Dynamic Mapping of Road Conditions using Smartphone Sensors and Machine Learning Techniques

Shahd Mohamed Abdel Gawad, Amr El Mougy, member, IEEE, and Menna Ahmed El-Meligy
Faculty of Media Engineering and Technology
German University in Cairo
Cairo, Egypt
{shahd.abdelgawad, amr.elmougy}@guc.edu.eg
menna.al-meligy@student.guc.edu.eg

***Abstract*— Road surface conditions can cause serious traffic accidents, often with tragic consequences. Thus, an efficient system for mapping road anomalies can significantly promote the safety of drivers and pedestrians. This paper proposes a novel road anomaly mapping system that is able to detect a wide variety of conditions with high accuracy. The smartphone's accelerometer and GPS sensors are used for detection to minimize infrastructure costs. In addition, to ensure the system is adaptive to different road conditions, pattern recognition techniques are used to automatically calculate the detection threshold. Furthermore, to compensate for GPS inaccuracies, reinforcement learning based on a proposed reward system is used to maximize confidence in the detected anomalies. The reward system is also able to forget anomalies that have been fixed. Moreover, the system is implemented in a distributed way between the smartphone and a cloud server to minimize cellular bandwidth usage, while still retaining the accuracy advantages of a centralized cloud. Live tests have been conducted to evaluate the performance of the system and the results show it is accurate under different driving conditions.**

***Index Terms*— Road surface conditions, pattern recognition, Reinforcement Learning, GPS, Smartphones, Accelerometer**

I. Introduction

Road surface conditions have been identified by several international organizations as one of the leading causes of traffic accidents around the world [1, 2]. Studies show that dynamic road conditions lead to unpredictable driving behavior and deterioration in the state of the vehicles, which may have an economic impact and may cause loss of life in certain situations. The impact is more pronounced in developing countries, due to the poorer status of the infrastructure.

Thus, developing a map of road conditions can have a significant positive effect on driver and pedestrian safety. However, this can be quite a challenging endeavor. First, road conditions may be dynamic. Roads may be affected by weather conditions (icy, slippery, etc.), natural deterioration, lack of regular maintenance, or civil works on underlying infrastructure (for example, maintenance of water pipes). Thus, pot holes or road bumps may appear overnight. Second, regular road maintenance may lead to the disappearance of certain anomalies. Even though this may be less serious, it still has to be considered in the road map to provide drivers with an acceptable level of trust in the system. Therefore, the system has to be dynamic, detecting new road anomalies and removing fixed ones. Third, the system has to achieve a suitable level of accuracy in order to provide drivers with actionable warning messages. Fourth, the map has to be able to detect a wide range of road conditions. This is particularly important in developing countries, where road statuses vary from smooth surfaces to dirt roads. Finally, the system has to be economically feasible to ensure wide deployment. Thus, it should not require significant infrastructure investments or costly vehicle modifications.

In the past few years, researchers have proposed various solutions to some of the aforementioned challenges. These solutions leveraged several technological tools such as road-side sensors, vehicular sensors, and smartphone sensors. Each of these tools has its own merits and drawbacks. For example, the system named "StreetScan" [3] uses vehicle-mounted sensors to monitor the roads efficiently and accurately. The system utilizes acoustic and radar sensors to measure tire vibrations and accordingly detect road surface anomalies such as potholes and bumps by their features. In addition, optical sensors are used to differentiate between the types of defects and confirm the validity of detected anomalies. Even though this system accurately detects road anomalies, it requires costly modifications to the vehicles, which may be prohibitive to some drivers.

Alternatively, a different approach proposed in [4] uses smartphone sensors to collect data, and then analyze this data to detect irregularities. The analysis consists mainly of signal processing techniques applied over the data from the smartphone sensors. First a low pass filter is applied over the signal to remove the low frequencies that might have been produced due to turbulence. Then, a speed filter is applied to remove high frequencies that were detected at vehicle velocity 0 km/hr, which are produced by false movements of the device in stationary state. Finally, a small-peaks filter is applied to remove any frequency below a certain threshold that might have been produced during speeding but do not resemble an anomaly. Afterwards, the produced signal resembles the road surface with the anomalies represented as the high frequencies. However, this approach produced a high percentage of false positives that would reduce the system's accuracy and trust.

To address this drawback, the authors in [5] proposed a second approach which also used a smartphone application to collect data using the accelerometer. This approach first determines a noise value upon opening the application. This value is then used to remove any irregularities within the readings that could have been caused by sudden false movements. Furthermore, readings are recorded with an interval of 1 second, and the difference between the maximum and minimum readings is calculated for each interval. The noise value is then subtracted from the result and compared to a saved threshold. If the value exceeds the threshold, the location with the high reading is recorded as a possible anomaly location. If the location is recorded more than 5 times, it is recorded as a definite anomaly spot. However, the approach relied upon manual input of certain system values such as noise, detection threshold, and the number of times a location needs to be recorded to be considered as a definite anomaly. It would need a large amount of training data and many trial-and-error experiments in order to produce accurate values. Even then, the values may not produce accurate results in all road scenarios.

A different approach that also utilizes smartphone sensors is proposed in [6]. Here, the smartphones' accelerometer sensor is also used to extract information. Then, derivatives for the extracted information are obtained to be used as features to produce a confidence score for each reading to be saved at a backend server. Further processing is done by running a background clustering algorithm over the saved confidence score data. Moreover, a fusing algorithm is applied to sets of readings that have the same location, producing a list of fused scores. A saved threshold is then used to determine if these fused results are anomalies. The entries that pass as anomalies are displayed over a map in the application to inform users of their existence. Although this approach uses derivatives which produce more accurate representation of signals, it still sustains certain inaccuracies. First, during fusion, the readings being fused have to have the same location to be considered as one. However, the smartphone's GPS has an inaccuracy rate of 3.3 meters [7]. Therefore, the system ends up producing many duplicated results, each with a low confidence score. An additional drawback in the system is the manual determination of the detection threshold, which may not be suitable for all scenarios, as mentioned before.

In this paper, we propose a system for accurate mapping of road surface conditions using smartphone sensors, namely GPS and accelerometer, in order to ensure cost-effectiveness. In particular, we address three main challenges: calculation of the detection threshold, detection accuracy of road anomalies (ensuring low false positives), and location accuracy of the detected anomalies. To address the first challenge, pattern recognition techniques [8] are used for automatic calculation of the detection threshold in different road conditions. This ensures that the proposed system works accurately in a wide variety of road conditions. To address the second and third challenges, reinforcement learning is used to accumulate rewards for detected anomalies. Furthermore, we design a reward system that is able to detect when an anomaly is fixed. Finally, we also ensure that the system minimizes mobile bandwidth consumption. This is done by distributing the analysis between the smartphone and the backend server. Thus, the smartphone calculates the detection threshold and analyzes the sensor data to detect anomalies. Only detected anomalies are relayed to the backend server, not the entire sensor output. At the server, reinforcement learning is implemented to improve accuracy.

The remaining sections of this paper are organized as follows: Section II presents the details of the proposed road mapping system; Section III provides key performance evaluation results; while Section IV offers some concluding remarks.

II. Road Mapping System

A. Overview

The proposed system is implemented in a distributed way over two main entities: The smartphones of the users, and a cloud-based backend server. At the smartphones, the functions of the system are implemented in a phone application (an Android application is proposed as a proof of concept). This application utilizes readings from the phone's accelerometer and GPS sensors to perform road anomaly detection. To support the detection mechanism, a perception algorithm [9] is also implemented in the application, and consists of a machine learning technique that calculates the detection threshold. Based on this threshold, the application is able to detect, not only the presence of an anomaly, but also its magnitude. The outputs of the functions of the application are the detected anomalies along with their locations, which are transmitted to the backend server.

At the cloud-based backend server, the detected anomalies are saved and forwarded to all other users of the system, where they are displayed on a map. The phone application also supports notifications to warn drivers of the presence of anomalies in their path. To calculate the confidence level of the detected anomalies, a reinforcement learning algorithm is implemented at the cloud server. This algorithm utilizes a proposed reward function, which is also able to "forget" anomalies which have not been reported for a period of time. This is implemented in order to detect maintenance work that fix road anomalies. The proposed system architecture is summarized in Fig. 1.

B. Detection Algorithm

In this section, the process of detection in the application is discussed in details. Inside the vehicle, the mobile device moves relative to the motion of the vehicle. This allows us to monitor the vehicle's motion using the smartphone's accelerometer, which is a motion sensor that measures acceleration readings in the 3 X, Y and Z axes. Here, the Z-axis readings indicate the vertical acceleration of the vehicle, given that the smartphone is placed horizontally on the dashboard, aligning its z-axis with that of the vehicle. Thus, any road bumps, potholes, or any other kind of road anomaly will be reflected as a jump in the Z-axis reading. This is depicted in Fig. 2.

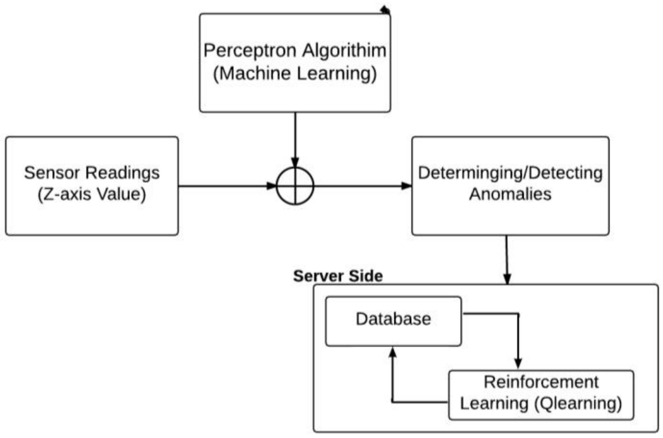

Perceptron Algorithim (Machine Learning)
Sensor Readings (Z-axis Value)
Determinging/Detecting Anomalies
Server Side
Database
Reinforcement Learning (Qlearning)

Fig. 1 Proposed System Architecture

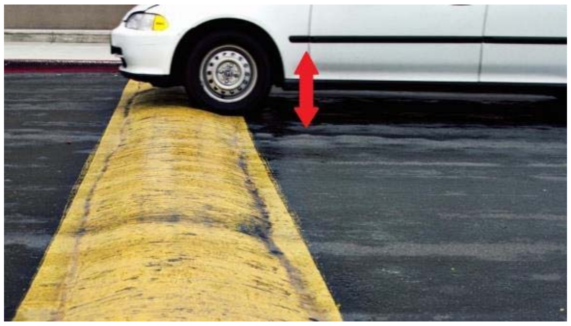
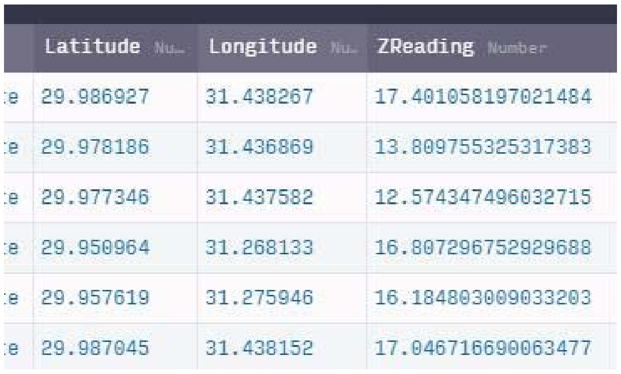


Fig. 2 Road bump causing a vertical change in the motion of the vehicle

As Fig. 2 shows, the car's acceleration in the vertical axis changes with variations of the road surface (for example, road bumps result in an increase in vertical acceleration while potholes result in a decrease in vertical acceleration). If this change is greater than a certain threshold, called the detection threshold, an anomaly is detected. Thus, the acceleration readings are monitored and compared against the defined detection threshold to detect anomalies. The algorithm that calculates the detection threshold is explained later. If a reading is determined to be an anomaly, it is sent to the cloud server along with its location, which is obtained from the smartphone's GPS sensor. A sample of the readings recorded at the server is shown in Fig. 3.



	Latitude Nu..	Longitude Nu..	ZReading Number
:e	29.986927	31.438267	17.401058197021484
:e	29.978186	31.436869	13.809755325317383
:e	29.977346	31.437582	12.574347496032715
:e	29.950964	31.268133	16.807296752929688
:e	29.957619	31.275946	16.184803009033203
:e	29.987045	31.438152	17.046716690063477

Fig. 3 Sample of anomalies recorded at the cloud server

C. *Dynamic Calculation of the Detection Threshold*

The detection threshold plays a critical role in determining the overall accuracy of the system. In order to detect anomalies in a wide variety of road conditions, the detection threshold would have to be dynamically calibrated. This is because an anomaly is detected when there is a large-enough change in the Z-axis readings. The threshold determines how large this change should be depending on the current condition of the road. For example, a smooth road would witness minor variations in the Z-axis readings. Therefore, the required change does not need to be very large to be considered an anomaly. On the other hand, bumpy roads would witness significant variations in Z-axis readings. Here, the change has to be relatively bigger in order to count as an anomaly. Thus, we propose an algorithm, which we named the *Perceptron Algorithm*, to dynamically calculate the detection threshold.

A perceptron is known as a basic unit of neural networks. The perceptron is used to represent neurons that receive "n-inputs", process them and produce a corresponding result. The number of inputs in our case is one since we are only considering the Z-axis readings from the accelerometer. The neural network then uses this input along with some other factors to calculate a "classifier line". This line is considered a splitting line between two groups, namely anomalies and non-anomalies. The line's equation is simply the Z-Reading (input) multiplied by a weight representing the slope of the line, in addition to a bias weight that represents the y-axis intercept of the line. Thus, Eqn(1) is obtained and is used to determine if a reading exceeds the line or not. Any reading above the line is considered as a detected anomaly.

Classifier Line = weight * Z-Reading + bias_Weight (1)

Where *weight* represents the slope of the detection line, *Z-Reading* is the input obtained from the accelerometer, and *bias_weight* represents the y-axis intercept. In order to obtain proper values of "weight" and "bias_weight" that produce the minimum detection error, an iterative training process was implemented. The process starts by collecting a large training data set for the learning procedure from various test drives and manually marking the anomalies along the trips. Therefore, by the end of the training period, a large table is obtained containing Z-Readings along with their anomaly labels, which is then used to train the system to automatically learn anomalies. Two initial values for "weight" and "bias_weight" are chosen using the collected Z-readings to calculate the detection line. Using the calculated line, the accuracy is determined by comparing the detected values to the calculated ones. This is done using Eqn(2).

Error = training_actual_Class – calculated_Class (2)

Where *training_actual_Class* is a Boolean that represents whether the entry is an anomaly or not, while *calculated_Class* is a Boolean that represents whether the calculation is an anomaly or not. If the error is not equal to zero, the values of "weight" and "bias_weight" are modified as follows

Weight += 0.1 * error * Z-Reading (3)

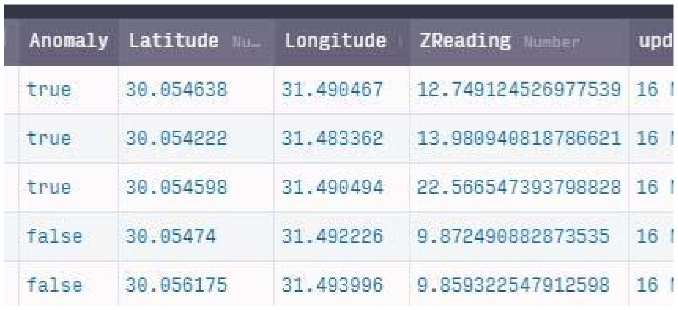
Bias_Weight += 0.1 * error **(4)**

The process is repeated until the error is zero. At the smartphone side, the calculated weights are retrieved from the cloud's database and used to calculate the classifier line.

Instead of awaiting the Z-Reading value to input into the equation and determine the correct class, the equation is reformatted according to Eqn(5). The weights are obtained and used to calculate the needed threshold that would be used to compare against each reading. If a value exceeds the threshold, the record is saved as an anomaly. With this reformation, the load over the client's side is reduced greatly, therefore improving the application's performance.

Threshold = (10- weight) / bias_Weight **(5)**

After such value is obtained, readings are monitored regularly and compared against the calculated threshold. If an anomaly is detected, the location is recorded as an anomaly spot and sent over to the database for processing. In Fig. 4, the resulting readings are shown after the learning algorithm has been applied with the proper classification obtained.



Anomaly	Latitude Nu…	Longitude	ZReading Number	upd
true	30.054638	31.490467	12.749124526977539	16
true	30.054222	31.483362	13.980940818786621	16
true	30.054598	31.490494	22.566547393798828	16
false	30.05474	31.492226	9.872490882873535	16
false	30.056175	31.493996	9.859322547912598	16

Fig. 4 Produced readings after training

After various test rides the accuracy of this algorithm was calculated. This was done by monitoring the count of true positive readings that have been entered into the database in the same location. The accuracy was found to be 70%. Thus, this accuracy value is used as the default confidence value for our detected readings. To improve the users' confidence about the detected events, the following reinforcement learning algorithm was proposed.

D. Accuracy Improvement through Reinforcement Learning

After a user's trip ends, a short process is done at the server side before saving the user's entries. Each entry is checked to determine whether it exists within the range of any other location saved in the database. The location range used is 3.3m, which is the GPS inaccuracy. The action taken by the server depends on whether or not the transmitted reading is within the range of a saved entry.

In the case of a reading that is not within the location range of a saved entry, a new entry is created in the database with an initial reward value of 70% - inferring that the anomaly has a 70% chance of existing, which is our system's initial accuracy level of anomaly detection. However, if the reading is found to be within the range of an entry in the database, the existing anomaly's reward value is updated, reinforcing its existence. This update is done using the following equation.

New_Reward = (new_Count *0.7) + old_Reward **(6)**

Where *old_Reward* is the existing record's reward value and *new_Count* is the number of users that passed by this location and recorded a detection. In order to determine *new_Count*, all vehicles that pass by a location where an anomaly had been previously detected have to transmit their Z-axis reading at this location. Using this approach, the value of *new_Count* would be consistently zero if an anomaly is fixed. If this is detected for a number of vehicles, the entry is deleted from the server's database. On the other hand, if the reward value reaches 100, it remains the same, implying that this anomaly exists with a possibility of 100%.

Furthermore, another mean of accuracy reinforcement is done to reduce the amount of false positives. Sudden external movements to the mobile device could result in an anomaly reading to be saved, however, no actual anomaly exists. In order to avoid such incident an additional check was done over the readings before being sent to the server to be saved. If a car is in a stationary state and an anomaly was detected, the reading would be ignored. With this check, the false positives produced in such cases would not exist, therefore improving the system's accuracy and reliability.

E. Detection of Fixed Anomalies

Having the system continuously updated with the anomalies' location, along with their confidence score, helps authorities locate anomalies. Therefore, information of each recorded anomaly has to be up-to-date, either by reinforcing its existence until it reaches 100% or by decreasing the confidence until it reaches 0% (in case the anomaly has been fixed). This kind of procedure is done regularly as an update over the entire database. Every 48 hours, a background job over the cloud is applied over the recorded entries. The functionality of the job is to reduce the confidence of records that have not been updated in the last 48 hours by 5%. If this case sustains, the confidence score continues to decrease until it reaches 0%. Once this occurs, the anomaly would be deleted from the database and will accordingly disappear from the map, implying that the anomaly/road problem has been fixed.

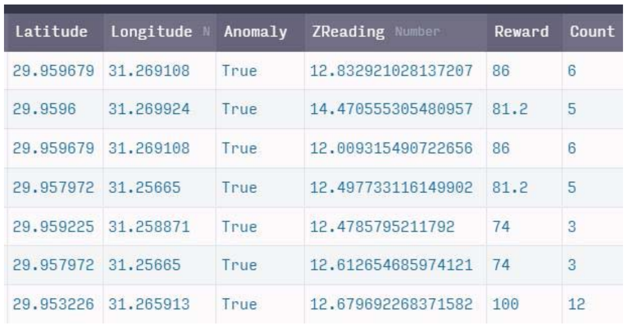
III. Experimental Results

A. Experimental Setup

The proposed system was tested using several types of Android smartphones and in several car models. The smartphone is always placed horizontally over the car's dashboard with the application running. The vehicles are driven over a specified route that has varying road surface conditions (potholes, speed bumps, dirt road). The actual locations of these anomalies are manually marked for comparison with the output of the system.

B. Results

The system was successfully able to detect anomalies no matter the speed of the vehicle during the moment of the incident. A few false entries were recorded; but this was handled by the reinforcement process done over the cloud side. This part of the system allows us to inform the user how confident the software is about each saved record. Therefore, if a false positive has been entered it would have a low confidence score, informing the user that the probability of existence for such record is very low. Furthermore, as time passes with no additional records for such false entry, its reward score would decrease until it is deleted from the database as explained before. Fig. 5 shows the recorded entries in the database. Each entry is saved with its exact geographical location, along with the Z-Reading value produced from passing through the anomaly. In addition, the confidence reward score is updated whenever a user passes over the anomaly along with the count of users passing.



Latitude	Longitude	Anomaly	ZReading	Reward	Count
29.959679	31.269108	True	12.832921028137207	86	6
29.9596	31.269924	True	14.470555305480957	81.2	5
29.959679	31.269108	True	12.009315490722656	86	6
29.957972	31.25665	True	12.497733116149902	81.2	5
29.959225	31.258871	True	12.4785795211792	74	3
29.957972	31.25665	True	12.612654685974121	74	3
29.953226	31.265913	True	12.679692268371582	100	12

Fig. 5 Recorded Entries in the database after testing

In Fig.6 the resulting map is shown, containing the anomaly detected by the system during more than one test drive resulting in an increased confidence score.

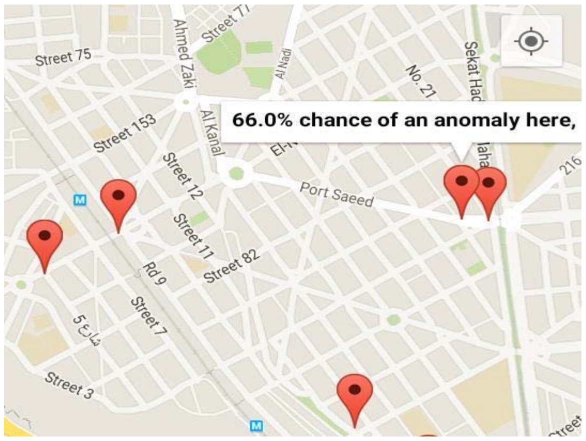


Fig.6 Resulting map after a test drive.

IV. Conclusions

In this paper, a system for mapping road anomalies was proposed. The system utilizes accelerometer and GPS sensors for anomaly detection. To improve accuracy, a pattern recognition algorithm was implemented to dynamically calculate the detection threshold. This enables our system to detect a wide range of road anomalies. Furthermore, a reinforcement learning algorithm based on a proposed reward function was used to compensate for the accuracy limitations of the GPS sensors. The proposed system was implemented in a distributed way between the smartphone and a centralized cloud server to reduce bandwidth consumption. Field tests were conducted to test the accuracy of the system, and results show the system achieves an accuracy that exceeds 70%.

References

[1] World Health Organization (WHO), *Global Status Report on Road Safety,* 2015.

[2] European Union Policy Department B, *EU Road Surfaces: Economic and Safety Impact of the Lack of Regular Road Maintenance,* 2014.

[3] http://insights.globalspec.com/article/2280/vehicle-mounted-sensors-monitor-road-conditions.

[4] A. Vittorio, V. Rosolino, I. Teresa, C.Vittoria, G. Vincenzo, and D. Francesco, "A mobile application for road surface quality control: UNIquALroad," *Journal of Social and Behavioral Sciences,* Vol. 54, 2012, pp. 1135-1144.

[5] A. Vittorio, V. Rosolino, I. Teresa, C.Vittoria, G. Vincenzo, and D.Francesco, "Automated sensing system for monitoring of road surface quality by mobile devices," *Journal of Social and Behavioral Sciences,* Vol. 111, 2014, pp. 242-251.

[6] A. Ghose, P. Biswas, C. Bhaumik, A. Pal, and A. Jha, "Road condition monitoring and alert application: using in-vehicle smartphone as Internet-connected sensor," *IEEE Conference on Pervasive Computing and Communications Workshops,* 2012, pp. 489-491.

[7] http://www.gps.gov/systems/gps/performance/accuracy/

[8] C. M. Bishop, *Pattern Recognition and Machine Learning,* Springer, 2006.

[9] S. I. Gallant, "Perceptron-based learning algorithms," *IEEE Transactions on Neural Networks,* Vol. 1, No. 2, 1990, pp. 179 – 191.