Senior Design Project Final Report for Year 2023

Pothole Detection System

Braeden Kurz (Computer Engineering)

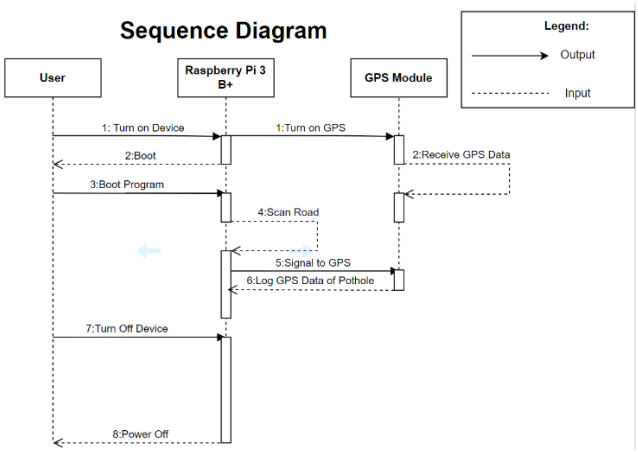
Faculty advisor: Dr. Regentova

1. Introduction

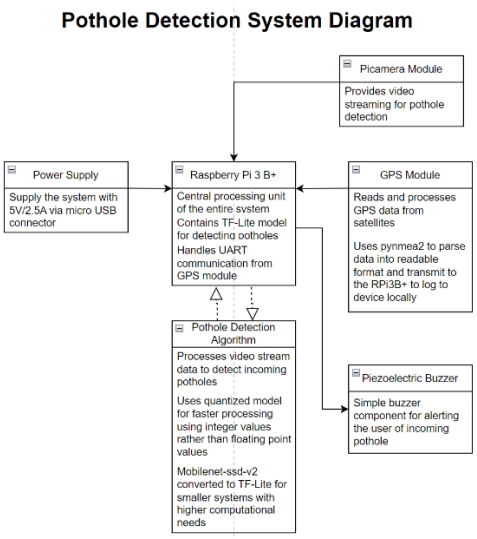
Potholes are a common problem on roads and highways, they can cause significant damage to vehicles, lead to accidents, and increase maintenance costs. Potholes are often difficult to detect, especially at high speeds or in poor lighting conditions. Drivers may not see potholes until it is too late, which can cause a safety hazard for drivers, passengers, and pedestrians. Potholes can also be inadvertently costly for cities and municipalities due to possible significant damage to roadways and infrastructure caused by traffic accidents if left untreated.

To address these issues, I am proposing a pothole detection system to help identify and locate potholes in real time. This system uses the TensorFlow-Lite (TF-Lite) machine learning library combined with a Raspberry Pi 3 and Raspberry Pi Camera Module to detect and track potholes, a piezoelectric buzzer to audibly alert drivers of their presence, and a NEO 6M GPS device to acquire global positioning system (GPS) data and log the geographical location of detected potholes locally to the device using universal asynchronous receiver transmission (UART) communication. By alerting users early, the Pothole Detection System can help prevent major accidents caused by potholes, and logging their geographical locations can help city management authorities prioritize and plan repairs more efficiently.

2. Proposed Solution and Accomplished Design



The diagram above is the updated sequence diagram of the system, which displays the order of events that occur when the system is operational. When power is supplied to the device the RPi module starts its boot process while simultaneously turning on the GPS device. While supplied with power, the GPS module searches for a satellite to fix onto to start reading relevant GPS information (we will know it is being tracked when the GPS module LED starts blinking). Next, the pothole detection program is loaded into RAM. The buzzer will beep for 100ms at first, indicating the PDS is operational and ready for detection! The PDS enters a program loop that continuously scans the road for potholes while the GPS module receives satellite data. Once a pothole is detected, the object detection script will label and draw a bounding box around the detected pothole with a confidence level indicating the level of certainty the object detected is a pothole, simultaneously grabbing the raw data from the GPS module and parsed into NMEA format and logs the longitude and latitude coordinates into the device locally. This program loop continues until the user decides to power off the device, shutting down the RPi, camera, and GPS modules.



Originally, I proposed to design the Pothole Detection System (PDS) using the YOLOv4-Tiny object detection library, but after testing the trained model, the frame rate (frames per second or FPS) of the program operated around 0.7 FPS, which is too slow for the program to operate. There is a tradeoff between speed and accuracy when designing object detection models, but since the Raspberry Pi 3 B+ is computationally limited, speed was a higher concern for this system. Since the PDS required a higher frame rate to operate properly, the SSD-Mobilenet-V2 model was chosen to replace the proposed YOLOv4-Tiny model, as it is employed on many embedded systems, can detect a higher number of objects compared to YOLO, and can achieve accuracy rates as high as 75.9 % mAP [1]. The camera is connected to the Raspberry Pi 3 Model B+ to record the video stream while the device remains on. The GPS module is constantly receiving GPS data from satellites while the system remains active but will not log data unless a pothole is detected. On startup, a piezoelectric buzzer will sound, indicating the system is ready

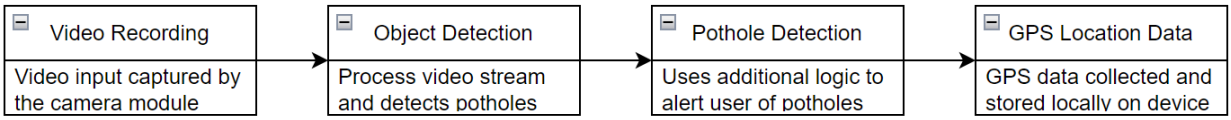
for use. If a pothole is detected by the camera within 10 meters, the device alerts the driver by activating a piezoelectric buzzer for 100ms. The GPS data received from the GPS module is then recorded, parsed, and logged to the device locally in a file named ‘pothole-log.txt’, which contains the longitude and latitude coordinates of the detected pothole. Since object detection is computationally demanding, a GPIO fan is installed to provide air circulation to the system to cool down when the temperature reaches above 65 degrees Celsius to limit power usage. Below are the proposed and implemented functions of the final design:

PDS Functionality

Proposed Functions	Final Design Implementaions
Detect potholes using a machine learning model and drawing bounding boxes around them.	Yes. A dataset containing nearly 700 pictures of well maintained roads and roads with potholes was used to train
Alert driver when a pothole is detected.	Yes. A piezoelectric buzzer is activated when a pothole is detected above a given confidence threshold (normally 0.5, but the threshold can be adjusted)
Log GPS data to a cloud-based server for further analysis.	Unfortunately, I was unable to incorporate the GSM module in my design due to complications in mobile communication, time constraints, and higher costs. Instead, the system uses a GPS module to log longitude and latitude data locally to the device.
Allow night-time detection capabilities	
Update travel route on pothole detection	As mentioned above, the GSM module was scrapped due to time, complexity, and high cost, so this function was not implemented. Since the main focus of the PDS was speed, this functionality would increase latency, thus slowing the frame rate of the system.
Detect other possible hazards (object collision detection, flash floods, fires, etc.)	Due to time constraints, this was not implemented. This could have been implemented by adding images of traffic hazards to the dataset and adding more labels to the labelmap.txt file (a file that lists the names of objects we want to detect). However, this would increase training time and, consequently, increase hardware usage. A larger dataset, depending on the images used, could possibly reduce the

	accuracy of the system as well.

Function breakdown and descriptions



Specification and design constraints

Purpose: A pothole detection system that alerts the driver when an upcoming pothole has been detected using a night vision camera and Raspberry Pi 3 Model B+ and uploads GPS location data of the pothole to a cloud database for reference and data visualization.

Behavior: PDS should begin scanning for potholes once powered on. The system is pre-trained with images of various potholes to accurately detect and alert the driver. PDS estimates the size of detected potholes (roughly 1ft-5ft wide) in every frame it records.

System Management Requirement: The system should provide object detection, GPS, and embedded systems functionality.

Data Analysis Requirement: The PDS uses a trained SSD-Mobilenet-v2 custom model to run an inferencing script to detect potholes by displaying a bounding box around them and the confidence level of its detection. The GPS module will use UART communication to transmit satellite data from the GPS module to the RPi for parsing, local logging, and displaying the data in the terminal.

Table of specification & design restrictions

Function module	Specification (use bullets)	Why chooses this specification?
Camera Module	• **Input Power:** 1.5V to 3.0V DC • **Peak Current:** 300mA • **Sensor:** OV5647 • **Connection:** 15 cm flat ribbon cable to 15-pin MIPI Camera	• Necessary to operate the camera module • Same as previous • Primarily daytime use • Cable necessary for interfacing on the Raspberry Pi 3 Model B+ • Allows for crisper images to capture. • Images need not be large • The potholes need to be detected at

	Serial Interface (CSI) connector • **Lens:** Fixed Focus Lens • **Angle of View:** 54° Horizontal x 41° Vertical • **Field of View:** 2.0 x 1.33m @ 2m • **Resolution:** 5 Megapixels, 2592 x 1944 • **Frame Rates:** 30fps@1080P, 60fps@720P, 90fps@480P • **Fixed Focus:** 1m to Infinity • **Dimensions:** 4.7 inches x 0.2 inches x 0.1 inches • **Weight:** 0.32 ounces **Operating Temperature:** -30 to+70 °C	a reasonable distance
Raspberry Pi 3 Model B+ Microcontroller	• **Processor:** Broadcom BCM2837B0, Cortex-A53 64-bit SoC @ 1.4GHz • **Memory:** 1GB LPDDR2 SDRAM • **Connectivity:** 2.4GHz and 5GHz IEEE 802.11.b/g/n/ac wireless, LAN, Bluetooth 4.2, BLE, Gigabit Ethernet over USB 2.0 (maximum throughput 300Mbps), 4 × USB 2.0 ports • **Access:** Extended 40-pin GPIO header • **Video & Sound:** 1	• **Processor:** Necessary for quickly processing signals • **Memory:** Storing the OS • **Connectivity:** Connecting to the internet to send GPS coordinates of detected pothole to cloud • **Access: N/A** • **Video & Sound:** Necessary for interfacing with the Raspberry Pi 3 Model B+ to install the OS and camera module • **Multimedia:** Necessary for interfacing the camera module • **SD Card Support:** The SD card contains the OS for operating the Raspberry Pi • **Input Power:** The Raspberry Pi needs this amount of power to function • **Dimensions:** Small and lightweight for easy transportation • **Weight:** Small and lightweight for easy transportation

	× full size HDMI, MIPI DSI display port, MIPI CSI camera port, 4 pole stereo output and composite video port • **Multimedia:** H.264, MPEG-4 decode (1080p30); H.264 encode (1080p30); OpenGL ES 1.1, 2.0 graphics • **SD Card Support:** Micro SD format for loading operating system and data storage • **Input Power:** 5V/2.5A DC via micro USB connector, 5V DC via GPIO header, Power over Ethernet (PoE)–enabled (requires separate PoE HAT) • **Dimensions:** 85 x 56 x 17 millimeters • **Weight:** 7.1 ounces • **Operating Temperature:** 0 to +50 °C	• **Operating Temperature:** Safe operating temperature to avoid damaging the board and harming users
GPS Module	• **Input:** 2.7V to 3.6V/10mA DC • **Output:** Raw GPS data • **Protocols:** National Electronic Marines Association (NMEA) 0183 • **Interfaces:** UART • **Time-to-First-Fix:** Cold-Start: 27s	• **Input:** The necessary power requirements to operate the GPS module • **Output:** This raw data will be parsed using the pynmea2 Python library to log the necessary data to our device • **Protocols:** A common data format used by global navigation satellite systems (GNSSs) that provide positioning, navigation, and timing services • **Interfaces:** This a the protocol we

	Warm-Start: 27s Hot-Start: 1s • **Default Baud Rate:** 9600 • **Dimensions:** 12x16mm, 250pcs/reel	use to communicate with the RPi module • **Time-to-First-Fix:** These are the min/max average times it takes for a satellite to fix onto the GPS module. It is desirable to reduce this time as much as possible, which can be done by taking the device outside in open-air rather than indoors. • **Default Baud Rate:** This is the baud rate we want to set the RPi serial port to so we can receive data at the right time • **Dimensions:** Necessary for producing a lightweight product
Power Supply (Micro USB)	• **Input:** 120 ACV (indoor use) or 24 DCV (car adapter) • **Output: 5.**0V/2.5A DC	• **Input:** The input voltage depends on where the user intends to utilize this system, whether indoors (for testing the functionality of the system) or outdoor use (using the car adapter or connected to a portable battery pack) • **Output:** The PDS requires a micro USB cable to connect to the Rpi3B+ module and the power supply must provide a 5.0V/2.5A DC supply for it to work properly
Entire prototype (All Components w/ Case and 5.0V/160mA DC Fan)	• **Input:** User credentials, video camera recording, power supply (Micro-USB), GPS satellite data • **Output:** Video stream w/ bounding boxes drawn around potholes detected, audible alarm from buzzer, GPS satellite data written to potholes-log.txt file locally • **Power Requirements:** 5V, 2.5A DC • **Data**	• **Input:** The RPi requires 5.0V/2.5A with a micro USB, which is necessary to operate the RPi module. The user must login to operate this device. The camera module needs to provide a constant stream to the RPi module to detect potholes. The GPS module begins reading data once it is tracked by a satellite (when the module is blinking a blue LED) • **Output:** The PDS provides a video stream when in operation, drawing bounding boxes around suspected potholes and labeling them with the label ‘Pothole’ and a confidence level (a percentage rating of how certain the system is in defining a suspected pothole). The system needs to alert drivers of upcoming

	Transmission Protocol: RS232 and UART TTL • **Pothole Detection Range:** ~10 meters • **Weight:** 7.2 ounces • **Constraints:** There should be enough light to detect the pothole and the vehicle should be moving below 45mph for the system to get a good quality video of the road.	potholes and their locations by activating the buzzer and logging their longitude and latitude coordinates to the device • **Power Requirements:** 5.0V/2.5A DC • **Data Transmission:** UART is used for the GPS module since the RPi and GPS module operate at difference clock rates, providing ease of data transmission • **Pothole Detection Range:** Early detection of potholes helps warn users with enough time to react • **Weight:** We want the PDS to be lightweight and easy to transport from place to place, whether it is used in cars, biking, walking, etc • **Constraints:** The camera needs enough light and speed to detect incoming potholes

Table of employed libraries/datasets/open sources:

Resource name	Resource type & description	Link or References
Pynmea2 Python Package	This Python library is used for parsing individual NMEA sentences into a NMEASentence object. This library is used in this project for parsing longitude and latitude data into a readable format that can be logged into a text file named 'pohtole-log.txt' on the device locally.	https://github.com/Knio/pynmea2
Opencv2 Python Package Version 3.4.11.41	An open-source Python library that includes computer vision algorithms. This library is essential for the pothole detection functionality.	https://github.com/opencv/opencv-python
GPSD – A GPS Service Daemon	A service daemon that monitors one or more GPSes or an Automatic Identification System (AIS) receiver. We use this	https://gpsd.io/index.html

	software to test the transmission of data between the GPS module and the RPi module and review its navigational and positional data.	
Raspberry Pi TensorFlow Package Version 2.8.0	An open-source software library for high-performance numerical computation across GPUs, CPUs, and TPUs. Real-time object detection is computationally demanding and TensorFlow allows training and inferencing processes to perform quickly. Version 2.8.0 is used in particular since the more recent versions have errors with other libraries.	https://github.com/tensorflow/tensorflow/releases/tag/v2.8.0
Picmaera Python Package	A software library that supports enumerating all camera devices available in the system. This library is necessary for operating the Picamera module	https://libcamera.org/docs.html
SSD-Mobilenet-v2 Machine Learning Model	A machine learning model based on Mobilenet V1 model that provides greater accuracy and speed on embedded platforms with limited resources. The model provides the necessary components for pothole detection.	**Library Download:** https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf2_detection_zoo.md **Supporting Documention:** https://ieeexplore-ieee-org.ezproxy.library.unlv.edu/stamp/stamp.jsp?tp=&arnumber=9219319
Pyserial Python Package	A software library that provides a backend for Python running on the Raspberry Pi Linux operating system. The library is used in this project to access the ttyAMA0 Serial0 port on the RPi module to receive incoming GPS data from the GPS module.	https://github.com/pyserial/pyserial/
Edje Electronics Tensorflow Object Detection Repository	A Github repository that provides the necessary libraries and step-by-step process to train and deploy a lightweight deep-learning model.	https://github.com/EdjeElectronics/TensorFlow-Lite-Object-Detection-on-Android-and-Raspberry-Pi
NEO-6M GPS	The author provides the base	https://sparklers-the-

Module with Raspberry Pi by Arjit Das	code for GPS setup and parsing raw satellite data.	makers.github.io/blog/robotics/use-neo-6m-module-with-raspberry-pi/
Confusion Matrix Generator	This repository was used to calculate the confusion matrix of the system with 69 test images. The repository also includes useful information, including accuracy, precision, recall, F1-Score, and so on.	https://github.com/DamianoP/confusionMatrixGenerator
Kaggle	Creating the dataset for detecting potholes. The dataset is divided into two categories: normal or well maintained roads and road that contain potholes. This division will help the system differentiate between well maintained roads and alert the user for potholes.	https://www.kaggle.com/datasets/atulyakumar98/pothole-detection-dataset

Table of employed standards (related to safety, materials, workload, power rate, data transmission, etc.):

Standard	Description	Link or References	Why this standard is employed/necessary?
IEEE Standard for Sensor Performance	This standard ensure that sensor performance data is consistent and accurate, which is important for many applications, including industrial process control, environmental monitoring, and medical diagnostics.	https://standards.ieee.org/ieee/2700/6770/	It helps to ensure that sensors are properly characterized and that their performance is well understood, which can lead to improved system performance and reliability.
Standard for Harmonization of Internet of Things (IoT) Devices and Systems	This standard defines a method for data sharing, interoperability, and security of messages over a	https://standards.ieee.org/ieee/1451.99/10355/	The standard defines a communication protocol and data model that allows for the exchange of information between smart transducers in a wireless network,

	network, where sensors, actuators and other devices can interoperate, regardless of underlying communication technology.		regardless of the manufacturer or communication technology used.
IEEE Draft Standard for Automotive System Image Quality	The standard provides a framework for evaluating various aspects of image quality, including resolution, color accuracy, noise, dynamic range, distortion, and more. It also includes guidelines for selecting appropriate test charts and equipment, as well as instructions for conducting tests in different lighting conditions.	https://standards.ieee.org/ieee/2020/6765/	Mainly used in the automotive industry to ensure that their imaging systems meet the necessary performance requirements for safety and reliability.
CC BY 4.0 License	Anyone is free to share, copy, and redistribute a work in any medium or format, and to adapt, remix, transform, and build upon the work for any purpose, even commercially. However, they must give appropriate credit to the original creator(s) and provide a link to the license.	https://creativecommons.org/licenses/by/4.0/legalcode	This license allows others to share, copy, and redistribute a work in any medium or format, and to adapt, remix, transform, and build upon the work for any purpose, even commercially, as long as they give appropriate credit to the original creator(s). This ensures that the original creator(s) receive proper recognition for their work, while also allowing others to benefit from and build upon it.

Table of hardware/budget use and all costs

Part Description	Function	Amount Needed / Unit Price	Subtotal	Purchase Link	Datasheet Link
Raspberry Pi 3 Model B+	Central controller for reading sensor data, alerting the driver and passengers of an upcoming pothole, and uploads the data to a cloud database for future reference and data visualization.	1/$35.00	$35.00	https://www.adafruit.com/product/3775?src=raspberrypi	https://datasheets.raspberrypi.com/rpi3/raspberry-pi-3-b-plus-product-brief.pdf
Arducam 5MP Camera Module OV5647	Camera module for scanning the road and detecting potholes in low-light conditions	1/$9.99	$9.99	https://www.amazon.com/Arducam-Megapixels-Sensor-OV5647-Raspberry/dp/B012V1HEP4?ref_=ast_sto_dp&th=1&psc=1	https://docs.arducam.com/Raspberry-Pi-Camera/Native-camera/source/OV5647DS.pdf
HiLetgo GY-NEO-6M GPS Module, 3V-5V w/ Ceramic Antenna	GPS module with ceramic antenna. Uses UART communication to	1/$8.99	$8.99	https://www.amazon.com/gp/product/B01D1D0F5M/ref=ppx_yo_dt_b_search_asin_title	https://content.u-blox.com/sites/default/files/products/documents/NEO-6_DataSheet_%28GP

	communicate with RPi and log positioning data of detected pothole and log it onto the device			?ie=UTF8&psc=1	S.G6-HW-09005%29.pdf
Jumper Wires (4in and 8in pack)	Connecting the modules to their appropriate pins to perform pothole detection and data uploading.	1/$6.49	$6.49	https://www.amazon.com/dp/B01L5ULRUA?ref_=dp_atch_dss_base_image	N/A
Cylewet 10Pcs 5V Active Buzzer	Buzzer component for alerting the driver of upcoming pothole	1/$6.98	$0.14	https://www.amazon.com/gp/product/B01N7NHSY6/ref=ewc_pr_img_1?smid=A2O4FZXIRZDLHA&psc=1	N/A
Dorhea Raspberry Pi Case Holder w/ Heat Sinks and Supporting Camera Installation	A case holder provide a more compact system and provide proper system cooling	1/$9.99	$9.99	https://www.amazon.com/dp/B07JBB9QSB?ref_=pe_386300_442618370_TE_sc_as_ri_0	N/A
Brushless DC Cooling Fan 3007S	A DC component installed	1/$2.99	$2.99	https://www.amazon.com/Brushl	N/A

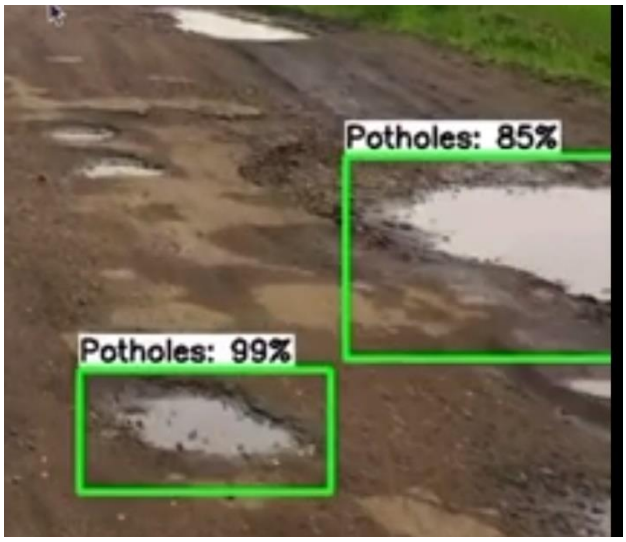
5.0V/160mA DC	into the RPi case to provide system cooling when the system gets too hot (System Temperature > 65.0 Degrees Celsius the fan turns on)			ess-Cooling-30x30x07mm-Sleeve-bearing-Skywalking/dp/B00JDXLXZ6	
Part Description	**Function**	**Amount Needed / Unit Price**	**Subtotal**	**Purchase Link**	**Datasheet Link**
Total:		$73.59			

3. Results Evaluation and Demonstration

Testing the Pothole Detection Algorithm with Recorded Video:

Demo video link: https://youtu.be/Ctj7EP3sql8

In this section, we test the accuracy and efficiency of the pothole detection system. I was unable to increase the playback speed of the actual video, so be sure to increase the playback speed on YouTube to 2x.



In the screenshot above, we can see two potholes are detected with confidence levels of 85% and 99%. One major area of concern is the distance in which potholes can be detected. Since we want to detect potholes as early as possible, speed and visibility play a large role in early detection, which can be improved by using a higher quality camera or training with higher quality images at the cost of higher computational demands. Overall, we can see the PDS is capable of identifying and labeling potholes and is overall confident in its labeling.

Calculating the Mean Average Precision (mAP) Confusion Matrix of the System:

Demo video link: N/A

Calculating mAP Value

```
/content/mAP
Calculating mAP at 0.50 IoU threshold...
71.18% = Potholes AP
mAP = 71.18%
Calculating mAP at 0.55 IoU threshold...
71.18% = Potholes AP
mAP = 71.18%
Calculating mAP at 0.60 IoU threshold...
71.18% = Potholes AP
mAP = 71.18%
Calculating mAP at 0.65 IoU threshold...
67.98% = Potholes AP
mAP = 67.98%
Calculating mAP at 0.70 IoU threshold...
66.70% = Potholes AP
mAP = 66.70%
Calculating mAP at 0.75 IoU threshold...
65.03% = Potholes AP
mAP = 65.03%
Calculating mAP at 0.80 IoU threshold...
65.03% = Potholes AP
mAP = 65.03%
Calculating mAP at 0.85 IoU threshold...
60.84% = Potholes AP
mAP = 60.84%
Calculating mAP at 0.90 IoU threshold...
51.53% = Potholes AP
mAP = 51.53%
Calculating mAP at 0.95 IoU threshold...
42.04% = Potholes AP
mAP = 42.04%

***mAP Results***

Class           Average mAP @ 0.5:0.95
----------------------------------------
Potholes                63.27%

Overall         63.27%
```

The mAP of this system was calculated using 69 images out of the 776 images, roughly 10% of the total dataset. The test images are never seen by the trained model, so we can evaluate its accuracy and precision without bias. This is calculated by calculating the Intersection Over Union (IOU), a ratio between the intersection of the predicted boxes over the actual, or ground-truth boxes, at different confidence thresholds and calculating the average. In the screenshot above, the mAP of the PDS is 63.27%, which is a great improvement compared to the YOLOv4-Tiny version with a mAP of 59.10%.

Confusion Matrix, Model Metrics, and Analyses Formulas

Confusion Matrix - Pothole Detection Test			
TARGET / OUTPUT	Pothole	Not Pothole	SUM
Pothole	17 24.64%	19 27.54%	36 47.22% 52.78%
Not Pothole	5 7.25%	28 40.58%	33 84.85% 15.15%
SUM	22 77.27% 22.73%	47 59.57% 40.43%	45 / 69 65.22% 34.78%

Class Name	Precision	1-Precision	Recall	1-Recall	f1-score
Pothole	0.4722	0.5278	0.7727	0.2273	0.5862
Not Pothole	0.8485	0.1515	0.5957	0.4043	0.7000
Accuracy	0.6522				
Misclassification Rate	0.3478				
Macro-F1	0.6431				
Weighted-F1	0.6637				

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN} \quad \text{Precision} = \frac{TP}{TP + FP}$$

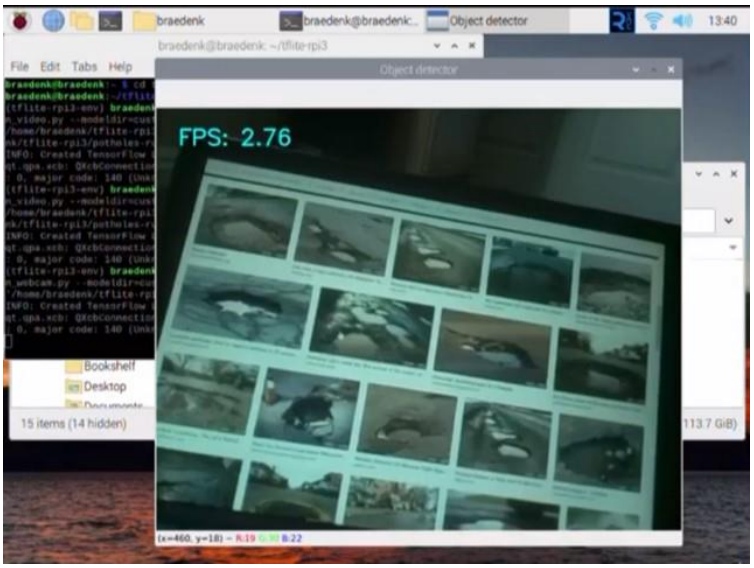
$$\text{Recall} = \frac{TP}{TP + FN} \quad \text{F1-Score} = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

The same 69 test images were then manually observed and analyzed using an object detection script for images to determine the true positive (TP), false positive (FP), true negative (TN) and false negative (FN) values to calculate the confusion matrix, which summarizes the performance of the system. In our case, a positive is a pothole and a negative is not a pothole. The confusion matrix is shown on the top left with the calculated metrics on the top right. By adding the number of true values over the total number of images, we get an accuracy rate of (17+28)/(17+18+19+5) = 45/69, or approximately 65.22%, about the same as our mAP value calculated from earlier. Other metrics, such as the precision, recall, and F1-Score can also be viewed above.

Real Time Pothole Detection Test with Images/Camera Module and RPi:

Demo video link: https://youtu.be/JVreNjIyj_A

Real Time Pothole Detection Using Images



In this demo, we are testing the speed and accuracy of the PDS in real time using pothole images for inferencing. As mentioned previously, the YOLOv4-Tiny model ran at a frame rate up to 0.7 FPS, which was too slow for operation unlike the current SSD-Mobilenet-v2 model operating in the 1.5 – 3.0 FPS range. The model was trained on Tensorflow and converted to TF-Lite for mobile applications and quantized, or rather, converted the floating values to integer-based values to improve latency and reduce peak memory usage.

Testing the Buzzer Alarm/Buzzer:

Demo video link: https://youtube.com/shorts/EHJMzl2ra3Y?feature=share

This demo captures the operation of the piezoelectric buzzer in the PDS. When the PDS detects a pothole, the buzzer emits and audible alarm for 100ms until the pothole is out of sight. The wiring of this component is shown below.

Wiring of the Piezoelectric Buzzer

Reading GPS Data/GPS Module:

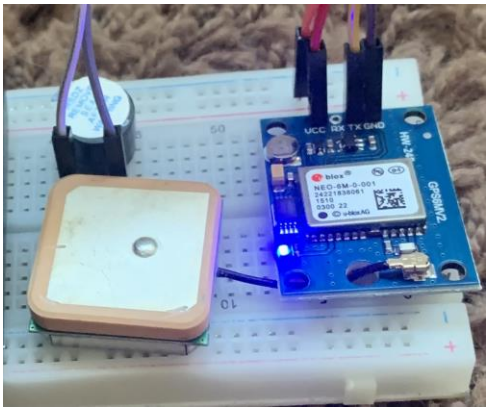
Using GPSMON: https://youtube.com/shorts/XwHX4bdys6A?feature=share

Using get_location.py: https://youtube.com/shorts/jM6C473XKY4?feature=share

Wiring of the NEO-6M GPS Module

The first video shows GPS data being displayed in pretty format. We also know the GPS module is functional when the blue LED on the board is blinking, indicating that a satellite/satellites are tracking this module, as shown in the image below.

GPS Module LED Blinking



While the GPS module is confirmed to operate properly, it is difficult to understand the data being transmitted to the RPi. In the second demo video we use a script named ‘get_location.py’ to parse through the data using the pynmea2 Python library to print the latitude and longitude coordinates of the GPS module into the terminal and log it into a ‘pothole-log.txt’ file as shown below. In the actual design, we want to only log the data when a pothole is detected, which is demonstrated in the final program.

Pothole-log.txt File for Logging Coordinate Data

pothole-log.txt

```
1  Latitude=36.272032 and Longitude=-115.199686Latitude=36.272032 and Longitude=-115.199686Latitı
```

Real Time Object Detection on the Road:

Demo video link: https://youtu.be/eViJFH5Mxow

Dashboard Placement of the PDS

I originally purchased a suction holder the PDS to hang from, but realized the screw was too small for the case and was unable to use it for this design. Nevertheless, I tested the PDS in a small neighborhood, but it falsely detected potholes in the area when there were hardly any. I was unable to configure the system properly for night mode driving, which may be the reason why it falsely detected potholes. The system was powered by a portable battery and then a car adapter, both of which met the power requirements of the system. The GPS module continued to blink, indicating that a satellite was tracking the location of the device, and the coordinates were being logged locally to the device. Unfortunately, since I was unable to connect to the system, I was unable to observe the frame rate of the device or the bounding boxes of the falsely detected potholes. One solution to improve nighttime usage could be training a model using greyscale images or images with varying levels of light.

4. Conclusions/Summary

There are many improvements that can be made with the Pothole Detection System, but the performance of this system can be improved by upgrading the Raspberry Pi module, replacing the GPS module with a GSM (Global System for Mobile Communications) module, and incorporate a Coral USB accelerator. I chose the Raspberry Pi 3 B+ for my design since I was already in possession of one, but the RPi 3B+ is a legacy system and compared to the more

recent Raspberry Pi 4, which has a faster CPU, GPU, and RAM, which could improve the frame rate of the object detection process. Since the PDS in unable to connect to the internet, the idea of storing GPS data to the cloud had to be scrapped, but using a GSM module can allow the PDS to connect to the GSM network and possibly transmit data wirelessly. For higher system complexity and budget costs, the GSM module can replace the GPS module as it can also provide positioning and navigation data, as well as connect to a network wirelessly. Lastly, we can dramatically increase the speed of object detection by using a Coral USB accelerator. The Coral USB provides an Edge Tensor Processing Unit (TPU) to the system, enabling high speed machine learning inferencing. However, my main concern with this design was maintaining an affordable budget, so I considered against it.

In the end, the final revision of the Pothole Detection System can detect potholes, sounding an alarm when a pothole is detected, and logging the positional data of the pothole into the device locally. While there were many more functions I wished to implement, such as general hazard detection, GPS map routing and night-time mode, I was unable to due to computational limitations, time constraints, complexity issues, and cost. Nonetheless, designing the Pothole Detection System gave me the opportunity to incorporate engineering design principles I have learned throughout my time at UNLV, as well as learn project design and management. I also learned new topics such as different machine learning models, training datasets, and the operation of GPS devices.

5. Contribution and Acknowledgement

Team member name	Contributions (use bullets)
Braeden Kurz	• Purchased the materials/components for the project. • Designed the schematic, process specification, and system diagram. • Trained the pothole dataset using Google Collaboratory with virtual TPU. • Authored comprehensive progress reports detailing the advancements made in the project

I want to thank Dr. Regentova for being available when I needed assistance, guidance and counselling, and thank you Dr. Ming for providing me with feedback on how I can improve my project, as well as providing comfort and moral support for my peers and I throughout this difficult semester.

References

[1] Y.-C. Chiu, C.-Y. Tsai, M.-D. Ruan, G.-Y. Shen, and T.-T. Lee, “Mobilenet-SSDv2: An Improved Object Detection Model for Embedded Systems,” *IEEE Xplore*, Aug. 01, 2020. https://ieeexplore.ieee.org/abstract/document/9219319

Appendix

SD-Pothole-Detection-System Github Repository: https://github.com/braeden-kurz/SD-Pothole-Detection-System