=0y=0 or 11 represent the two class labels (recall that the sigmoid activation function outputs values in [0,1][0,1]; if we were using a tanh activation function, we would instead use -1 and +1 to denote the labels). For regression problems, we first scale our outputs to ensure that they lie in the [0,1][0,1]range (or if we were using a tanh activation function, then the [−1,1][−1,1] range).

Our goal is to minimize J(W,b)J(W,b) as a function of WW and bb. To train our neural network, we will initialize each parameter W(l)ijWij(l) and each b(l)ibi(l) to a small random value near zero (say according to a Normal(0,ϵ2)Normal(0,ϵ2)distribution for some small ϵϵ, say 0.010.01), and then apply an optimization algorithm such as batch gradient descent. Since J(W,b)J(W,b) is a non-convex function, gradient descent is susceptible to local optima; however, in practice gradient descent usually works fairly well. Finally, note that it is important to initialize the parameters randomly, rather than to all 0's. If all the parameters start off at identical values, then all the hidden layer units will end up learning the same function of the input (more formally, W(1)ijWij(1)will be the same for all values of ii, so that a(2)1=a(2)2=a(2)3=…a1(2)=a2(2)=a3(2)=… for any input xx). The random initialization serves the purpose of symmetry breaking.

One iteration of gradient descent updates the parameters W,bW,b as follows:

W(l)ijb(l)i=W(l)ij−α∂∂W(l)ijJ(W,b)=b(l)i−α∂∂b(l)iJ(W,b)Wij(l)=Wij(l)−α∂∂Wij(l)J(W,b)bi(l)=bi(l)−α∂∂bi(l)J(W,b)

Where αα is the learning rate. The key step is computing the partial derivatives above. We will now describe the back propagation algorithm, which gives an efficient way to compute these partial derivatives.

We will first describe how back propagation can be used to compute ∂∂W(l)ijJ(W,b;x,y)∂∂Wij(l)J(W,b;x,y) and ∂∂b(l)iJ(W,b;x,y)∂∂bi(l)J(W,b;x,y), the partial derivatives of the cost function J(W,b;x,y)J(W,b;x,y) defined with respect to a single example (x,y)(x,y). Once we can compute these, we see that the derivative of the overall cost function J(W,b)J(W,b) can be computed as:

∂∂W(l)ijJ(W,b)∂∂b(l)iJ(W,b)=[1m∑i=1m∂∂W(l)ijJ(W,b;x(i),y(i))]+λW(l)ij=1m∑i=1m∂∂b(l)iJ(W,b;x(i),y(i))∂∂Wij(l)J(W,b)=[1m∑i=1m∂∂Wij(l)J(W,b;x(i),y(i))]+λWij(l)∂∂bi(l)J(W,b)=1m∑i=1m∂∂bi(l)J(W,b;x(i),y(i))

The two lines above differ slightly because weight decay is applied to WW but not bb. The intuition behind the back propagation algorithm is as follows. Given a training example (x,y)(x,y), we will first run a "forward pass" to compute all the activations

throughout the network, including the output value of the hypothesis hW,b(x)hW,b(x). Then, for each node ii in layer ll, we would like to compute an "error term" δ(l)iδi(l) that measures how much that node was "responsible" for any errors in our output. For an output node, we can directly measure the difference between the network's activation and the true target value, and use that to define δ(nl)iδi(nl) (where layer nlnl is the output layer). How about hidden units? For those, we will compute δ(l)iδi(l) based on a weighted average of the error terms of the nodes that uses a(l)iai(l) as an input. In detail, here is the back propagation algorithm:

1. Perform a feed forward pass, computing the activations for layers L2L2, L3L3, and so on up to the output layer LnlLnl.
2. For each output unit ii in layer nlnl (the output layer), set

   δ(nl)i=∂∂z(nl)i12‖y−hW,b(x)‖2=−(yi−a(nl)i)·f′(z(nl)i)δi(nl)=∂∂zi(nl)12‖y−hW,b(x)‖2
   =−(yi−ai(nl))·f′(zi(nl))

3. For l=nl−1,nl−2,nl−3,…,2l=nl−1,nl−2,nl−3,…,2
   For each node ii in layer ll, set
   δ(l)i=(∑sl+1j=1W(l)jiδ(l+1)j)f′(z(l)i)δi(l)=(∑j=1sl+1Wji(l)δj(l+1))f′(zi(l))
4. Compute the desired partial derivatives, which are given as:

∂∂W(l)ijJ(W,b;x,y)∂∂b(l)iJ(W,b;x,y)=a(l)jδ(l+1)i=δ(l+1)i.∂∂Wij(l)J(W,b;x,y)=aj(l)δi(l+1)∂∂b
i(l)J(W,b;x,y)=δi(l+1).

Finally, we can also re-write the algorithm using matrix-vectorial notation. We will use "·" to denote the element-wise product operator (denoted .* in Matlab or Octave, and also called the Hadamard product), so that if a=b·ca=b·c, then ai=biciai=bici. Similar to how we extended the definition of f(·)f(·) to apply element-wise to vectors, we also do the same for f′(·)f′(·) (so that f′([z1,z2,z3])=[f′(z1),f′(z2),f′(z3)]f′([z1,z2,z3])=[f′(z1),f′(z2),f′(z3)]).
The algorithm can then be written:

1. Perform a feed forward pass, computing the activations for layers L2L2, L3L3, up to the output layer LnlLnl, using the equations defining the forward propagation steps
2. For the output layer (layer nlnl), set

   δ(nl)=−(y−a(nl))·f′(z(nl))δ(nl)=−(y−a(nl))·f′(z(nl))

3. For l=nl−1,nl−2,nl−3,…,2l=nl−1,nl−2,nl−3,…,2, set

   δ(l)=((W(l))Tδ(l+1))·f′(z(l))δ(l)=((W(l))Tδ(l+1))·f′(z(l))

4. Compute the desired partial derivatives:

∇W(l)J(W,b;x,y)∇b(l)J(W,b;x,y)=δ(l+1)(a(l))T,=δ(l+1).∇W(l)J(W,b;x,y)=δ(l+1)(a(l))T,∇b(l)J
(W,b;x,y)=δ(l+1).

Implementation Note: In steps 2 and 3 above, we need to compute f′(z(l)i)f′(zi(l)) for each value of ii. Assuming f(z)f(z) is the sigmoid activation function, we would already have a(l)iai(l) stored away from the forward pass through the network. Thus, using the expression that we worked out earlier for f′(z)f′(z), we can compute this as f′(z(l)i)=a(l)i(1−a(l)i)f′(zi(l))=ai(l)(1−ai(l)).

Finally, we are ready to describe the full gradient descent algorithm. In the pseudo-code below, ΔW(l)ΔW(l)is a matrix (of the same dimension as W(l)W(l)), and Δb(l)Δb(l) is a vector (of the same dimension as b(l)b(l)). Note that in this notation, ”ΔW(l)ΔW(l)” is a matrix, and in particular it isn’t ”ΔΔ times W(l)W(l).” We implement one iteration of batch gradient descent as follows:

1. Set ΔW(l):=0ΔW(l):=0, Δb(l):=0Δb(l):=0 (matrix/vector of zeros) for all ll.
2. For i=1i=1 to mm,
    1. Use back propagation to compute

        ∇W(l)J(W,b;x,y)∇W(l)J(W,b;x,y) and ∇b(l)J(W,b;x,y)∇b(l)J(W,b;x,y).

    2. Set ΔW(l):=ΔW(l)+∇W(l)J(W,b;x,y)ΔW(l):=ΔW(l)+∇W(l)J(W,b;x,y).
    3. Set Δb(l):=Δb(l)+∇b(l)J(W,b;x,y)Δb(l):=Δb(l)+∇b(l)J(W,b;x,y).
3. Update the parameters:

W(l)b(l)=W(l)−α[(1mΔW(l))+λW(l)]=b(l)−α[1mΔb(l)]W(l)=W(l)−α[(1mΔW(l))+λW(l)]b(l)=b(l)−α[1mΔb(l)]

To train our neural network, we can now repeatedly take steps of gradient descent to reduce our cost function J(W,b)J(W,b).

CHAPTER-7

PROGRAM OF ROADTRACK

7.1. Program

```
#include <SoftwareSerial.h>

#include <AcceleroMMA7361.h>

AcceleroMMA7361 accelero;

SoftwareSerial SIM900 (7, 8); // configure software serial port RXIN=7, TXpin=8

//Initializing the required variables

char latitude [15];

char longitude[15];

char altitude[6];

char date[16];

char time[7];

char satellites[3];

char speedOTG[10];

char course[10];

char frame[200];

int q=0;

int x;

int y;

int z;

void setup() {

  //pinMode(REDLed, OUTPUT);

  //pinMode(GREENLed, OUTPUT);

 // pinMode(GREENLed, OUTPUT);

   SIM900.begin(9600);     //9600 baud rate communication is used for communicating with
SIM module using AT Commands
```

```
  Serial.begin(115200);   //115200 baud rate communication is used printing on serial monitor
  Serial.print("power up" );
  delay(300);
  accelero.begin();  //starting the accelerometer library
  accelero.setSensitivity(HIGH);  //sets the sensitivity to +/-6G
  accelero.calibrate(); //calibrates the accelerometer sensor
}
void loop()
{
int answer=0;
    x = accelero.getXRaw(); //imports the x direction values from the accelerometer library
    y = accelero.getYRaw(); //imports the y direction values from the accelerometer library
    z = accelero.getZRaw(); //imports the z direction values from the accelerometer library
  Serial.println("SubmitGPSRequest - started" );
  SubmitGPSRequest();
  Serial.println("SubmitHttpRequest - started" );
  SubmitHttpRequest();
  Serial.println("SubmitHttpRequest - finished" );
  delay(100);
    Serial.println("SubmitHttpandGpsRequest - started" );
    SubmitHttpandGpsRequest();
  Serial.println("SubmitHttpandGpsRequest - finished" );
  delay(500);
//x=x+40;
//y=y+60;
}
void SubmitGPSRequest()
```

```
{
  SIM900.println("AT"); // Signal quality check
  delay(100);
  ShowSerialData();
  SIM900.println("AT+CGNSPWR=1");
  delay(100);
  ShowSerialData();
  SIM900.println("AT+CGNSSEQ=\"RMC\"");
  delay(100);
  ShowSerialData();
  /*Serial.println(latitude);
  delay(100);
  Serial.println(satellites);
  delay(100);
  Serial.println(date);
  delay(100);*/
  }
void SubmitHttpRequest()
{
  SIM900.println("AT+CSQ"); // Signal quality check
  delay(100);
  ShowSerialData();// this code is to show the data from gprs shield, in order to easily see the
process of how the gprs shield submit a http request, and the following is for this purpose too.
  //SIM900.println("AT+CGATT?"); //Attach or Detach from GPRS Support
  delay(100);
  ShowSerialData();
  SIM900.println("AT+SAPBR=3,1,\"Contype\",\"GPRS\"");//setting the SAPBR, the
connection type is using gprs
```

```
  //AT+SAPBR=3,1,"Contype","GPRS"
  delay(100);
  ShowSerialData();
  SIM900.println("AT+SAPBR=3,1,\"APN\",\"www\"");//setting the APN, Access point
name string
  //AT+SAPBR=3,1,"APN","www"
  delay(400);
  ShowSerialData();
  SIM900.println("AT+SAPBR=1,1");//setting the SAPBR
  delay(200);
  ShowSerialData();
  SIM900.println("AT+HTTPINIT"); //init the HTTP request
  delay(200);
  ShowSerialData();
SIM900.println("AT+HTTPPARA=\"CID\",1"); //init the HTTP request
//AT+HTTPPARA="CID",1
  delay(200);
  ShowSerialData();
delay(100);
}
void SubmitHttpandGpsRequest(){
  int counter = 0;
  int answer=0;
  memset(frame, '\0', 100);
  SIM900.println("AT+CGNSINF");
do{
  if(SIM900.available() != 0){
```

```
            frame[counter] = SIM900.read();
            counter++;
            // check if the desired answer is in the response of the module
            if (strstr(frame, "OK") != NULL)
            {
                answer = 1;
            }
        }
        // Waits for the asnwer with time out
}
while((answer == 0));
    delay(100);
    /*frame[counter-3] = '\0';
    // Parses the string
    strtok(frame, ",");
    strcpy(longitude,strtok(NULL, ",")); // Gets longitude
    strcpy(latitude,strtok(NULL, ",")); // Gets latitude
    strcpy(altitude,strtok(NULL, ".")); // Gets altitude
    strtok(NULL, ",");
    strcpy(date,strtok(NULL, ".")); // Gets date
    strtok(NULL, ",");
    strtok(NULL, ",");
    strcpy(satellites,strtok(NULL, ",")); // Gets satellites
    strcpy(speedOTG,strtok(NULL, ",")); // Gets speed over ground. Unit is knots.
    strcpy(course,strtok(NULL, "\r")); // Gets course*/
 /* delay(1000);
  Serial.println("u are awesome");
```

```
  delay(100);
  Serial.println("i want to compete with spacex");
  delay(100);*/
size_t len = strlen(frame);
    memmove(frame, frame+23, len - 23 + 1);
  Serial.println(frame);
  delay(100);
  /*Serial.println("Let me think now");
  delay(100);*/
  int p=0;
/*for(p=23;p<32;p++){
  Serial.print(frame[p]);
  }
  Serial.println(" ");
for(p=33;p<42;p++){
  Serial.print(frame[p]);
  }
  Serial.println(" ");*/
SIM900.print("AT+HTTPPARA=\"URL\",\"http://datadrop.wolframcloud.com/api/v1.0/Add
?bin=kPDw3EV&x=");// setting the httppara, the second parameter is the website you want to
access
  //SIM900.print(x);
for(p=23;p<32;p++){
  SIM900.print(frame[p]);
  }
  SIM900.print("&y=");
  //SIM900.print(y);
for(p=33;p<42;p++){
```

```
  SIM900.print(frame[p]);
  }
SIM900.print("&a=");
SIM900.print(x);
SIM900.print("&b=");
SIM900.print(y);
SIM900.print("&c=");
SIM900.print(z);
  SIM900.println("\"");
  delay(100);
  //Serial.println("submitted url");
  //ShowSerialData();
  SIM900.println("AT+HTTPACTION=0");//submit the request
  delay(10000);//the delay is very important, the delay time is base on the return from the website, if the return data is very large, the time required longer.
  //while(!SIM900.available());
  //SubmitHttpandGpsRequest();
  //ShowSerialData();
  /*SIM900.println("AT+HTTPREAD");// read the data from the website you access
  delay(300);
  changeLed();
  ShowSerialData();*/
  }
  void ShowSerialData()
{
  while(SIM900.available()!=0)
    Serial.write(char (SIM900.read()));}
```

CHAPTER-8

WORKING PRINCIPLE

In our project Arduino is connected to all the peripherals (Accelerometer sensor and GPS-GSM module). Accelerometer is used to measure the Vibrations of the road continuously and these values are read from analog pins A0, A1 and A2.GPS-GSM module is connected serially through Pin 7 and Pin 8, where Pin 7 is used as RX-Pin and Pin 8 is used vvas TX-Pin. Pin 7 and 8 are used as TX and RX pins by using Software Serial predefined function of Arduino.

Using AT commands the SIM808 module is first powered up, then defining the last NMEA sentence that parsed, then Read GNSS navigation information, then Set URC reporting every 2(1-255) GNSS fix, then send Command to GNSS and then Send NMEA data to AT UART. Now GPS and GPRS are ready for use.

GPS values are taken at regular intervals just after acquiring the vibration signal, these GPS Geo Coordinates are embedded with the Vibration signal and stored in an array. With the help of GPRS the data acquired is sent to Wolfram cloud by making an HTTP request and the data is collected in the Cloud in Real time.

The data collected in the cloud is segregated and the vibration data is analyzed primarily .The vibration signal is classified among pre-defined by the Neural Network developed by us.

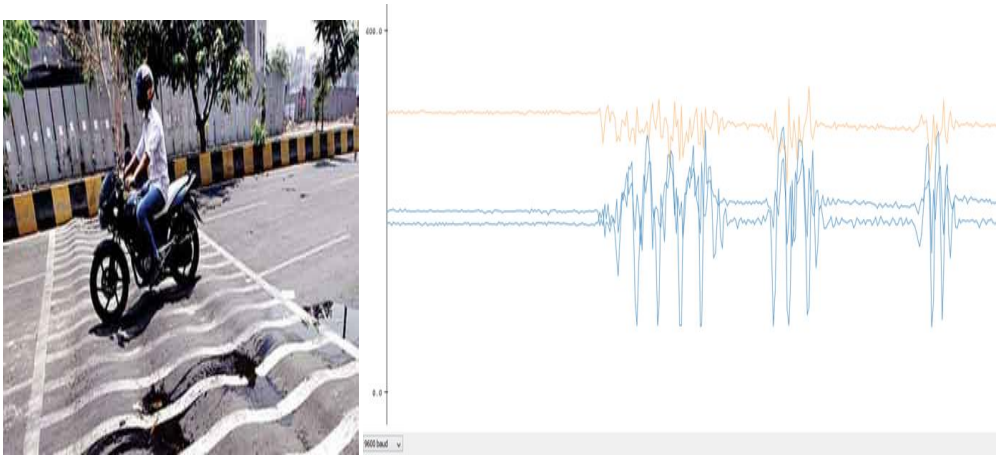

Fig.8.1. Vibrations obtained on speed breakers type roads

Fig8.1 is the vibration data that is obtained when vehicle is traversing through multiple speed breakers where we can observe Two dimensional axis's (X and Y) are experiencing the vibration and Z being unaffected .This pattern is saved in server as the pre-defined classifier for speed breakers.

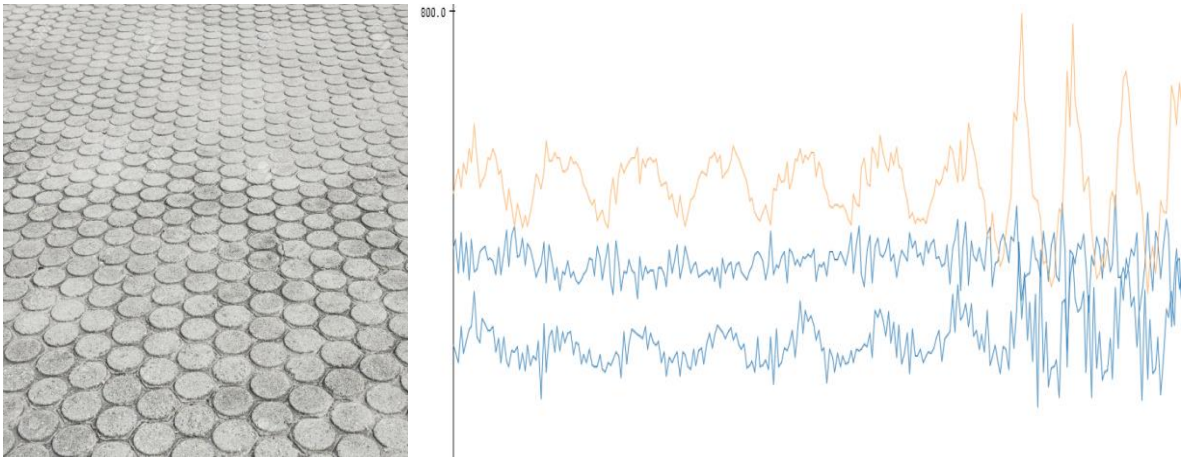

Fig.8.2. Vibrations obtained on gravel type roads

Fig 8.2 is the vibration data that is obtained when vehicle is traversing through a pattern road, in this case we observe a pattern of vibration this is saved as a pre-defined classifier for patterned road.

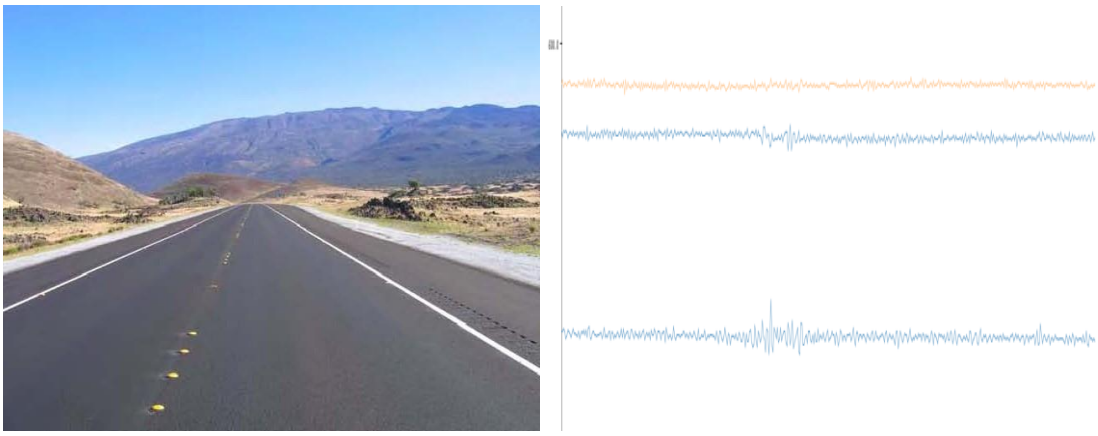

Fig.8.3. Vibrations obtained on asphalt type roads

Fig 8.3 is the vibration data that is obtained when vehicle is traversing on the smooth road, where we observe a perfect smooth wave form.

After the classification of vibration signal a specific colour is assigned based on roughness of road and then with the help of Wolfram Mathematica Vibration data associated with geo-coordinates are plotted on Map.

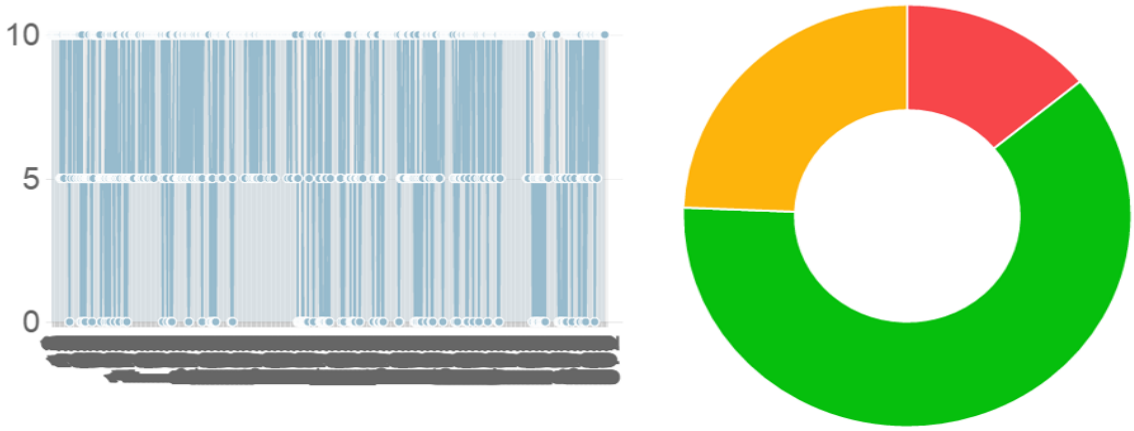

Fig.8.4. Road Quality Charts

This is the Quality chart that is designed to represent the Quality of Roads.

In the first chart we can observe number of points that have road quality below 5 and number of points that have road quality above 5.

In the second chart we have mapped all the road quality data on the pie chart.

This Quality chart has been designed for the Web application that we have built.

CHAPTER-9

RESULTS AND DISCUSSIONS

9.1. Result

After the classification of vibration signal a specific colour is assigned based on roughness of road and then with the help of Wolfram Mathematica Vibration data associated with geo-coordinates are plotted on Map.

All the data is processed and classified in accordance to the pre-defined parameters. All the data that is collected over the period of travel is analyzed in the cloud and is represented over the map in terms of easy-to-understand depiction.

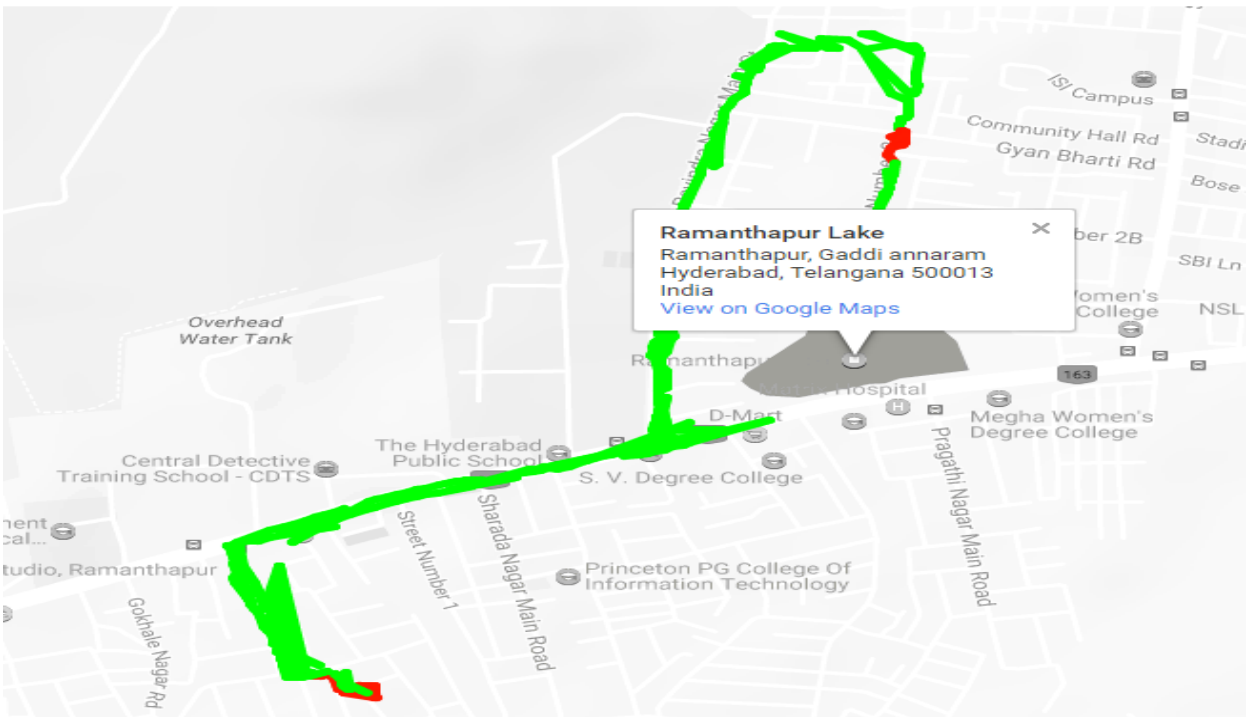

Fig.9.1.Output of RoadTrack

Here, experimentally RoadTrack is implemented and the geo locations are viewed in the geo graphics in terms of longitude and latitude values. Experimental results are obtained in the road route from Ramanthapur TV Studio to Ramanthapur Lake. (View on Google Maps)

The green path indicates that the roads are smooth and are ideal for the user to travel.

The orange path indicates that the roads are not ideal and may cause discomfort to the user. All the processed data is later used as reference for further received data to compare and update the previously stored data. This helps in creating and maintaining a real-time system that constantly improves with use and gets better in prediction over time.

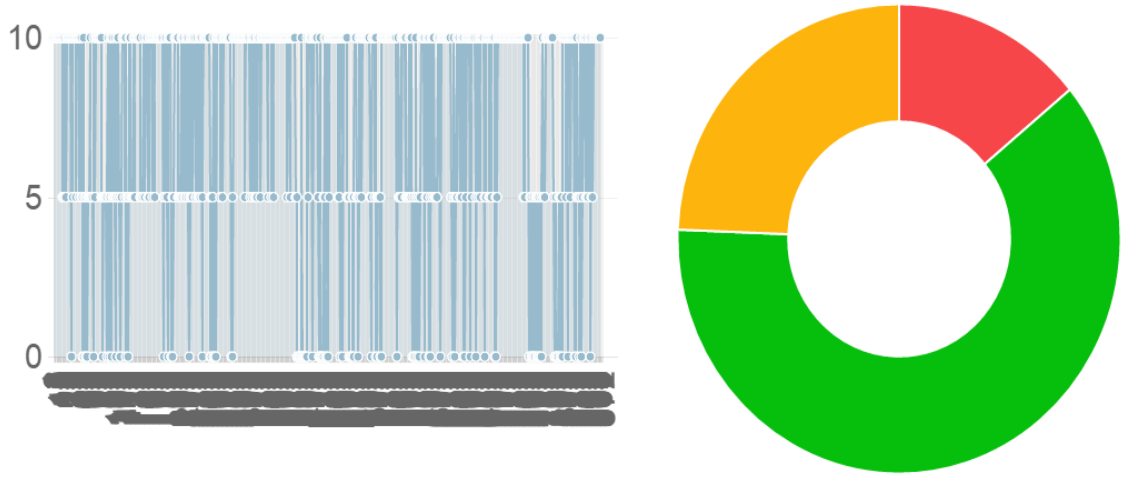

Fig.9.2. Road Quality Chart

Road Quality is also rated on a scale of 0-10 by the designed webpage application. We use a consistent color scale ranging from red (low) to green (high) to quickly convey to users where a rating falls on the spectrum of worst to best routes. (See scale below).

0	Worst	6	Above Average
1	Very Poor	7	Significantly Above Average
2	Poor	8	Good
3	Significantly Below Average	9	Very Good
4	Below Average	10	Best
5	Average		

CHAPTER-10

SUMMARY

10.1. Advantages

- Accurate & Efficient data parsing system
- Intelligent Backend system powered by Artificial Intelligence
- Cloud based, can be used from anywhere in the world and quick(Real time)
- Low hardware requirements
- Web accessible.

10.2. Applications

- It is Great help for cyclists to plan their route in hassle free route
- It can be used by Packers and movers while transportation
- While carrying fragile Articles this application of great use
- Great help in Medical sector
- Can be chosen for faster and safe transport route
- Can be used by Governments and Municipalities in improving the road connectivity

  Ex: Pradhan Mantri Gram Sadak Yojana, NHAI
- More Importantly

  This attribute is unscathed by giants like Google maps, Apple Maps, Yandex Maps And achieved by us, It can be embedded in their product.

CHAPTER-11
CONCLUSION

11.1. Conclusion

RoadTrack now detects all the road attributes and maps them perfectly. Artificial Intelligence powered classification is working accurately and Learning with the new data. Remote server is optimized for High data processing.

Developed a webpage for mapping the Road Quality with statistics.

11.2. Future Scope

- Route planning algorithm
- Can be deployed on Smart phone by an APP
    - This will reduce the dependencies
    - Help crowd sourcing

References

Reference Text books:

1. Introduction to Autonomous Mobile Robots (Intelligent Robotics and Autonomous Agents series) second edition.

   By Davide Scaramuzza, Roland Siegwart & Illah R. Nourbakhsh

2. Probabilistic Robotics (Intelligent Robotics and Autonomous Agents series edition)

   By Sebastian Thrun, Wolfram Burgard & Dieter Fox

Reference Websites:

1. http://omms.nic.in/
2. https://www.wolfram.com/datadrop/quick-reference/home/
3. https://www.wolfram.com/datadrop/quick-reference/arduino/
4. https://www.cooking-hacks.com/projects/arduino-realtime-gps-gprs-vehicle-tracking/
5. http://www.deekshith.in/
6. http://ieeexplore.ieee.org/document/7057827/?reload=true