AUTONOMOUS POTHOLE DETECTION AND CATALOGING FOR BIKES

Team 62 - Harshvardhan Bhatia, Jesse Chen, Andy Sun
ECE 445 Project Proposal Spring 2018
TA: Xinrui Zhu

1 Introduction

1.1 Objective

Potholes are an issue which plague cities all around the world. While damaging to cars, potholes are particularly dangerous, even fatal, for bikers and can lead to millions of dollars in lawsuits for a city if not patched [1]. Bikers need to be very aware of the surroundings around them while on the road, but with so much going on, it can be easy to miss potholes, both at day and at night. However, for cities to be able to fix potholes, they need to first know where they are. Currently, the city of Champaign is starting to utilize a phone application for reporting problems like potholes, but few bikers are going to stop, get off their bikes, and pull out their phone to fill in a report [2]. Most cities simply utilize a telephone number or website for pothole reporting, which is even more inconvenient.

Our project aims to ease the issues of both pothole cataloging for municipalities as well as pothole warning for bikers. We aim to do this by creating a device which can detect potholes via computer vision (for long-range detection) as well as accelerometers (for potholes actually hit). In addition, a user can report a pothole they ride past by pressing a button. Once a pothole is detected by the system, the device's current GPS location and time is sent to a database for municipalities to access. Devices will also host a local copy of this database, which will be used to warn bikers via haptic feedback if they are riding towards an area with many potholes. In addition, when the computer vision portion detects a pothole, the rider will also be alerted, in case they didn't spot it themselves.

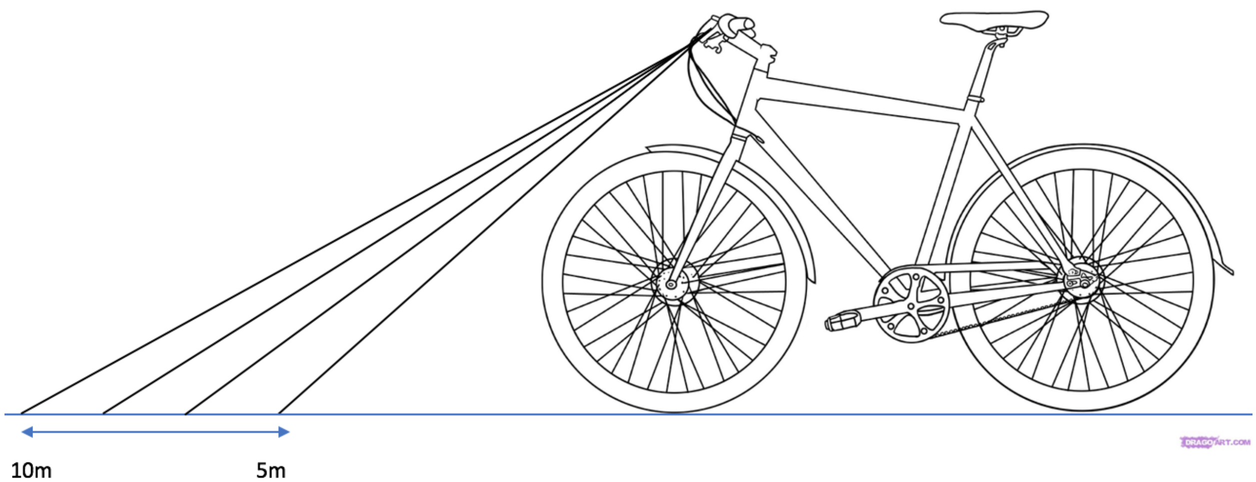
1.2 Background

Research papers have been written about pothole cataloging techniques, but commercial products have been few and far between. In addition, all research papers that we found focused upon the use case for cars. There was one commercial product which advertised obstacle warning for bikes, but its crowdfunding campaign ultimately failed [3]. Generally, these papers describe specific parts of a pothole detection system, such as computer vision techniques [4], laser imaging techniques [5], or accelerometer-based techniques [6]. The high cost of laser-imaging techniques makes them impractical and computer-vision based techniques suffer from mediocre accuracy and false positives. No systems that we found combine these techniques. In addition, these papers generally fail to describe any type of pothole warning system, which we believe would be useful for both drivers and bikers.

By combining various aspects of these techniques, we hope to create a holistic system which will enable quick, easy pothole cataloging while also warning the biker of nearby potholes in order to prevent accidents.

1.3 Physical Design Concept

The device will be attached to the handlebar of bike. As shown in Figure 1 below the camera will scan the road ahead (upto 10 meters). The accelerometer will catch potholes that the biker rides over. Potholes that the biker sees can be reported using the pothole reporting button.



https://www.dragoart.com/tuts/3307/1/1/how-to-draw-a-bike.htm

Figure 1: Device scanning road while attached to bike (not to scale)

1.4 High-level requirements list

- Pothole identification must take no longer than 5 seconds for any method.
- The device must be able to grab and send location data upon pothole detection.
- The device must be able to warn user of nearby potholes using database and current location info.

2 Design

Our design consists of 3 main blocks, a power module, a control/feedback module, and a computer vision modules. The power module supplies power to the other modules and ensures they can run for the duration of the bike ride. The control/feedback module tracks the current location of the device, transmits the location of detected potholes, analyzes motion data to do pothole detection, and offers haptic feedback when the rider encounters an area with potholes. The computer vision module detects potholes visually.

The power module accesses the power bank via a USB to barrel jack connector (5.5 mm outer diameter, 2.1 mm inner diameter). It will be located on the same PCB as the control module and will therefore provide power via a PCB trace. However, the computer vision module is on a seperate board. Therefore, we put another barrel jack connector in the power module to provide 5V power to the computer vision module, which also accepts power via barrel jack. The computer vision module consists of a dedicated MCU and camera. It outputs its detection status to a header pin, which will connect to another header pin on the control module via a standard wire. It then feeds into a GPIO pin on the control module MCU. When that pin goes high, it will trigger an interrupt and let the control module MCU that a pothole has been detected.

2.1 Block Diagram

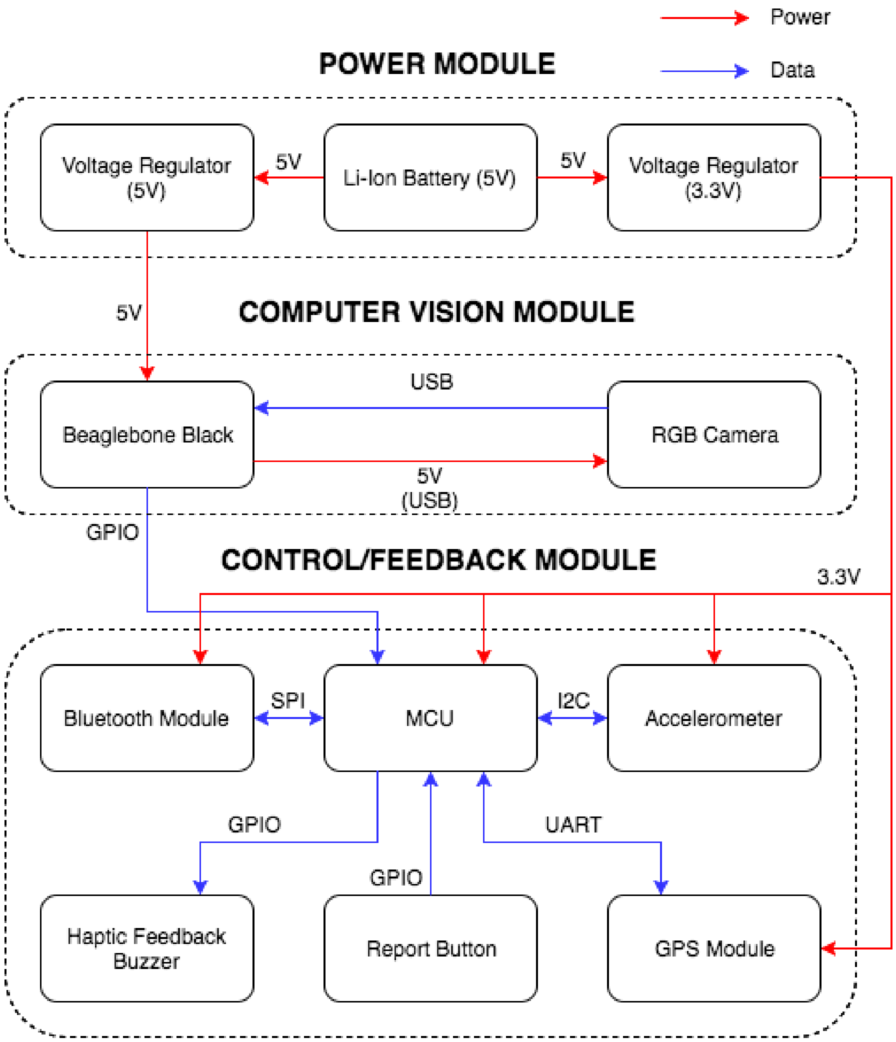


Figure 2: Block Diagram

2.2 Schematics / PCB

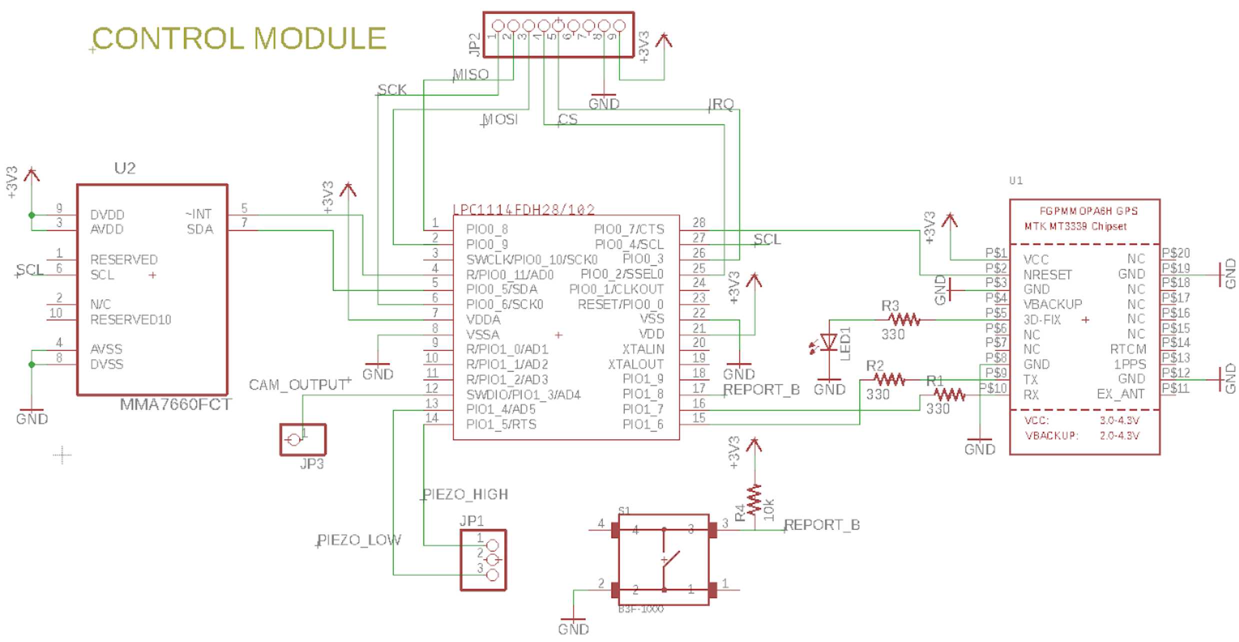


Figure 3: Control Module Schematic

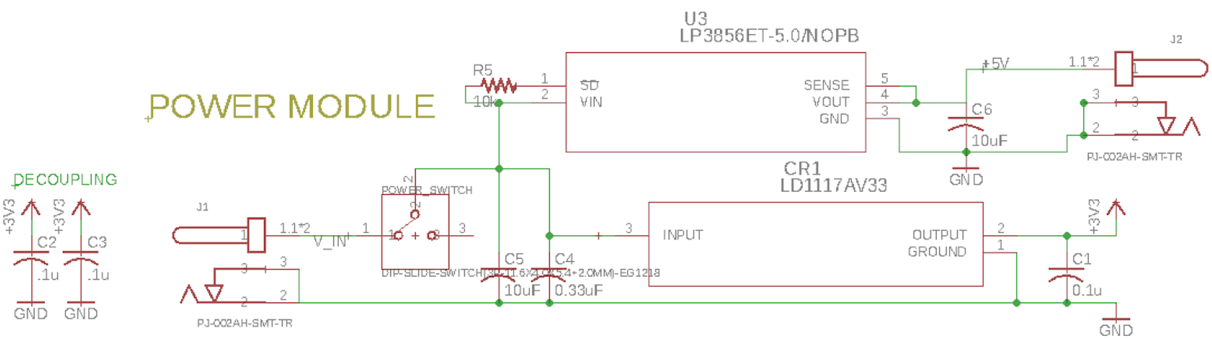


Figure 4: Power Module Schematic

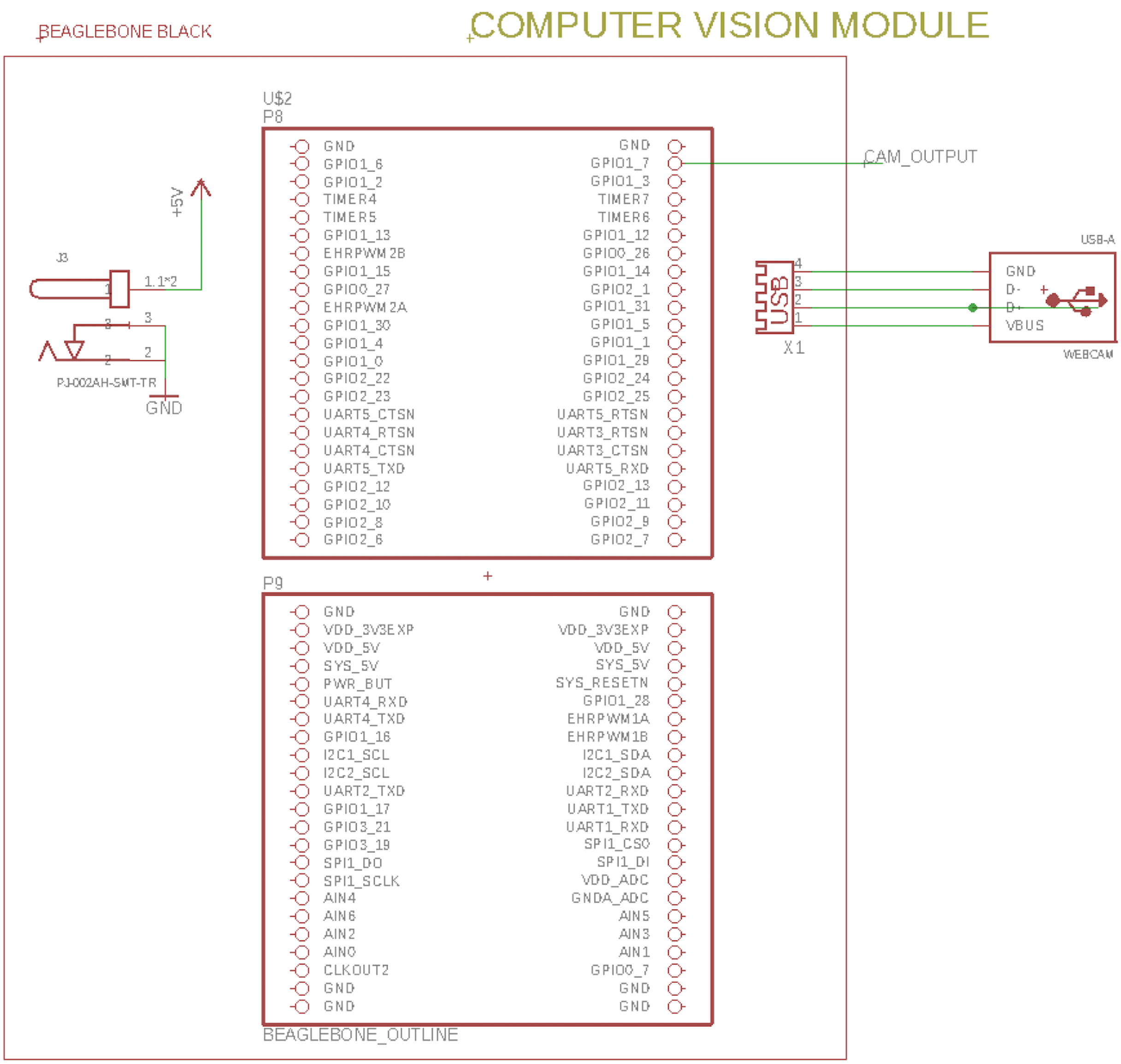


Figure 5: Computer Vision Module Schematic

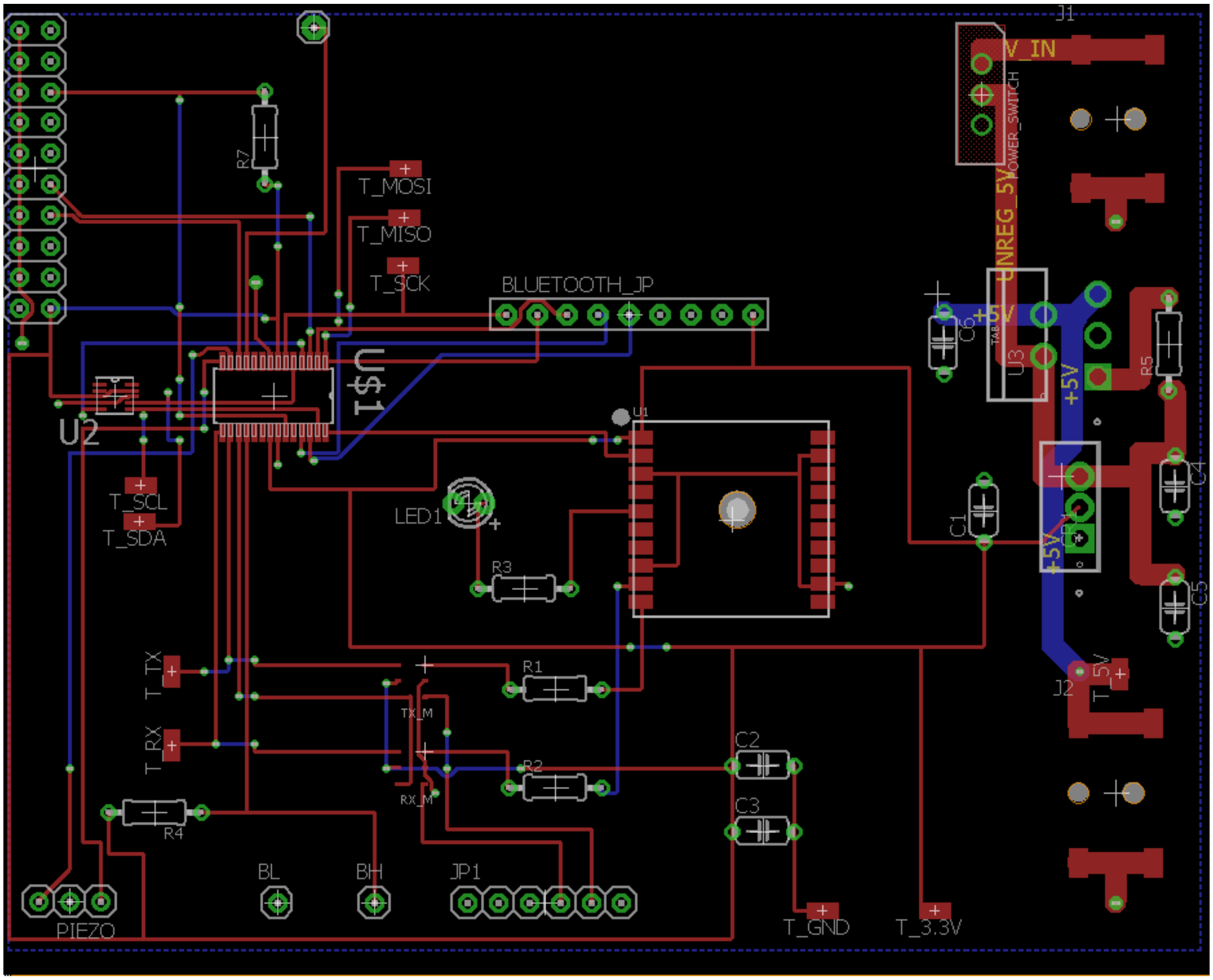

J1
V_IN
POWER_SWITCH
UNREG_5V
T_MOSI
T_MISO
T_SCK
BLUETOOTH_JP
R7
U$1
U2
U1
U3
C6
R5
+5V
C4
C1
C5
LED1
R3
T_SCL
T_SDA
T_TX
T_RX
R1
R2
TX_M
RX_M
C2
C3
J2
T_5V
R4
PIEZO
BL
BH
JP1
T_GND
T_3.3V

Figure 6: PCB

2.3 Power Module

Description

As this will be a mobile product, we will require a mobile power source. We plan on utilizing a rechargeable USB power bank which operates at 5V. This will also feed into a voltage regulator for any modules which operate at 3.3V.

2.3.1 Li-Ion Battery

Functional Overview

The battery supplies mobile power for our project. It will supply power at 5V and should also be able to provide at least 2A of current. However, we do not place this in our requirements list as this is more a feature of the product than it is a design requirement. Below we have compiled a power consumption table. Based upon this, we have calculated a typical power consumption of 7.598 W. For three hours of use, this equals 22.794 Wh. Applying this to a 5V battery, we would require a battery with a capacity of at least 4558.8 mAh for three hours of use, hence our requirement for a 5000 mAh battery below.

Total Power Consumption

Part	Voltage (V)			Current(mA)			Power(mW)		
	min	typ	max	min	typ	max	min	typ	max
GPS Module	-	3.3	-	-	20[1]	-	-	66	-
Bluetooth Module	-	3.3	-	0.0006	0.0026	16	0.0018	0.0078	48
Piezo Buzzer	3	3.3	5	0	0	4	0	0	13.2
Accelerometer (@ 120 samples/sec)	2.4	3.3	3.6	-	.294	-	.71	.97	1.1
MCU	1.8	3.3	3.6	-	9	-	16.2	29.7	32.4
Beaglebone Black	-	5	-	1200	1500	2000	6000	7500	10000
USB Webcam	-	5	-	.120	-	.220	0.6	0.85[2]	1.1
TOTAL POWER VALUES (mW)							6084	7598	10,162
TOTAL POWER VALUES (W)							6.084	7.598	10.162

* If min and max values are not provided in datasheet typical values will be used for power calculations
[1] Typical current consumption during tracking shown, as that is the state the module will usually be in. Current consumption during acquisition is 25 mA.
[2] Typical power consumption calculated by assuming average current as being mean of minimum and maximum currents.

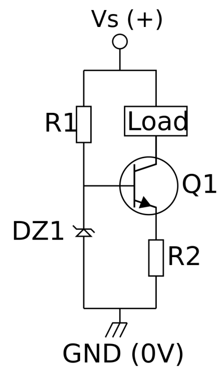


Figure 7: Constant current load circuit

Requirements	Verifications	Points
> 5000 mAh of capacity	1. Connect fully-charged battery (as indicated by LED on commercial power bank) 2. Discharge battery at 500 mA +/- 5% for 10 hours using constant current load circuit 3. Check if commercial power bank will still allow current to flow using ammeter, as they will automatically shut current off when voltage is too low.	2

2.3.2 Voltage Regulator

Functional Overview

The voltage regulators maintains a constant voltage for smooth operation of our device. There is a 3.3V regulator for smooth operation of the control module, while a 5V regulator provides some short-circuit and reverse polarity protection to the battery, as well as a smooth voltage to the computer vision module.

Requirements	Verifications	Points
Provide 3.3V +/- 5% output from 4.7V - 5.3 V source at 0.5A +/- 5%	1. Connect the input to a power supply 2. Draw 0.5A from power supply using constant current load circuit 3. Sweep the power supply from 4.7 to 5.3 V 4. Measure the output is 3.3V +/- 5% at 0.5A +/- 5% with a multimeter	2.5

Provide 5V +/- 5% output from 5V - 5.3 V source at 2.5A +/- 5%	1. Connect the input to a power supply 2. Draw 2.5A from power supply using constant current load circuit 3. Sweep the power supply from 5 to 5.3 V 4. Measure the output is 5V +/- 5% at 2.5A +/- 5% with a multimeter	2.5

2.4 Control/Feedback Module

Description

The control/feedback unit takes care of processing pothole detection events, recording and sending their location, as well as alerting the user if their GPS location is near a pothole or if the computer vision module detects a pothole.The accelerometer will communicate with the MCU via an I2C bus. The GPS module will use UART and the bluetooth module will use SPI. The different communication protocols were chosen in part due to cost and taking advantage of parts available from the class and available libraries.

2.4.1 MCU

Functional Overview

The MCU takes care of polling/processing the accelerometer data in order to detect potholes, as well as processing pothole detection events from the report button and computer vision module. It takes care of the pothole reporting mechanism when a pothole is detected in addition to determining if a user is approaching a pothole and warning them appropriately. Figure 8 and 9 portray the functionality of the MCU.

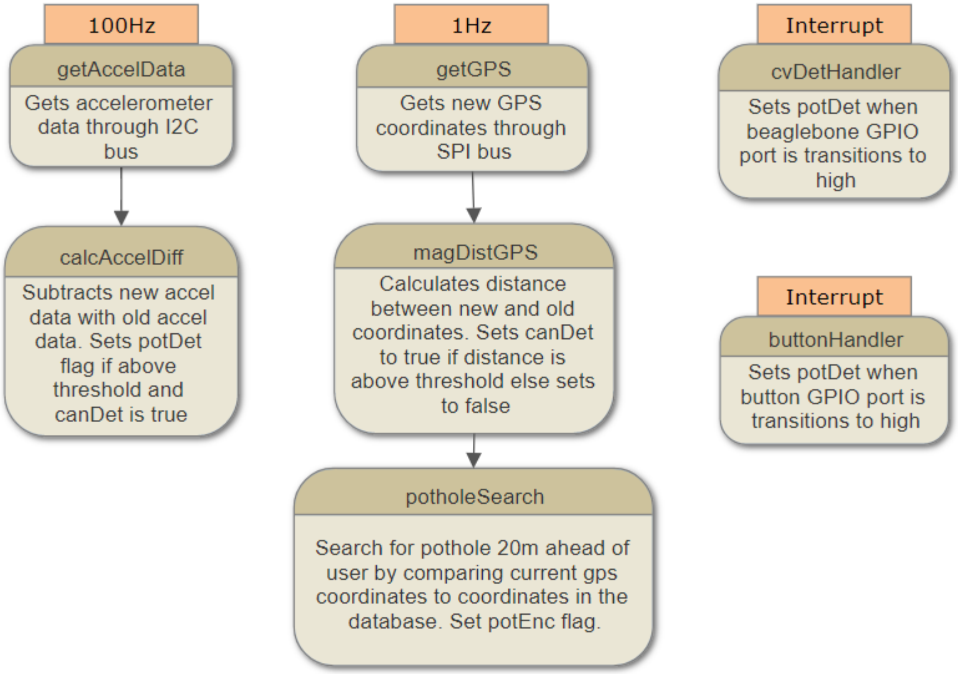


Figure 8: Interrupts and Timed Functions

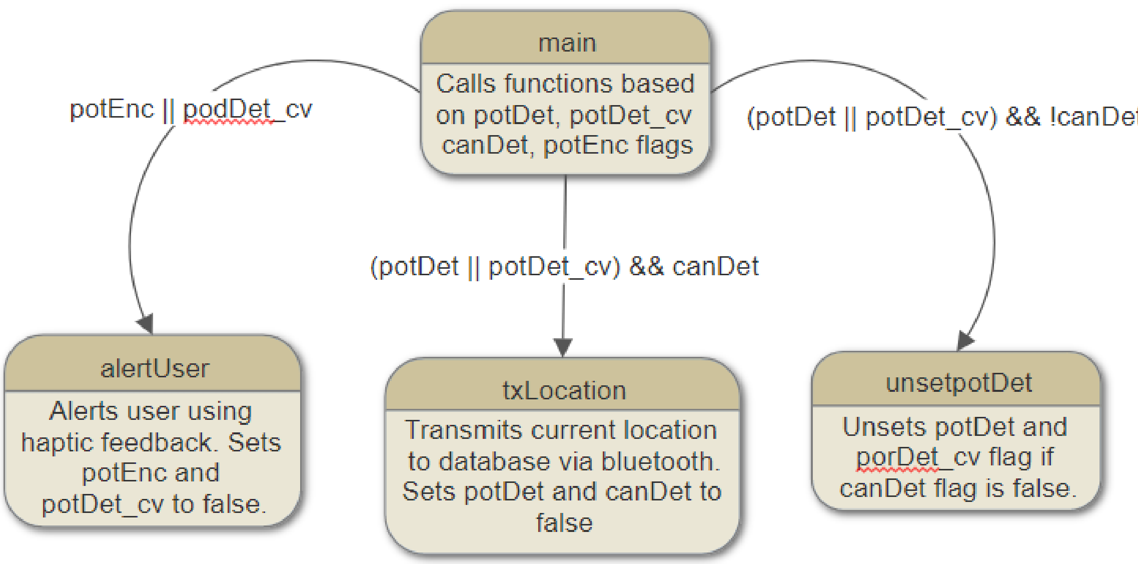


Figure 9: Main Function

Requirements	Verifications	Points
Must be able to communicate over I2C in at least standard mode (100 kHz)	1. Set-up I2C on MCU. 2. Set up oscilloscope to trigger mode 3. Connect positive probe to SCL pin 4. Connect negative probe to GND 5. Send any data via the I2C bus 6. When oscilloscope is triggered find the frequency of the clock waveform by measuring any 2 rising edges.	5
Must be to communicate over UART at 9600 baud	1. Connect MCU to a computer terminal using UART and PuTTY 2. Set PuTTY baud rate to 9600 and make connection 3. Send a test string from MCU to computer upon button press 4. Ensure that string on terminal matches test string sent	3

2.4.2 MCU + Button

Functional Overview

The button provides a method for a user to easily report a pothole if it is missed by both the computer vision module and accelerometer.

Requirements	Verifications	Points
Button and debouncing software should register press within 500 ms	1. Connect button to MCU GPIO pin 2. Set MCU to output to computer terminal via UART if press is detected 3. Start timer and press button simultaneously 4. Wait for output to appear on terminal 5. Once output appears, stop timer 6. Check if time allotted is less than 500 ms	3
Button should register press accurately 19 out of 20 times	1. Connect button to MCU GPIO pin 2. Set MCU to output to computer terminal via UART if press is detected 3. Press button 20 times 4. Check if at least 19 outputs are displayed on terminal	2

2.4.3 MCU + Bluetooth Module

Functional Overview

The bluetooth module allows our device to connect to a phone in order to send pothole locations to a remote database upon pothole detection, regardless of the method.

Requirements	Verifications	Points
Must have range greater than 2m	1. Place the receiver 2m away from the module 2. Ensure that phone can see module while it is advertising using Bluetooth scanner app	1
Must transmit latest GPS location after pothole detection within 1s	1. Connect phone to bluetooth module 2. Open Serial Bluetooth Terminal app 3. Start timer and simulate pothole detection using user report button 4. Stop timer once coordinate appears on phone 5. Ensure time allotted is less than 1s	5

2.4.4 MCU + GPS

Functional Overview

The GPS module will be used to determine the user's current location in relation to nearby potholes as well as the locations of potholes detected by the system.

Requirements	Verifications	Points
Location accuracy of 5m (best case, open sky scenario)	1. Determine the precise longitude and latitude of a location using Google Maps (such as a street lamp). 2. Take the GPS module there. 3. Compare that to the output of the GPS module.	2
Warn user if they are within 25 m of a known pothole and are traveling towards it	1. Manually set pothole location using user report button 2. Mark location 25 m away from pothole 3. Walk 30 m away, then start walking towards pothole 4. See if haptic feedback sensor goes off	5

2.4.5 MCU + Accelerometer

Functional Overview

The accelerometer allows for detection of potholes that the user runs over. Specifications and function have been chosen from the following paper [7]. This device will be using the Z-DIFF mode which compares new accelerometer data to old data. If the new accelerometer data passes a certain threshold value, the MCU count that as a pothole detected. The figure below demonstrates this method.

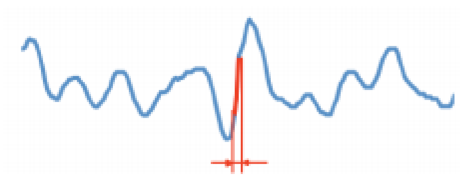


Figure 10: Z-DIFF method plot representation [7]

The actual threshold value will be decided after a series of tests that will compare the true pothole detection percentage to false pothole detection percentage.

Requirements	Verifications	Points
Must have sampling rate of 100 Hz	1. Connect accelerometer to MCU 2. Create timer interrupt at a 100 Hz on the MCU 3. Set AMSR[2:0] of accelerometer to 111 (120 samples/sec) 4. Set accelerometer to Active Mode 5. Read either X,Y or Z register at every MCU interrupt while moving the accelerometer 6. Output data to any platform and check for consecutive duplicate values	2
±1g range with .1g sensitivity	1. Measure the weight of the accelerometer 2. Place it on one end of a balance and put known weights on the other side 3. Record the output of the accelerometer and compare with the expected computed values.	1
MCU must detect potholes with the accelerometer. True Positive rate of 80%	1. Find 3 potholes 2. Run bike through pothole track at least 5 times 3. Keep track of each detected event as well as each true positive event 4. Divide true positives by detected events to get rate	4

2.4.6 MCU + Piezo Buzzer

Functional Overview

The buzzer serves as a method of warning the user of incoming potholes. It will be activated if the user is approaching a pothole described in the database or if the computer vision module has detected a pothole. There will be two different buzzing patterns depending on which of the two cases activates the buzzer. It will be placed in an armband to be worn around the user's wrist.

Requirements	Verifications	
Must be able to be detected on bare skin	1. Connect buzzer to GPIO pin on MCU 2. Place the piezo buzzer on wrist 3. Set GPIO pin high 4. Observe if buzzer can be felt	1
Must warn user within 100 ms of pothole detection from computer vision module	1. Connect control module to computer via PuTTY and UART-USB connector 2. Place function to output system clock count when the GPIO pin connected to the computer vision module goes HIGH 3. Place function to output system clock count when code to activate buzzer is entered. 4. Attach SPDT switch and hardware debouncer to GPIO pin normally connected to computer vision module. Attach GND to one input and 3.3V to the other. 5. Set switch to GND. 6. Run control module normally. 7. Simulate computer vision pothole detection by flipping switch to 3.3V and immediately flipping back to GND. 8. Look at output values on computer terminal. Calculate time elapsed based upon MCU clock speed and compare to 100ms.	5

2.5 Computer Vision Module

Description

The computer vision unit analyzes visual data to detect potholes and communicates that information to the main microcontroller. This unit detects potholes before they are hit, in contrast to the accelerometer.

2.5.1 RGB Sensor

Functional Overview

Our camera will capture image data for pothole detection and send it to the the detection MCU for processing.

2.5.2 Detection Microcontroller

Functional Overview

We will use a dedicated microcontroller for image processing to ensure it can be completed as rapidly as possible. The detection microcontroller will notify the main control microcontroller whenever a pothole is detected.

Requirements	Verification	
The camera captures at least 10 frames per second	1. Connect the camera over USB to a computer and capture 5 seconds of video. 2. Use a python program to count the number of frames and divide by the video length	1
The module should have a false negative rate of at most 30% and a false positive rate of at most 25%.	1. Bike the system past 5 potholes 2. Record the outputs of the camera module 3. Repeat 3 times for each pothole	3

2.5.3 Detection Algorithm

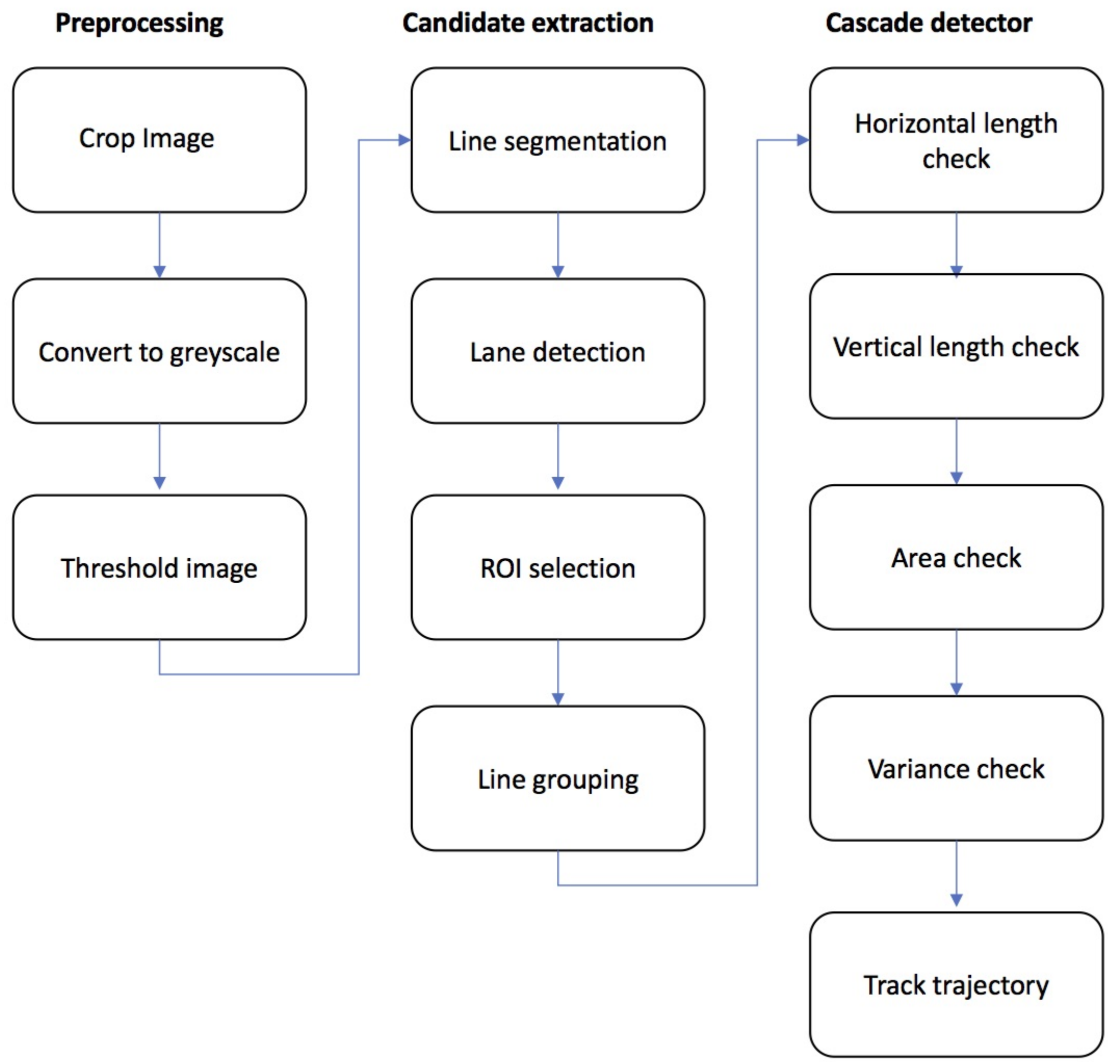


Figure 11: Detection algorithm flowchart reproduced from [5]

We plan to implement the algorithm described in "Pothole Detection System Using a Black-box Camera"[5]. In the preprocessing step, the input image is cropped and converted to greyscale to reduce the computational resources needed. The image is then thresholded to retain only the darker regions as possible pothole candidates. In candidate extraction, regions of interest are selected by grouping highly continuous dark areas that are within the lanes of the road or pathway. The cascade detector checks whether the properties like size and variance in intensity match those of a pothole.

2.6 Tolerance Analysis

Our project is a bicycle mounted pothole logger which uses a camera to identify potholes. Therefore, it is critical for our detection module to both be fast and accurate. It has to quickly identify potholes before the rider passes by them and should be accurate to avoid logging non-potholes or not logging real potholes. For the detection module to meet both targets, it depends on receiving clear images from the camera to analyze. The images it receives should not be blurry, should be well exposed, and should have minimal noise. In addition, the detection algorithm should be fast and accurate as well.

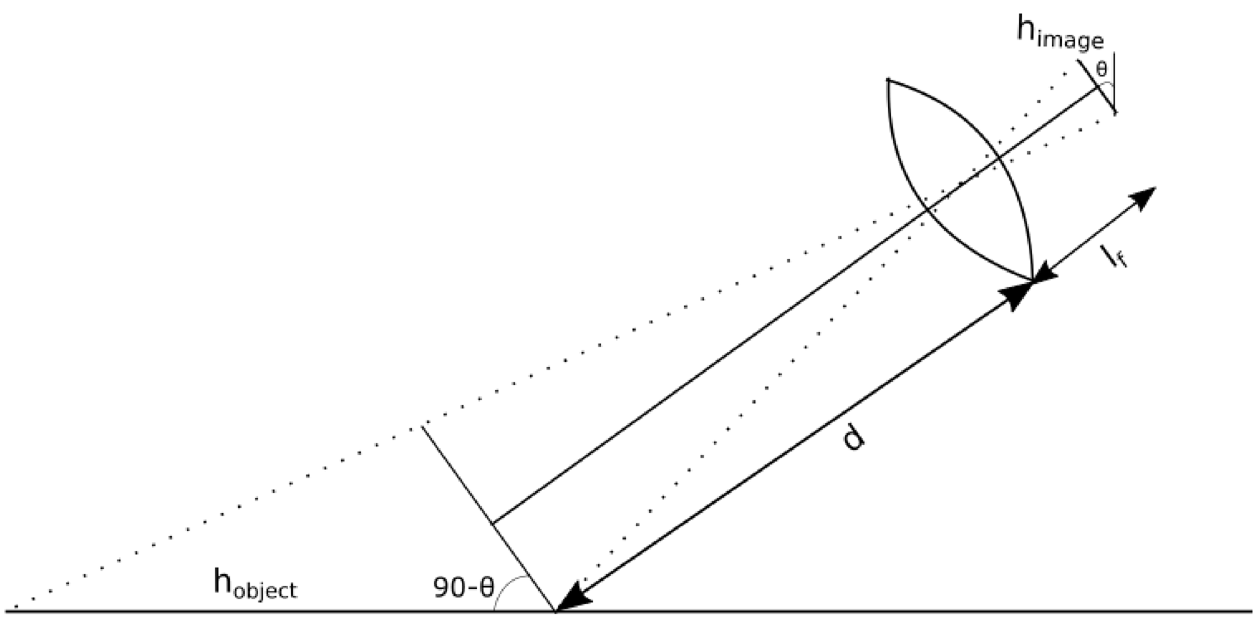


Figure 12: Lens relation with similar triangles

Since our project is going to be used while in motion, it is important for the camera to have a fast shutter speed to minimize blurriness. We can use the behavior of lenses to find an estimate of blurriness as a function of shutter speed. Using similar triangles we can obtain the following relationship.

$$\frac{h_{object}}{d} = \frac{h_{image}}{l_f}$$

h_{object} is the distance an object travels, h_{image} is the distance on the sensor, d is the distance from the sensor to the object and l_f is the focal length. For our setup shown in figure 12 we can rewrite the equation as

$$h_{image} = \frac{v*ss*sin\theta*l_f}{d}$$

Where v is the bike velocity, ss is the shutter speed, and θ is the angle of the camera. Given that the average biker travels at about 5 m/s [9], our ideal pothole detection distance is 3m, and our current camera pixel size is 3 µm, we can obtain the plot in figure 13. The number of pixels is the number of pixels the image moves during one capture and is found by dividing h_{image} by the pixel size. Since our current camera also has a focal length of 2mm, we can see that at 1/800 s shutter speed, and a camera mounted at 45 degrees or less, will record a pothole on the ground with minimal blurring. Figure 14 shows at 30 degrees and 1/1000s shutter speed, we have a roughly 50% tolerance on pixel size. If the pixel size turns out to be less than 1.5 µm we can either increase the shutter speed even farther or decrease the camera mount angle. However, finding a faster shutter may be difficult and decreasing the camera mount angle will decrease the size of the pothole as it appears on the image, which could affect the ability of our classifier to identify the pothole. At twice the average velocity, the blurring still remains relatively low.

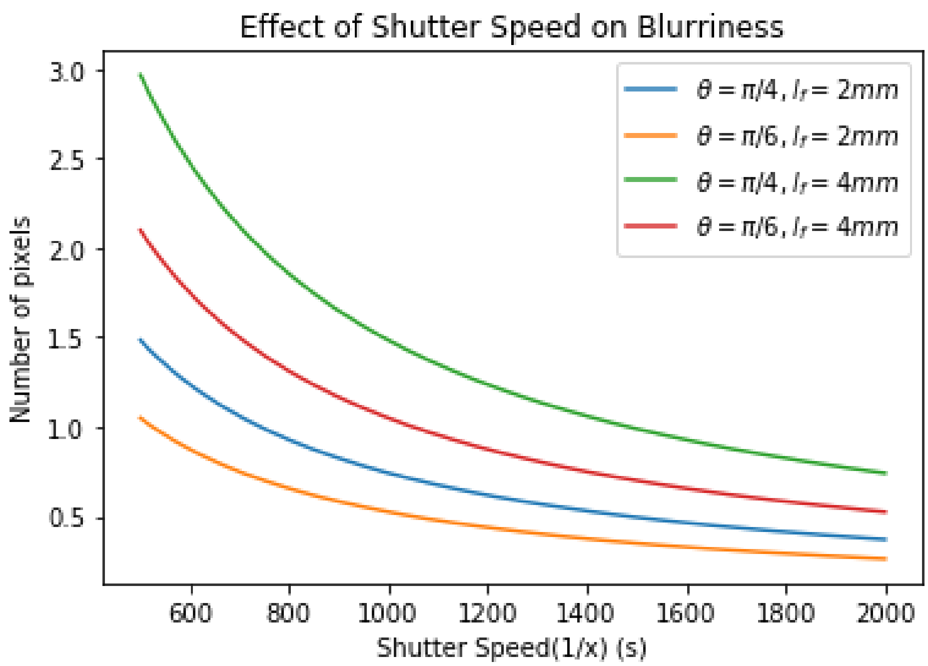


Figure 13: Plot of shutter speed vs blurriness

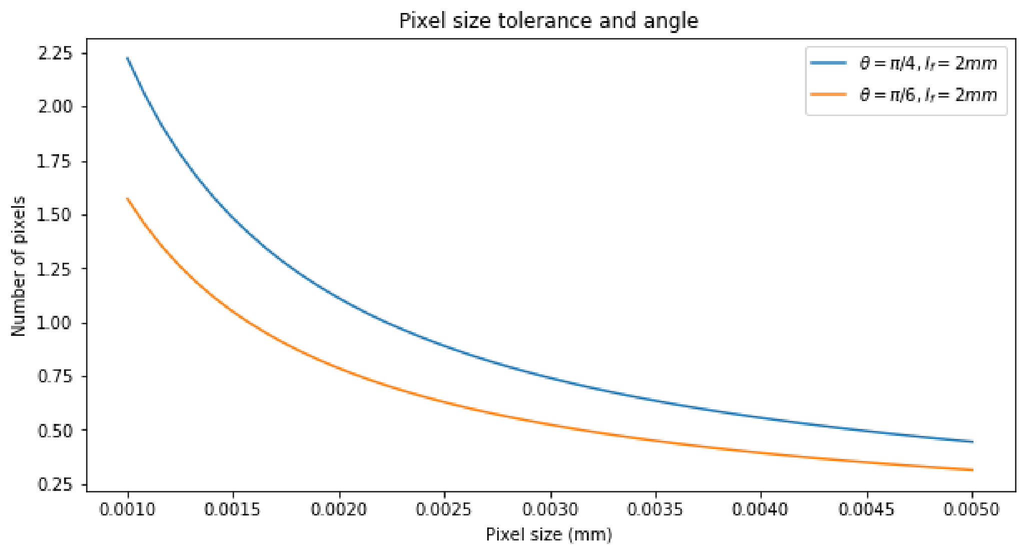


Figure 14: Plot of pixel size vs blurriness for different angles

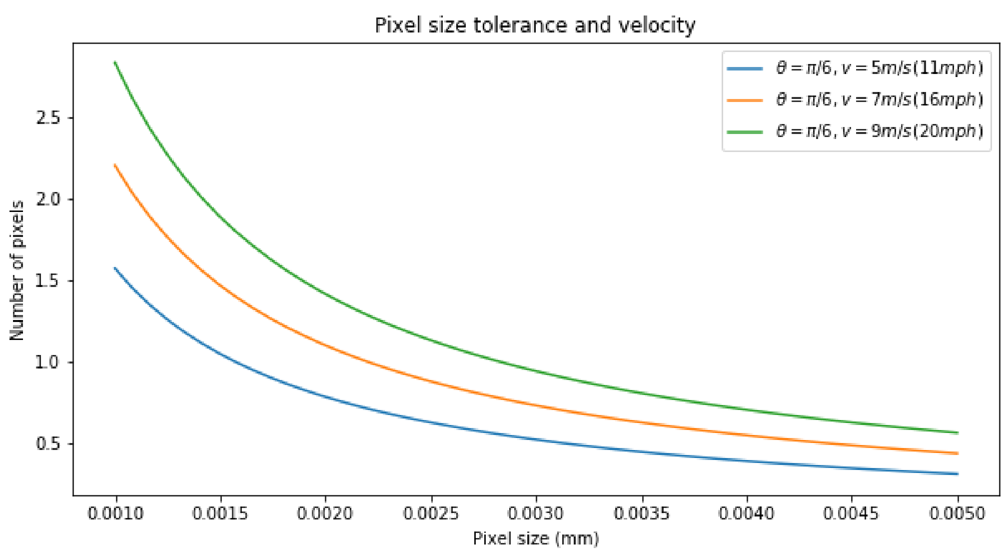


Figure 14: Plot of pixel size vs blurriness for different velocities

The tradeoff for a fast shutter speed is less exposure, so we will either need to increase the ISO value or increase the aperture size. ISO refers to the sensitivity of the sensor and aperture size the size of the opening to the lens. Increasing ISO will make the image brighter, but will also make it more noisey. A low SNR could decrease the accuracy of our detection algorithm, although steps can be taken to improve SNR. We could denoise in software or average multiple images together to reduce noise. Aperture size is usually fixed for the types of cameras we're looking at. Increasing aperture size will increase the brightness, but at the cost of depth of field. Less of the image will be in focus, so this will have to be compensated by a faster frame rate to capture more images. Shutter speed, ISO, and aperture size are all related. Doubling shutter speed has the same effect as taking ½ of the ISO value or a decrease by a factor of square root of two in aperture size. Therefore, changing one value can be compensated by changing another. As a general rule, for broad daylight conditions with an aperture size of f/16 and a shutter speed of 1/1000s we will want an ISO of 1000 [10]. Since the aperture size is most likely dependent on which camera we choose, the ISO will need to be adjusted accordingly to maintain proper exposure. Figure 16 shows the relationship between aperture size and ISO to maintain the same exposure.

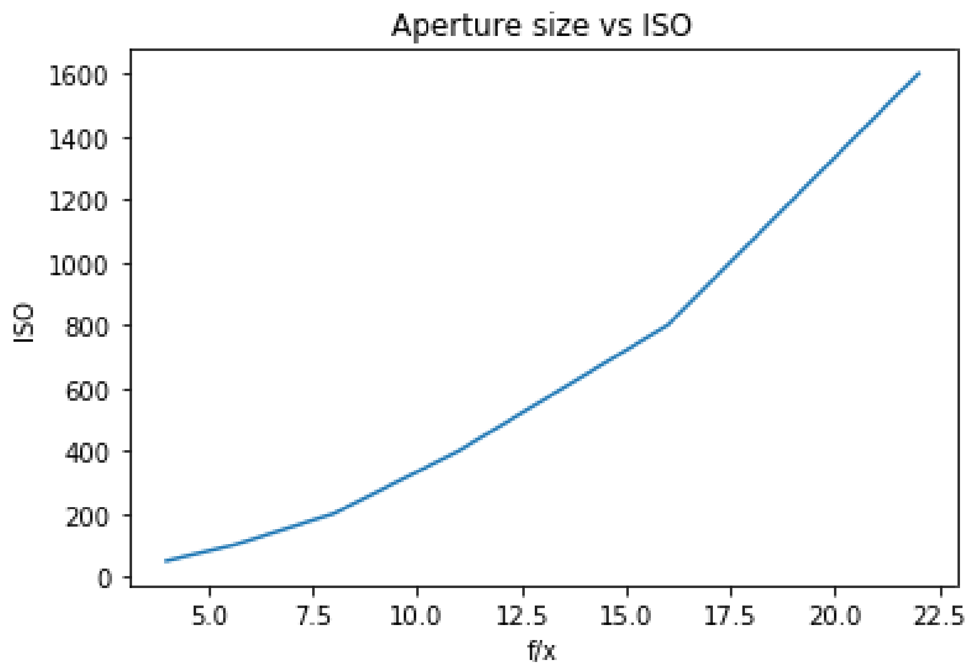


Figure 16: Plot of equal exposure ISO and aperture size

Our detection speed is an important factor as well. We will want to reduce the computational complexity as much as possible with the possible trade off of accuracy. One example is during preprocessing we build a histogram to pick a threshold for our binary classifier. If we sampled every other pixel to build the histogram instead of using every pixel we could decrease the computational time of this step by a half. As long as there are no extremely rapid changes in brightness, our accuracy should not be strongly affected. In this way we can consider decreasing the computational time of other steps as well.

2.7 Risk Analysis

We believe the computer vision block presents the biggest risk of failure for our project. Speed is a significant issue as image processing tasks can be computationally intensive. We will need to carefully select suitable hardware and write our algorithms to be as efficient as possible.

The accuracy of the computer vision is also an area of concern. There are many unforeseen variables in the real world the can affect the performance of our unit. In addition we will also have to do a lot of testing to find the ideal parameters for our algorithm to respond to these conditions. Existing computer vision algorithms have only found success in broad, daylight conditions. In addition, these algorithms have been designed for pothole detection on automobiles, which would offer a smoother camera feed than on a bike.

3 Costs and Schedule

3.1 Cost

Part	Distributer	Part Number or Name (if applicable)	Cost
GPS Module	Adafruit	FGPMMOPA6H	$30
Button	Adafruit	B3F-1000	$2.50 for pack of 20
Bluetooth Module	Adafruit	Bluefruit LE SPI Friend	$17.50
Piezo Buzzer	Adafruit	Buzzer 5V - Breadboard friendly	$0.95
Accelerometer	Digikey	MMA7660FCT	$1.81
MCU	Digikey	LPC1114FDH28	$2.58
Computer Vision MCU	Digikey	BeagleBone Black	$56.25
USB Webcam	Amazon		$45.99
Battery	Amazon	Anker PowerCore II Slim	$25.99
3.3V Voltage Regulator	Digikey	LD1117AV33	$0.53
5V Voltage Regulator	Digikey	LP3856ET-5.0/NOPB	$5.04
Barrel Jack Connector x 2	Digikey	PJ-002AH-SMT-TR	$1.44 per unit
Labor[1]	N/A	N/A	$45,000
TOTAL COSTS			$30,186.98

Labor Cost = $3\ people \times 30\$/hour \times 20\ hours/week \times 10\ weeks \times 2.5 = \$45,000$

3.2 Schedule

Week	Andy	Harsh	Jesse
2/5	Work on project proposal	Work on project proposal	Work on project proposal
2/12	Research computer vision Work on design document	Research accelerometer and MCU Work on design document	Research power, GPS, and bluetooth Work on design document
2/19	Finish design document Order camera	Finish design document Order accelerometer, MCU	Finish design document Order GPS, bluetooth module, buzzer, button, battery, voltage regulators, barrel jack connectors, etc
2/26	Complete first python prototype of detection algorithm	Build circuit on breadboard Set up I2C protocol Set up Accelerometer tests	Write SPI and UART protocols Build initial test circuit on breadboard
3/5	Optimize algorithm to improve speed and accuracy	Design PCB and order Finalize Accelerometer pothole detection algorithm Design PCB case	Write firmware to communicate with GPS and store location data
3/12	Write bluetooth logging app for phone	Assemble and test PCB Assemble and test PCB Case Begin to integrate GPS module to MCU Start to implement control workflow	Write firmware to communicate with Bluetooth
3/19 (spring break)			
3/26	Copy program onto MCU	Begin testing pothole detection system Integrate Bluetooth with control workflow	Design algorithm to detect if user is approaching stored potholes based upon current location

4/2	Continue to debug and integrate detection unit	Test, debug and optimize system	Write piezo buzzer and pothole report button code Test and debug GPS, bluetooth, and haptic feedback modules
4/9	Test, debug and optimize system	Test, debug and optimize system	Test and debug pothole detection module interface with feedback modules
4/16 (mock demo)	Test to ensure complete functionality	Test to ensure complete functionality	Test to ensure complete functionality
4/23	Start final report Prepare for final demo	Start final report Prepare for final demo	Start final report Prepare for final demo
4/30	Complete final report	Complete final report	Complete final report

4 Ethics and Safety

If we perfect our project to be able to do real time pothole detection with ample warning time, it may cause a sense of complacency in the user. They may become more prone to distraction if they believe they can completely rely on the system which is a dangerous situation.

Since our project is designed to be used outside, there is a chance it will be exposed to rain or other precipitation that could damage the components. To mitigate this, we will test our case to ensure it is at least IP13 compliant.

We will be using a power bank as a power source which will most likely contain a lithium ion battery. Since we are purchasing a commercially available power bank it should satisfy high safety standards for charging. The USB interface makes a reverse polarity error unlikely, due to the physical design of USB connectors. Physical damage to the lithium battery could potentially start a chemical fire so we will ensure the battery remains far away from the moving mechanical parts of the bike. We will also ensure the house the battery in a location away from potential collisions and provide protection from any collisions that do occur. There is a switch controlled by the user to shut off power to the system completely. Even though the power bank we purchased should have built in circuit protection, we have also attached a 5V regulator with built in short-circuit and overcurrent protection to the battery output, which will shut off power in case any danger arises. A voltage regulator also protects the rest of the circuit from damage in case a reverse polarity event does occur, as the voltage regulator will be the only component damaged.

The continuous recording of the road could mean that people who do not want to be recorded can be in public. This could cause an ethical issue, however we believe that this device can be compared to dashcams. Our device will also be continuously deleting video as its been processed.

We will follow the IEEE code of ethics while we pursue our project, especially, points 1,3,7, and 10. We will “hold paramount the safety, health, and welfare of the public,” “be honest and realistic in stating claims or estimates based on available data,” “seek, accept, and offer honest criticism of technical work, acknowledge and correct errors, credit properly the contributions of others,” and “assist colleagues and co-workers in their professional development and support them in following [the] code of ethics” [8]

References

[1] R. Annis, "$6.5 Million Settlement Given to LA Cyclist for Injuries from Pothole", *Bicycling*, 2017. [Online]. Available: https://www.bicycling.com/news/when-cyclists-sue-the-city. [Accessed: 08- Feb- 2018].
[2] "Report A Problem - City of Champaign", *City of Champaign*. [Online]. Available: http://champaignil.gov/public-works/report-a-problem/. [Accessed: 08- Feb- 2018].
[3] "Byxee - the smart active safety device for bicycle", *Indiegogo*, 2015. [Online]. Available: https://www.indiegogo.com/projects/byxee-the-smart-active-safety-device-for-bicycle. [Accessed: 08- Feb- 2018].
[4] Y. Jo and S. Ryu, "Pothole Detection System Using a Black-box Camera", *Sensors*, vol. 15, no. 12, pp. 29316-29331, 2015.
[5] S. Yamuna, "Automatic Pothole Detection System using Laser", *International Journal for Research in Applied Science and Engineering Technology*, vol., no., pp. 1226-1231, 2017.
[6] H. Wang, C. Chen, D. Cheng, C. Lin and C. Lo, "A Real-Time Pothole Detection Approach for Intelligent Transportation System", *Mathematical Problems in Engineering*, vol. 2015, pp. 1-7, 2015.
[7] A. Mednis, G. Strazdins, R. Zviedris, G. Kanonirs and L. Selavo, "Real time pothole detection using Android smartphones with accelerometers," *2011 International Conference on Distributed Computing in Sensor Systems and Workshops (DCOSS)*, Barcelona, 2011, pp. 1-6.
doi: 10.1109/DCOSS.2011.5982206
[8] "IEEE Code of Ethics", *IEEE*. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html. [Accessed: 20- Feb- 2018].
[9] R. Haas, “The Average Bike Riding Speed”, *Livestrong*. [Online]. Available: https://www.livestrong.com/article/413599-the-average-bike-riding-speed/. [Accessed: 22-Feb-2018].
[10] Fotosiamo, “Photography Essentials: The Sunny 16 Rule”, *SLRLounge.* [Online]. Available: https://www.slrlounge.com/photography-essentials-the-sunny-16-rule/. [Accessed: 22-Feb-2018].