NAME OF THE STUDENT	:	P.PRUDHVI KRISHNA
ROLL NUMBER	:	AP20110020021
BRANCH / SECTION	:	ECE – A
NAME OF THE SUBJECT	:	ELECTRONIC WORK SHOP ECE_123
NAME OF THE PROJECT	:	IOT ENABLED STRUCTURAL HEALTH MONITORING ROADS

SRM UNIVERSITY - AP

A Project Based on

IOT ENABLED STRUCTURAL HEALTH MONITORING ROADS

IMPLEMENTING ON BOTH

‘ RASPBEERY-PI AND ARDUINO-UNO ’

BRANCH / SECTION	:	ECE – A
PROJECT GROUP	:	6
PROFESSOR	:	GOUTAM RANA SIR
MENTOR	:	UDAYA SANKAR SIR
NAME OF THE SUBJECT	:	ELECTRONIC WORK SHOP ECE_123
NAME OF THE PROJECT	:	IOT ENABLED STRUCTURAL HEALTH MONITORING ROADS

ABSTRACT

As we see one of the main significant issue is Road Damages all over the Country.
So many Streets and Roads contribute a Note Worthy part to the Nation's Economy.

For Example, we see Potholes increasing day by day due to Heavy Vehicles, Some cause of Pollution and Etc. factors involved in Road Damaging.

These are not indicating the drivers to stay away from this Potholes, Damages. It also harms the vehicle, additionally deviate's the drivers to look after roads and leads to many mischievous things.

This Project talks about Pothole Identification Methods that have been created and proposes a practical answer for distinguish the Potholes on streets and roads and give the Information to the Road Management System (Government).

So, It Provides the Information where the Potholes are present.
Raspberry Pi and Arduino UNO are two Boards where we can Implement the Project with Accelerometer Sensor and GPS Sensor, Etc...

These are utilized to Distinguish the Potholes out and about, the land area directions of the Potholes are recognized.
The Detected Information will be collected and it will show the Pothole width and Geographic Area (Location using GPS), and stored in the database.
This Stored Information will reach the Road Safety Management and they will look after the Damages.
Also, An android applications can be Implemented for safety of drivers so they be careful to avoid mishaps.

INDEX

INTRODUCTION

INDIA, the second most crowded Country in the World and a quickly developing economy as we known to have a huge system of streets.

Roads are the main Source for transportation in India today.
They convey very nearly 90 percent of nation's traveler activity and 65 percent of its cargo.
But we see that the large portion of the streets in India are thin and congested with poor surface quality and Road Damages occurs with Potholes.

Regardless of where you are in India, driving is a breath-holding and very difficult part including, possibly life debilitating issues.
The potholes happen because of over whelming precipitation or because of the substantial vehicles out and about.
These potholes turn into the principle purpose behind the street mischances.

Road Surface Monitoring (RSM) is the one of the way toward Identifying the misery on cleared or uneven road surfaces.
Also we can have LVDT sensor and Ride Height Sensors which helps us in Monitoring the Road Surfaces.

Early recognition of street surface breaks can help and support before the repair costs turn out to be too high.

AIM / OF THE PROJECT

The main aim of the Project is to detect Road Surface Anomalies, such as Potholes, Cracks, and Bumps, which affect driving comfort and on-road safety.

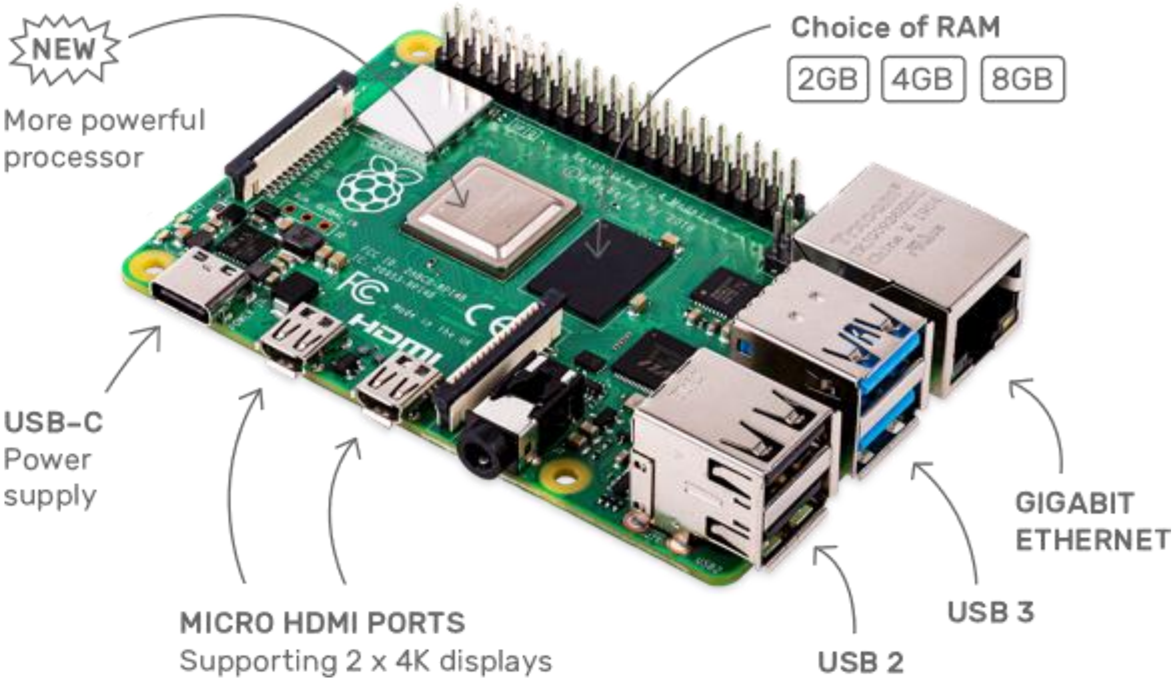
ADVANTAGE'S OF THE PROJECT

- Helps in Monitoring the Roads
- Gives the Information to Road Safety Management
- Tracks the Location and Acceleration
- Ensures Safety
- Increase the life span of Roads
- It reduces the Burden to Government and also early recognition of road surface that can help and support before the repair costs turn out to be too high.

DIS-ADVANTAGE'S OF THE PROJECT

- There are no much Dis-Advantages
- Some Sensors are too Cost to Implement them
- Care full with Data Privacy
- Dependency on Proper Connections

RASPBERRY - PI



The Raspberry Pi is a series of small single-board computers developed in the United Kingdom by the Raspberry Pi Foundation to promote the teaching of basic computer science in schools and in developing countries.

USE OF RASPBERRY-PI

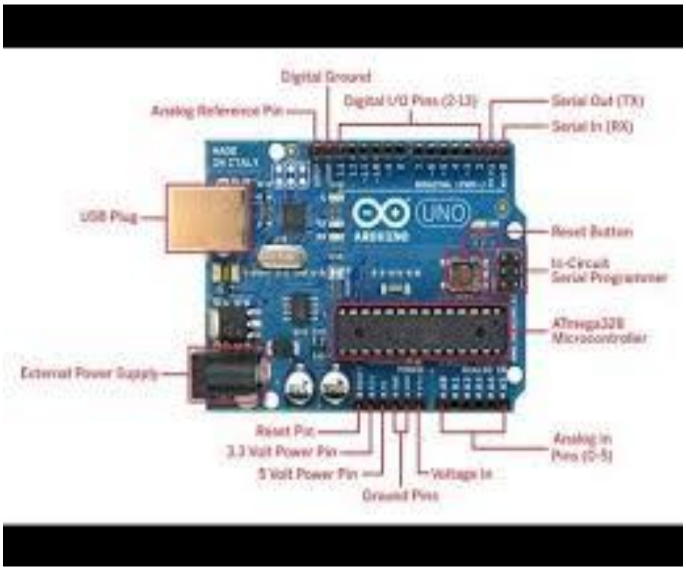
- Very Low Cost ($25 – Rs 1550/- for Model A & $35 –Rs 2200/- for Model B/B+) (moderate if compared to Arduino)
- Great tool for Learning Programming, Computers & Concepts of Embedded Linux, etc
- Consumes less than 5W of Power
- Supports Full HD Video Output (1080p), Multiple USB Ports.

WORKING OF RASPBERRY-PI

An SD card inserted into the slot on the board acts as the hard drive for the **Raspberry Pi**. It is powered by USB and the video output can be hooked up to a traditional RCA TV set, a more modern monitor, or even a TV using the HDMI port.

CPU speed ranges from 700 MHz to 1.2 GHz for the Pi 3 and on board memory range from 256 MB to 1 GB RAM.

ARDUINO



This is a micro controller based on the ATmega 328P.
There are total of 20 pins (0-19) out of which 6 are analog inputs, 14 are digital input output pins (6 pins provide PWM voltage) which can also be used as general purpose pins, a ceramic resonator of frequency 16 MHz, an USB connection, a power jack and a reset button.
It has an operating voltage of 5V. It contains everything needed to support a microcontroller.

DIFFERENCE BETWEEN RASPBERRY-PI AND ARDUINO-UNO

Raspberry Pi is a general-purpose Computer (Mini PC), usually with a Linux operating system, and the ability to run multiple programs. It is more complicated to use than an Arduino Slightly costlier Suitable for complex IOT applications	**Arduino** is a micro-controller based board. Micro-controllers are simple computer that can run one program at a time, over and over again. It is very easy to use Extremely cheap Not advisable for complex IOT applications

COST	
RASPBERRY – PI → 3,500 RS	ARDUINO – UNO → 300 RS

SELECTION OF SENSOR'S AND IMPLEMENTING THEM

RE-SEARCH WORK

Prudhvikrishna_pavuluri@srmap.edu.in → RIDE HEIGHT SENSOR

→ NON-CONTACT :
RHL4 Miniature Laser Ride Height Sensor

→ Laser position sensor - RF603

Vishnu_sharma@srmap.edu.in → LVDT
[Linear Variable Differential Transformation]

Rohithkumar_ankam@srmap.edu.in → Model SVD Oscillation Shock Sensor

naskar_ayan@srmap.edu.in → SW 420
→ ATXL 365

veerakumarr_v@srmap.edu.in → **Road surface anomaly detection process.**
There are different kinds of sensor data that can be obtained from smartphone sensors.
Motion sensor types include:
Accelerometer, gyroscope, linear accelerometer, and rotation.
Position sensor types include:
GPS, manometer, and rotation.

We will see this all Sensors and Implement the Best

SENSORS

Sensor types
Road condition sensors are either active or passive and include embedded pavement sensors, remote surface sensors, and vehicle mounted sensors.

ACTIVE AND PASSIVE

Active sensors require an external power source, sometime described as excitation voltage. They may be hard-wired or battery operated.

Passive sensors detect changes in the surrounding environment and output a signal that is proportional to the measured variable without the need for a power source. Examples of passive sensors include thermocouples. Thermocouples detect a thermally induced voltage that is produced by joining two dissimilar conductors with opposing thermoelectric coefficients.

Embedded Pavement Sensor

Embedded pavement sensor are typically fixed-in-pace in a drilled cavity using a quick setting epoxy. The sensor face is designed to endure harsh conditions in high traffic areas. The sensor face mounts flush to the pavement surface and measures near surface environmental properties.

Vehicle mounted sensors, like remote surface sensors, rely on remote sensing techniques to detect near surface environmental properties. Vehicle mounted sensors allow operators to survey larger areas and detect real-time surface conditions.

MATERIAL'S USED

- RASPBERRY – PI
- ARDUINO – UNO
- ADXL – 345

By Using These we have Done Real Simulation

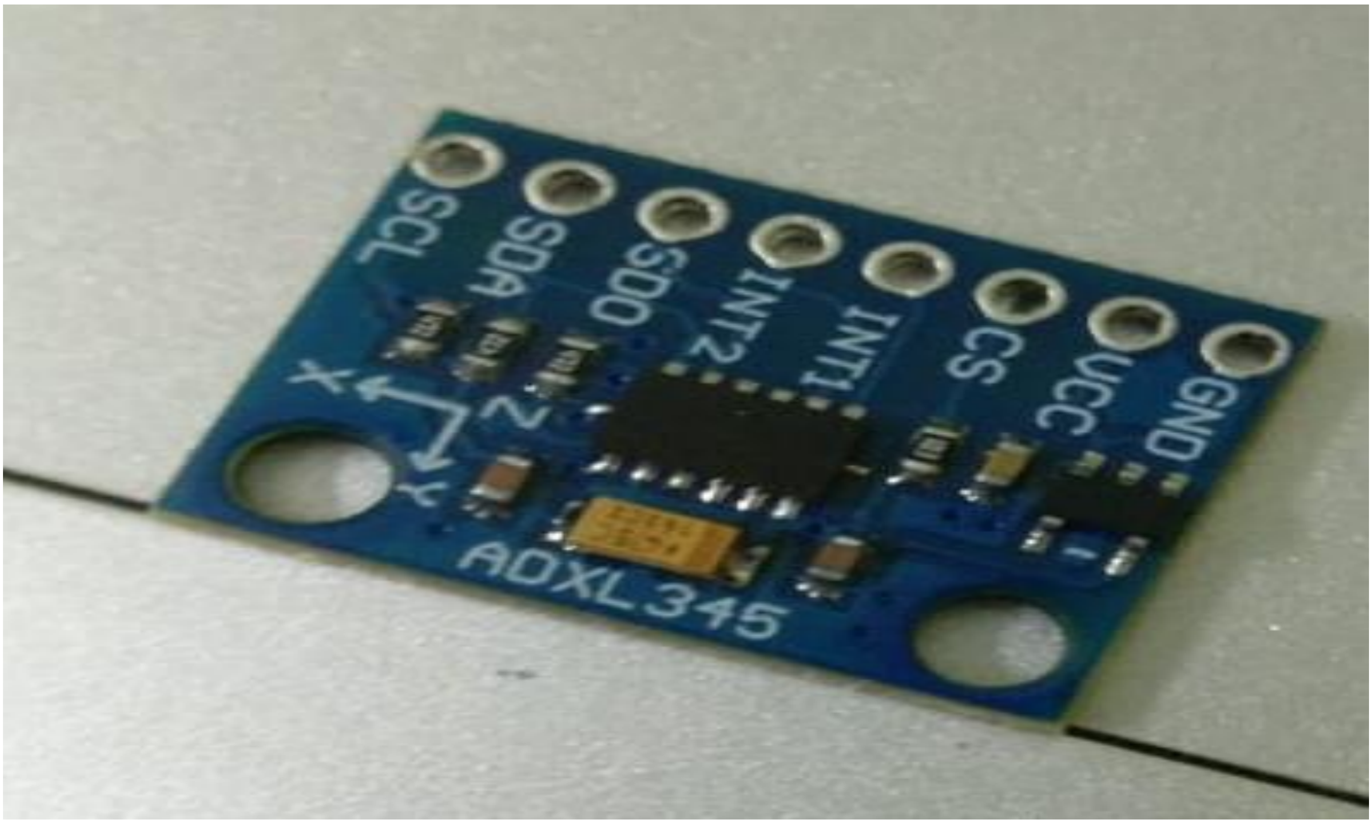
ADDITIONAL MATERIAL'S USED

- RIDE HEIGHT LASER SENSOR
- LVDT SENSOR
- ULTRASONIC HC-SR 04 SENSOR
- SHOCK SENSOR
- SW 420
- ATXL – 365
- GYROSCOPE
- GPS
- MANO METER SENSOR

ADXL - 345

ACCELEROMETER SENSOR

ADXL-345 IC from Analog Devices is the brain of this module.
The ADXL-345 is a small, thin, low power, complete 3-axis accelerometer with signal conditioned voltage outputs. The product measures acceleration with a minimum full-scale range of ± 16g.
ADXL-335 Accelerometer is an analog accelerometer therefore it works on the principle of capacitive sensing. In capacitive sensing accelerometer, when it is moved in any direction then its capacitance is changed.
When this capacitance is changed then its analog voltages are changed which is sensed by its interfacing controller.



GPS – MODULE

The GPS Position Sensor uses the Global Positioning System.
The System consistes of more than 24 satellites that broadcasts ranging signals and other necessary data.
For the Position Sensor to determine its own position and velocity, it must receive data from atleast three satelllites.
The satellites broadcasts ranging signals and necessary data for the Position Sensor to determine its own position and velocity as we already seen above.
The Position Sensor only receives data from the satellites; it doesn't transmit any signal.
GPS modules contain tiny processors and antennas that directly receive data sent by satellites through dedicated RF frequencies.
From there, it'll receive timestamp from each visible satellites, along with other pieces of data.

GPS - ERROR

GPS provides the longitude and longitude values of a location.
It is used to detect the location of potholes to users. It has an error of 3.3 meters .
This error must be minimized to get the exact location where events to be detected have occurred.
It may also be possible to miss some GPS data in urban areas among tunnels and tall buildings. A method to minimize the localization error is still an open problem.

Implementation on Arduino Board

CODES WITH ITS APPLICATION AND DESCRIPTION

We have Implemented Two Codes

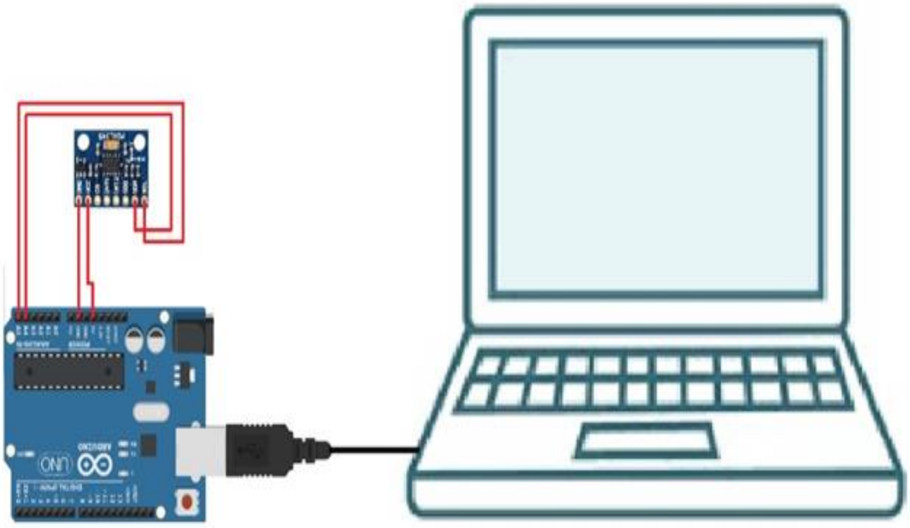
CODE - 1

```
#include <Wire.h>  //Accessing data from wire library to use I2C device//
int ADXL345 = 0x53; //Declaring variable ADXL345 as 0x53 which also the address of
sensor
int X_out, Y_out, Z_out;
void setup()
 {

Serial.begin(9600);
 Wire.begin();
 Wire.beginTransmission(ADXL345);
 Wire.write(0x2D); //Wire write is used to access the address of resistors of the I2C
device
 Wire.write(8); //To set the registers of the I2C device high.
 Wire.endTransmission();
 delay(10);
}

void loop()
{
Wire.beginTransmission(ADXL345); //Called the previously declared ADXL 345 function
Wire.write(0x32); //Used to write in the registor of address 0x32
Wire.endTransmission(false);
Wire.requestFrom(ADXL345, 6, true);
//Using Wire.read() twice as I2c device used give data in 13 bit format and Arduno can
process inly 8 bit at a time
X_out = ( Wire.read() | Wire.read() << 8);
Y_out = ( Wire.read() | Wire.read() << 8);
Z_out = ( Wire.read() | Wire.read() << 8);
```

```
//Printing the out put in serial monitor to further analyse the data

Serial.print(X_out);
Serial.print(" ");
Serial.print(Y_out);
Serial.print(" ");
 Serial.println(Z_out);
 delay(100);
}
```

APPLICATION AND DESCRIPTION

ADXL 345 is an I2C device and to controll I2C devices in arduino we get the necessary datas from wire library.
A variable ADXL 345 is decleared with a hex value Ox53 which is also the adress of this I2C device.
To run any I2C device we have to write in the register we make the use of 6 register in total of ADXL 345 here Ox2D and 0x32 used in the code are the adress of two of those registers.8 is (0000 1000) which is to measure high ,after this we get our device in measure mode.
Now the accelerometer gives the reading of acceleration of three axis in 13 bit, but arduino can process 8 bit data thats why we send the data in two goes,
We print the output of the data given by ADXL345 in the format of x,y,z. whose 1 unit reading is equivalent to 1g(9.81m/ss).



CONNECTIONS:-

ADXL -345	ARDUINO UNO
VCC	5V
GND	GND
SDA	A4
SCL	A5

WRITTEN OUT PUT

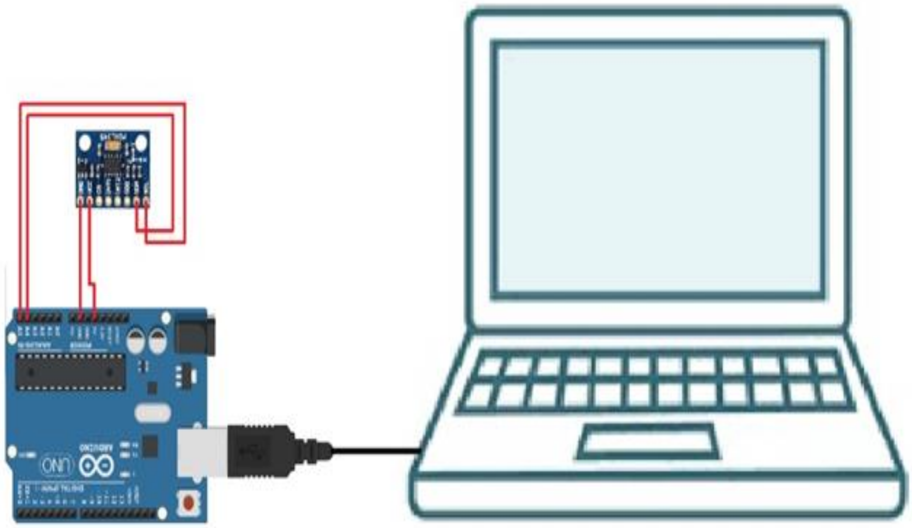
14:31:31.658 -> 1 1 1
14:31:31.761 -> 0 0 0
14:31:31.863 -> -2 0 1
14:31:31.964 -> 0 0 1
14:31:32.065 -> -2 -2 0
14:31:32,168 -> 1 1 1
14:31:32,272 -> -1 0 0
14:31:32.376 -> -1 1 1
14:31:32.478 -> 0 0 1
14:31:32,582 -> -2 -2 1
14:31:32,686 -> 1 1 1
14:31:32.788 -> -1 0 1
14:31:32.891 -> -2 1 1
14:31:32.993 -> 0 0 1
14:31:33.097 -> -2 -2 1
14:31:33.199 -> -1 1 1
14:31:33.301 -> -2 -1 0
14:31:33.403 -> 0 1 1
14:31:33.507 -> 0 0 1
14:31:33.609 -> -2 -2 1
14:31:33.677 -> -1 1 1
14:31:33.813 -> -1 -1 0
14:31:33,914 -> 0 1 1
14:31:33.983 -> 0 1 0
14:31:34,087 -> -2 0 1
14:31:34,188 -> -1 0 1
14:31:34.291 -> -1 -2 1
14:31:34,393 -> 1 1 1
14:31:34,494 -> 0 0 0
14:31:34,598 -> -2 1 1
14:31:34.701 -> 0 0 1

CODE – 2

```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_ADXL345_U.h>
Adafruit_ADXL345_Unified accel = Adafruit_ADXL345_Unified();
void setup()
{
  Serial.begin(9600);
  if (!accel.begin())
  {
    Serial.println("No valid sensor found");
    while (1);
  }
}
void loop()
{
  sensors_event_t event;
  accel.getEvent(&event);
  Serial.print("X: "); Serial.print(event.acceleration.x); Serial.print("");
  Serial.print("Y: "); Serial.print(event.acceleration.y); Serial.print("");
  Serial.print("Z: "); Serial.print(event.acceleration.z); Serial.print("");
  Serial.println("m/s^2 ");
  delay(500);
}
```

APPLICATION AND DESCRIPTION

For this code unified Driver has been enabled to access the necessary data from Adafruit_Sensor.h,Adafruit_ADXL345_U.h library to use accelerometer sensor.sensor is benig identified using accel.begin function .Inside the loop we print the acceleration values of respective x ,y,z axis .We print the output of the data given by ADXL345 in the format of x,y,z. whose 1 unit reading is equivalent to 1g(9.81m/ss).



CONNECTIONS:-

ADXL -345	ARDUINO UNO
VCC	5V
GND	GND
SDA	A4
SCL	A5

WRITTEN OUT PUT

22:21:29.793 -> X: 10.59 Y: -0.31 Z: -2.24m/s*2
22:21:30.254 -> X: 11.26 Y: -0.24 Z: -2.39m/s*2
22:21:30.775 -> X: 10.59 Y: 0.27 Z: -1.33m/s*2
22:21:31.276 -> X: -0.04 Y: -0.04 Z: -0.04m/s*2
22:21:31.895 -> X: -0.04 Y: -0.04 Z: -0.04m/s*2
22:21:32.398 -> X: -0.04 Y: -0.04 Z: -0.04m/s*2
22:21:32.887 -> X: -0.04 Y: -0.04 Z: -0.04m/s*2
22:21:33.411 -> X: 10.04 Y: 2.002 Z: -0.39m/s*2
22:21:33.909 -> X: -0.04 Y: -0.04 Z: -0.04m/s*2
22:21:35.483 -> X: -0.04 Y: -0.04 Z: -0.04m/s*2
22:21:36.337 -> X: -0.04 Y: -0.04 Z: -0.04m/s*2
22:21:36.841 -> X: 10.00 Y: 3.37 Z: -2.51m/s*2
22:21:37.431 -> X: 8.51 Y: 0.90 Z: 2.00m/s*2
22:21:38.016 -> X: -0.04 Y: -0.04 Z: -0.04m/s*2
22:21:38.615 -> X: 0.00 Y: 2.90 Z: 3.02m/s*2
22:21:39.162 -> X: -0.04 Y: -0.04 Z: -0.04m/s*2
22:21:39.201 -> X: 9.81 Y: -3.02 Z: 3.61m/s*2
22:21:39.734 -> X: -0.04 Y: -0.04 Z: -0.04m/s*2
22:21:40.240 -> X: -0.04 Y: -0.04 Z: -0.04m/s*2
22:21:40.725 -> X: 7.22 Y: -6.39 Z: 4.24m/s*2
22:21:41.297 -> X: 0.00 Y: 2.20 Z: 4.75m/s*2
22:21:41.495 -> X: 9.06 Y: 2.24 Z: 2.00m/s*2
22:21:42.011 -> X: -0.04 Y: -0.04 Z: -0.04m/s*2
22:21:42.124 -> X: -0.04 Y: -0.04 Z: 2.71m/s*2
22:21:42.608 -> X: -0.04 Y: -0.04 Z: -0.04m/s*2
22:21:43.113 -> X: -0.04 Y: -0.04 Z: -0.04m/s*2
22:21:43.634 -> X: 10.83 Y: 4.392 Z: -0.24m/s*2
22:21:44.165 -> X: -0.04 Y: -0.04 Z: -0.04m/s*2
22:21:44.647 -> X: -0.04 Y: -0.04 Z: -0.04m/s*2

OUT PUT RESOLUTION

ADXL-345 supports output resolution of 10-bit for +/- 2g measurement range, 11-bit for +/- 4g, 12-bit for +/- 8g and 13-bit for +/- 16g.

The default resolution is 10-bit for all measurement ranges.

SENSITIVITY

With the default resolution of 10-bit, the ADXL345 has typical sensitivity of 3.9 mg/LSB for default measurement range (i.e. +/- 2g), 7.8 mg/LSB for +/- 4g, 15.6 mg/LSB for +/- 8g and 31.2 mg/LSB for +/- 16g measurement range. This means that ADXL345 with default 10-bit resolution selected can detect minimum change of acceleration of 3.822 cm/s^2 (3.9 * 9.8/1000 * 100) for +/- 2g, 7.64 cm/s^2 for +/- 4g, 15.28 cm/s^2 for +/- 8g, and 30.57 cm/s^2 for +/- 16g range.

SIMULATION VIDEOS

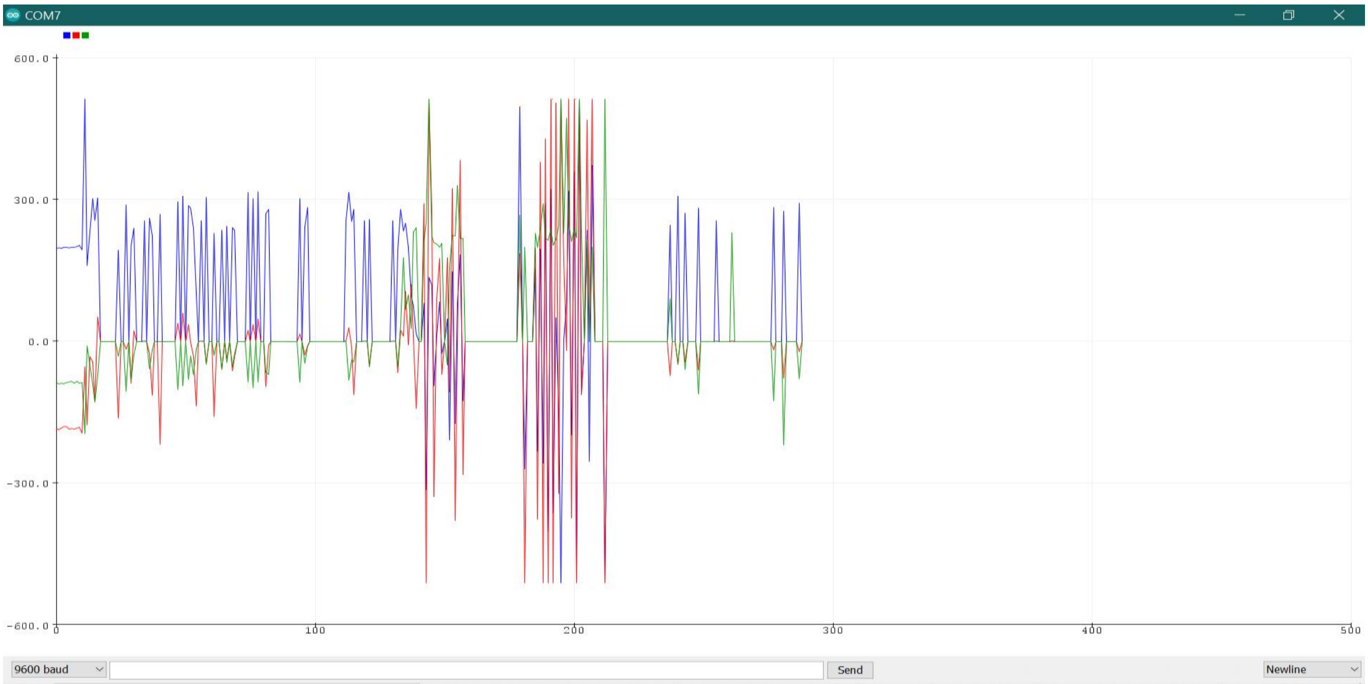
USING CODE-1 VALUES AND IMPLEMENTATION

https://drive.google.com/file/d/1CFOxQj7o0Alc-SLbgCvVO7b8YXDJleYM/view?usp=drivesdk

USING CODE-2 VALUES AND IMPLEMENTATION

https://drive.google.com/file/d/1UcXpfzrEBZpLL-MzdLLpq1pUNl4rkUx_/view?usp=drivesdk

GRAPHICAL REPRESENTATION



SIMULATION VIDEO LINK

https://drive.google.com/file/d/1j_EVoEoBiTNqoaxgIbzyysgU2Ye3--MU/view?usp=drivesdk

PROCEDURE AND WORKING PRINCIPLE

The entire project is controlled by Using the Arduino.
Download Arudino UNO from the browser.
Arduino is an open-source hardware and software company in this project.
Now take the Arduino Board and Place it on the Table then take the Jumper Wires and connect one end to the Arduino Board and the other end of the wire to the Accelerometer Sensor (ADXL-345) Connections are shown Below.

CONNECTING THE WIRES

ARDUINO - UNO		ADXL - 345
5 V	⟷	VCC
GND	⟷	GND
A 4	⟷	SDA
A 5	⟷	SCL

CONCLUSION

This project is a frugal method of gathering invaluable data as it uses rudimentary and compact devices such as Arduino Uno and ADXL345, the timestamps in the results provide useful information about the users motion, when a plethora of these devices are used, roads can be surveyed with ease and provide real time information about road health.

SWITCHING TO RASPBERRY – PI

The device when implemented in Arduino Uno serves its purpose but rudimentary devices have their limits, one such glaring limit is that Arduino requires a PC connected to it, also Arduino is an 8 bit microcontroller and the data sent by I2C device (ADXL345) is of 13 bits, hence the data must be sent from the sensor twice so that 8 bit microcontroller Arduino can process it.

These limitations hinder the project's application having understood this, our team simultaneously tried implementing this project in a **Raspberry Pi** computer which is an **SBC** (Single Board Computer) that uses a 64 bit processor, it was selected as it requires no additional computers and the data can be transferred wirelessly, the project when developed in Raspberry Pi provides versatility as better hardware can be used to gather data, further it allows the option of mounting the device directly onto smaller vehicles such as cycles and bikes.

The information gathered can be stored in cloud which when analysed meticulously can provide indispensable data such as traffic patterns and vehicular accidents and their frequencies in certain locations helping predict and prevent road hazards.

\

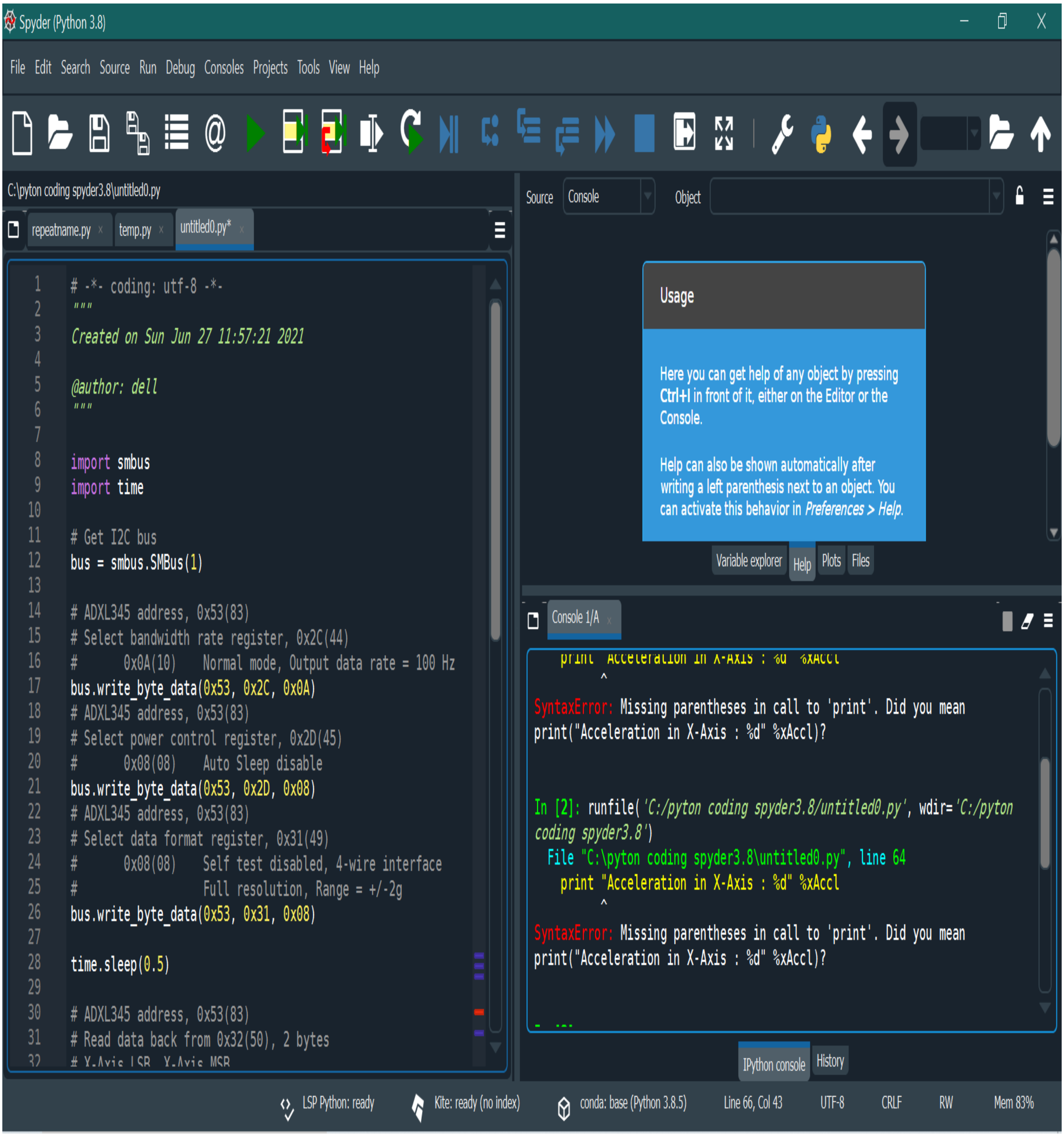
Implementation on Raspberry - Pi

IMPLEMENTING CODE IN PHYTON

```
import smbus
import time

# Get I2C bus
bus = smbus.SMBus(1)

# ADXL345 address, 0x53(83)
# Select bandwidth rate register, 0x2C(44)
#               0x0A(10)      Normal mode, Output data rate = 100 Hz
bus.write_byte_data(0x53, 0x2C, 0x0A)
# ADXL345 address, 0x53(83)
# Select power control register, 0x2D(45)
#               0x08(08)      Auto Sleep disable
bus.write_byte_data(0x53, 0x2D, 0x08)
# ADXL345 address, 0x53(83)
# Select data format register, 0x31(49)
#               0x08(08)      Self test disabled, 4-wire interface
#                                     Full resolution, Range = +/-2g
bus.write_byte_data(0x53, 0x31, 0x08)

time.sleep(0.5)

# ADXL345 address, 0x53(83)
# Read data back from 0x32(50), 2 bytes
# X-Axis LSB, X-Axis MSB
data0 = bus.read_byte_data(0x53, 0x32)
data1 = bus.read_byte_data(0x53, 0x33)

# Convert the data to 10-bits
xAccl = ((data1 & 0x03) * 256) + data0
if xAccl > 511 :
        xAccl -= 1024
```

```
# ADXL345 address, 0x53(83)
# Read data back from 0x34(52), 2 bytes
# Y-Axis LSB, Y-Axis MSB
data0 = bus.read_byte_data(0x53, 0x34)
data1 = bus.read_byte_data(0x53, 0x35)

# Convert the data to 10-bits
yAccl = ((data1 & 0x03) * 256) + data0
if yAccl > 511 :
	yAccl -= 1024

# ADXL345 address, 0x53(83)
# Read data back from 0x36(54), 2 bytes
# Z-Axis LSB, Z-Axis MSB
data0 = bus.read_byte_data(0x53, 0x36)
data1 = bus.read_byte_data(0x53, 0x37)

# Convert the data to 10-bits
zAccl = ((data1 & 0x03) * 256) + data0
if zAccl > 511 :
	zAccl -= 1024

# Output data to screen
print "Acceleration in X-Axis : %d" %xAccl
print "Acceleration in Y-Axis : %d" %yAccl
print "Acceleration in Z-Axis : %d" %zAccl
```

PYTHON PROGRAM



As we are not having the availability of so many sensors and also Raspberry – Pi we have Tried with Arduino – UNO
Also we don’t have a Proper Simulation Links To do the Complete work in Raspberry – Pi.
As we can do it if we can get the required materials and Objects needed

END USER OF THIS PROJECT

Now a days Transport Facilities and Transportation Increases Rapidly Every Family as a Two wheeler (Bike or a Cycle) or Four Wheeler (Car or Van) Etc..
So this was a very Easy and Usefull as well, We can just fix the Board to it and the data will automatically transfers to the Road Management Systems

FUTURE SCOPE

Sharing data in Social media is also the major advantage because it is viewed by government authorities.
The maintenance of road becomes easy and accidents can be avoided.

In future, by using various algorithms for the development of self calibration functionality for further analyzing and quantifying the distress in an engineering manner.

PRECAUTIONS

- Carefull While Handling Accelerometer Senor and other Sensors
- Connect the Jumper Wires Properly at the both Ends
- Be Carefull in Data Privacy
- Attach or Fix the Device Carefully to the Vehicle so it may not Deviate or fall from the Vehicles

CONCLUSION

This study proposes a real-time pothole detection.
Roads are needed to be monitored continuously for roughness and other anomalies to avoid inconvenience to the road users.
Road monitoring can also help to predict the estimated arrival time from one place to another.
This Project presents a detailed survey of methods for detecting road conditions.
Form the survey, it is noted that the most commonly used sensors are accelerometer and GPS which we can Implement them by using Arduino Board or Raspberry – Pi