REALTIME POTHOLE DETECTION

17BCE018

17BCE019

Why Pothole Detection ?

Dangerous road surface conditions are major distractions for safe and comfortable transportation.

To ensure road surface quality it should be monitored continuously and repaired as necessary.

PROJECT REQUIREMENTS

- Detect potholes in real time
- Use a generic Android OS based smart-phone that has built-in accelerometer sensors
- The system should be able to run on different smartphone models with different parameters
- Utilization of all smart-phone resources for pothole detection only is not acceptable
- System should have self-calibration functionality

HOW TO DETECT POTHOLES?

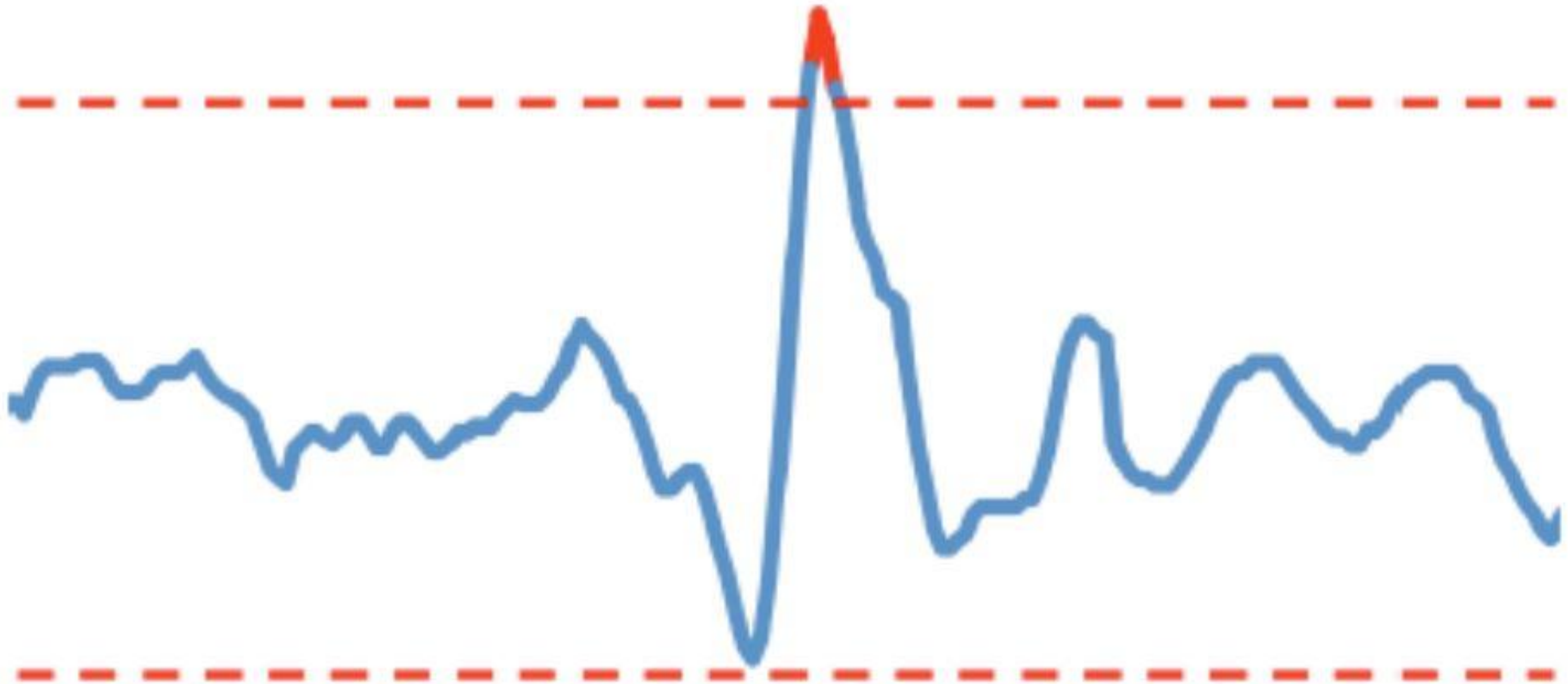
IMAGE DETECTION	SENSOR DETECTION
MORE ACCURATE	LESS ACCURATE
VERY HIGH RESOURCE REQUIREMENT	LOW RESOURCE REQUIREMENT

METHODS OF POTHOLE DETECTION

- Z-THRESHOLD
- Z-DIFFERENCE
- STANDARD DEVIATION
- G-ZERO

Z-threshold method

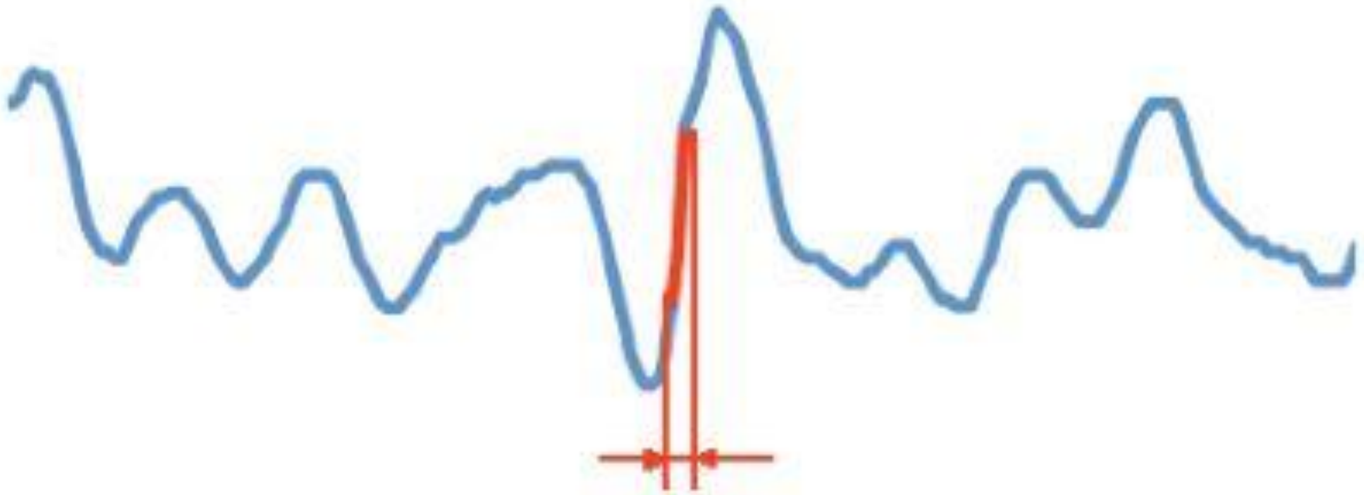
In this method we decide a threshold of acceleration reading on the z axis and when the value crosses the threshold pothole there is a possibility of a pothole.



Z- THRESHOLD

Z-difference method

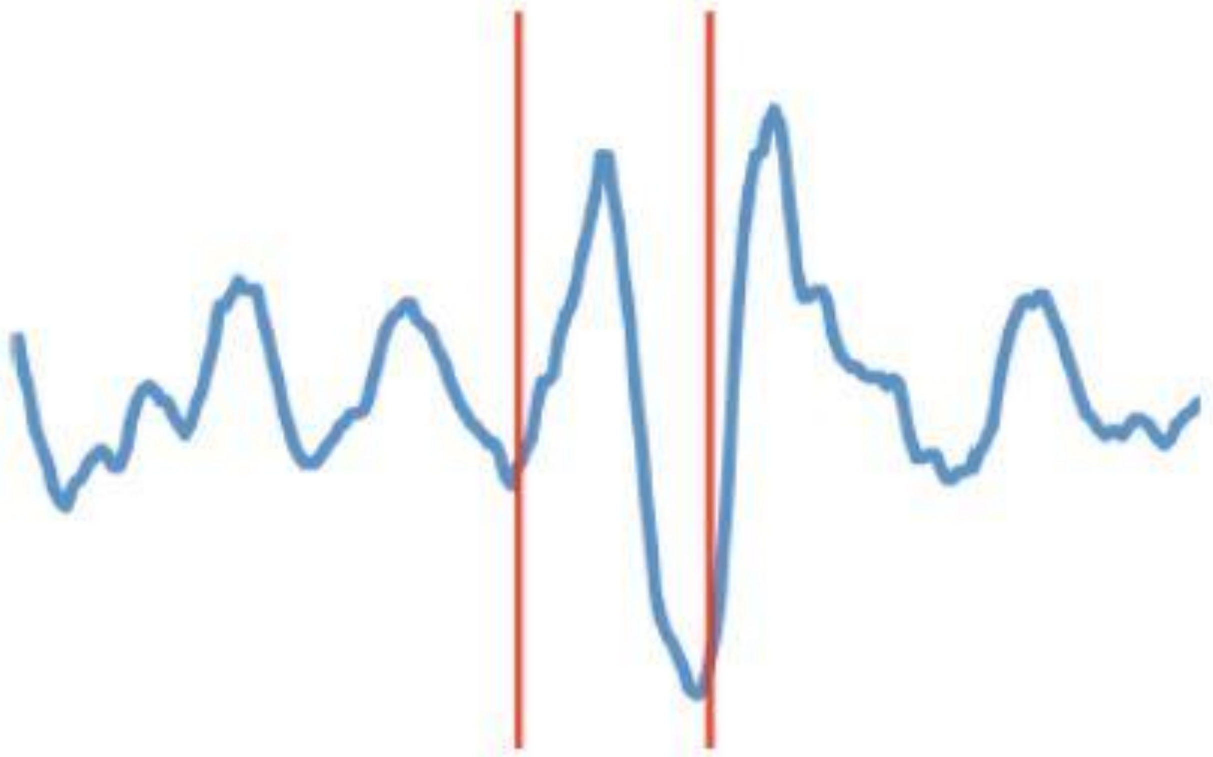
In this method we decide an threshold difference of acceleration along z axis between two consecutive readings and when the value of difference crosses the threshold pothole is predicted.



Z-DIFFERENCE

Standard-deviation method

In this method we decide a window of time , and for that window we calculate the standard deviation of all the acceleration readings and when the value crosses the threshold for standard deviation pothole is predicted.

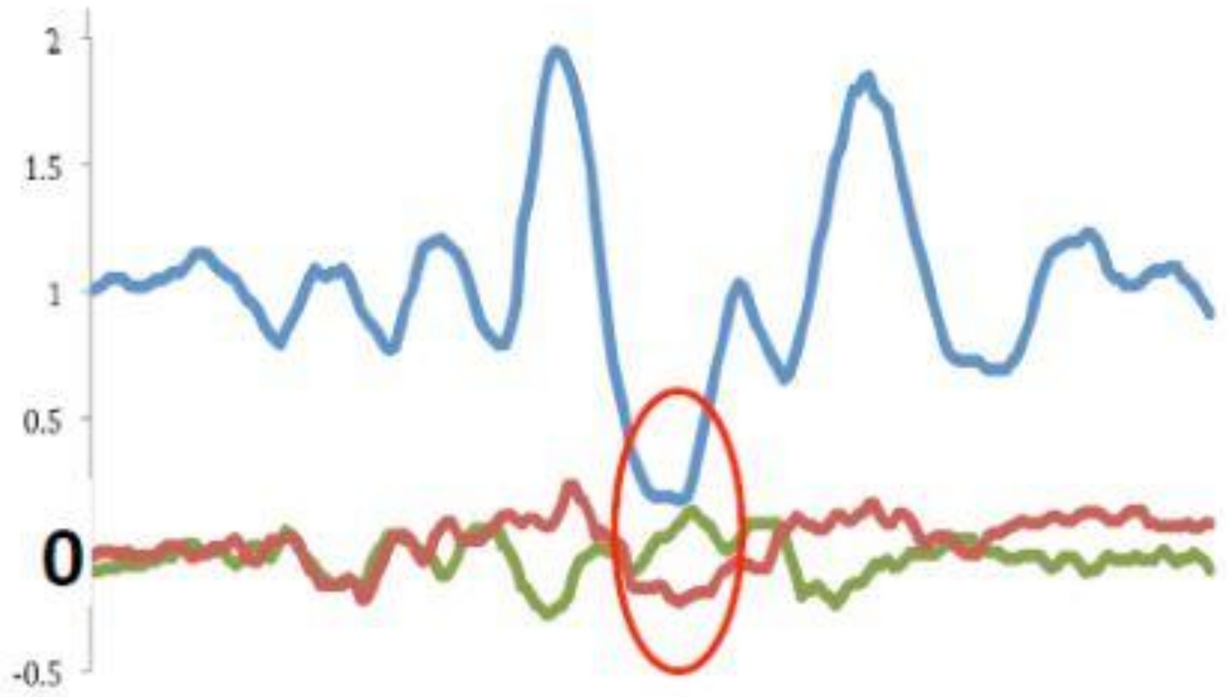


STANDARD DEVIATION

G-zero method

When the vehicle is entering or leaving pothole it is in temporary free fall due to which the accelerometer readings become zero.

So when readings of all axes are near to zero we predict a pothole.

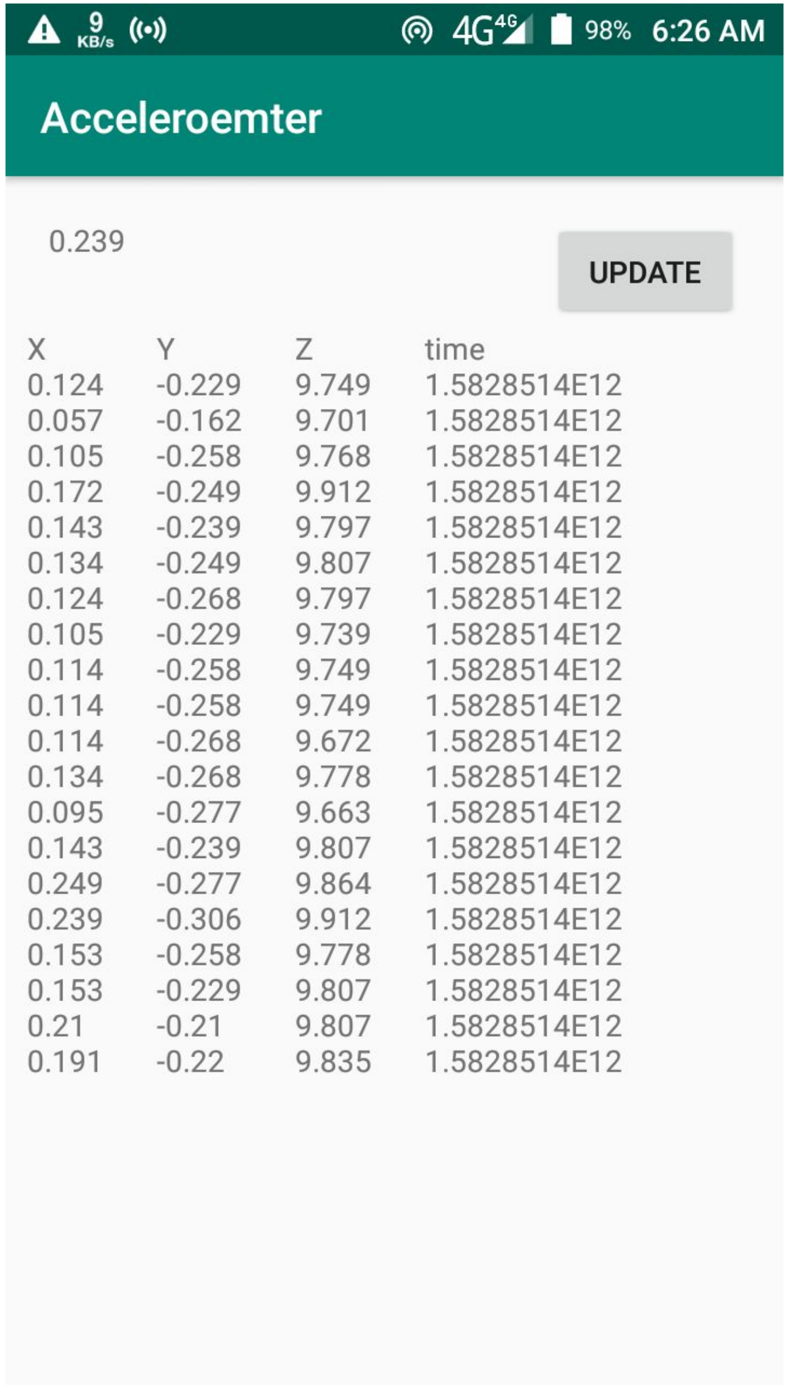


G-ZERO

Data From Accelerometer

Last 20 readings are shown when update button is clicked.

Device was kept still so there are no major changes in the values of readings from accelerometer.

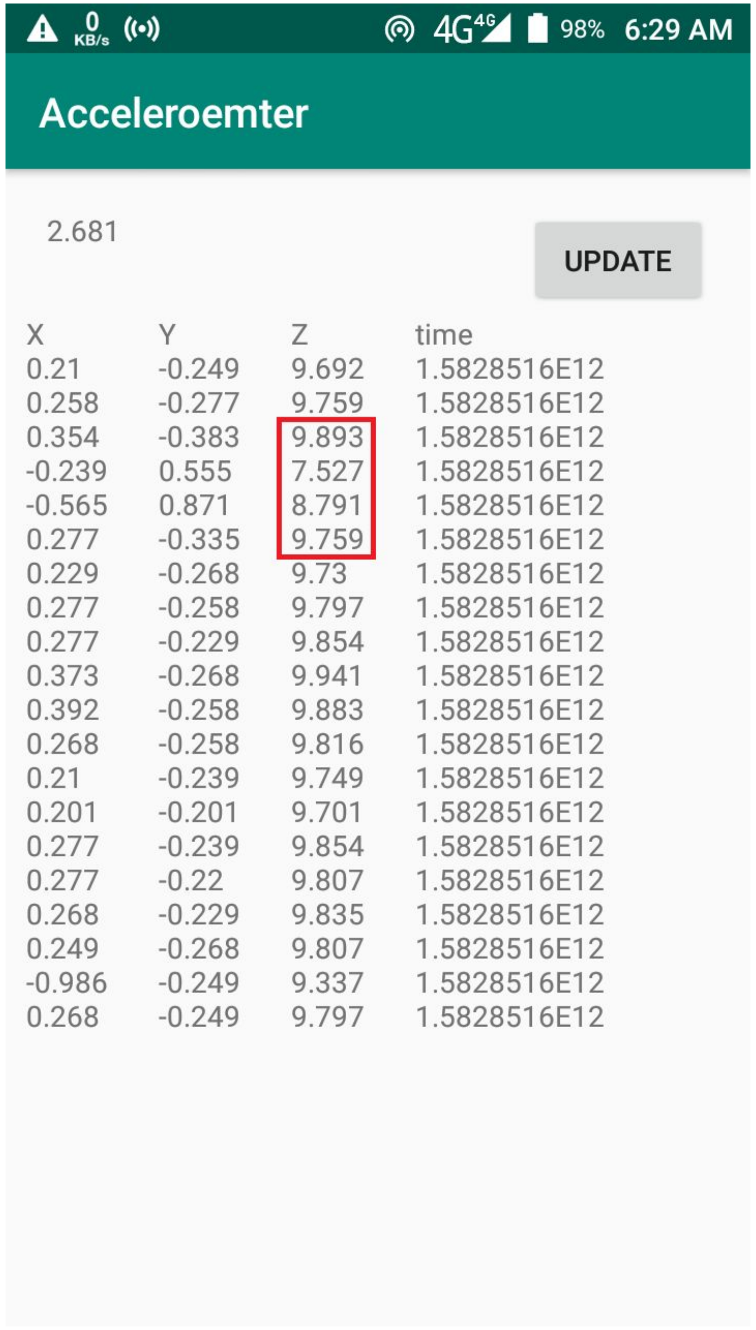


0.239

X	Y	Z	time
0.124	-0.229	9.749	1.5828514E12
0.057	-0.162	9.701	1.5828514E12
0.105	-0.258	9.768	1.5828514E12
0.172	-0.249	9.912	1.5828514E12
0.143	-0.239	9.797	1.5828514E12
0.134	-0.249	9.807	1.5828514E12
0.124	-0.268	9.797	1.5828514E12
0.105	-0.229	9.739	1.5828514E12
0.114	-0.258	9.749	1.5828514E12
0.114	-0.258	9.749	1.5828514E12
0.114	-0.268	9.672	1.5828514E12
0.134	-0.268	9.778	1.5828514E12
0.095	-0.277	9.663	1.5828514E12
0.143	-0.239	9.807	1.5828514E12
0.249	-0.277	9.864	1.5828514E12
0.239	-0.306	9.912	1.5828514E12
0.153	-0.258	9.778	1.5828514E12
0.153	-0.229	9.807	1.5828514E12
0.21	-0.21	9.807	1.5828514E12
0.191	-0.22	9.835	1.5828514E12

Data From Accelerometer

Device was allowed to fall by a height of 10-15 cms, and noticeable variation can be seen in the readings of Z-axis as shown in red box.

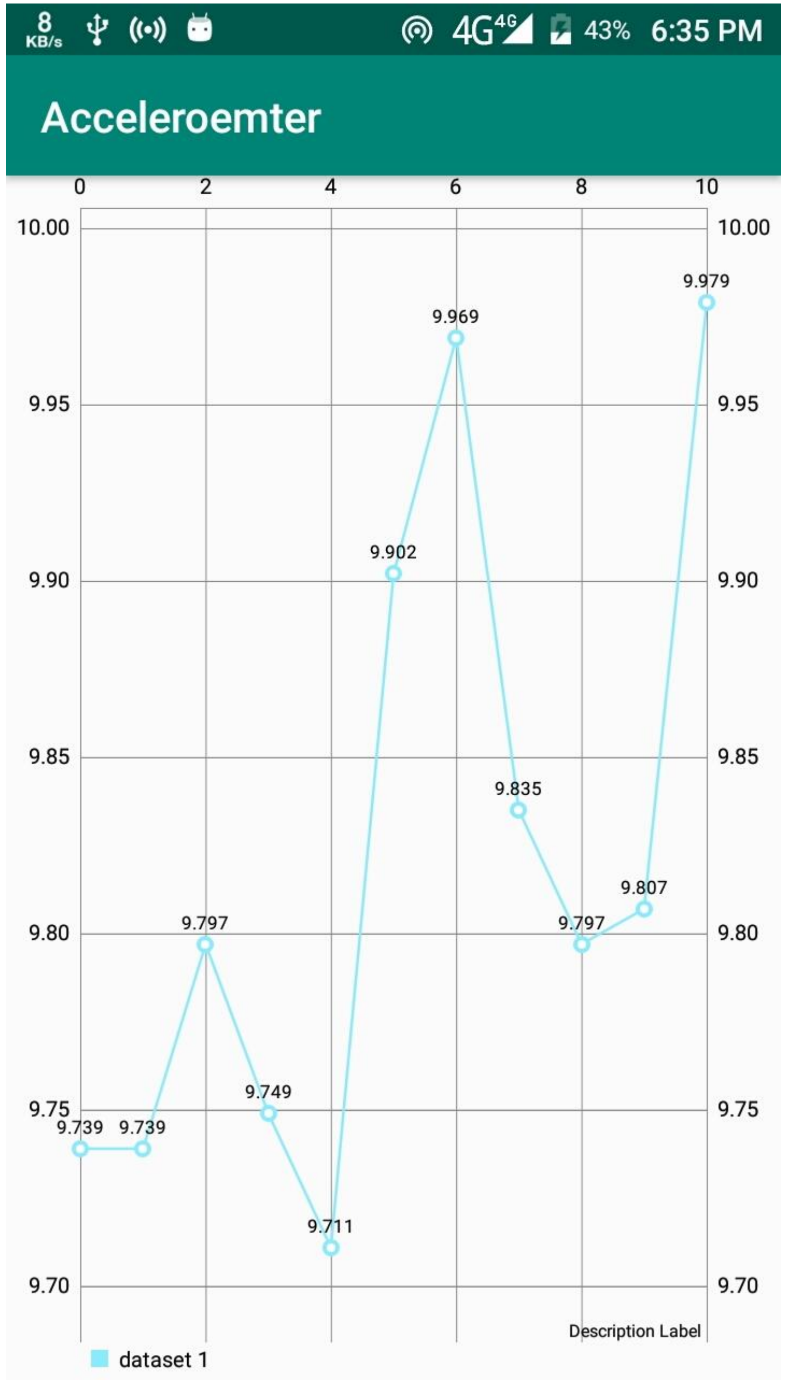


2.681

X	Y	Z	time
0.21	-0.249	9.692	1.5828516E12
0.258	-0.277	9.759	1.5828516E12
0.354	-0.383	9.893	1.5828516E12
-0.239	0.555	7.527	1.5828516E12
-0.565	0.871	8.791	1.5828516E12
0.277	-0.335	9.759	1.5828516E12
0.229	-0.268	9.73	1.5828516E12
0.277	-0.258	9.797	1.5828516E12
0.277	-0.229	9.854	1.5828516E12
0.373	-0.268	9.941	1.5828516E12
0.392	-0.258	9.883	1.5828516E12
0.268	-0.258	9.816	1.5828516E12
0.21	-0.239	9.749	1.5828516E12
0.201	-0.201	9.701	1.5828516E12
0.277	-0.239	9.854	1.5828516E12
0.277	-0.22	9.807	1.5828516E12
0.268	-0.229	9.835	1.5828516E12
0.249	-0.268	9.807	1.5828516E12
-0.986	-0.249	9.337	1.5828516E12
0.268	-0.249	9.797	1.5828516E12

Plot of Readings

This plot shows the reading taken from the accelerometer in real time.



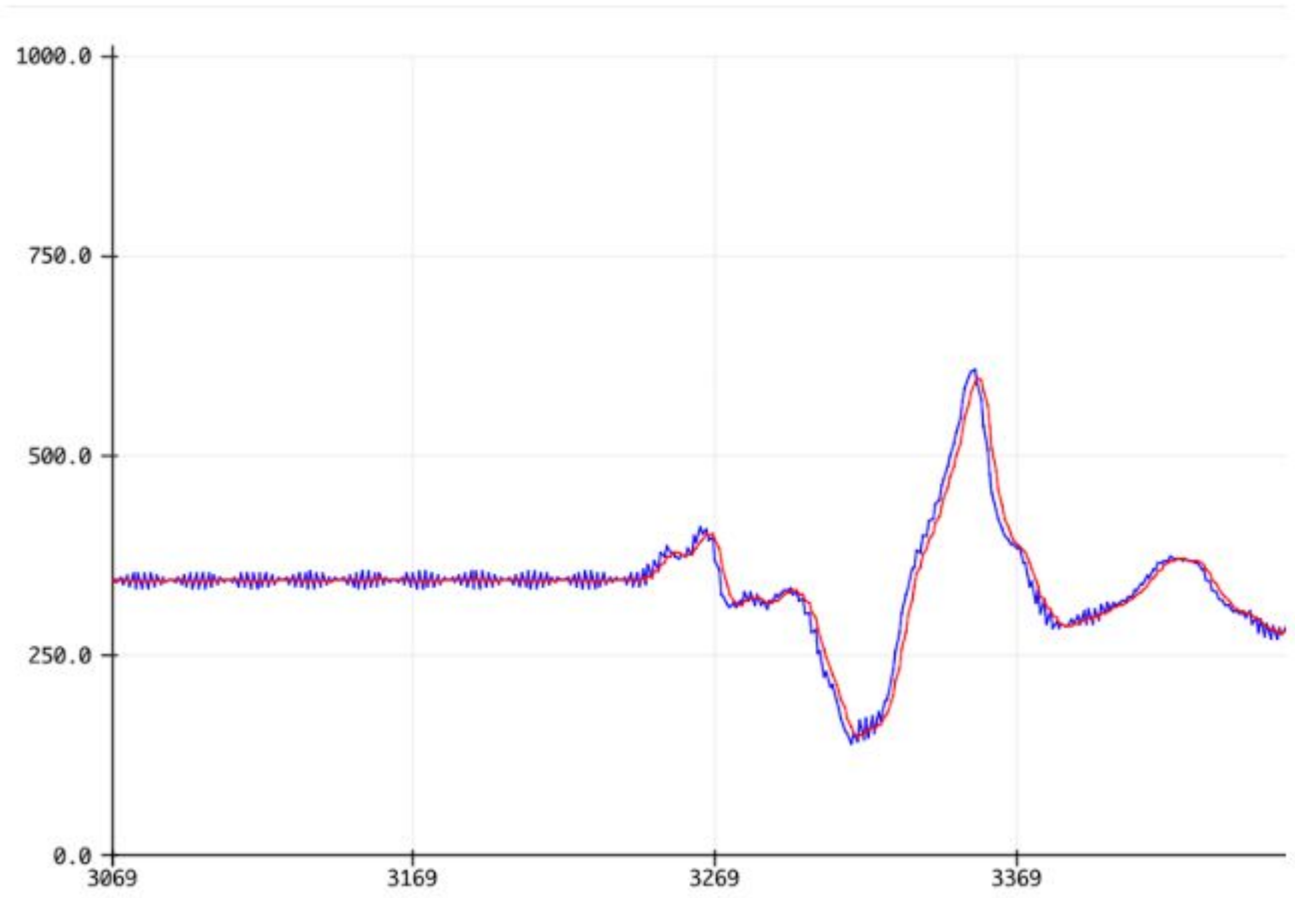
Data Preprocessing

Data obtained from sensors is noisy and cannot be used directly for analysis and prediction of potholes, thus preprocessing of data is necessary. In preprocessing data from sensors is properly formatted and smoothed, the noises from data are filtered out.

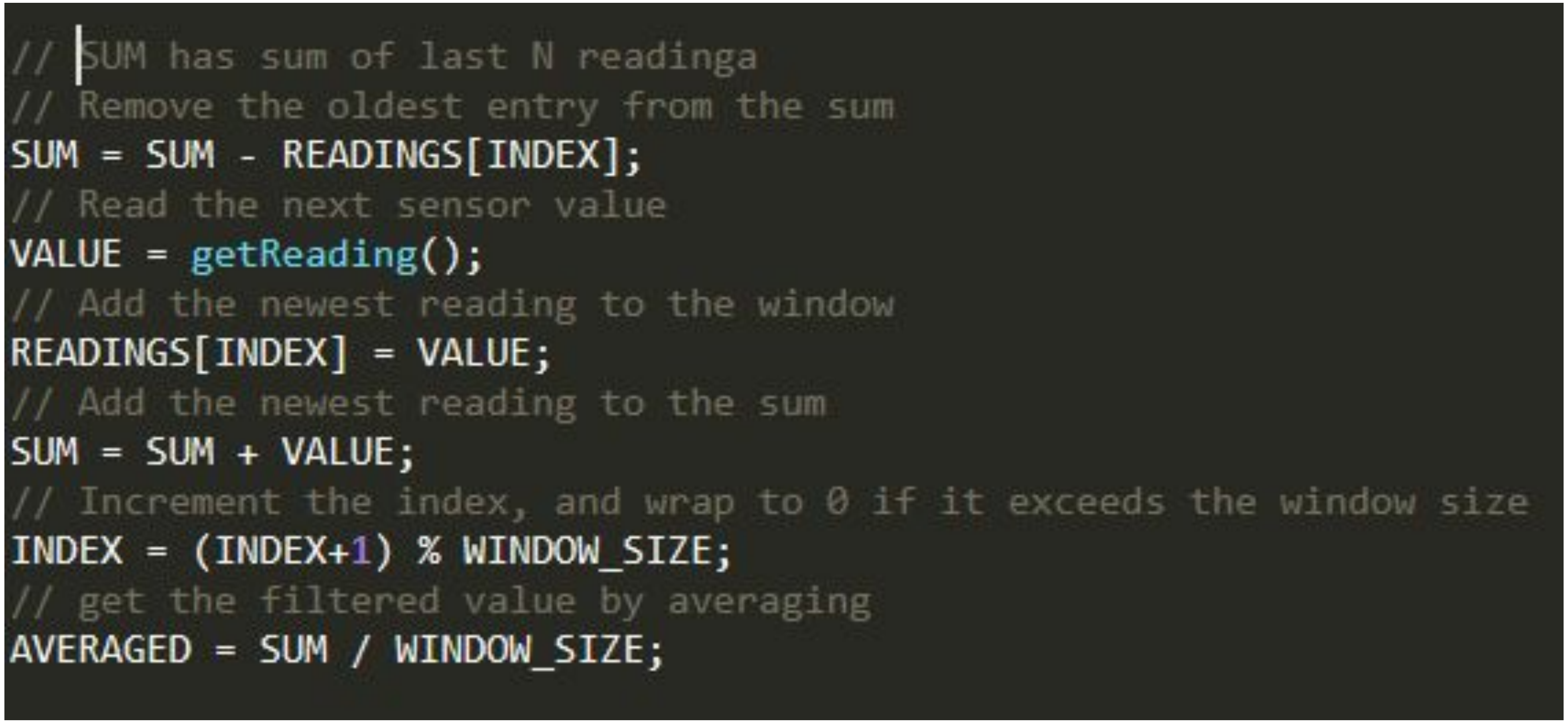
We used moving average filtering for noise removal and euler angles for data reorientation.

Moving Average Filtering

Blue graph represents input noisy data, and after applying moving average filtering we get the smoothed and noise reduced data represented by red graph.



Moving Average Filtering



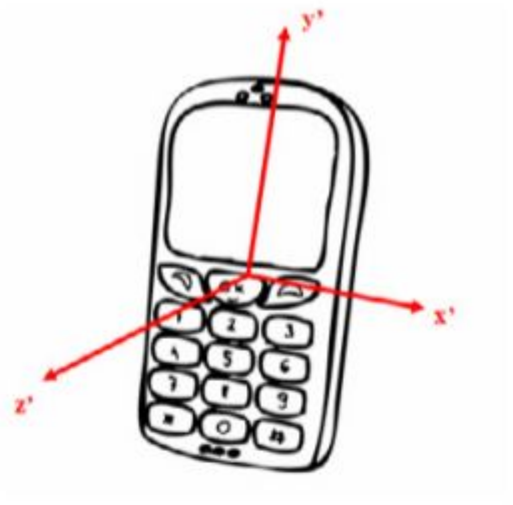
```
// SUM has sum of last N readinga
// Remove the oldest entry from the sum
SUM = SUM - READINGS[INDEX];
// Read the next sensor value
VALUE = getReading();
// Add the newest reading to the window
READINGS[INDEX] = VALUE;
// Add the newest reading to the sum
SUM = SUM + VALUE;
// Increment the index, and wrap to 0 if it exceeds the window size
INDEX = (INDEX+1) % WINDOW_SIZE;
// get the filtered value by averaging
AVERAGED = SUM / WINDOW_SIZE;
```

Eulerian reorientation

In order to determine the accelerations undergone by the vehicle when there are potholes, the accelerometer must detect what happens in the direction perpendicular to the vehicle.

the x axis identifies the longitudinal direction, the y axis the transverse one and the z axis the perpendicular direction to the xy plane

To detect road anomalies the z axis direction must correspond with the z' axis direction. If this condition occurs, the accelerometer is well oriented, otherwise it is not well oriented and it must be reoriented.



Calculating Euler angles

For vehicle at rest the values of acceleration are

$$a_x = 0\,m/s^2;\;\; a_y = 0\,m/s^2;\;\; a_z = 9.81\,m/s^2 = 1g$$

alpha and beta are the euler angles

$$\alpha = tan^{-1}\left(a_y' / a_z'\right) \qquad \beta = tan^{-1}\left(-a_x' / \left(\sqrt{(a_y')^2 + (a_z')^2}\right)\right)$$

Calulating reoriented acceleration

Now using the euler angle and unoriented values of acceleration, oriented values of acceleration can be found using below equations, where c represents cosine of angle and s represents the sine of angle.

$$a_{xreor} = c\beta a_x' + s\beta s_\alpha a_y' + c_\alpha s\beta a_z';$$

$$a_{yreor} = c_\alpha a_y' - s_\alpha a_z';$$

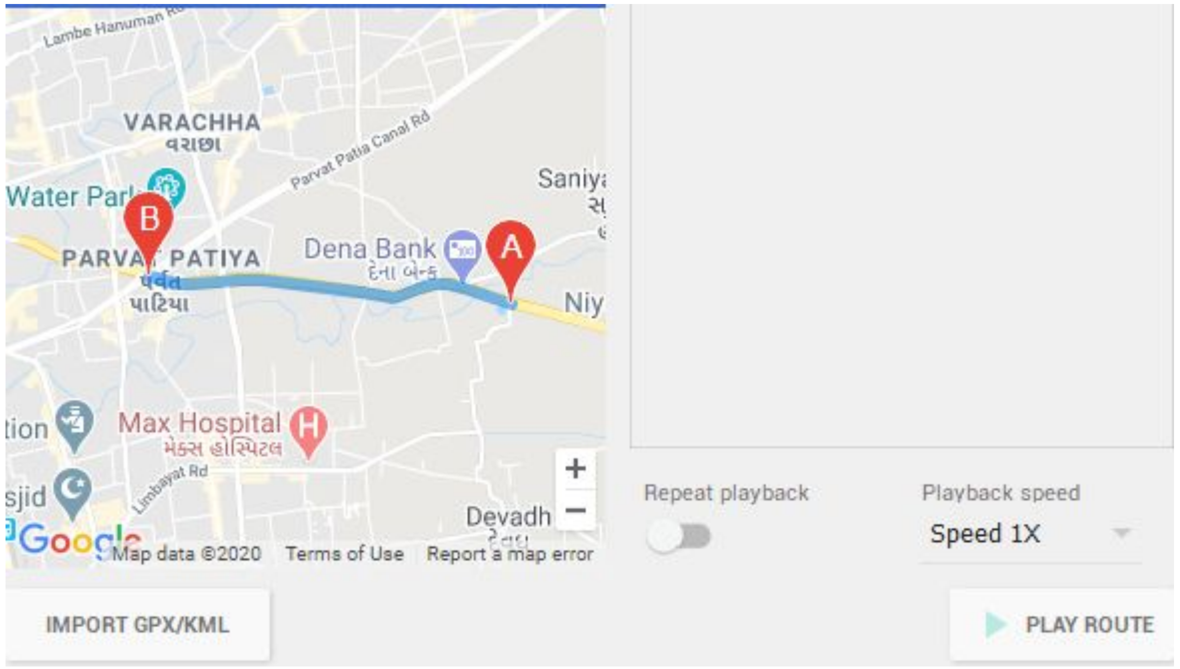
$$a_{zreor} = -s\beta a_x' + c\beta s_\alpha a_y' + c\beta c_\alpha a_z'$$

Virtual traversal

- To virtually travel with a vehicle, we used android studio's inbuilt facility of going from source A to destination B.

- We can control the speed of the vehicle dynamically to create proper effect of any pothole, crack or bump.

Application Description

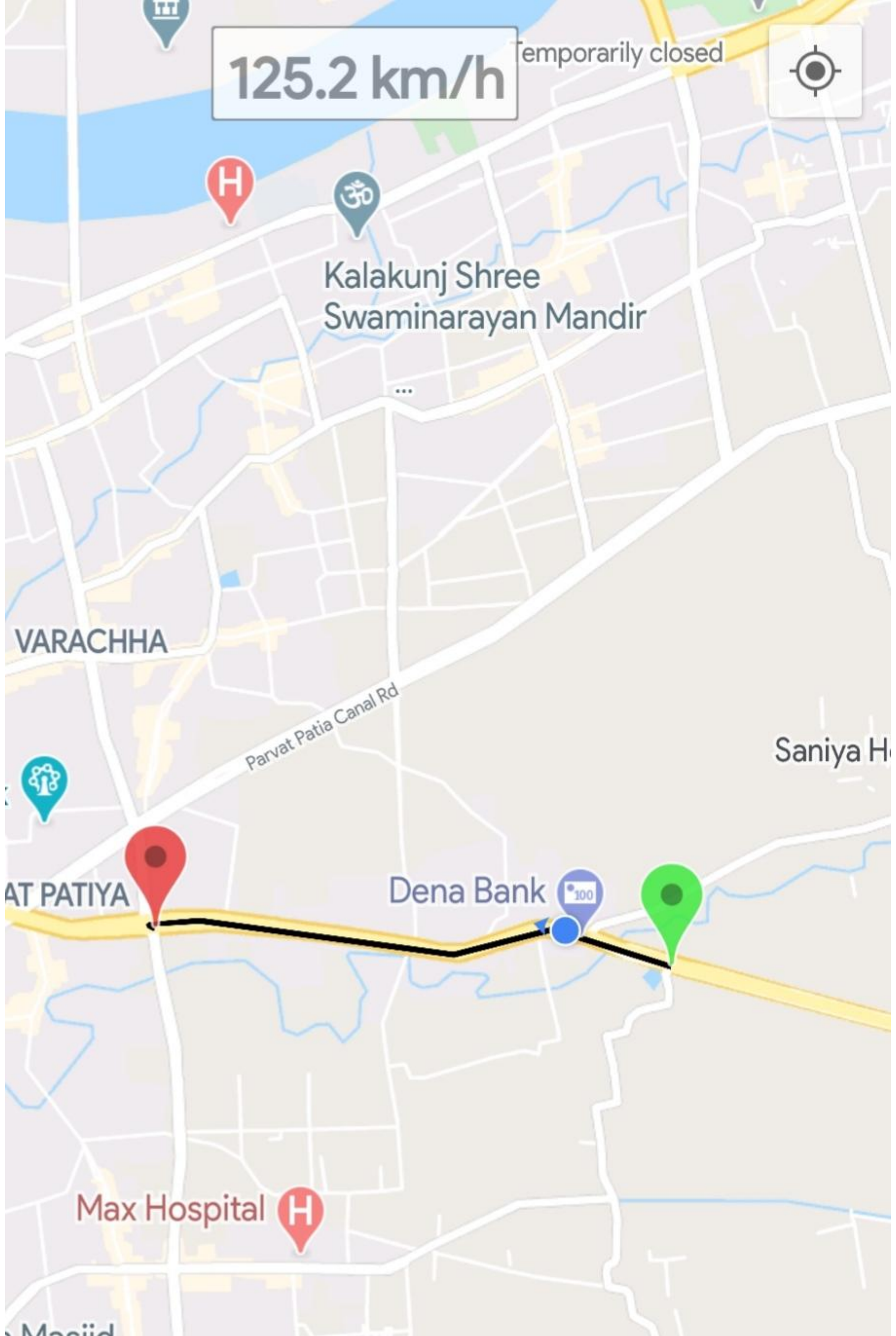


Application interface in traversal

- As we can see in the image, we have marked the source(yellow marker), destination(red marker) and route between them(black polylines).

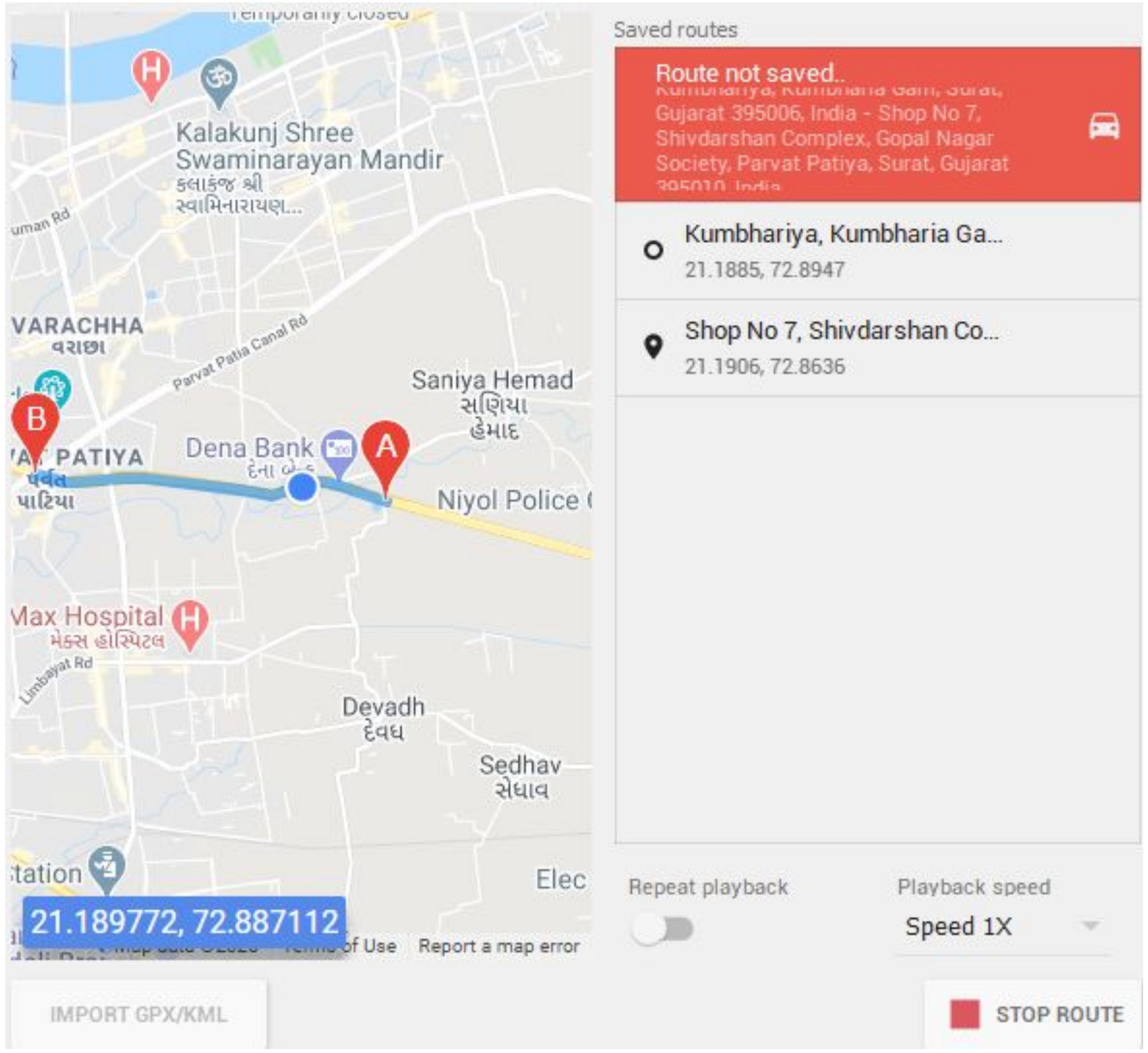
- You can see speed of device at top of the screen and the current location as blue marker.

- user can use the upper right corner button to zoom-in at his current location.



Pothole Detection

- We have reduced the travelling speed to its original speed compared to 3X speed.
- With that, we shake phone little bit to make illusion of pothole.
- Both the changes are detected and once the algorithm identifies it as a pothole, the latitude and longitude of that place will be stored in database to apply red marker to that point later.



21.189772, 72.887112

Saved routes

Route not saved..
Kumbhariya, Kumbharia Gam, Surat, Gujarat 395006, India - Shop No 7, Shivdarshan Complex, Gopal Nagar Society, Parvat Patiya, Surat, Gujarat 395010, India

Kumbhariya, Kumbharia Ga...
21.1885, 72.8947

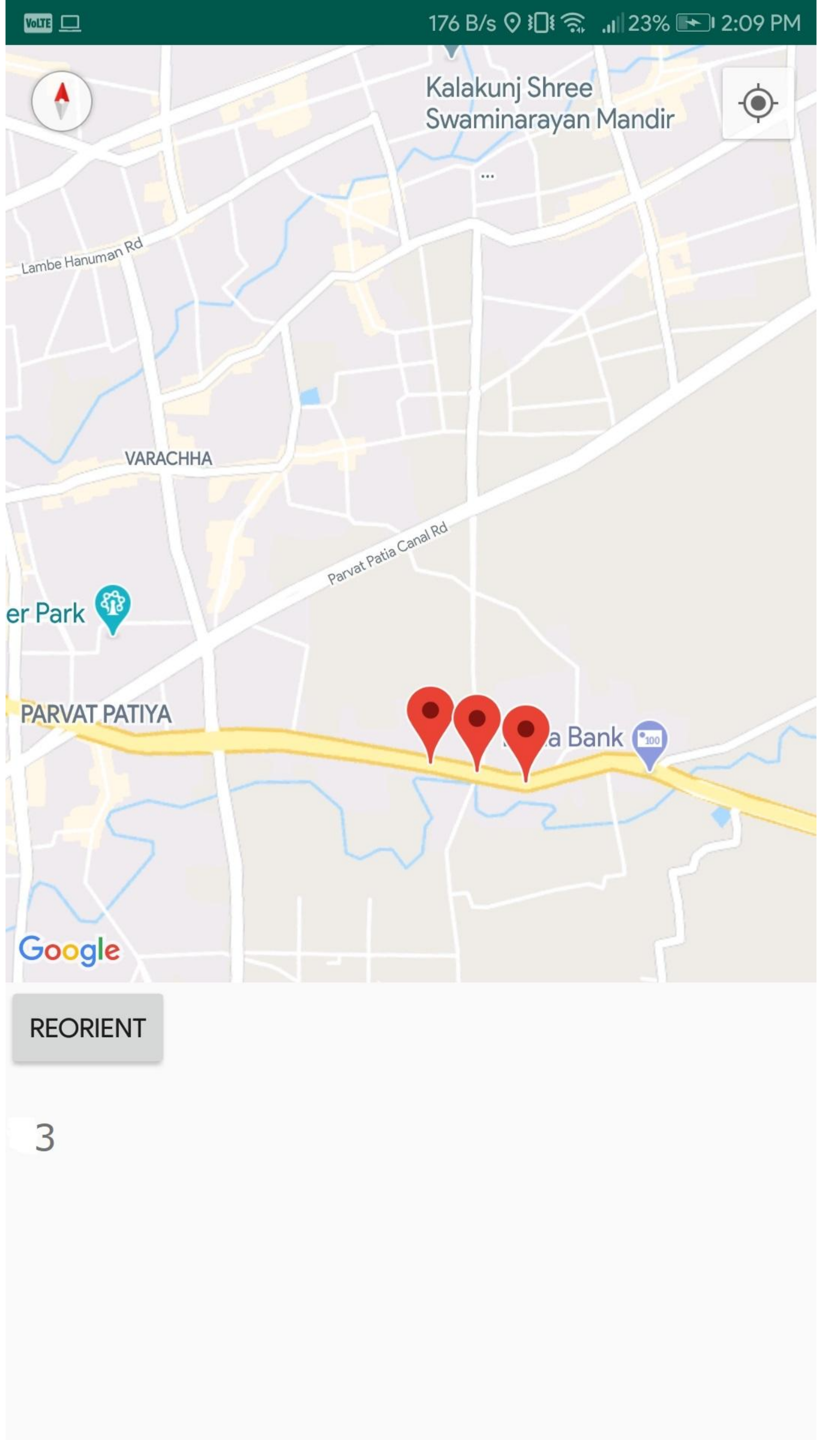
Shop No 7, Shivdarshan Co...
21.1906, 72.8636

Repeat playback

Playback speed
Speed 1X

Displaying Potholes

- In entire route, if user changes orientation of smartphone, he can recalibrate the new orientation by clicking on **reorient** button.
- While travelling, algorithm keeps track of number of potholes detected and it stores them in database.
- Once the marked location is reached, position of potholes encountered via that route, and number of potholes are displayed.



END