Team 27

Road Roughness Mapping

First Semester Report

Luke Bacopoulos, Aymane El Jerari, Balaji Sathyanarayanan, Jason Wang, Haoming Yi

Executive Summary — Many roads in Boston are littered with potholes, remnants of poor repair work and generally rough surfaces. As a result, vehicle drivers are burdened with huge damage costs and the suspension/bumper of vehicles face significant damages. Vehicle drivers and pedestrians are injured and fatal accidents occur in some cases as well. This calls for a smart system which would grade road roughness and provide the relevant pothole information to the user. The final deliverable will be a system that comprises an application, sensors, machine learning code, and a cloud service which work together to give the user a safe driving experience. The system is briefly described below and outlined in detail in the next few sections of this report. The system will assess road roughness and display the locations of potholes on a clean user interface. The sensors (accelerometer, GPS) and camera in the smartphone collect pothole/location data and the aggregated data is uploaded to the cloud. Multiple machine learning models are trained on the collected data to predict the grade of roughness. The pothole information is displayed on our application integrated with Google Maps. Cloud computing and deep learning are growing fields in the technology industry right now. Enterprises are moving their workloads to the cloud for cheap and effective management of resources. With the advent of big data, neural networks have become prevalent and are used extensively in many of the applications that we use everyday. Our project trains multiple models for learning patterns in the collected dataset and information is stored on a cloud service for secure and easy storage. We value our user's privacy and many privacy features are included in our system for secure data collection.

- *Luke Bacopoulos is with the Department of Electrical and Computer Engineering, Boston University, Boston, MA 02215. Email: baco@bu.edu*
- *Aymane El Jerari is with the Department of Electrical and Computer Engineering, Boston University, Boston, MA 02215. Email: aymaneli@bu.edu*
- *Balaji Sathyanarayanan is with the Department of Electrical and Computer Engineering, Boston University, Boston, MA 02215. Email: balajis@bu.edu*
- *Jason Wang is with the Department of Electrical and Computer Engineering, Boston University, Boston, MA 02215. Email: jaswang@bu.edu*
- *Haoming Yi is with the Department of Electrical and Computer Engineering, Boston University, Boston, MA 02215. Email: yiha@bu.edu*

1 Introduction

Governments invest a huge amount of money in maintaining and improving the country's road transportation system. It is hard to imagine a life without an effective and safe network of roads since commuting is a daily necessity in all of our lives. The roads are in a good condition when they are newly built. But, they become susceptible to damages as vehicles pass over them. The situation gets worse due to extreme weather conditions like heavy rain and snow. Hence, it is understandable that roads are filled with potholes and layered with rough surfaces. As mentioned in the executive summary, potholes and rough roads damage the vehicle and commuters are left with huge repair and maintenance costs. In some cases, accidents and deaths also occur. By and large, potholes pose the problem of leaving commuters with an unsafe and rough driving experience.

The number of vehicles sold in the US is around 12 million in 2021. The roads become rougher and more prone to damages as more vehicles travel over them. The drivers in the US spend around three billion dollars fixing damages caused by potholes. It is possible to drastically reduce the damage costs by monitoring, maintaining, and repairing potholes frequently using a smart system.

The problems outlined above require a solution which would ensure a safe and smooth experience for commuters. Our system aims at locating potholes and uploading the location and severity (grade of roughness) of the potholes to a clean user interface. The pothole information on our software application is regularly updated as we collect more data. Our users can view the pothole data on our app and then choose an optimal and smooth path for their travel. It is not enough if we just locate and display the information on an interface. The relevant authorities must be able to do repairs and close the potholes every once in a while. The authorities visually inspect roads once in three years and do repairs if necessary. An effective pothole detection system is not yet in place for the government authorities to leverage for locating and monitoring potholes. Our system would serve that exact purpose and incentivize the authorities to frequently locate, repair, and monitor potholes.

The user mounts his smartphone on the vehicle's dashboard. The application on the phone collects pothole data (location, accelerometer data, and images) using the smartphone's accelerometer, GPS, and camera. The information is uploaded to AWS (Amazon's cloud platform) for storage and processing. Multiple machine learning models are trained on the collected images for predicting the roughness or severity of the potholes. A times series model is also trained on the accelerometer data to supplement the results from the neural networks. The pothole location and severity are displayed on the application using Google Maps API. The user will be able to explore the map and locate potholes easily.

The React Native application will display pothole data (location and roughness) on a map for the user to view. The user will be able to avoid the roads with potholes and choose the smoothest possible path to the destination. An extra feature is included where the user can manually upload images and locations of potholes encountered while walking or traveling. The government authorities can view the application and send a team to repair the roads. Our system collects new data as more customers use our product and updates the pothole information displayed on the map. The authorities can also flag a particular pothole as repaired through the application once they have covered the pothole.

Even though a number of similar products exist, we strongly believe that our product has the edge. Privacy concerns must be addressed and our product asks for permission before collecting data. The user is in control of the data collection process. Moreover, the user can also input the region within which data collection must take place. The data is uploaded to the cloud and hence, we do not have to worry about storing information on a physical drive. Multiple machine learning models are trained on the raw dataset to extract the relevant and important information regarding pothole roughness.

2 Concept Development

The problem of potholes and rough roads has plagued modern day infrastructure since the beginning of automobiles. It is particularly rough in areas with cold winters, such as New England, and more specifically, Boston. Boston has some of the roughest roads in the US, with little systems in place for efficient maintenance and repairs. Our customer, Professor Alan Pisano, noticed the inconvenience of this issue through his daily commute to Boston University. And it isn't only attributable to potholes. There is currently no way to know if a certain section of a road is currently in an undesirable condition, such as when construction is in progress, leaving large steel plates and otherwise bump-inducing objects or surfaces in the road. It can be very unpredictable, and many commuters would be happy to know whether their daily route contains any disturbances. This specifically may not be the most pressing issue, but it would definitely be nice to know, so you can take an alternative route. Sure, there are existing systems that may be able to warn users about road imperfections such as Waze, but this and similar systems rely on a crowd-sourced model in which the mass of users need to manually input locations where they noticed the rough roads. This leads to other issues such as having people using their smartphones while driving, as well as it being a somewhat unreliable system in the first place. When you rely on the manual input of users, you have to take into account that a percentage of users will simply not do it. This is where an automated system works to solve the issue. If a user's job is to simply activate data collection and leave their phone be, there arises no issues that normally would due to a crowd-sourced system. It is also much safer due to drivers not being distracted by manual inputs.

Circling back to the problematic nature of potholes, cities and governments currently do not have road infrastructure at the top of their dockets. Roads are often left neglected, with little ways to hold these systems accountable to repairs. This can be due to the lack of information available to the cities on the state of roads. The current system Boston uses to inform the cities about rough roads is a very simple system in which you go on a Boston city website to report rough road conditions at a location chosen on google maps. This poses the same problem as described with Waze and similar navigation apps, where it relies on the manual input of users. With an automated system and enough users gathering data, there should be a significant margin of data on rough roads compared to the current data available. With this, it can be brought to city agencies in order to hold them accountable to the maintenance of their infrastructure.

Our client requested that we create a monitoring device that can be attached to any automobile and contains various sensors that can detect the state of the roads the user is driving on. The device would have to be affordable and accessible, with minimal hassle to the user in operating the system. He also noted the importance of user privacy, due to the sensitive nature of collecting a person's GPS data, meaning that the aggregation of data is of utmost importance. Given this information, our team spent the first few weeks brainstorming on how this will be best implemented. We initially focused mostly on our client's request to create a hardware based device, and how we were going to implement this. We considered various sensors currently available, such as accelerometers, cameras, and LIDAR in pairing with some sort of microcontroller such as a Raspberry PI, but soon looked outside of the box to other methods of data collection. This was due to the interest of time, but also because we thought that there may be a more efficient way to implement this system. We came across the idea of scrapping the hardware aspect completely, and transforming the idea into a full software based project. This made sense for a multitude of reasons. First, all of the sensors that we have considered for use are already implemented in devices that are carried by most people, every day: smartphones. Smartphones have streamlined ways of accessing their accelerometers as well as their cameras in application development. This approach also cuts out a "middle man" in the microcontroller, which would most likely have to upload data to a user's smartphone anyway. If not, we would have to implement a system where the user manually uploads their data after a drive, meaning more work on the user end and less efficiency. It also makes the streaming of data much more reliable by cutting out the communication between microcontroller and smartphone. It also saves budget and time immensely. We can spend the resources that would otherwise be used to design a device that most people would already have elsewhere.

A software-implemented system would still have to have a formatted system in order to ensure the proper collection of data. Along with the application, users must be provided with a uniform place to prop their phone while driving. Images from various different locations on the dashboard being uploaded to a machine learning algorithm may lead to improper classification of potholes, which would be a critical failure in our system. Therefore, some sort of introduction or tutorial for the proper (but simple) use of our application must be presented. Being a software based project, the user interface needs to be streamlined and simple enough for anyone to use, with minimal extraneous devices or items needed for use. Our technical engineering requirements for the project are outlined in **Appendix 1**.

3 SYSTEM DESCRIPTION

Our system starts with the user placing their smartphone in a rigid dashboard mount with the camera facing to the road in front of them. The application, which will be free for users to download, will display both a map of their location, markers from previous data collection available for viewing, and a button that begins the collection of data. The user may allow the app to run in the background while this data collection is activated. As the user drives along their normal route, our system will be collecting various data through various means. First, the smartphone's camera will take photos of the road at a specified interval, and upload these images to our database. Meanwhile, accelerometer data will be streamed at another specified interval along with latitude and longitude coordinates. These values will then be uploaded to a cloud storage service provider in order to train the machine learning algorithms. Our project's machine learning component will consist of two sets of models trained using two data modalities. The first set of models consists of convolutional neural networks (image segmentation, object detection) that are trained using a dataset of pothole images. The second set consists of a model that is trained using three channel time series data collected from the smartphone's built in accelerometer. Given our client's requirements for the project, the time series model is the more important one, as it allows us to translate from user accelerometer data to a road roughness metric. Our pothole recognition model is a secondary feature that can be helpful in flagging pothole GPS coordinates when the weather allows it. The images will be evaluated, and each photo will be classified as containing a pothole or not. Meanwhile, the stream of accelerometer data will be checked for abnormalities, which may occur in the X, Y, or Z directions, depending on the orientation of the smartphone. Abnormalities will be graded on a scale. Classified images as well as graded accelerometer data at specified locations will then be uploaded back to the application, in which a function will accept the roughness grade of positively classified locations as well as their corresponding latitude and longitude coordinates. These values will then be used in order to place markers on the front-page map for easy viewing by any user. As for the user interface, it will most prominently contain the map with markers easily viewable. It will also contain a button which is used to begin and stop the collection of data. A concept image of our UI can be found in *Appendix 3: Figure 4*. Our application is written in JavaScript using the REACT-Native framework, with data streaming provided by Amazon Kinesis. We will use Amazon S3 to store the data, and data will be deleted upon negative classifications. This will allow for more efficient use of our system, and will leave room for the unpredictable nature of road roughness, as the state of a road on a person's daily commute can change overnight. A block diagram of our system can be found in *Appendix 3: Figure 5*.

4 FIRST SEMESTER PROGRESS

The pothole images collected have to be processed and trained to predict the severity of the potholes. Some of the images collected will contain pothole(s) and those images have to be identified using a deep neural network. We implemented and trained an Inception V3 image classifier to classify images as pothole images or not. The neural net was trained on a different dataset to ensure functionality and accuracy metrics were printed out. A good pothole dataset for image classification could not be found and hence, we trained image segmentation and object detection models.

The expectations we had set for ourselves at the beginning of the semester and the team contract were exceeded by the result we were able to achieve for this class. In fact, we successfully trained a convolutional neural network using the pothole600 dataset we found online.

First, we wrote code to automatically organize the dataset and prepare it for the training step. We also added functions to augment the data from ~600 images to more than 6000, effectively increasing the dataset by a factor of 10. Doing so not only allowed us to generate more training data, but also increased the data variance. By introducing random gaussian blur and rotations to the images, the model would have to learn the inherent image features that represent a pothole.

We trained the Feature Pyramid Network using resnet32 to encode the input images. The training has shown to be successful as the current IOU metric is ~0.86. After performing a training, validation, and testing split (80%, 10%, 10% respectively) I used the testing images that the model has never seen before. As we can see the model predictions are almost perfect replicas of the ground truth.

We believe that it would be possible to make the model reach an IOU in the nineties, but we are currently investigating the possibility of running these models on edge smartphones. Theoretically speaking, FPN's 24 million parameters would require 4 bytes per parameter * 24 million parameters = 100 megabytes of storage. We believe that the model could easily fit on any modern smartphone.

The model was trained on a dataset of closeup images of potholes. This means that the model was only able to learn the image patterns specific to the dataset. We can see that the model does not perform very well with outlier input data. The model has a hard time with sidewalks, white crosswalk paint, etc... Due to the limited nature of the dataset, the model did not have the chance to be exposed to a wide variety of situations. An example output of our model can be found in *Appendix 3: Figure 3*.
These accomplishments are significant because they lay the foundation of this project's computer vision component. With a fully trained pothole recognition model that uses around 100MB, we can run inference locally on the user's mobile device. Furthermore, this success allows me to focus more on the second time series deep learning model that will be able to classify road roughness using 3 channel accelerometer data.

We are training another image segmentation model and an object detection model for predicting the grade of roughness of potholes. These models would serve as a backup to the initially implemented feature pyramid network.

In terms of the mobile app, we're aiming to achieve the collection and storage of three types of data: photographs, accelerometer readings and GPS coordinates. We've separated the 3 functions into three apps for now so we can better divide up the work and keep each section separate.

For the photographs app, it collects the photographs at a predetermined rate and the photographs are stored locally on the phone within the app. As the phone is mounted on the windshield of the car, the app will be able to record the photographs of the road and eventually that data will be fed into the machine learning algorithm to determine the existence of a pothole.

For the accelerometer app, it collects the accelerometer readings in real time and the readings are displayed on the app. While running, the app will be able to record the accelerometer readings throughout the collection period and will be fed into a different machine learning algorithm that analyzes the peaks and valleys of the data plotted against time and determine when exactly a pothole is hit.

For the GPS app, it collects GPS coordinates once in a while and corresponds the coordinates to the photographs and accelerometer readings. These will help determine where exactly the potholes are and with the help of the Google Maps API, we will be able to show our road roughness data on a map which will help the commuters to choose a best path, not unlike the traffic filter on Google Maps.

5 TECHNICAL PLAN

Task 1. Integrate mobile apps
The current photographs app, accelerometer app and the GPS app should be integrated into one single app with one single repository. The app should display all data in an orderly fashion and store the data in an easily accessible way.

Task 2. Set up cloud storage
The cloud storage should be set up so that the now unified data collection mobile app will be able to upload the data onto the cloud with an API provided by the cloud services. The data stored on the cloud needs to be organized and easily retrieved for later analysis.

Task 3. Start data collection
The data collection app, once connected to the cloud, will be distributed to the faculty who will be helping us collect road roughness data, The app will be distributed on both Android and IOS for maximum audience. The faculty will collect the data as part of their daily commuting routine with no extra work.

Task 4. Train accelerometer ML algorithm
A machine learning algorithm will be trained to process the accelerometer data by looking at the trends and outliers to determine when exactly a pothole or other road imperfections are hit. This algorithm will train on datasets found on the internet first, and then fine-tuned with the data we've collected.

Task 5. Start data analysis
The accelerometer ML algorithm, once tuned, along with the photograph ML algorithm will start analyzing the data collected on both sensors, and this process will be automated with human supervision. The final conclusive results regarding when and where a road imperfection is hit will be generated.

Task 6. Adjust ML algorithms
While analyzing the data, the ML algorithms might not be able to handle the real world data as smoothly as hand-picked online datasets. Therefore, to make the algorithms more robust, they will be further adjusted and fine-tuned until a satisfactory result is guaranteed. The evaluation of the algorithms will be done by human effort.

Task 7. Composite processed data
The classification results from the ML algorithms will be composited into an online database, which will allow both humans and our final mobile application to access with ease and minimal lag.

Task 8. Build user-end app
The user-end app will be built using the Google Maps API, allowing us to overlay the processed road roughness data onto a map. Graphically, this will look like any other filter that the users can turn on on Google Maps, such as the real-time traffic filter. The app will access the data we've processed and stored onto the cloud.

Task 9. Test user-end app
The user-end app, once finished, will be presented as the final product of our senior design project. We will test the app by using it on a car driving around Boston. Ideally, this app will show us the road roughness and allow us to pick a route and enhance the driving experience.

6 BUDGET ESTIMATE

We will be utilizing Amazon's S3 Simple Storage Service to store our accelerometer data and pictures in preparation to be trained. We estimated that 5 TB would be sufficient, and thus at $0.023/GB, the total pricing comes out to $115.

For faculty members who will be testing our product, their phones must be mounted on the vehicle's dashboard in order for the camera to take photos. We decided to buy one phone mount for each member that will be part of the testing period. At $20 per mount, the total pricing comes out to $240.

A table outlining the projected budget can be found in *Appendix 3: Figure 6*.

7 ATTACHMENTS

7.1 Appendix 1 – Engineering Requirements

Requirement	Metric
Accessibility	System limited to two physical items: smartphone, mount
Privacy	No data is traceable to an individual user.
Correct Classification	ML metrics correct >70% of the time
Data	System able to store at minimum 5 Terabyte of data
Map Accuracy	Markers displayed within 3 meters of actual location

Figure 1 – List of engineering requirements

7.2 Appendix 2 – Gantt Chart

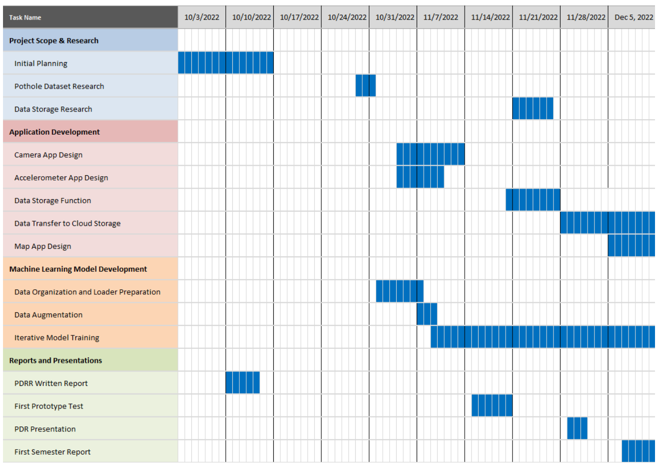

Figure 2 – Gantt Chart

7.3 Appendix 3 – Other Appendices

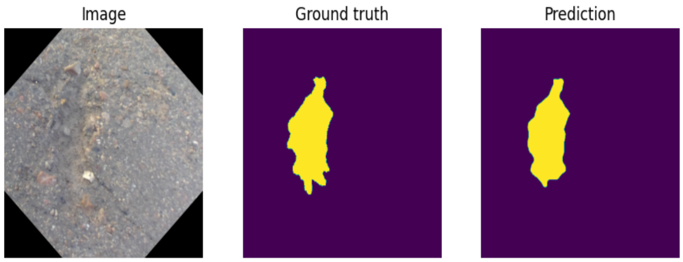

Figure 3 – Classification Example

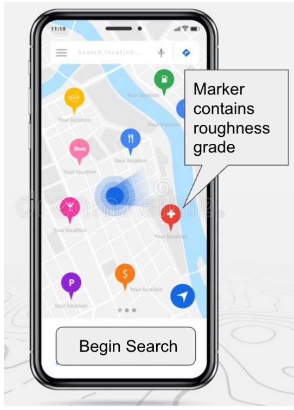

Figure 4 – User Interface Concept Image

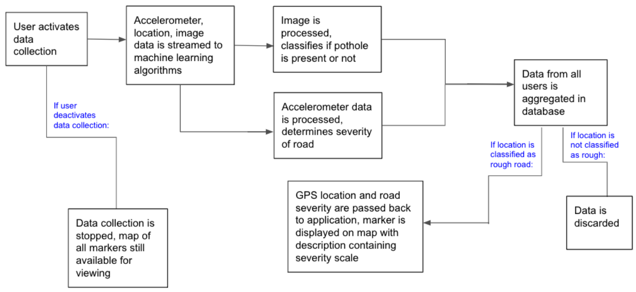

Figure 5 – System Block Diagram

Item	Description	Cost
1	AWS Storage ($23/TB)	$115
2	Dashboard phone mounts	$240
	Total Cost	$355

Figure 6 – Projected Budget

REFERENCES

[1] Matt has been a car fanatic since he was young hence why his favorite video game series is Gran Turismo. He loves writing about any and all things automotive. But he loves to help people the most. "How Many New Cars Are Sold Each Year? [Updated 2022]." AxleWise, 25 Aug. 2022, axlewise.com/car-sales-stats/.

[2] "Pothole Damage Costs US Drivers $3 Billion a Year." QuoteWizard, https://quotewizard.com/news/pothole-damage-costs-us-driv

ers-3-billion-dollars-per-year#:~:text=According%20to%20AAA%2C%20U.S.%20drivers,fixing%20damage%20caused%20by%20potholes.