**Boston University**

**Electrical & Computer Engineering**

**EC464 Senior Design 2**

**Second Prototype Test Report**

**EZRider**

**by**

**Team 27**

**Team Members:**

Aymane El Jerari

Luke Bacopoulos

Jason Wang

Haoming Yi

Balaji Sathyanarayanan

**Equipment and Setup**

We purchased 5 vehicle dashboard mounts for accelerometer data collection. One of the dashboard mounts was placed on the table for testing purposes. A smartphone was attached to the mount to simulate the capabilities of our React Native application.

The application displayed the accelerometer data with the help of the smartphone’s built-in accelerometer. The data was stable (no observable numerical variations in the x, y, and z axes). The device was subjected to manual forces (shaking, pushing, pulling, etc.) while being fixed on the mount. Consequently, we could observe new accelerometer values on the application. This situation is representative of what would occur when the vehicle goes over a pothole on the road.

The equipment and setup are divided into two parts: the smartphone will be placed in its designated dashboard mount with the app opened, a laptop will be running the expo server to host the app, and a second laptop will be used to run a script that pulls data from our s3 bucket; the Python notebooks and scripts implementing the image classification and object detection models. The smartphone collects accelerometer data after the application user approves the data collection process. The data is uploaded to the S3 bucket on AWS. A python script downloads all the data from the S3 bucket for training the machine learning model. The ML model is trained on the accelerometer data and the YOLO object detection model is trained on image data found online. The severity and location of the potholes are presented on a map running on our mobile application.

We also illustrated the characteristics of the image datasets used for training the Inception V3 and YOLO neural networks. The code implementing the models was made available on Google Colab notebook and Github.

**Detailed Measurements taken**

For the accelerometer data, we measured the success of our application by looking at two factors: if we could collect data manually using newly implemented buttons, and if data could be successfully uploaded and subsequently downloaded from our S3 bucket. We set up an iPhone in a rigid mount, and tested out pressing “start recording”, then “stop recording”, then both “delete data” and “upload data”. Upon checking the S3 bucket, it showed that the app had acted correctly, only saving one json file of data taken in the interval where we had pressed start, stop, and upload. The next step was ensuring the data could be downloaded locally to any computer to be trained, so we took 10 measurements of accelerometer data and ran the python script we wrote on two computers. Both computers showed 10 downloaded json files with the same data sets that were in the corresponding files in the bucket.

Currently, our application collects only accelerometer data. We are working on implementing the image collection feature for the app. We trained 3 neural networks (image segmentation, Inception V3 image classification, YOLO object detection) on image datasets found online. We tested the neural networks by analyzing its accuracy on the test data. For example, the Inception V3 net gave 91% accuracy on the test dataset with binary cross entropy loss. Once our application is ready to collect images, we will test and inference our models on the collected images.

The datasets used for training were shown to the graders. A few samples from the dataset (positive and negative samples) were produced and the characteristics of the dataset (number of samples, lighting, size of images, etc.) were highlighted.

**Conclusions**

Our application was able to successfully collect accelerometer data and subsequently upload it to our S3 bucket. The data is downloaded from the bucket to any local computer by running a simple python script. We demonstrated extended capabilities (“start recording”, “delete data”) of our application and showed our clean user interface with a seamless data collection feature.

The datasets used for training the image segmentation, image classification, and object detection models were shown. The models were successfully trained and produced more than 80% accuracy on the corresponding test datasets.

As shown on the [GitHub](https://github.com/aymane-eljerari/pothole-localization) repository, we are aggregating different datasets to finetune an object detection model to identify potholes from images. Our YOLO model should be able to draw a bounding box around the pothole in images taken from a car’s dashboard.