TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

PULCHOWK CAMPUS

THESIS NO.: 071MSCS668

A Novel Method for Real Time Pothole Detection System using Smartphones with Accelerometers

BY:

SUJIT MAHARJAN

A THESIS

SUBMITTED TO THE DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF SCIENCE IN COMPUTER SYSTEM AND KNOWLEDGE ENGINEERING

DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING

APRIL, 2017

A Novel Method for Real Time Pothole Detection System using Smartphones with Accelerometers

By

Sujit Maharjan

071/MSCS/668

Thesis Supervisor

Dr. Basanta Joshi

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Computer System and Knowledge Engineering

Department of Electronics and Computer Engineering

Institute of Engineering, Pulchowk Campus

Tribhuvan University

Pulchowk, Lalitpur, Nepal

APRIL, 2017

COPYRIGHT

TRIBHUVAN UNIVERSITY
INSTITUTE OF ENGINEERING
PULCHOWK CAMPUS, PULCHOWK
DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING

The undersigned certify that they have read, and recommended to the Institute of Engineering for acceptance, a thesis report entitled —"A Novel Method for Real Time Pothole Detection System using Smartphones with Accelerometers" submitted by Mr Sujit Maharjan in partial fulfillment of the requirement for the degree of Master of Science in Computer System and Knowledge Engineering.

.....
Supervisor, Dr. Basanta Joshi
Assistant Professor
Department of Electronics and Computer Engineering

.....
External Examiner, Mr. Adesh Khadka
IT Director
Ministry of Finance

.....
Committee Chairperson, Dr. Subarna Shakya
Professor
Department of Electronics and Computer Engineering

Department Acceptance

The thesis entitled "**A Novel Method for Real Time Pothole Detection System using Smartphones with Accelerometers**", submitted by **Sujit Maharjan** in partial fulfillment of the requirement for the award of the degree of — Master of Science in Computer System and Knowledge Engineering‖ has been accepted as a bonafide record of work independently carried out by him in the department.

.....
Dr. Dibakar Raj Pant
Head of the Department
Department of Electronics and Computer Engineering,
Pulchowk Campus
Institute of Engineering,
Tribhuvan University,
Nepal.

ACKNOWLEDGEMENT

With great pleasure that I would like to thank my respected supervisor Asst. Professor Dr. Basanta Joshi for his expert guidance and mentorship. Also, I would like to express my sincere gratitude to our Head of Department Dr. Dibakar Raj Pant, Prof. Dr. Shashidhar Ram Joshi, Prof. Dr. Subarna Shakya and Sanjeeb Prasad Panday (Ph.D.) for their encouragement and valuable suggestions.

I would like to thank our Program Coordinator (MSCSKE) Dr. Aman Shakya for his encouragement and precious guidance. Similarly, I am pleased to thank my external examiner Adesh Khadka for his valuable feedback and review in this work.

Finally, I would like to thanks all my friends for their incredible help and suggestions from the selection of this topic to the completion.

Sincerely,
Sujit Maharjan
(071/MSCS/668)

ABSTRACT

With the popularity of smartphones, alot of problem are being solved through the use of mobile applications. One of the problem is road accidents which occurs due to mainly the poor condition of road. The authorities need to keep the record about the condition of the roads to be able to maintain the roads in good condition. Mobile phone with embedded powerful sensors such as accelerometers, gyroscopes etc make it very powerful so this thesis tries to exploit these sensors to develop a system capable of automatically detecting the potholes in realtime. Algorithms like Z-THRESH, Z-DIFF, STDEV(Z) and Z-ZERO are widely used to detect potholes in realtime but they have very high false positive result when used individually with threshold values kept constant. In this thesis, the threshold values are made dynamic and it takes the value as a function of speed. Also, the potholes are detected based on result of all four algorithms. The users only need to enable the application while driving and all the data are sent to the server and the potholes are mapped in web in the realtime. The system developed is evaluated using data obtained from crawdad.org dataset and also verified on the dataset generated through android application which show better result than previous work.

Keywords: Pothole, Z-DIFF, Z-THRESH, STDEV(Z), G-ZERO, Road Monitor

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF EQUATIONS

Abbreviations

GPS Global Positioning System

1. INTRODUCTION

1.1. Background

Dangerous road surface conditions are major distractions for safe and comfortable transportation. Both drivers and road maintainers are interested in fixing them as soon as possible. However, these conditions have to be identified first. One approach to road damage detection is to use human reports to central authorities. While it has the highest accuracy, assuming that people are fair, it also has the most human interaction and is not comprehensive. Statistical analysis can be used to estimate damage probabilities of road segments based on their usage intensity. Integration of vibration and vehicle counting sensors in the pavement are used for statistical data collection [1]. Surface analysis methods using Ground Penetrating Radar (GPR) have been developed [2] and commercial products do exist [3]. Unfortunately, this technology is using expensive equipment and therefore limits its accessibility. As an alternative, participatory sensing has the potential to increase the collected data resolution and scope. The simplest method might be to collect photos of road damage and hazards taken by the participants and to upload them to a central server. However, this requires strong participation and interaction from the users as well as manual image analysis. I believe that an automated approach for detecting potholes with little or no human interaction is more promising. This would ensure more comprehensive survey data with less errors caused by human factors than generated by mere enthusiasm of the participants. An automated survey approach could be carried out by either customized embedded sensing devices or smart-phones. While the former has more sensing capabilities and adaptation potential, the popularity of smart-phones makes the latter approach very appealing in terms of practical usability. To create a successful road surface monitoring system accepted by wide user community, it is important to make it attractive for the users - to provide added value without a significant process overhead. Therefore the system can be distributed as service, which can also used for traffic information systems, such as Waze [4], which use real-time traffic information, collected by participatory sensing approach.

Although contemporary smart-phones have high processing power and considerable memory, the detection system is recommended to avoid resource-intensive detection methods and to preserve initial user interface responsiveness. Automated embedded sensing systems, including smartphones, have two general classes of sensors to be used for pothole detection: microphone and accelerometers. In this thesis, I want to focus on accelerometer data processing for pothole detection. This solution extends works of [5] and [1], and will be implemented on Android OS [6].

1.2. Problem Definition

Potholes are shallow pits on a road's surface, caused by activities like erosion, weather, traffic and some other factors. These anomalies when accumulated in the transportation system, constitutes to major problems. These problems, even though appear to be less significant at an individual level, constitute to major problems when taken in cumulative, collective and large- scale manner. The problems constituted by these potholes result in low fuel economy, accidents, traffic coagulation and so on, which have an adverse impact on the economy of a country and day to day life of citizens.

1.3. Objective

To build a novel method for real time pothole detection system that is applicable in smartphones.

1.4. Scope of Work

Road is one of the important infrastructure of development. The authorities invest huge amount of budget in the construction and maintenance of road infrastructure. The quality of the road degrades due to heating from sun, rainfall and other external agents and creates physical damages creating potholes.

Many reports have suggested that potholes is one of the major causes of road accidents. And transport authorities have to pay high attention in monitoring these roads. Many highly sophisticated instruments have be built but they cannot provide information at the real time. With the popularity and advancement in smartphone technologies, it is possible to monitor road condition with smartphone application in realtime.

The scope of this thesis work is to develop model that can real time pothole detection system using smartphone with accelerometer.

2. LITERATURE REVIEW

There are several vehicular sensing systems for pothole detection. Some of these systems use accelerometers for data acquisition. This section contains a short review of pothole detection algorithms implemented in such accelerometer-based systems. In addition the feasibility for implementation of these systems on platforms with limited hardware and software resources, such as Android based smart-phone, is considered.

BusNet [7] system developed at University of Colombo is using Crossbow MICAz motes and several sensor boards including accelerometer and GPS as hardware platform. This system does not have the functionality for real time data processing. The data is collected and stored locally for trans-mission through wireless network to collection nodes located at the bus stations for later processing. The only algorithm related to pothole detection is based on sensing acceleration and is used to start the data collection to save the limited storage space.

Pothole Patrol system [8] developed at Massachusetts Institute of Technology is using a specific hardware/software platform – Linux powered Soekris 4801 embedded computers with external accelerometers (sampling rate 380Hz) and an external GPS. Their pothole detection algorithm is based on simple machine-learning approach using X and Z axis acceleration and the vehicle velocity data as input. The algorithm consists of five consecutive filters: speed, high-pass, z-peak,xz-ratio and speed vs. z ratio. Each filter is used as a rejecter of one or more event types not related to potholes such as door slams or railway crossings. Additional training process is executed for optimal tuning of the last three filters.

Nericell and Traffic Sense [10] systems developed at Microsoft Research India are using Windows Mobile OS powered smart-phones as hardware/software platform with an array of external sensors such as accelerometers (sampling rate 310Hz),microphones and GPS. Their algorithms for pothole detection z-sus (for speeds <25km/h) and z-peak (for speeds ≥25km/h)are based on simple threshold-based heuristics. Additional algorithm virtual reorientation is used to compensate arbitrary orientation of the smart-phone during driving in the vehicle.

A system developed at National Taiwan University [11] is using motorcycle-based mobile phones HTC Diamond asa hardware platform with built-in accelerometers (sampling rate ≤25Hz) and external GPS. Their approach for pothole detection is based on supervised and unsupervised machine learning methods. Client side tasks include filtering, segmentation and feature extraction. Server side tasks use two learning models - support vector machine and a smooth road model.Road abnormality detection is performed using histograms ofa sequence of triaxial and overall acceleration data segments with different windows sizes representing data from 0.5-2.0seconds of driving time.

Researchers from University of Jyvaskyla propose a pothole detection approach in the context of offline data mining [12].Accelerometer data (sampling rate 38Hz) is preprocessed using band-pass filters with frequency range 0.5-6.0Hz, a sliding window with different functions such as Chebyshev, Hamming, Taylor and normalization in the range [0,1]. The next step is feature extraction such as mean, peak-to-peak ratio,root mean square, stan-

dard deviation, variance, power spectrum density and wavelet packet decomposition. Reducing of the number of the features is done using backward and forward selection, genetic algorithm and support vector machine using principal component analysis. Although the test results of the proposed approach show good performance, it is not suitable for full implementation on a device with limited hardware and software resources. Nevertheless, some of the described methods could be useful for real time data processing.

Simple threshold based algorithms such as z-sus, z-peak etc. undoubtedly are suitable for implementation on Android based smartphones. However, the available hardware and software resources on this platform are capable of more complex algorithms with better pothole detection parameters.

A system developed by Artis Mednis and his team[1] have used the constant threshold to detect the potholes separately. They have used used Z-THRESH, Z-DIFF, G-ZERO and STDEV(Z) separately to detect the potholes. The paper has suggested to combine the results of different algorithms and calculate dynamic value of threshold for the given algorithms.

3. METHODOLOGY

3.1. System Overview

The basic block diagram of the program execution can be shown in figure 3.1 and explained below.

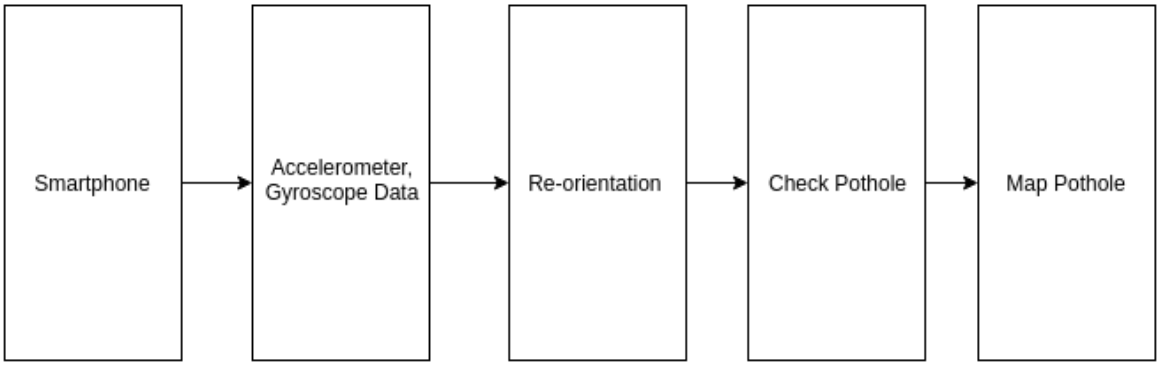

Figure 3.1: A Schematic Block Diagram of the System

3.1.1. Smartphones / Data Source

Smartphones are being more popular day by day and it is also getting cheaper. The smartphones has sensors like accelerometer and gyroscope which can measure the acceleration on its three axis and the orientation of the device. The smartphone is used as the data source of the detection of the pothole.

However for initial system modeling and system validation the dataset available at the crawdad.org is used as the data source. The crawdad dataset has already reoriented dataset so we can directly apply pothole algorithms on the pothole dataset.

3.1.2. Reorientation

The accelerometer reading and gyroscope reading can be combined to get actual accelerations in real X, Y and Z axis. The reorientation algorithm is applied and accelerations are calculated in the android phone itself.

3.1.3. Check Pothole

The pothole is check based on novel method and if the smartphone finds any pothole, it informs the about the occurrence of the pothole at the location to the central server and saved to database. However, currently all the accelerometer reading is sent to the server for analysis.

3.1.4. Map Pothole

The central server has an web access through which we can see that the occurance of the potholes in the road.

3.2. Algorithms

3.2.1. Virtual Re-orientation

The 3-axis parameters obtained from the accelerometer are largely affected by the angle of inclination or position of the device. As such, this can lead to errors in the values especially because users cannot be controlled over how or where to place their phones. Introducing the magnetometer, we would get the magnetic gravity vectors, which is then combined with the 3-axis values and used to compute the correct orientation of the device, hence, the name "virtual" re-orientation. This module is responsible for carrying out the virtual re-orientation operation using the algorithm in the following subsection.

3.2.2. Virtual Re-orientation Algorithm

1. Get Rotation Matrix based on the gravity and magnetic vectors
2. Invert Rotation Matrix
3. Get Linear acceleration Vectors in Device coordinate system
4. Multiply Rotation Matrix by linear acceleration vector.

3.2.3. Device Speed Calculation

The speed of the device can be obtained through android api which uses GPS for calculating speed. Also, we can calculate speed using accelerometer data as:

$$v(t) = \int_{t=0}^{t} \text{acceleration dt} \tag{3.2.1}$$

where v(t) is the velocity at the time t.

3.2.4. Z-THRESH Algorithm

Events are represented by measurements with values exceeding specified threshold levels.

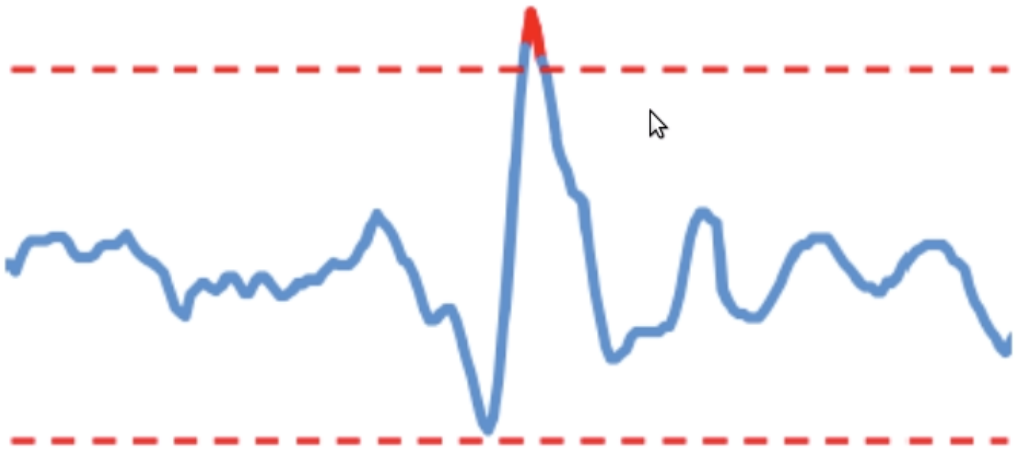

Figure 3.2: Z-THRESH Algorithm

Papers have suggested Z-THRESH algorithm was more accurate with threshold values between 0.1-1.0g.

The proposed method is the calculation of the dynamic threshold and the dynamic threshold can be found as the function of speed.

$$Z - THRESH \quad Threshold = 0.1 + (\frac{2}{1 + e^{-s*a}} - 1) * 0.9 \tag{3.2.2}$$

where s is the speed of the vehicle, a is adjustment factor. The value of a will be found out by experiment.

3.2.5. Z-DIFF Algorithm

Pothole detection algorithm Z-DIFF. Events are represented by consecutive measurements with difference value above specific threshold level.

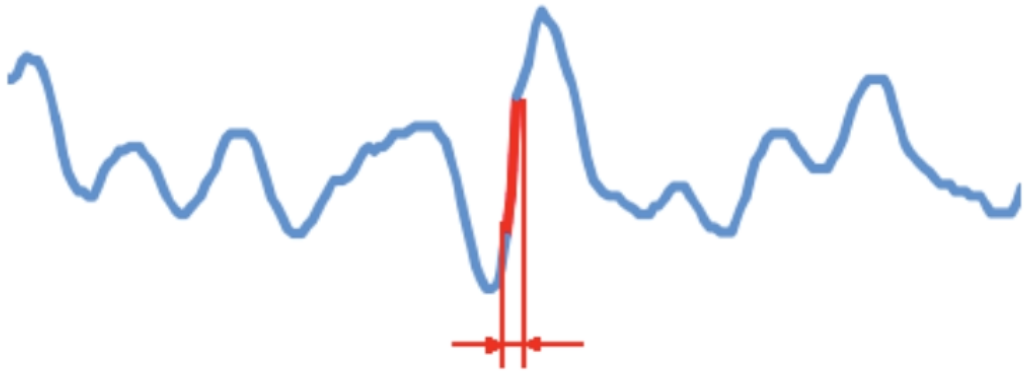

Figure 3.3: Z-DIFF Algorithm

Threshold values between 0.1g and 0.8g were more accurate for tuning the Z-DIFF algorithm.

The proposed method is the calculation of the dynamic threshold and the dynamic threshold can be found as the function of speed

$$Z - DIFF \quad Threshold = 0.1 + (\frac{2}{1 + e^{-s*a}} - 1) * 0.7 \tag{3.2.3}$$

where s is the speed of the vehicle, a is adjustment factor. The value of a will be found out by experiment.

3.2.6. Z-STDEV(Z) Algorithm

Pothole detection algorithm STDEV(Z). Events are represented by measurements with standard deviation value above specific threshold level.

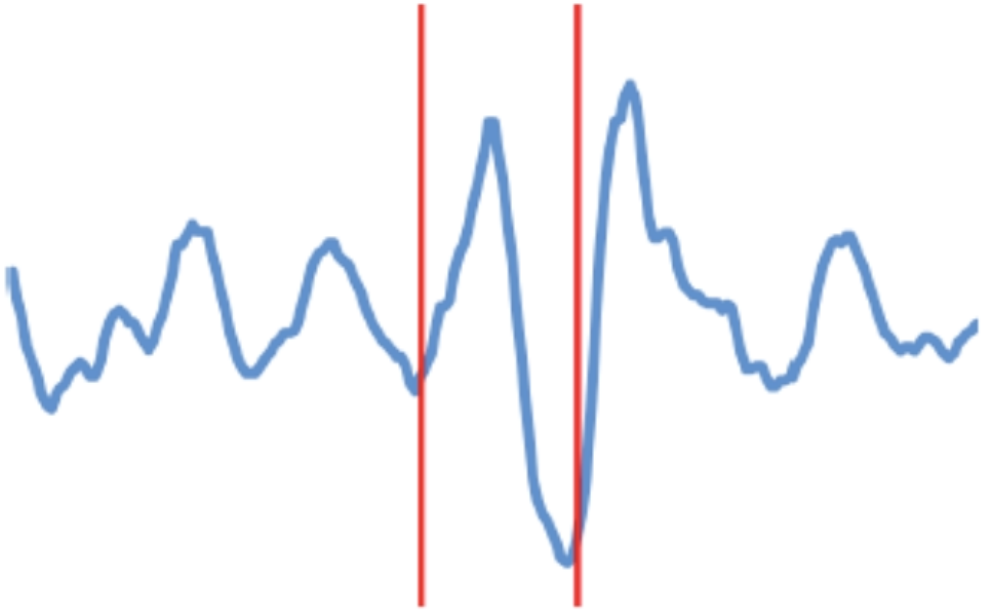

Figure 3.4: Z-THRESH Algorithm

STDEV(Z) uses a window range of 4-80 samples and ranges from 0.2 to 0.8.

The proposed method is the calculation of the dynamic threshold and the dynamic threshold can be found as the function of speed

$$STDEV(Z) \quad Threshold = 0.2 + (\frac{2}{1 + e^{-s*a}} - 1) * 0.6 \tag{3.2.4}$$

where s is the speed of the vehicle, a is adjustment factor. The value of a will be found out by experiment.

3.2.7. G-ZERO Algorithm

Events are represented by tuple of measurements with all three axis values below specific threshold level.

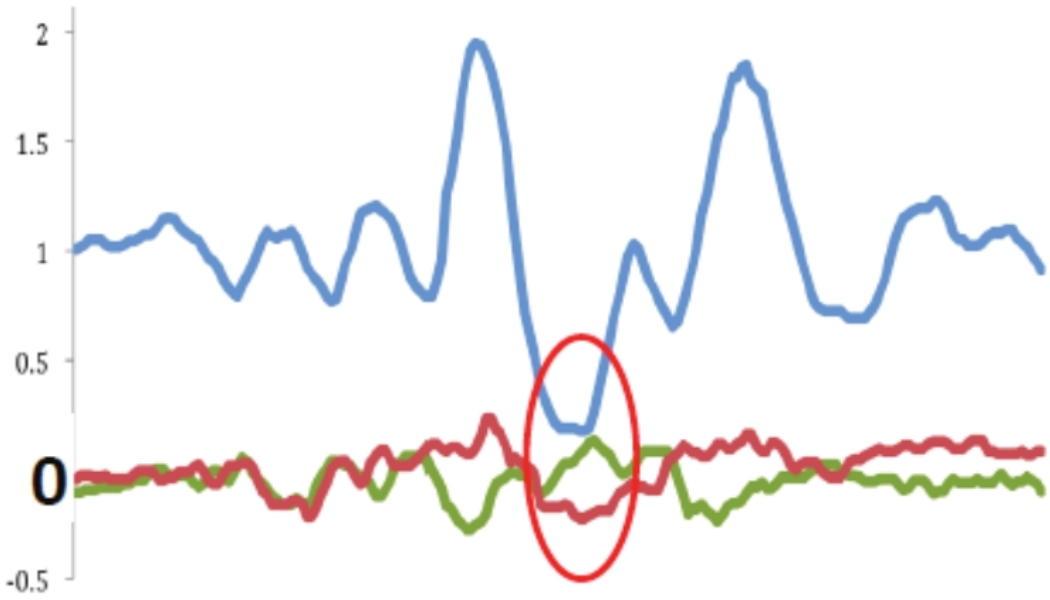

Figure 3.5: G-ZERO Algorithm

Threshold value between 0.1-0.2g is usually taken. The proposed method is the calculation of the dynamic threshold and the dynamic threshold can be found as the function of speed.

$$G-ZERO \quad Threshold = 0.1 + (\frac{2}{1+e^{-s*a}} - 1) * 0.1 \tag{3.2.5}$$

where s is the speed of the vehicle, a is adjustment factor. The value of a will be found out by experiment.

3.2.8. Novel Approach

In the Novel Approach, the sample size of 20 is taken as previous papers has got optimum results at this sample size. Also, result comes better when we select the sample size 20. The algorithms have very false positives so they cannot be used individually. The novel algorithm uses the combined decision of the four algorithms. This helps to reduce the false negative and increase confidence in the pothole detected. The trade off will be some of the true positive will also be missed which is detected by individual algorithm.

1. Make a sliding window of size 20 samples.
2. For each window, check the Z-THRESH, Z-DIFF, STDEV(Z) and G-ZERO in the window.
3. If all the algorithms mark it as pothole then mark it pothole.
4. Else there is not pothole

3.3. Data Collection

The accelerometer data is available at Crawdad which is an online senor dataset repository through www.crawdad.org/jiit/accelerometer. The data set consists of accelerometer values collected through Android phones when driven on different vehicles. Also, an android application will be build to record the acceleration data in the roads that has potholes in it.

3.3.1. Crawdad Dataset

The Crawdad dataset consists of accelerometer samples collected through Android phones when driven on different vehicles. The dataset consists of 678 Km of drive data, which involved 22 different drivers, 5 different types of vehicles (bus, auto rickshaw, cycle rickshaw, motorcycle, and car) and 4 smartphones.

3.4. Tools

For doing this thesis, following tools have been used

- Git for Version Controlling
- Android Studio and sublime IDEs for coding
- Sharelatex for documentation
- Mongodb for storing locations of potholes
- Python for server side scripting, algorithm testing and visualization
- Leafletjs for mapping and visualization

3.5. Verification and Validation

The data available in Crawdad is been used to verify the model. Also, the data collected from the android application will be used to verify the model constructed.

The result of the system is measured based on the following parameters.

3.5.1. Accuracy

Accuracy is the degree to which the result of a measurement, calculation, or specification conforms to the correct value or a standard. It gives the nearness of a calculation to the true value. The accuracy of the system is given by formula below:

$$\text{Accuracy} = \frac{\text{No. of true hit}}{\text{Total number of potholes in the route}} \tag{3.5.1}$$

3.5.2. Precision

Precision is the number of true data found by a system to the total number of true detected by the system. The precision of the system is given by formula below:

$$\text{Precision} = \frac{\text{No. of true hit}}{\text{Total number of pothole detected}} \tag{3.5.2}$$

3.5.3. False Positive Rate

The false positive show how often the false potholes are detected by the system. The false positive rat of the system is given by formula below:

$$\text{False Positive Rate} = \frac{\text{No. of false hit}}{\text{Total number of pothole detected}} \tag{3.5.3}$$

4. RESULTS AND ANALYSIS

4.1. Data Sources

The algorithms have been tested on two different dataset. The first dataset is the Crawdad accelerometer dataset which consists of the road acceleromter data near New Delhi, India. The second dataset is generated using custom built android application.

4.1.1. Crawdad Dataset

The crawdad dataset consisted accelerometer data recorded in Indian roads mainly in Delhi and nearby locations. The data were recorded using four smartphones and each smartphone has dataset of 10 routes. They had used different device to take the accelerometer reading and mark the potholes. The data were merged later based on the timestamp to make single data file. The data were recorded from the Samsung Galaxy Mini android phone. The dataset also had information about the speed breakers but those speed breakers are also treated as potholes in this system.

The dataset consisted information of time, acceleration in x-axis, y-axis and z-axis, longitude, latitude and marking at the pothole. The sample of dataset is as follows:

time,x-axis,y-axis,z-axis,Pothole,Speedbreaker,lat,lng
25.864 , -1.84 , 5.67 , 7.05,0,0,28.63540334453045 , 77.36752894642822
25.899 , -1.99 , 5.82 , 7.66,0,0,28.63540334453045 , 77.36752894642822

4.1.1.1 Details of the routes in the Crawdad dataset used

Vehicles	Pothole Counts	Place
Car	41	Gurgaon to Rajokri, 14km
Motorbike	68	Shipra Suncity to Fortis Hospital, 23km
Bus	59	Sultanpuri to Safdurjang, 26km

Table 4.1: Detail of the route used in analysis

4.1.2. Dataset Collected from android smartphone

The second dataset was collected by building native android application and traveling in the roads in Kathmandu valley. All the data were collected from Asus Google Nexus 7 (2013) tablet and the potholes were marked using media button in the android microphone. The vechicles used for collection of accelerometer data in road were:

4.1.2.1 Zotye T200 Mini SUV

The car had MacPherson struts Suspension Front and Five-link independent Suspension Rear.

4.1.2.2 Honda CB Unicorn 150

The motorbike had telescopic front suspension and loaded hydraulic type (Monoshock) rear suspension spring.

4.1.2.3 Ashok Leyland (Sajha Yatayat)

The bus had air suspension with shock absorber front suspension and air suspension with shock absorber rear suspension.
Details of the route is in table 4.2

4.1.2.4 Details of routes for collected dataset

Vehicles	Pothole Counts	Place
Car	32	Chobhar to Pulchowk, 6km
Motorbike	107	Lagankhel to Budhanilkhantha, 14km
Bus	54	Lagankhel to Budhanilkhantha, 14km

Table 4.2: Detail of the data collected routes used in analysis

4.2. Analysis using crawdad dataset

4.2.1. Finding constant value 'a' of individual algorithms using crawdad dataset

The individual algorithms were run on different value of 'a' and the line chart of true positive, false positive and false negative were plotted to find out the optimum value of constant 'a'.

The result of the plot is as given below.

4.2.1.1 For STDEV(Z) Algorithm

Each of the algorithms were first run alone for hundreds of times and an the constant value 'a' was determined.

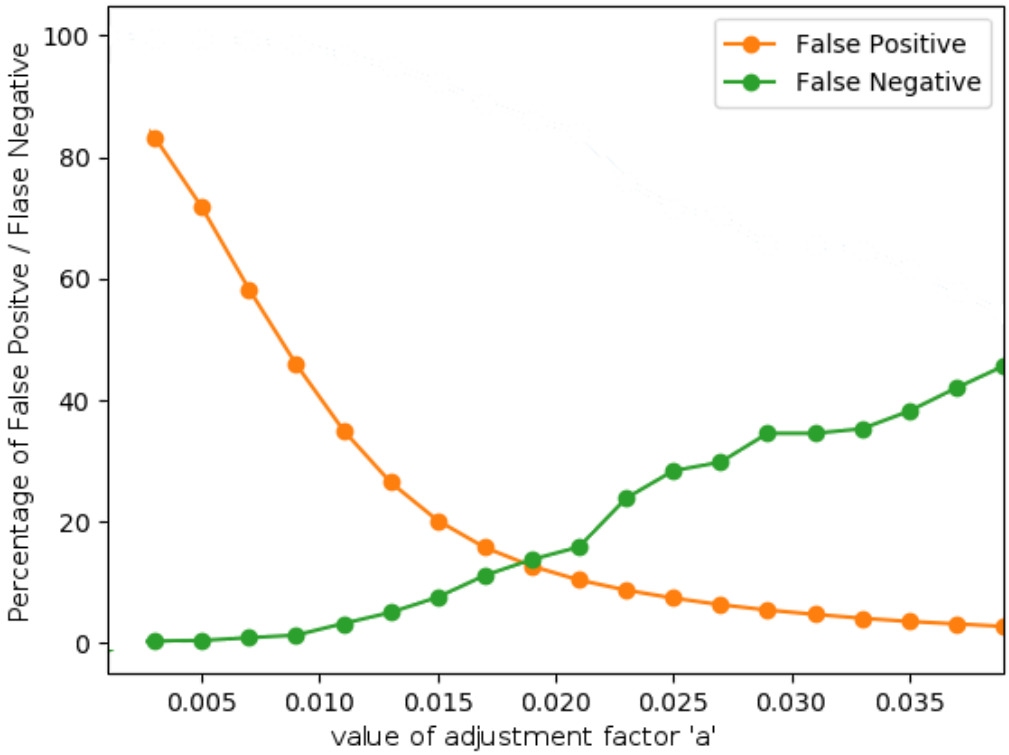

Figure 4.1: Determining value of adjustment factor 'a' for STDEV(Z) Algorithm

From the figure 4.1, a = 0.018 is selected as optimal at which there is 82% true positives and 18% false positives

4.2.1.2 For Z-THRESH Algorithm

The Z-THRESH Algorithm was run alone for range of of values of a. The optimum result was obtained for a=0.022.

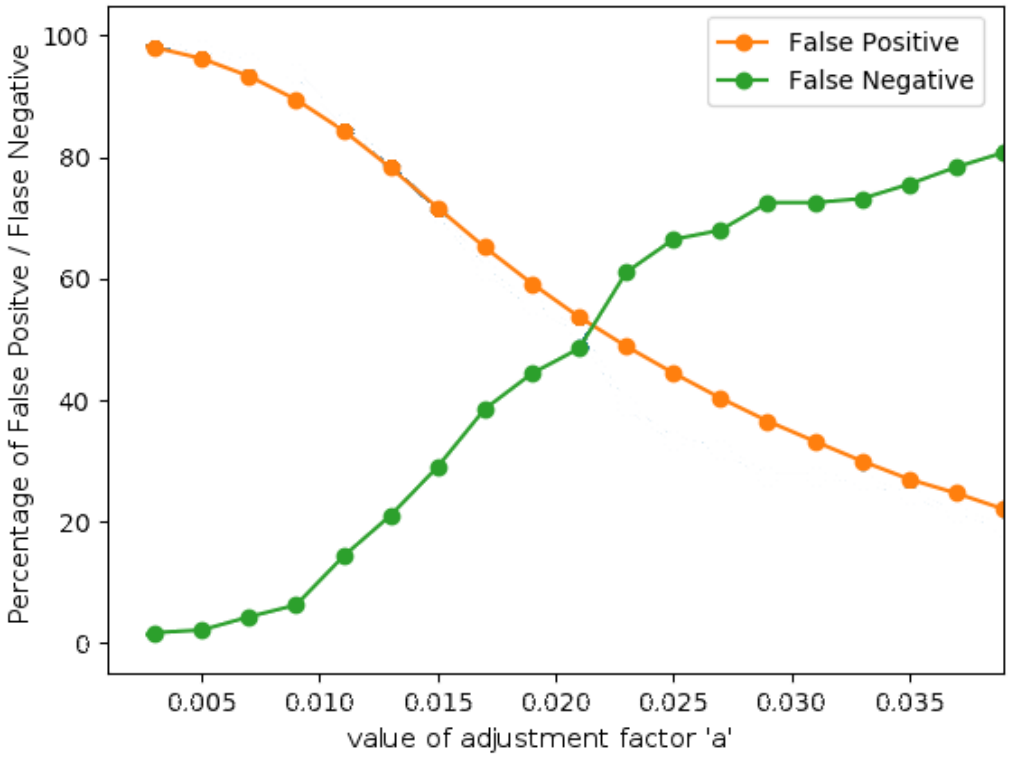

Figure 4.2: Determining value of adjustment factor 'a' for Z-THRESH Algorithm

From the figure 4.2, a = 0.022 is selected as optimal value at which there is 50% true positives and 50% false positives.

4.2.1.3 For Z-DIFF Algorithm

The Z-DIFF Algorithm was run alone for range of of values of a. The optimum result was obtained for a=0.022 .

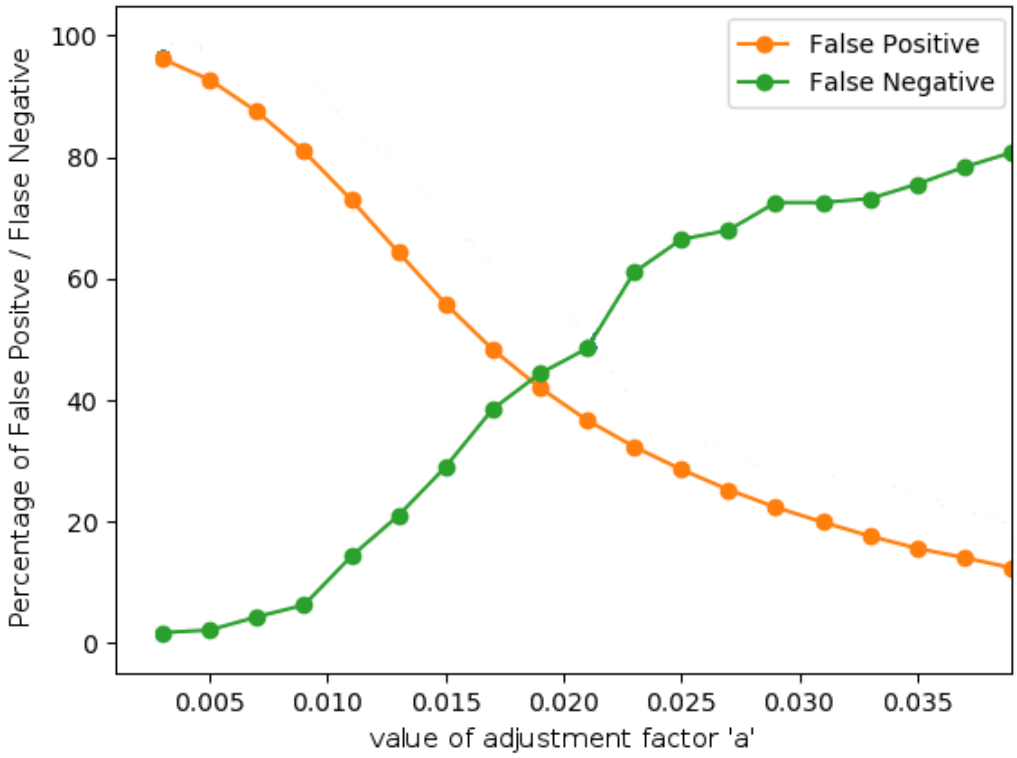

Figure 4.3: Determining value of adjustment factor 'a' for Z-DIFF Algorithm

From the figure 4.3, a = 0.022 is selected as optimal value at which there is 50% true positives and 50% false positives.

4.2.1.4 For G-ZERO Algorithm

The G-ZERO Algorithm was run alone for range of of values of a. The optimum result was obtained for a=0.016.

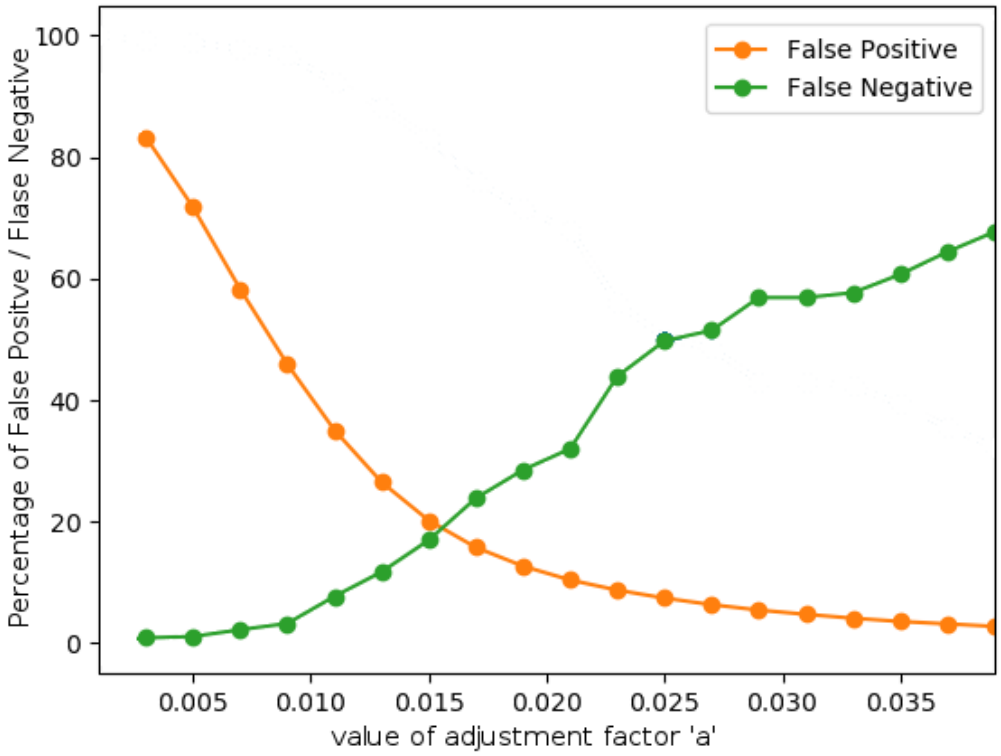

Figure 4.4: Determining value of adjustment factor 'a' for G-ZERO Algorithm

From the figure 4.3, a = 0.016 is selected as optimal value at which there is 80% true positives and 20% false positives.

4.2.2. Comparison of Constant Threshold and Dynamic Threshold in individual algorithms in Crawdad dataset

The algorithms were run on route for the car which was driven from Gurgaon to Rajokri and the result is tabulated below.

Algorithms	**Dynamic Threshold**		**Constant Threshold**	
	Detected	True Hit	Detected	True Hit
STDEV(Z)	106	35	116	35
Z-THRESH	154	29	181	37
Z-DIFF	122	37	178	41
G-ZERO	145	38	158	40

Table 4.3: Comparison of individual algorithms with constant and dynamic threshold

4.2.2.1 Evaluation of results in STDEV(Z)

Algorithms	Dynamic Threshold	Constant Threshold
Accuracy	85 %	85 %
Precision	33 %	30 %
False Positive Rate	66 %	69 %

Table 4.4: Comparison of Individual algorithms with constant and dynamic threshold using STDEV(Z)

4.2.2.2 Evaluation of results in Z-THRESH

Algorithms	Dynamic Threshold	Constant Threshold
Accuracy	70 %	90 %
Precision	18 %	20 %
False Positive Rate	81 %	79 %

Table 4.5: Comparison of Individual algorithms with constant and dynamic threshold using Z-THRESH

4.2.2.3 Evaluation of results in Z-DIFF

Algorithms	Dynamic Threshold	Constant Threshold
Accuracy	92 %	97 %
Precision	26 %	25 %
False Positive Rate	73 %	74 %

Table 4.6: Comparison of Individual algorithms with constant and dynamic threshold using Z-DIFF

4.2.2.4 Evaluation of results in G-ZERO

Algorithms	Dynamic Threshold	Constant Threshold
Accuracy	90 %	100 %
Precision	30 %	23 %
False Positive Rate	69 %	76 %

Table 4.7: Comparison of Individual algorithms with constant and dynamic threshold using G-ZERO

4.2.3. Results of Dynamic Threshold in different vehicles in Crawdad dataset

The algorithms were run on route for the car which was droven from Gurgaon to Rajokri and the result is tabulated below.

Algorithms	Motorbike		Bus	
	Detected	True Hit	Detected	True Hit
STDEV(Z)	137	54	121	45
Z-THRESH	214	67	168	49
Z-DIFF	231	61	198	37
G-ZERO	211	59	144	36

Table 4.8: Comparison of individual algorithms with constant and dynamic threshold

4.2.3.1 Evaluation of results in STDEV(Z)

Algorithms	Car	Motorbike	Bus
Accuracy	85 %	79 %	76 %
Precision	33 %	39 %	37 %
False Positive Rate	66 %	60 %	62 %

Table 4.9: Comparison of STDEV(Z) in different vehicles in Crawdad dataset

4.2.3.2 Evaluation of results in Z-THRESH

Algorithms	Car	Motorbike	Bus
Accuracy	70 %	98 %	83 %
Precision	18 %	31 %	29 %
False Positive Rate	81 %	68 %	70 %

Table 4.10: Comparison of Z-THRESH(Z) in different vehicles in Crawdad dataset

4.2.3.3 Evaluation of results in Z-DIFF

Algorithms	Car	Motorbike	Bus
Accuracy	92 %	89 %	62 %
Precision	26 %	26 %	18 %
False Positive Rate	73 %	73 %	81 %

Table 4.11: Comparison of Z-DIFF in different vehicles in Crawdad dataset

4.2.3.4 Evaluation of results in G-ZERO

Algorithms	Car	Motorbike	Bus
Accuracy	90 %	86 %	61 %
Precision	30 %	27 %	25 %
False Positive Rate	69 %	72 %	75 %

Table 4.12: Comparison of G-ZERO in different vehicles in Crawdad dataset

4.2.3.5 Result on applying Novel algorithm on different vehicles

Vehicle	Detected	True Hit
Car	48	28
Motorbike	71	43
Bus	62	29

Table 4.13: Result on applying novel algorithm on different vehicles

4.2.3.6 Evaluation of results in Novel method

Algorithms	**Car**	**Motorbike**	**Bus**
Accuracy	68 %	63 %	49 %
Precision	48 %	60 %	46 %
False Positive Rate	41 %	39 %	53 %

Table 4.14: Comparison of Novel algorithm in different vehicles in Crawdad dataset

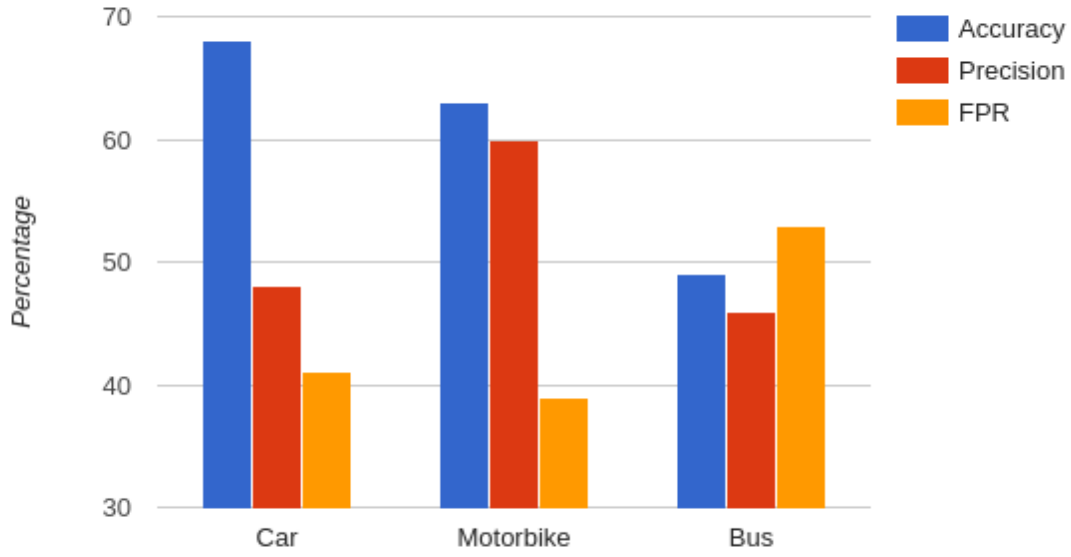

Figure 4.5: Comparison of algorithm in vehicles

4.2.4. Mapping of Potholes from Crawdad dataset

The novel algorithm was applied in the Crawdad dataset and the potholes were detected and mapped in its locations.

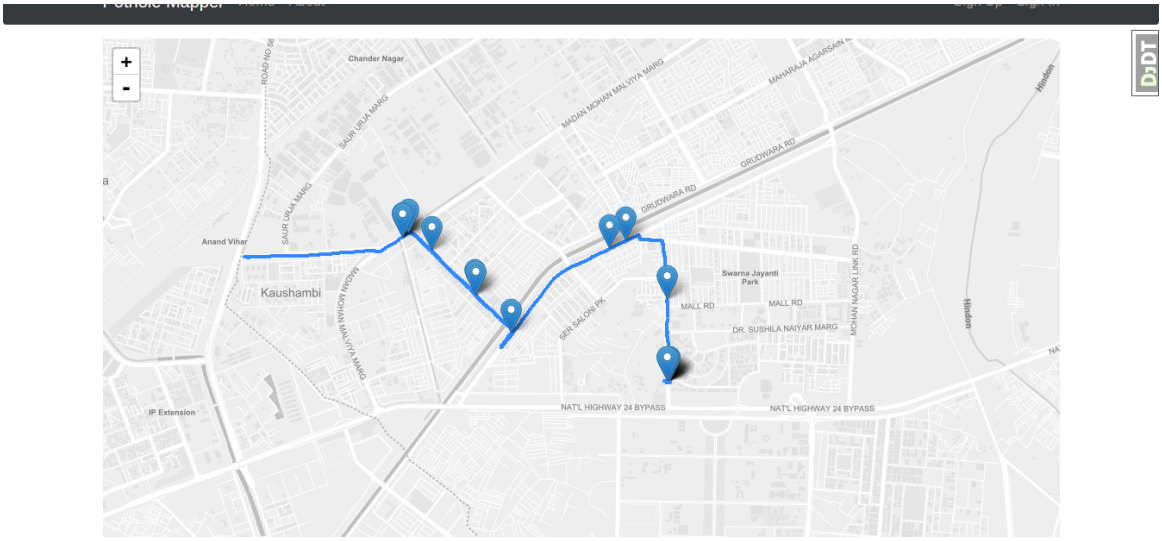

Figure 4.6: Mapping of the detected potholes from Novel Algorithm in Noida City

4.3. Analysis using collected dataset from smartphone

The android application developed to detect pothole is live in beta version at https://play.google.com/store/apps/details?id=np.net.sujit.pthole&hl=en and the potholes detected from the application is mapped at http://potholesmap.com.

4.3.1. Android App for collection of accelerometer data

An android application has been built to collect the accelerometer reading while driving. The user should press on the start button while starting the drive and again press stop while driving is completed. The data collected during the drive will be process on the device in a background process and only the detected potholes are sent to the server.

Android app start up screen is as shown in figure 4.7

Figure 4.7: Screenshot of android application to take accelerometer reading from smartphone

4.3.2. Evaluation of Virtual Reorientation Algorithm

An android application was built and the acceleration data was passed into the virtual reorientation algorithm. The accelerometer reading from the android device is in Figure 4.8

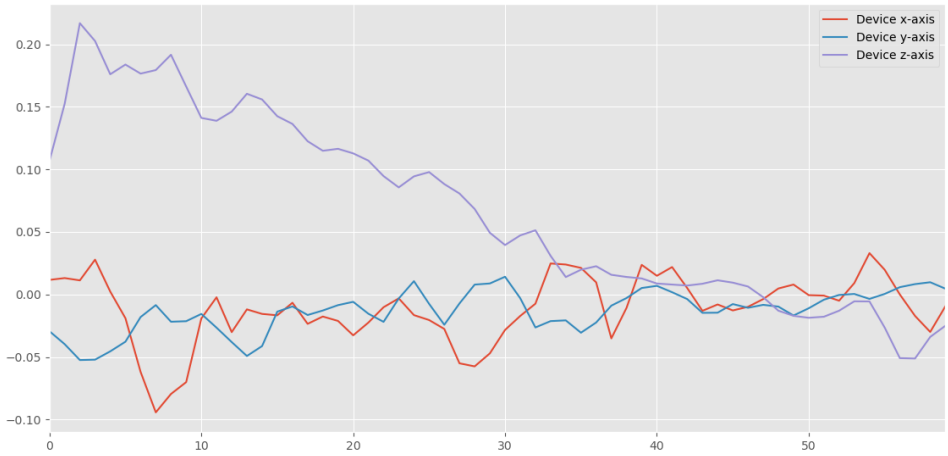

Figure 4.8: Accelerometer reading before Virtual Reorientation Algorithm

The output after the reorientation is in figure 4.9

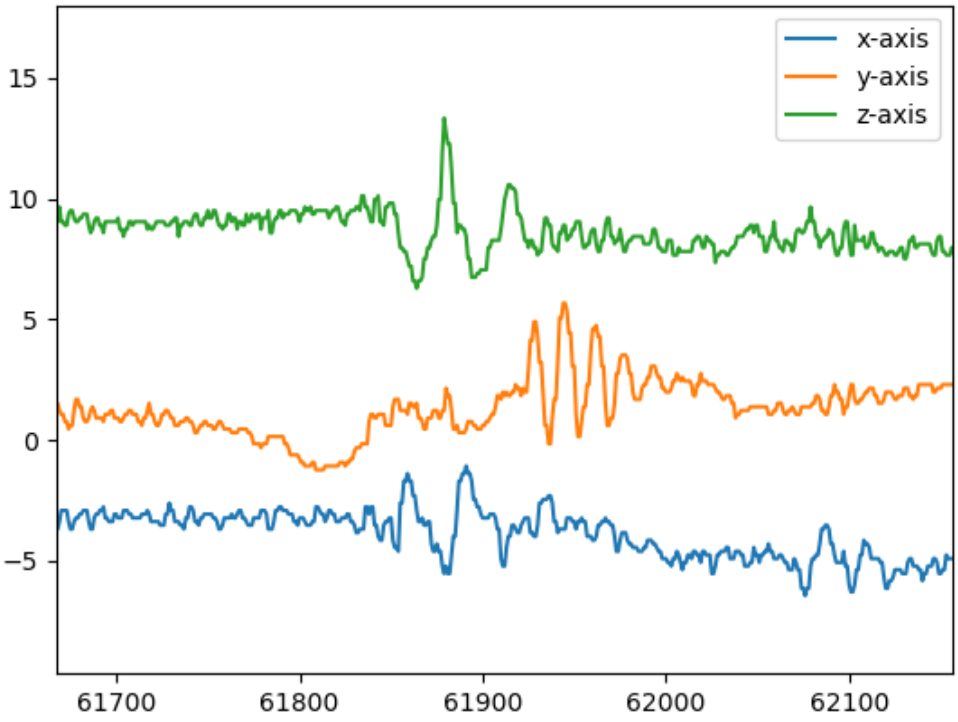

Figure 4.9: Evaluating the Virtual Reorientation Algorithm

An android app was built to collect accelerometer data and the potholes were also marked at the time of reading. The data were collected from three different types of vehicles as below.

4.3.3. Comparison of Constant Threshold and Dynamic Threshold in individual algorithms in collected dataset

The algorithms were run on route for the car which was driven from Jalabinayak to Pulchowk and the result is tabulated below.

Algorithms	**Dynamic Threshold**		**Constant Threshold**	
	Detected	True Hit	Detected	True Hit
STDEV(Z)	92	27	120	27
Z-THRESH	139	32	179	30
Z-DIFF	170	26	197	28
G-ZERO	145	27	188	26

Table 4.15: Comparison of individual algorithms with constant and dynamic threshold

4.3.3.1 Evaluation of results in STDEV(Z)

Algorithms	Dynamic Threshold	Constant Threshold
Accuracy	84 %	84 %
Precision	29 %	22 %
False Positive Rate	70 %	77 %

Table 4.16: Comparison of individual algorithms with constant and dynamic threshold using STDEV(Z)

4.3.3.2 Evaluation of results in Z-THRESH

Algorithms	Dynamic Threshold	Constant Threshold
Accuracy	100 %	93 %
Precision	23 %	16 %
False Positive Rate	76 %	83 %

Table 4.17: Comparison of individual algorithms with constant and dynamic threshold using Z-THRESH

4.3.3.3 Evaluation of results in Z-DIFF

Algorithms	Dynamic Threshold	Constant Threshold
Accuracy	81 %	87 %
Precision	15 %	14 %
False Positive Rate	84 %	85 %

Table 4.18: Comparison of individual algorithms with constant and dynamic threshold using Z-DIFF

4.3.3.4 Evaluation of results in G-ZERO

Algorithms	Dynamic Threshold	Constant Threshold
Accuracy	84 %	81 %
Precision	18 %	13 %
False Positive Rate	81 %	86 %

Table 4.19: Comparison of individual algorithms with constant and dynamic threshold using G-ZERO

4.3.4. Results of Dynamic Threshold in different vehicles in collected dataset

The algorithms were run on different vehicles with dynamic threshold in car, motorbike and bus and the result is tabulated below.

Algorithms	**Motorbike**		**Bus**	
	Detected	True Hit	Detected	True Hit
STDEV(Z)	238	98	157	41
Z-THRESH	467	102	245	47
Z-DIFF	347	87	224	54
G-ZERO	345	107	247	44

Table 4.20: Comparison of individual algorithms with constant and dynamic threshold

4.3.4.1 Evaluation of results in STDEV(Z)

Algorithms	**Car**	**Motorbike**	**Bus**
Accuracy	84 %	91 %	75 %
Precision	29 %	41 %	26 %
False Positive Rate	70 %	58 %	73 %

Table 4.21: Comparison of STDEV(Z) in different vehicles in collected dataset

4.3.4.2 Evaluation of results in Z-THRESH

Algorithms	**Car**	**Motorbike**	**Bus**
Accuracy	100 %	95 %	87 %
Precision	23 %	21 %	32 %
False Positive Rate	76 %	78 %	67 %

Table 4.22: Comparison of Z-THRESH(Z) in different vehicles in collected dataset

4.3.4.3 Evaluation of results in Z-DIFF

Algorithms	**Car**	**Motorbike**	**Bus**
Accuracy	81 %	81 %	100 %
Precision	15 %	25 %	24 %
False Positive Rate	84 %	74 %	75 %

Table 4.23: Comparison of Z-DIFF in different vehicles in collected dataset

4.3.4.4 Evaluation of results in G-ZERO

Algorithms	**Car**	**Motorbike**	**Bus**
Accuracy	84 %	100 %	81 %
Precision	18 %	31 %	17 %
False Positive Rate	81 %	68 %	82 %

Table 4.24: Comparison of G-ZERO in different vehicles in Collected dataset

4.3.5. Result on applying Novel algorithm on different vehicles

Vehicle	Detected	True Hit
Car	51	24
Motorbike	134	79
Bus	76	28

Table 4.25: Result on applying novel algorithm on different vehicles

4.3.5.1 Evaluation of results in Novel method

Algorithms	**Car**	**Motorbike**	**Bus**
Accuracy	75 %	73 %	51 %
Precision	47 %	58 %	36 %
False Positive Rate	52 %	41 %	63 %

Table 4.26: Comparison of Novel algorithm in different vehicles in collected dataset

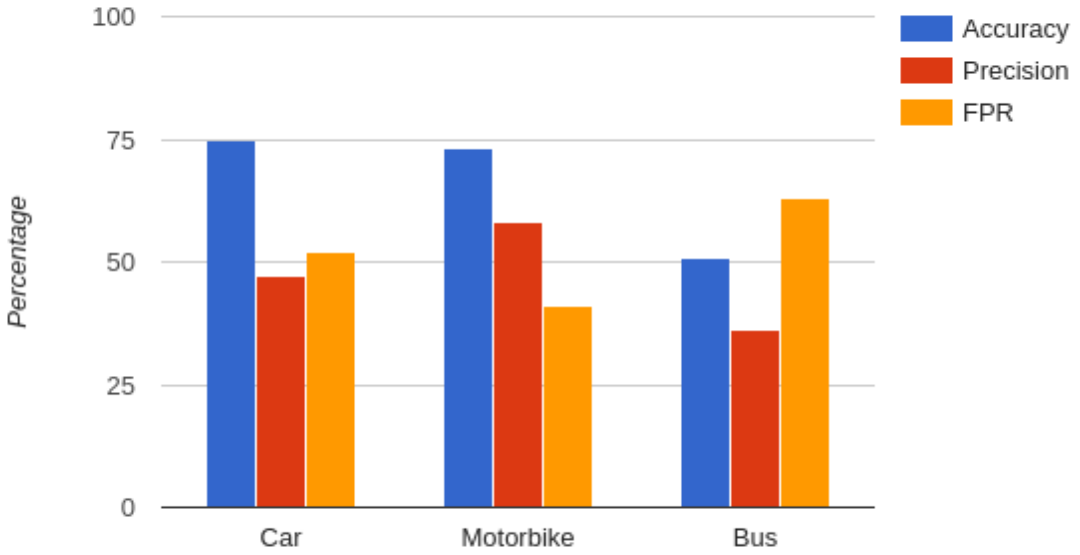

Figure 4.10: Comparison of algorithm in vehicles

4.3.6. Mapping of Potholes from collected dataset

The novel algorithm was applied in the data collected from android application and the potholes were detected and mapped in its locations.

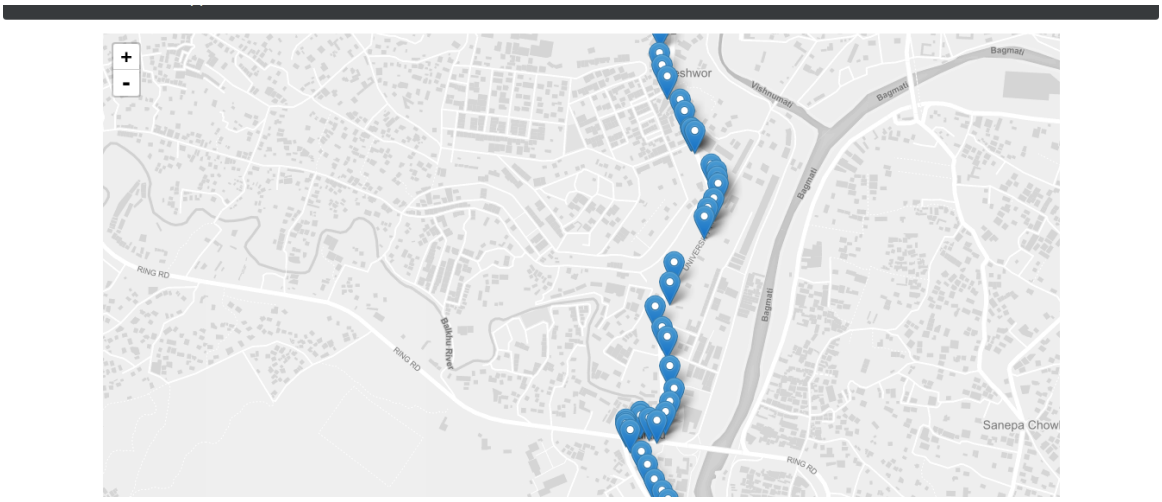

Figure 4.11: Mapping of the detected potholes from Novel Algorithm in collected data near Balkhu

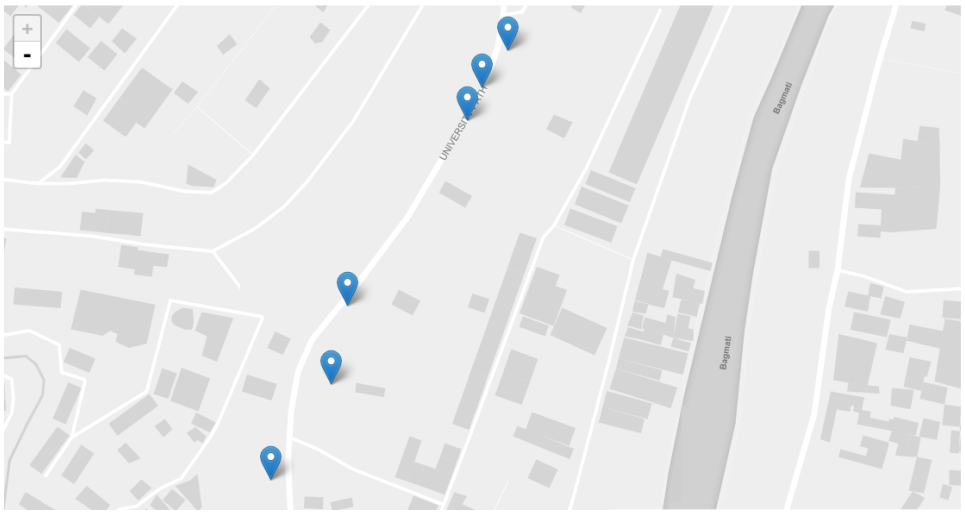

Figure 4.12: Mapping of the detected potholes from Novel Algorithm in collected data in closed up view at Sundharighat

4.4. Computational Complexity

The computational complexity of the system is O(1) to check a pothole in an window and the average time required to check pothole is 5 millisecond and send it to server in realtime. So, the system is very lightweight and can be easily implemented in current version of smartphones without draining much power from smartphones.

4.5. Power Consumption

Energy consumption is one of the key factor for feasibility of smartphone application. The application was run for a fixed interval of time and the energy consumption was measured. The result is as follows:
Running Time = 1 Hour
Decrease in Battery = 13%
Total battery Capacity = 3800mAH
Then, Total Charge Consumed = 0.13 * 3800 = 494 mAH
So, the power consumption rate is about 494 mAH/hour

5. CONCLUSION AND RECOMMENDATIONS

5.1. Conclusion

In this work, a novel method after combining four algorithms was developed to detect the potholes in real time using smartphones. The adjustment factor 'a' was calculated to implement the dynamic thresholding. The dynamic thresholding helped to reduce the false positives in each algorithms. The combination of algorithms decreased the rate of false positives which increased the precision of the system to find pothole with loss in some accuracy.

Previously, the Z-THRESH, Z-DIFF, G-ZERO and STDEV(Z) methods are individually used with constant thresholds. They had false positive rate of about 60% to 80% with accuracy ranging from 70% to 95% when used individually with constant thresholds. However, the novel algorithm reduced the false positive rate to 40% to 50% with accuracy of about 75%. The novel system increased the precision of the system from 20% when used individually to 52% with novel approach.

Thus, the system can be used to find the potholes more precisely in real time. The system can be further extended to collected potholes data from many users and mark potholes based on larger set of data.

5.2. Limitations

Although the app can detect the potholes but till it has some limitations which are listed below:

- Due to the limited accuracy and precision of GPS system, it is not possible to exactly locate the potholes.
- Also the vehicles are in motion and the GPS system takes some time to compute the location, the potholes may be drifted from real location.
- The system is not able to classify the size and type of the potholes.
- In the current system the user have to start the application while starting driving. However, it would be more comfortable for user if the application could automatically recognize the driving status.

5.3. Future Enhancement

The potholes detected from the single device may not be sufficient to confirm the existence of pothole because the potholes can be seen due to some false positive.
The system can be made more accurate by combining the pothole reports from many users and give a final verdict. In this way the possibility of false positive will be minimized.
As of now, the system only checks the potholes but we also can use the collected motion and location data to predict the traffic jams in the road.

References

[1] Mednis, A., Strazdins, G., Zviedris, R., Kanonirs, G., Selavo, L. (2011, June). Real time pothole detection using android smartphones with accelerometers. In Distributed Computing in Sensor Systems and Workshops (DCOSS), 2011 International Conference on (pp. 1-6). IEEE.

[2] H. Jol, Ground penetrating radar: Theory and applications. Elsevier Science Ltd, 2009.

[3] "Roadscanners oy." [Online]. Available: http://www.roadscanners.com/

[4] "Real-time maps and traffic information based on the wisdom of the crowd." [Online]. Available: http://world.waze.com/

[5] G. Strazdins, A. Mednis, G. Kanonirs, R. Zviedris, and L. Selavo, "Towards Vehicular Sensor Networks with Android Smartphones for Road Surface Monitoring," in CONET'11, CPSWeek'11, 2011.

[6] "Android OS." [Online]. Available: http://android.com

[7] K. De Zoysa, C. Keppitiyagama, G. P. Seneviratne, and W. W. A. T. Shihan, "A public transport system based sensor network for road surface condition monitoring," in Proceedings of the 2007 workshop on Networked systems for developing regions, ser. NSDR '07. New York, NY, USA: ACM, 2007, pp. 9:1–9:6. [Online]. Available: http://doi.acm.org/10.1145/1326571.1326585

[8] J. Eriksson, L. Girod, B. Hull, R. Newton, S. Madden, and H. Balakrishnan, "The pothole patrol: using a mobile sensor network for road surface monitoring," in Proceeding of the 6th international conference on Mobile systems, applications, and services, ser. MobiSys '08. New York, NY, USA: ACM, 2008, pp. 29–39.

[9] P. Mohan, V. N. Padmanabhan, and R. Ramjee, "Nericell: using mobile smartphones for rich monitoring of road and traffic conditions," in Proceedings of the 6th ACM conference on Embedded network sensor systems, ser. SenSys '08. New York, NY, USA: ACM, 2008, pp. 357–358. [Online]. Available: http://doi.acm.org/10.1145/1460412.1460450

[10] "Trafficsense: Rich monitoring of road and traffic conditions using mobile smartphones," Microsoft Research, Tech. Rep. MSR-TR-2008-59, April 2008.

[11] Y.-c. Tai, C.-w. Chan, and J. Y.-j. Hsu, "Automatic road anomaly detection using smart mobile device," in Proceedings of the 2010 Conference on Technologies and Applications of Artificial Intelligence (TAAI2010), November 2010.

[12] H. Hautakangas and J. Nieminen, "Data mining for pothole detection." Presented at the Pro gradu seminar, University of Jyvaaskyla February 201

[13] Victor Akinwande, Kayode Adewole, Olayiwola Bello, Abimbola Akintola (2015, August). Automatic and real-time Pothole detection and Traffic monitoring system using Smartphone Technology. University of Ilorin, 2011 International Conference on (pp. 1-8).

A. ROUTES AND POTHOLE MAPS

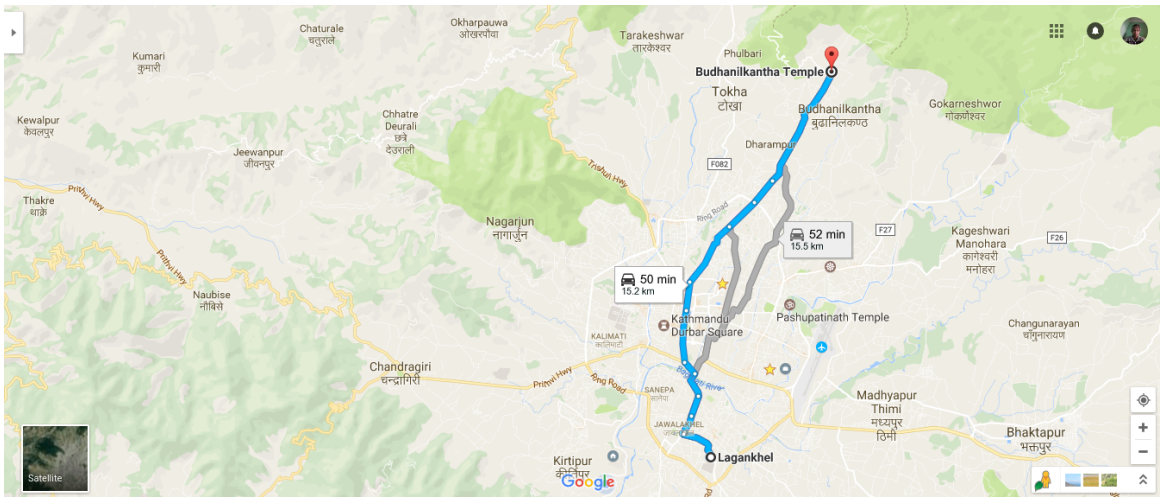

Figure A.1: Route from Lagankhel to Budhanilkantha

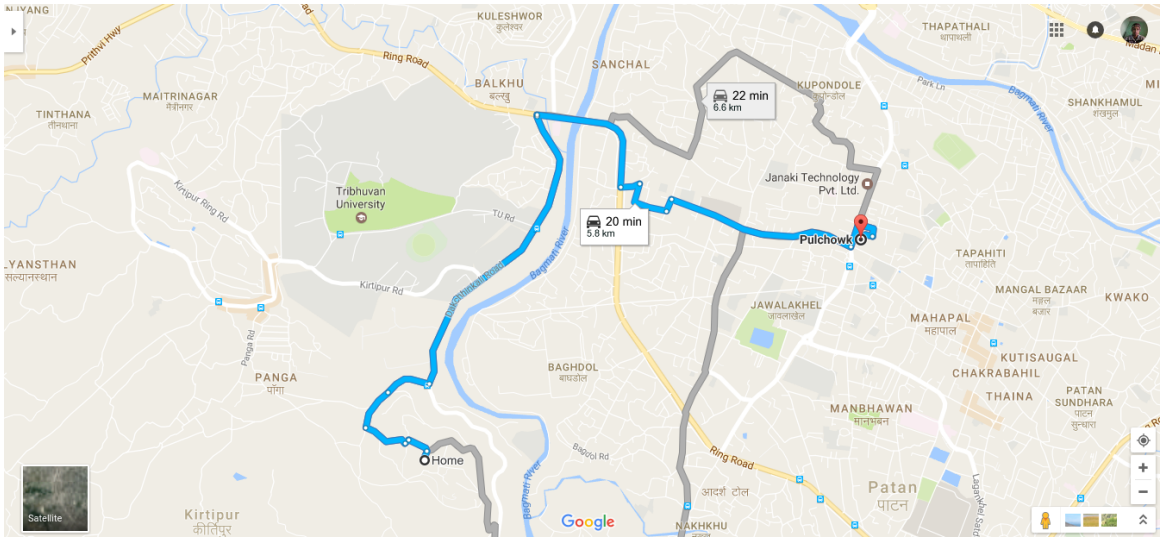

Figure A.2: Route from Chobhar to Pulchowk Campus

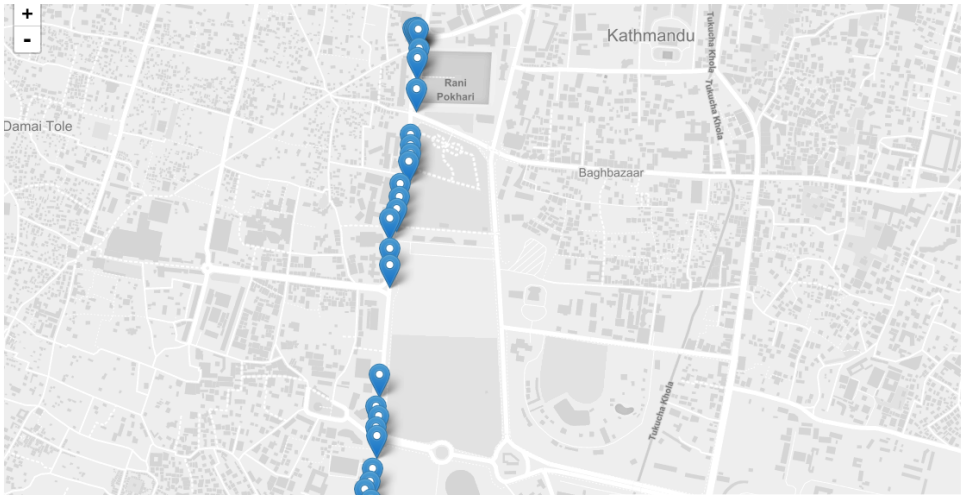

Figure A.3: Potholes marked between Tirpureshwor and Jamal

B. SAMPLE DATASET

time,x-axis,y-axis,z-axis,Pothole,Speedbreaker,lat,lng
25.864 , -1.84 , 5.67 , 7.05,0,0,28.63540334453045 , 77.36752894642822
25.899 , -1.99 , 5.82 , 7.66,0,0,28.63540334453045 , 77.36752894642822
25.915 , -1.84 , 5.520005 , 7.51,0,0,28.63540334453045 , 77.36752894642822
25.93502 , -1.84 , 5.520005 , 7.51,0,0,28.63540334453045 , 77.36752894642822
25.95502 , -1.84 , 5.060005 , 7.350005,0,0,28.63540334453045 , 77.36752894642822
25.976 , -1.84 , 5.060005 , 7.350005,0,0,28.63540334453045 , 77.36752894642822
25.995 , -1.53 , 4.600005 , 7.51,0,0,28.63540334453045 , 77.36752894642822
26.015 , -1.69 , 4.9 , 7.51,0,0,28.63540334453045 , 77.36752894642822
26.035 , -1.69 , 4.9 , 7.51,0,0,28.63540334453045 , 77.36752894642822
26.055 , -1.69 , 5.36 , 7.51,0,0,28.63540334453045 , 77.36752894642822
26.075 , -1.380001 , 5.060005 , 7.51,0,0,28.63540334453045 , 77.36752894642822
26.095 , -1.380001 , 5.060005 , 7.51,0,0,28.63540334453045 , 77.36752894642822
26.11502 , -1.84 , 5.060005 , 7.66,0,0,28.63540334453045 , 77.36752894642822
26.135 , -1.84 , 5.060005 , 7.66,0,0,28.63540334453045 , 77.36752894642822
26.155 , -1.84 , 5.21 , 7.810005,0,0,28.63540334453045 , 77.36752894642822
26.175 , -1.380001 , 5.060005 , 7.66,0,0,28.63540334453045 , 77.36752894642822
26.198 , -1.380001 , 5.060005 , 7.66,0,0,28.63540334453045 , 77.36752894642822
26.21702 , -1.23 , 5.21 , 7.810005,0,0,28.63540334453045 , 77.36752894642822
26.262 , -1.07 , 5.21 , 7.66,0,0,28.63540334453045 , 77.36752894642822
26.28 , -1.07 , 5.520005 , 7.51,0,0,28.63540334453045 , 77.36752894642822
26.30402 , -1.07 , 5.520005 , 7.51,0,0,28.63540334453045 , 77.36752894642822
26.327 , -1.23 , 5.520005 , 7.51,0,0,28.63540334453045 , 77.36752894642822
26.346 , -1.23 , 5.21 , 7.51,0,0,28.63540334453045 , 77.36752894642822
26.36202 , -1.23 , 5.21 , 7.51,0,0,28.63540334453045 , 77.36752894642822
26.395 , -1.07 , 5.21 , 7.350005,0,0,28.63540334453045 , 77.36752894642822
26.415 , -1.23 , 5.520005 , 7.350005,0,0,28.63540334453045 , 77.36752894642822
26.43502 , -1.23 , 5.36 , 7.51,0,0,28.63540334453045 , 77.36752894642822
26.45502 , -1.23 , 5.36 , 7.51,0,0,28.63540334453045 , 77.36752894642822
26.476 , -1.07 , 5.36 , 7.51,0,0,28.63540334453045 , 77.36752894642822
26.495 , -0.77 , 5.520005 , 7.51,0,0,28.63540334453045 , 77.36752894642822
26.515 , -0.77 , 5.520005 , 7.51,0,0,28.63540334453045 , 77.36752894642822
26.535 , -0.77 , 5.82 , 7.66,0,0,28.63540334453045 , 77.36752894642822
26.555 , -0.77 , 5.82 , 7.66,0,0,28.63540334453045 , 77.36752894642822
26.575 , -0.61 , 5.67 , 7.350005,0,0,28.63540334453045 , 77.36752894642822
26.595 , -0.77 , 5.520005 , 7.51,0,0,28.63540334453045 , 77.36752894642822
26.61502 , -0.77 , 5.520005 , 7.51,0,0,28.63540334453045 , 77.36752894642822
26.635 , -0.92 , 5.36 , 7.51,0,0,28.63540334453045 , 77.36752894642822
26.655 , -1.07 , 5.36 , 7.66,0,0,28.63540334453045 , 77.36752894642822
26.675 , -1.07 , 5.36 , 7.66,0,0,28.63540334453045 , 77.36752894642822
26.695 , -1.07 , 5.21 , 7.66,0,0,28.63540334453045 , 77.36752894642822
26.715 , -1.07 , 5.21 , 7.66,0,0,28.63540334453045 , 77.36752894642822
26.736 , -1.07 , 5.36 , 7.51,0,0,28.63540334453045 , 77.36752894642822
26.755 , -1.07 , 5.36 , 7.51,0,0,28.63540334453045 , 77.36752894642822
26.77502 , -1.07 , 5.36 , 7.51,0,0,28.63540334453045 , 77.36752894642822
26.795 , -0.92 , 5.36 , 7.51,0,0,28.63540334453045 , 77.36752894642822

26.815 , -0.92 , 5.36 , 7.66,0,0,28.63540334453045 , 77.36752894642822
26.835 , -0.92 , 5.36 , 7.66,0,0,28.63540334453045 , 77.36752894642822
26.855 , -0.92 , 5.520005 , 7.66,0,0,28.63540334453045 , 77.36752894642822
26.876 , -0.92 , 5.520005 , 7.66,0,0,28.63540334453045 , 77.36752894642822
26.895 , -0.92 , 5.36 , 7.66,0,0,28.63540334453045 , 77.36752894642822
26.915 , -1.07 , 5.36 , 7.66,0,0,28.63540334453045 , 77.36752894642822
26.93502 , -1.07 , 5.36 , 7.66,0,0,28.63540334453045 , 77.36752894642822
26.95502 , -0.92 , 5.21 , 7.66,0,0,28.63540334453045 , 77.36752894642822
26.976 , -0.92 , 5.21 , 7.66,0,0,28.63540334453045 , 77.36752894642822
26.995 , -1.23 , 5.36 , 7.66,0,0,28.63540334453045 , 77.36752894642822
27.023 , -1.23 , 5.21 , 7.51,0,0,28.63540334453045 , 77.36752894642822
27.041 , -1.23 , 5.21 , 7.51,0,0,28.63540334453045 , 77.36752894642822
27.06002 , -1.23 , 5.21 , 7.51,0,0,28.63540334453045 , 77.36752894642822
27.078 , -1.380001 , 5.36 , 7.66,0,0,28.63540334453045 , 77.36752894642822
27.097 , -1.380001 , 5.36 , 7.66,0,0,28.63540334453045 , 77.36752894642822
27.11502 , -0.77 , 5.21 , 7.51,0,0,28.63540334453045 , 77.36752894642822
27.138 , -0.92 , 5.36 , 7.51,0,0,28.63540334453045 , 77.36752894642822
27.155 , -0.92 , 5.36 , 7.51,0,0,28.63540334453045 , 77.36752894642822
27.175 , -1.380001 , 5.21 , 7.51,0,0,28.63540334453045 , 77.36752894642822
27.2 , -1.380001 , 5.21 , 7.51,0,0,28.63540334453045 , 77.36752894642822
27.221 , -0.77 , 5.060005 , 7.810005,0,0,28.63540334453045 , 77.36752894642822
27.259 , -0.61 , 4.9 , 7.97,0,0,28.63540334453045 , 77.36752894642822
27.277 , -1.380001 , 4.75 , 7.350005,0,0,28.63540334453045 , 77.36752894642822
27.29802 , -1.380001 , 4.75 , 7.350005,0,0,28.63540334453045 , 77.36752894642822
27.315 , -1.380001 , 4.9 , 8.12001,0,0,28.63540334453045 , 77.36752894642822
27.335 , -1.380001 , 5.060005 , 7.66,0,0,28.63540334453045 , 77.36752894642822
27.358 , -1.380001 , 5.060005 , 7.66,0,0,28.63540334453045 , 77.36752894642822
27.381 , -1.23 , 5.36 , 7.97,0,0,28.63540334453045 , 77.36752894642822
27.415 , -1.07 , 5.36 , 7.810005,0,0,28.63540334453045 , 77.36752894642822
27.43502 , -1.23 , 5.060005 , 7.810005,0,0,28.63540334453045 , 77.36752894642822
27.456 , -1.23 , 5.060005 , 7.810005,0,0,28.63540334453045 , 77.36752894642822
27.475 , -1.380001 , 4.9 , 7.66,0,0,28.63540334453045 , 77.36752894642822
27.495 , -1.53 , 4.9 , 7.66,0,0,28.63540334453045 , 77.36752894642822
27.515 , -1.53 , 4.9 , 7.66,0,0,28.63540334453045 , 77.36752894642822
27.535 , -1.53 , 4.75 , 7.97,0,0,28.63540334453045 , 77.36752894642822
27.555 , -1.53 , 4.75 , 7.97,0,0,28.63540334453045 , 77.36752894642822
27.575 , -1.380001 , 4.9 , 7.810005,0,0,28.63540334453045 , 77.36752894642822
27.596 , -1.23 , 5.060005 , 7.66,0,0,28.63540334453045 , 77.36752894642822
27.61502 , -1.23 , 5.060005 , 7.66,0,0,28.63540334453045 , 77.36752894642822
27.635 , -1.380001 , 5.36 , 7.810005,0,0,28.63540334453045 , 77.36752894642822
27.655 , -1.07 , 5.21 , 7.810005,0,0,28.63540334453045 , 77.36752894642822
27.675 , -1.07 , 5.21 , 7.810005,0,0,28.63540334453045 , 77.36752894642822
27.695 , -1.07 , 5.060005 , 7.66,0,0,28.63540334453045 , 77.36752894642822
27.726 , -1.380001 , 5.36 , 7.810005,0,0,28.63540334453045 , 77.36752894642822
27.757 , -0.92 , 5.21 , 7.810005,0,0,28.63540334453045 , 77.36752894642822
27.777 , -0.92 , 5.21 , 7.810005,0,0,28.63540334453045 , 77.36752894642822
27.795 , -1.07 , 5.060005 , 7.810005,0,0,28.63540334453045 , 77.36752894642822
27.815 , -0.77 , 4.75 , 8.12001,0,0,28.63540334453045 , 77.36752894642822

27.835 , -0.77 , 4.75 , 8.12001,0,0,28.63540334453045 , 77.36752894642822
27.855 , -0.46 , 4.600005 , 7.97,0,0,28.63540334453045 , 77.36752894642822
27.875 , -0.46 , 4.600005 , 7.97,0,0,28.63540334453045 , 77.36752894642822
27.895 , -0.77 , 4.44 , 7.97,0,0,28.63540334453045 , 77.36752894642822
27.915 , -0.92 , 4.14 , 8.43,0,0,28.63540334453045 , 77.36752894642822
27.93502 , -0.92 , 4.14 , 8.43,0,0,28.63540334453045 , 77.36752894642822
27.95502 , -0.46 , 4.14 , 8.27,0,0,28.63540334453045 , 77.36752894642822
27.976 , -0.46 , 4.29 , 8.27,0,0,28.63540334453045 , 77.36752894642822
27.995 , -0.61 , 4.29 , 8.27,0,0,28.63540334453045 , 77.36752894642822
28.015 , -0.46 , 3.98 , 8.12001,0,0,28.63540334453045 , 77.36752894642822
28.035 , -0.46 , 3.98 , 8.12001,0,0,28.63540334453045 , 77.36752894642822
28.055 , -0.31 , 4.600005 , 7.97,0,0,28.63540334453045 , 77.36752894642822
28.075 , -0.46 , 4.600005 , 7.66,0,0,28.63540334453045 , 77.36752894642822
28.095 , -0.46 , 4.600005 , 7.66,0,0,28.63540334453045 , 77.36752894642822
28.11502 , -0.31 , 4.600005 , 7.97,0,0,28.63540334453045 , 77.36752894642822
28.135 , -0.31 , 4.600005 , 7.97,0,0,28.63540334453045 , 77.36752894642822
28.155 , -0.15 , 4.9 , 7.97,0,0,28.63540334453045 , 77.36752894642822
28.175 , -0.15 , 5.060005 , 7.810005,0,0,28.63540334453045 , 77.36752894642822
28.201 , -0.31 , 4.75 , 7.66,0,0,28.63540334453045 , 77.36752894642822
28.228 , -0.31 , 4.75 , 7.66,0,0,28.63540334453045 , 77.36752894642822
28.255 , -0.31 , 4.75 , 7.97,0,0,28.63540334453045 , 77.36752894642822
28.28102 , -0.15 , 4.9 , 8.12001,0,0,28.63540334453045 , 77.36752894642822
28.314 , -0.31 , 4.75 , 7.97,0,0,28.63540334453045 , 77.36752894642822
28.33902 , -0.31 , 4.600005 , 7.97,0,0,28.63540334453045 , 77.36752894642822
28.377 , -0.31 , 4.9 , 8.12001,0,0,28.63540334453045 , 77.36752894642822
28.39702 , -0.31 , 4.9 , 8.12001,0,0,28.63540334453045 , 77.36752894642822
28.422 , -0.31 , 5.060005 , 8.12001,0,0,28.63540334453045 , 77.36752894642822
28.445 , -0.31 , 4.600005 , 8.12001,0,0,28.63540334453045 , 77.36752894642822
28.468 , 0.15 , 5.060005 , 7.97,0,0,28.63540334453045 , 77.36752894642822
28.492 , 0.15 , 5.21 , 7.66,0,0,28.63540334453045 , 77.36752894642822
28.511 , 0.15 , 5.21 , 7.66,0,0,28.63540334453045 , 77.36752894642822
28.529 , 0.31 , 5.060005 , 8.12001,0,0,28.63540334453045 , 77.36752894642822
28.547 , 0.31 , 5.060005 , 8.12001,0,0,28.63540334453045 , 77.36752894642822
28.578 , 0.15 , 5.060005 , 7.810005,0,0,28.63540334453045 , 77.36752894642822
28.598 , 0.15 , 5.060005 , 7.810005,0,0,28.63540334453045 , 77.36752894642822
28.61802 , 0.15 , 5.060005 , 7.810005,0,0,28.63540334453045 , 77.36752894642822
28.639 , -0.15 , 4.75 , 7.66,0,0,28.63540334453045 , 77.36752894642822
28.655 , -0.15 , 4.75 , 7.810005,0,0,28.63540334453045 , 77.36752894642822
28.675 , -0.15 , 4.75 , 7.810005,0,0,28.63540334453045 , 77.36752894642822
28.695 , -0.15 , 4.75 , 7.97,0,0,28.63540334453045 , 77.36752894642822
28.715 , -0.15 , 4.75 , 7.97,0,0,28.63540334453045 , 77.36752894642822
28.735 , -0.15 , 4.75 , 7.97,0,0,28.63540334453045 , 77.36752894642822
The full dataset is provided in the cd binded with this report.