Rear Collision Bicycle Warning System

By

Justin Davis,
Seongwoo Kang,
and
Gus Kroll

Final Report for ECE 445, Senior Design, Spring 2021

TA: Xihang Wu

5 May 2021

Project No. 71

Abstract

In this project, we implemented a Rear Collision Bicycle Warning System. The project combines proactive and reactive approaches to cyclist safety. The proactive approach utilizes a radar sensor that provides speed and distance information about vehicles approaching the cyclist from behind. This information is then used to determine if the alert devices should be triggered. The alert devices consist of an LED to alert drivers and a haptic motor to alert the cyclist. The reactive approach utilizes a camera and microphone to record video and audio data in the case of an accident. This data is saved to an SD card from which the user may retrieve the data after an accident has occurred. Throughout this project we faced many difficulties in integrating the hardware and software components, however, we were able to complete the core functionality of both the proactive and reactive approaches.

Contents

1. Introduction

This chapter details the motivation and background of our project as well as similar products that already exist on the market and how our project improves upon them. In the rest of this report, we will describe the overall design of our project and how we adapted the initial design throughout the course of the semester. We will also explain how we verified that each component of our project met the initial requirements that we set for them during the design phase. Then, we will give an overview of the costs of this project and provide an analysis of this project's viability on the market. Finally, we will summarize our concluding thoughts and the overall results of our project.

1.1 Objective

According to the Insurance Institute for Highway Safety (IIHS), 854 cyclists were killed and more than 47,000 cyclists were injured in motor vehicle crashes in 2018 [1]. Since 2010, the number of cycling fatalities per year has increased by 38% [2]. It has also been shown that the likelihood of cyclists suffering a severe injury resulting from a motor vehicle accident is substantially higher with SUVs and that the overall likelihood of a collision occurring is also significantly higher for hybrids and electric vehicles [1]. With an increasing number of SUVs on US roads and the growing adoption of electric vehicles, the number and severity of motor vehicle accidents involving cyclists is likely to increase in the short-term.

To counteract this expected rise in the number of accidents and fatalities in bicycle-motor vehicle collisions, we propose and implement an affordable rear collision warning system for bicycles. This unit is able to detect vehicles behind the cyclists using a radar sensor and notify them of a vehicle's presence via a vibrational sensory alert through the use of a haptic motor. The device also includes an integrated tail light that flashes when the system detects a vehicle in an attempt to alert the driver of the cyclist's presence on the road. The device also functions as a sort of "black box" for the rider. The system includes a camera and microphone which are always on and are triggered to save video and audio data in the case of an accident or near-accident. The system also allows riders to focus on what is happening in front of them, giving them the peace of mind that our device will warn them about what is going on behind them.

1.2 Background

There are currently just a few bike dash cams and only a single rear collision warning product available on the market. No one has yet combined these two features into an all-in-one bike safety device. Garmin's product has a detection range of 150 m and requires the use of your smartphone or external screen to give users notifications. This system is very expensive and also

beyond the specification of what a more casual rider needs. Fly6's camera system is currently available for pre-order and costs $229, yet only has a 4 hour battery life.

In order to reduce the likelihood of an accident and catch aggressive drivers, integrating these two systems into our product that is more affordable and easy to use along with a proposed longer battery life should make our device more appealing to a wider group of cyclists. Our implementation reduces the detection range to 100 m, with the assumption that riders will not be riding in areas where the maximum speed a car will be traveling is greater than 60 mph. This will give a cyclist a minimum of 3 seconds notification that a vehicle is approaching from behind. Then, in the event of a collision or near-collision, the built-in camera and microphone save the preceding video and audio data to be retrieved and used as evidence after the incident.

1.3 High-Level Requirements

During the design phase of our device, we identified three high-level requirements that we were able to achieve the core functionality of throughout the implementation of our project. The high-level requirements that we identified map neatly onto our overall objectives of detecting oncoming vehicles, warning the cyclist and driver of each other's presence, and saving relevant data about collisions and near-collisions in the case of an incident outlined in the previous sections.

The high-level requirements that guided our design and implementation of the device are as follows:

- Detect objects coming towards the device at a distance of at least 75 m, while maintaining a false positive rate of less than 15%.
- If the user selects trigger mode, the device is able to trigger video and audio save based on detection of collision or near-collision with the cyclist, while continuing to record (if the sensor was not damaged) until stopped by the user or memory is exhausted.
- Upon detection of an object, the time to trigger a response to the alert devices should be less than 750 ms.

These three high-level requirements served as our guiding principles throughout the entire design and implementation phases of the project. Whenever we were in doubt about what decisions to make, we referred back to these requirements in order to adapt our design and implementation to meet these goals.

2. Design

This chapter details the entire design process of our overall system. We start by providing information about the overall system through the block diagram and physical design. Then we give an overview of how the initial design came to fruition for each component and how this design changed over the course of the semester. Finally, we give a more detailed description of how each component was designed and implemented.

2.1 Block Diagram and Physical Design

This section simply displays the block diagram and physical design of the project for reference. This section also outlines all of the major blocks in our final design. In Figure 1, we have the block diagram which displays all of the major components and modules in our design. The block diagram shows that the design consists of four main modules which are power, sensors/data, control, and user interface (UI). Figure 2 shows the physical design of our device after implementation. Notice that the physical design differs somewhat from our block diagram because we were not able to complete all of the components outlined by our initial design in time. However, we were able to complete the core functionality of our project both in hardware and software.

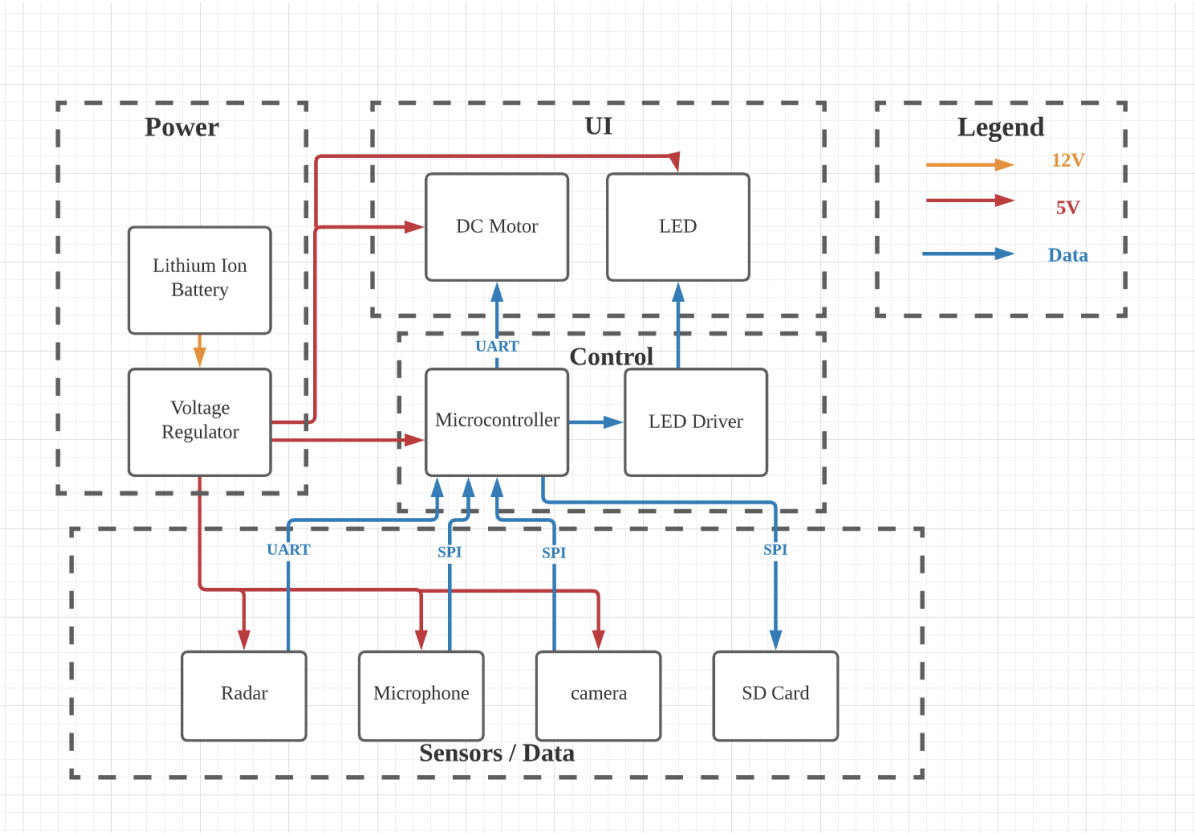


Figure 1: Block diagram of the initial overall design. Design consists of four main modules; power, sensors/data, control, UI.

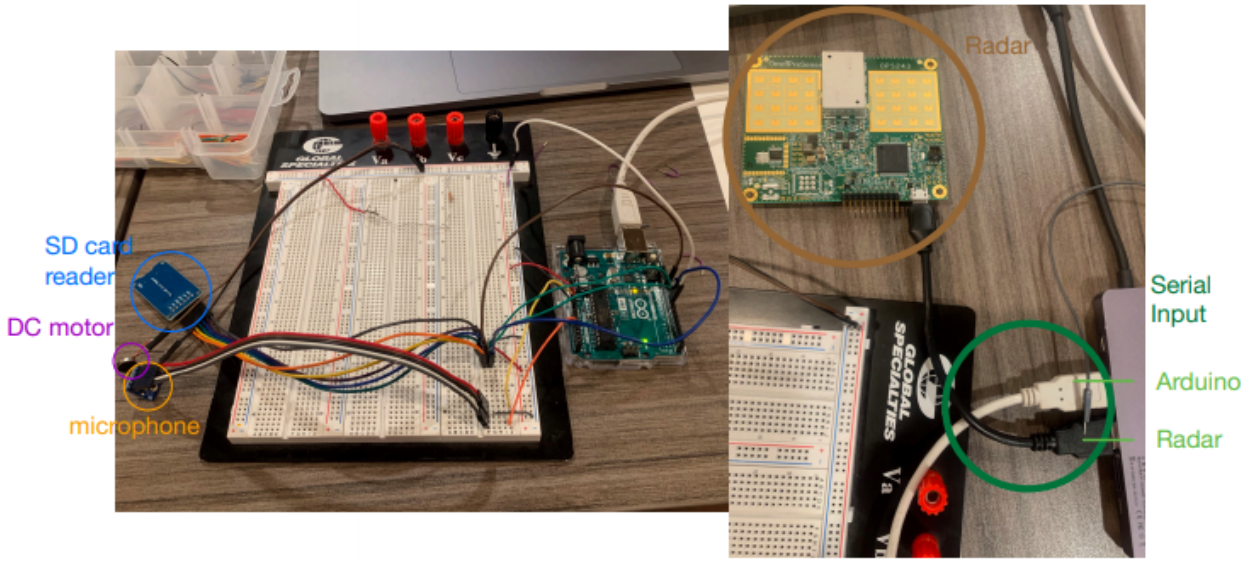


Figure 2: Physical design of the device. The left image shows the SD card reader, haptic motor, LED, and microphone used for alert, data collection, and data retrieval. The right image shows the serial input used for the arduino and radar to allow us to program our device.

Although our physical project build does not complete all the specifications of the block diagram, it does achieve the core functionality of the project through the use of the components displayed in Figure 2. The microphone and SD card reader allow us to collect and retrieve data in the case of an accident. The haptic motor and LED allow us to alert the driver and cyclist of each other's presence. The radar, arduino, and computer allow us to detect incoming objects and trigger our alerts if certain thresholds are met. The technical details of each of these modules will be discussed in the following sections.

2.2 Design Procedure

This section outlines the major design decisions we made for each block at the most general level. This section also describes potential alternative approaches to our initial design and why we chose the method that we chose for implementation. Finally, this section provides an explanation of why certain components failed and how we could have saved time by using different approaches from our initial design.

2.2.1 Power

Our initial design for the power supply utilized a battery and a dual output linear regulator with an input of 7.2 V and outputs at 5 V and 3.3 V. We faced significant challenges with getting our power supply to work with the rest of the overall system, despite designing the power supply to work with these components. After trying several different approaches such as designing our own circuit initially and then using existing products, we were not able to get this module to work properly in the overall system, though it did work nominally. This stage of the

implementation process cost us a lot of time and effort. This process was especially difficult and time consuming because it required us to order several different PCB iterations and components that delayed our implementation process because of shipping time.

2.2.2 Sensors/Data

This module consists of four main components. These components are the radar sensor, microphone, camera, and SD card/card reader. The radar sensor is used to detect oncoming vehicles. We chose radar because ultrasonic sensors, though cheap, can be quite unreliable in outdoor conditions compared to radar. We also felt that radar would be preferable to lidar because lidar sensors are often extremely expensive. We chose to use a microphone and camera for data collection. On their own, these two devices can capture only part of the evidence in the case of an accident. For this reason, after some deliberating during the design phase, we decided to use both devices in our final design so that we could capture audio and video data. Although we were not able to integrate the camera into our final build, we were able to verify its ability to capture images. Finally, we decided to use an SD card to save the data recorded by our data collection devices. The ability to save this data is obviously critical because without it, we have no way to ensure the cyclist's ability to prove their innocence in the case of an accident.

2.2.3 Control

This module consisted of a microcontroller that we could use to communicate with all of our alert, data collection, data storage, and sensor devices. Figure 3 depicts the general control flow of our code that ran on this microcontroller. In Figure 3, we can see that the general control flow consists of six major components. The first component is to initiate the device. The second component is to turn the radar, microphone, and camera on. The third is to determine if there is an object incoming. The fourth is to alert the cyclist and driver of each other's presence using the alert devices if an incoming object was detected. The fifth is to determine if the speed detected is over some set threshold. The sixth is to record the video and audio data for the next 10 seconds to the SD card if the detected speed was over the set threshold. In all cases, the flow of the control eventually returns to the state where our devices are turned on and looking for incoming objects. The microcontroller was not able to be implemented into our final build due to complications that we will discuss in following sections.

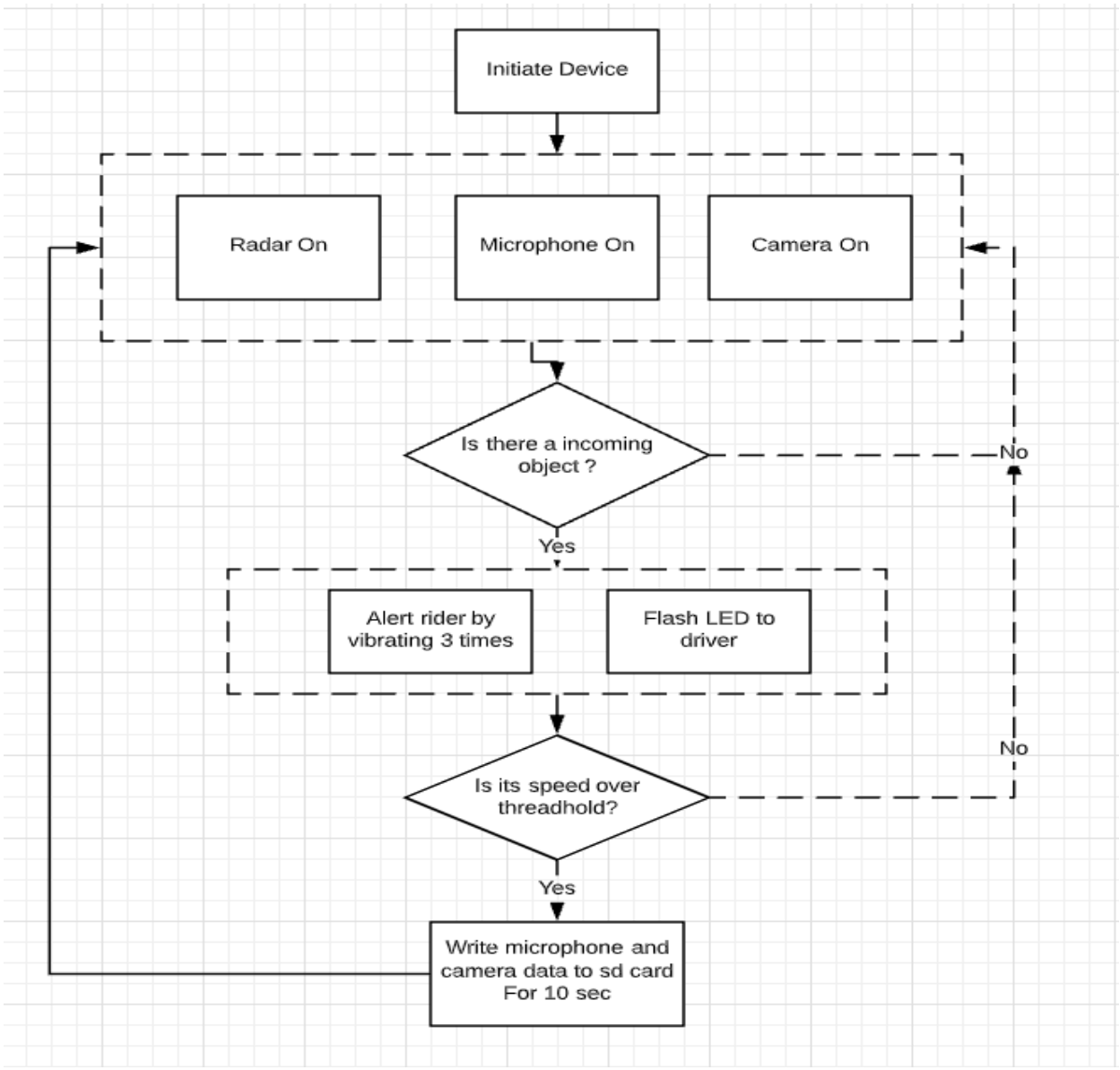


Figure 3: Control flow. This diagram depicts the control flow of the software for our system. The general principle is to always be checking if there are incoming objects. If there aren't, then we continue checking. If there are, then we trigger our alert and data collection devices.

2.2.4 User Interface

This module consists of our two alert devices. The first alert device is a haptic motor and the second alert device is an LED. We require two alert devices because one is used to notify the cyclist of the driver and the other is used to notify the driver of the cyclist. The haptic motor vibrates when a vehicle is detected to alert the cyclist of the driver. The LED flashes when a vehicle is detected to alert the driver of the cyclist. We considered several other alert mechanisms for our device such as an audible alert or a display screen. However, in the case of the audible alert, we found that this would probably be annoying to the user as well as

surrounding pedestrians and other cyclists. In the case of the display screen, we found that this would require an entirely different device which would be much more expensive. Thus, using a display screen instead of a vibrational motor would defeat the purpose of our cheap, all in one device.

2.3 Design Details

This section provides a more detailed description of all of the major blocks and components in our design. This section also describes some of the challenges we faced in implementing these components and how we overcame these challenges. For components that we were not able to complete, we give a detailed description of why these components failed and how we could have fixed them if we had more time to complete the project.

2.3.1 Power

Our power module design consists of a 7.4 V battery with a dual voltage regulator. We wanted the battery to power the device for up to 3 hours. The battery uses JST plug in to provide 7.4 V for 2200 mAh. Assuming our system draws 1 A at maximum, it should be able to power the device for up to 3 hours as required. Also, since our sensor modules make use of two different voltages (3.3 V & 5 V), we needed two voltage regulators. Originally, we were planning to use two voltage regulators based on the circuit diagram in Figure 4.

However, in order to save spacing on our PCB and avoid possible error in the circuit, we have decided to use the existing dual voltage regulator (TPS767D301MPWPREP) from Texas Instruments that can provide a fixed voltage of 3.3 V and another voltage that can be set to a range from 1.5 V - 5 V.

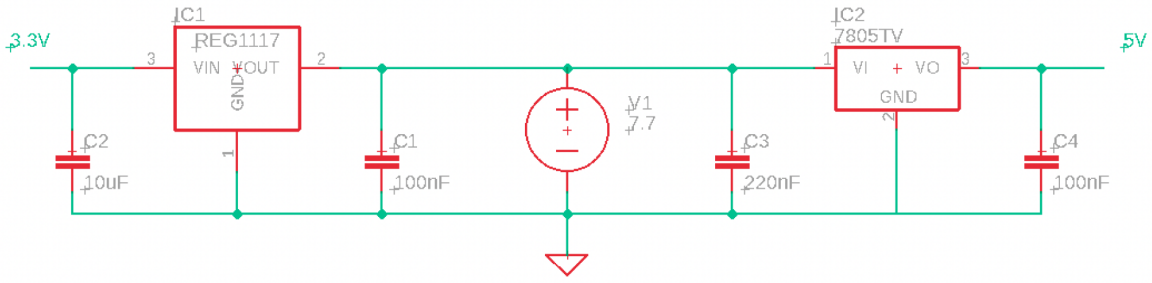
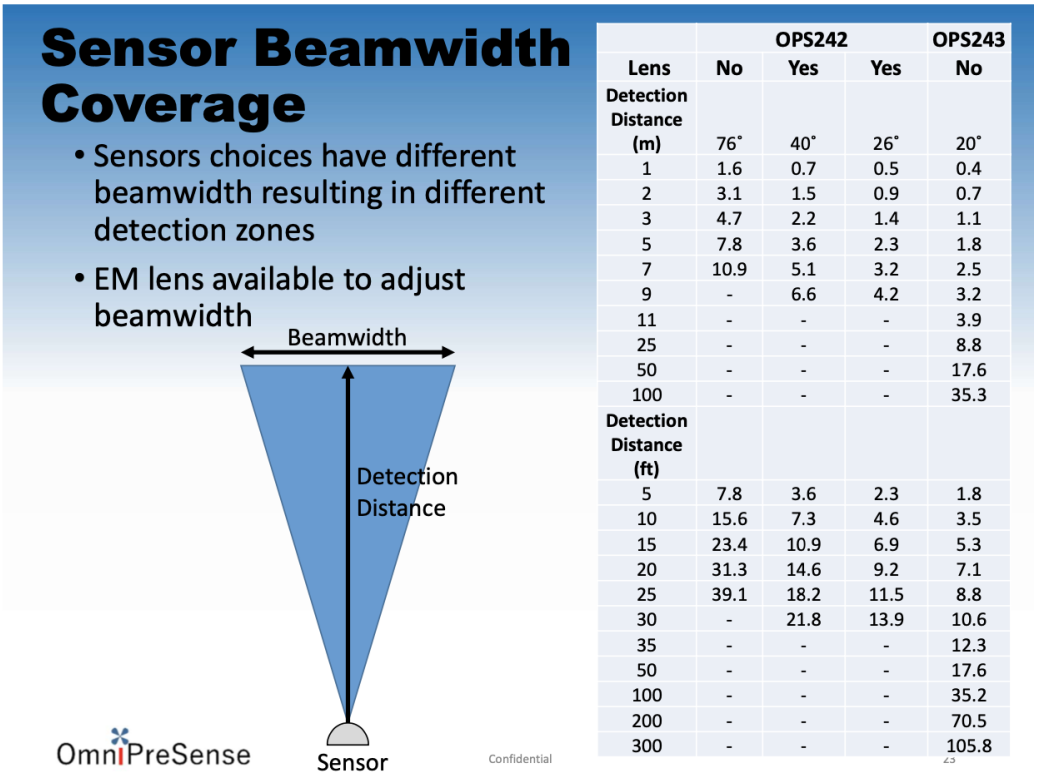


Figure 4: Dual voltage regulator circuit. We were originally planning to use this circuit in our project build, but abandoned it after finding an existing circuit that achieved the same functionality due to time constraints.



	OPS242			OPS243
Lens	**No**	**Yes**	**Yes**	**No**
Detection Distance (m)	76°	40°	26°	20°
1	1.6	0.7	0.5	0.4
2	3.1	1.5	0.9	0.7
3	4.7	2.2	1.4	1.1
5	7.8	3.6	2.3	1.8
7	10.9	5.1	3.2	2.5
9	-	6.6	4.2	3.2
11	-	-	-	3.9
25	-	-	-	8.8
50	-	-	-	17.6
100	-	-	-	35.3
Detection Distance (ft)				
5	7.8	3.6	2.3	1.8
10	15.6	7.3	4.6	3.5
15	23.4	10.9	6.9	5.3
20	31.3	14.6	9.2	7.1
25	39.1	18.2	11.5	8.8
30	-	21.8	13.9	10.6
35	-	-	-	12.3
50	-	-	-	17.6
100	-	-	-	35.2
200	-	-	-	70.5
300	-	-	-	105.8

Figure 5: OPS243 radar beamwidth chart. This is the variable beamwidth chart provided by the manufacturer for the radar module that we chose to use in our final design. Detection distance is given in meters (above) and feet (below).

Even though we could not utilize the power module in the final project build due to time constraints and shipping delays, we were able to test it nominally, which will be described in Chapter 3.

2.3.2 Sensors/Data

We wanted our device to be able to detect objects less than or equal to 75 m away. The best radar option to achieve this was the OPS243 radar from Omnipresence, which can detect objects up to 100 m away with varying beamwidth coverage as shown in Figure 5, provided by the manufacturer [3].

Using the distance and speed of the detected object provided by the radar, we calculate the intensity of vibration necessary for the haptic motor. If this intensity meets a threshold that we set in our code, we trigger our alert devices with the appropriate intensity. We consider vehicles that are close and fast to be the most dangerous and vehicles that are far and slow to be the least dangerous. Based on this design principle, we calculate the required intensity using the following equation,

$$I = \left(\frac{d_{max} - d_{current}}{d_{max}} + \frac{v_{current}}{v_{max}}\right) / 2$$

where I is the required intensity, d_{max} is the maximum sensing distance, $d_{current}$ is the distance of the currently sensed object, v_{max} is the maximum sensing velocity, and $v_{current}$ is the velocity of the currently sensed object. This equation outputs a value in the range 0-1, which is then multiped by 255 to determine the strength of the motor through PWM. Note that if the output intensity is less than 0 due to an object moving away from the device, we output an intensity of 0 to indicate that no danger is detected.

Actual code that preprocesses the data from the radar that was done in python and the code used in the arduino to trigger our alert devices and save data recorded by the microphone is provided in Appendices B and C in Figures 7 and 8, respectively.

In addition to the radar, we planned to incorporate a microphone, camera, and SD card reader modules in the overall system. We chose the OV7670 camera module, Hiletgo SD card reader module, and Adafruit MEMS Microphone module because they are known to interact well with the microcontroller that we chose (Atmega328-PU) initially. Even though we were able to integrate the system with the microphone and SD card reader, we could not integrate the camera due to lack of design consideration. The camera that we chose required too many pins for our physical design, and our microcontroller simply could not process the video data or radar data provided to it fast enough.

2.3.3 Control

In order to process the data from the radar, trigger our alert devices, and communicate with other sensor modules to save relevant data in the case of an accident, we originally planned to use the Atmega328-PU, which is the microcontroller used in arduino. We had various difficulties when trying to program and integrate the microcontroller with the various sensor and alert modules. Namely, the microcontroller could not clear the data provided to it by the radar fast enough. Eventually, we decided to use a computer to receive and process the data from the radar and an arduino to trigger the system's alert and sensor modules based on the data processed by the computer. This stage of implementation cost us the most time because we tried using multiple microcontrollers and could not achieve our originally intended design. If we were to do this project again, we would have used a more powerful microcontroller such as a Raspberry Pi initially and completed the rest of the project using that.

2.3.4 User Interface

The User Interface (UI) module includes a haptic motor and LED. The vibrating motor is connected to arduino, and its power is controlled by the intensity that is calculated by the data input from the radar. The LED is also connected to the arduino and is triggered only when the intensity reaches or exceeds a set threshold. For testing purposes, we used a standard vibrational motor from Seeed Technologies and standard LEDs from Adafruit. If we were to actually deploy our product to the market, we would have used more powerful devices for the UI module. We simply chose these cheaper options to reduce the cost of our overall design during testing and implementation. Additionally, these devices were adequate to verify the functionality of our overall hardware and software implementations.

3. Verification

This chapter discusses the testing of our completed project and the major blocks. We discuss how we verified each component of our project based on the requirements and verifications tables that we made during the design phase of our project. For components that we were not able to verify or integrate into our final project build, we describe why these components failed and how we would have changed them to work in our overall system if we had more time. All of the requirements and verifications are specified in Table 2 located in Appendix A.

3.1 Power

In order to test that 3.3 V and 5 V is provided reliably, we had to solder the dual voltage regulator to the PCB. However, because the size of the regulator was smaller than we had expected, we failed to solder the regulator onto the PCB. If we had more time, we would have reverted to the original design where we make use of two voltage regulators.

3.2 Sensors/Data

This section details how we verified all of our various sensor components. These components include the radar, camera, microphone, and SD card.

3.2.1 Radar

In order to test that the radar correctly detects an object 75 m away and provides reliable input, we connected the radar to the computer and observed the serial data input. Both indoors and outdoors, the radar detected all objects. Also, we could program the radar using the provided API to fit the needs of our system.

3.2.2 Other Sensors

All other sensors (Camera, Microphone, SD Card Reader) were tested on a breadboard using an arduino. We had connected both the microphone and SD Card Reader so that the sound data could be saved to the SD card. We have confirmed its functionality by retrieving the data through a computer. Once we retrieved the sound data from the SD card onto a computer, we used existing software to convert this data into a WAV file and then into a MP3 file. Finally, we listened to this data and verified that it was correct.

The camera module was tested based on a tutorial provided online using the circuit pinout in Figure 6 provided by the manufacturer [4]. We found that the camera could capture image data, however, we were unable to implement the camera into our final project build.

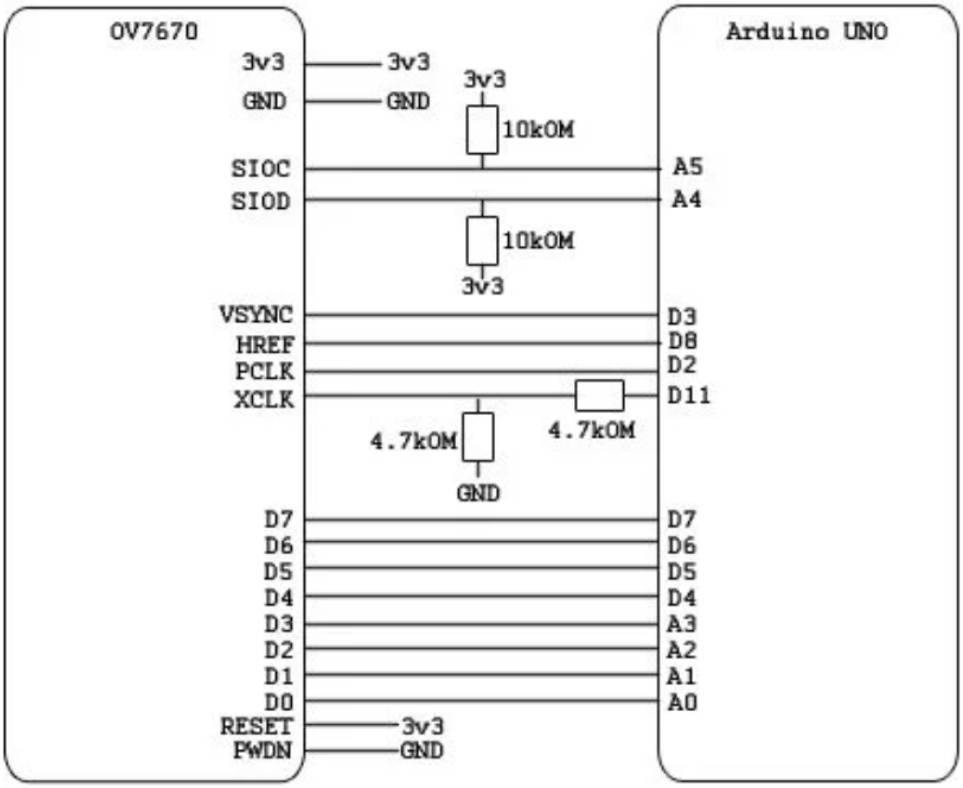


Figure 6: Camera pinout. This diagram was provided by the manufacturer of the camera module we used. We utilized this pinout diagram to verify the functionality of the camera module.

3.3 Control

Programming and testing the Atmega328-PU was one of the main difficulties that we encountered in this project. Initially, we attempted to program the microcontroller using FTDI Breakout from Sparkfun. However, despite numerous attempts with various circuits, we could not program nor bootload the microcontroller. As a result, we decided to use arduino which uses the Atmega328. If we had more time, we would have used the IC chip mount so that we could solder the mount onto a PCB and simply program the microcontroller using the arduino instead of having to design and insert a separate module for programming.

3.4 User Interface

The LED and DC motor for vibration were tested simply by connecting them to a power source and testing their strengths. The strengths of both the LED and DC motor were weaker than we had expected. Even though we had anticipated this and ordered 4 identical motors, they were still not strong enough to reliably alert the rider through the seatpost on a moving bike. Since it is difficult to determine the strength of a haptic motor by number, it would have been better to have one motor as a comparable device.

4. Costs

This chapter provides an overview of all the components we used in our final project design and their respective costs. Additionally we estimate the cost of labor for each of our group members. Finally, using all of this information, we provide a brief analysis of our project's commercial viability in terms of the total cost of components, labor, and mass production.

4.1 Parts

Table 1 provides a list of parts used in our proposed final project build and their respective costs. Table 1 gives a complete outline of all of the costs of all of the components if we had completed this build in time to give an accurate view of how much our system would cost.

Table 1: Costs of All Components in Final Build

Part	Manufacturer	Cost	Quantity	Total Cost
OPS243	Omnipresence	$229.00	1	$229.00
OV7670	HiLetgo	$4.49	1	$4.49
Micro SD TF Card module	HiLetgo	$2.00	1	$2.00
Microphone breakout	Adafruit	$4.95	1	$4.95
LED (pack of 5)	Adafruit	$3.95	1	$3.95
Dual voltage regulator (TPS767D301MPWPREP)	Texas Instruments	$11.35	1	$11.35
DC motor (vibration)	Seeed Tech	$1.20	4	$4.80
Atmega328-PU	Microchip	$2.30	1	$2.30
Battery	VIDAR	$13.99	1	$13.99
16Mhz crystal	Sparkfun	$0.95	1	$0.95
Serial Breakout	Sparkfun	$15.95	1	$15.95
			Total:	$293.73

4.2 Labor and Commercial Viability

An entry-level Electrical Engineer/Computer Engineer makes about $85,000 annually, which can be converted into an average hourly rate of $40. Based on this data, we can compute the estimated cost of labor to produce our product,

$$C = 3(\text{Engineers}) * \$40/\text{hr} * 10\text{hrs/wk} * 12\text{wk} * 2.5(\text{overhead factor}) = \$36{,}000$$

where *C* represents the total cost of labor to produce our final product. Although $36,000 is a relatively small investment for R&D, we determined that the product we completed in this project would not be commercially viable due to the costs of the components we used. In order to make our product commercially viable, we would have to reduce the cost of the radar we used in our final design, since this component is by far the most expensive. In order to do this, we would probably need to design our own radar module from scratch because pre-built radar modules that currently exist on the market are often very expensive and come equipped with many unnecessary functions that drive up the cost.

5. Conclusion

In this chapter we will give our concluding thoughts about the project as well as provide a summary of our accomplishments, uncertainties, ethical considerations, and proposed future work. We will also discuss the societal impacts of our project and how it would change traffic behavior in general if it became a popular product on the market.

5.1 Accomplishments

In terms of our accomplishments, we are proud to say that despite major complications when integrating our hardware and software that cost us significant amounts of time, we were indeed able to complete the core functionalities of our project as outlined in the high-level requirements. As discussed in previous chapters, we were able to integrate a radar with a detection distance of up to 100 m into our overall design. Additionally, we were able to process the data provided by the radar to acquire speed and distance information about oncoming objects in order to determine how far and how fast an object is approaching the device. We then used this information to trigger our alert and data collection devices based on thresholds that we set in our software during the implementation and testing phases. During testing, we determined that the response time of all of our devices fell in line with the high-level requirements that we set at the beginning of the design phase. Finally, we were able to successfully record data from our microphone, save that data to our SD card, and retrieve that information in a format that can be converted to usable formats by using existing software on the internet. Although we had trouble getting our camera to work in the overall system because of hardware limitations, we were able to verify the functionality and retrievability of our camera and video data with respect to the SD card.

5.2 Uncertainties

Due to time constraints, we were left with two major uncertainties at the end of the implementation phase. The first and most obvious uncertainty is that we were not able to integrate our entire system onto a single PCB and contain the entire device within a case that could be mounted to a bicycle. The second is that we were not able to get the camera functioning within our overall system. Our inability to complete these portions of the project can mainly be attributed to shipping delays, faulty PCB designs, and hardware challenges that we faced when trying to integrate the software with the hardware. Upon starting this project, our group had a noticeable lack of experience with hardware overall. However, over the course of the semester, we were all able to increase our knowledge of hardware significantly. This increased level of knowledge about hardware would lead us to make different decisions during the design phase if we were to do this project again. The most important change we would make during the design phase that would save us a great amount of time and effort would be to select a different, more powerful microcontroller such as a Raspberry Pi so that we could meet

the necessary requirements to process both video and audio data fast enough. This change alone would have given us enough time to complete one more draft of the PCB design to mirror the circuit implementation on our final breadboard that we then would have been able to integrate into our final system. We also would have had enough time to fit all of our components into a case and mount it to a bicycle if we had made this change during the design phase. Finally, although the alert devices we used for our final demonstration were sufficient to demonstrate the functionality of our system, if we were to actually deploy our product on the market we would have used a larger LED and haptic motor.

5.3 Ethical Considerations

Throughout design, implementation, and testing we had to consider several ethical problems that could arise throughout the implementation and use of our device. These considerations, along with our high-level requirements, guided the important decisions we made throughout the entire process of this project. Had we not carefully outlined these ethical considerations at the beginning of the semester, we could have caused serious bodily harm to ourselves during implementation and testing.

The first ethical consideration has to do with the haptic motor that we use to alert the cyclist of incoming vehicles. Our device makes use of a haptic motor attached to the bottom of the bike seat to alert the user of incoming vehicles. During our design and implementation, we had to ensure that we output a variable vibrational intensity to the motor in order to ensure that the motor would not interfere with the cyclist's ability to ride. For this reason, we had to moderate the power of vibration carefully during testing.

The second ethical consideration has to do with the case we would use to contain our device. Our device is meant to be used outdoors attached to a moving bicycle, so it needs to be able to sustain a significant amount of dust and water along with impact in the case of a collision. Thus, the device should be encased in a protective box that meets the IP65 [5] enclosure standards.

The third ethical consideration has to do with working with various electrical components, especially lithium ion batteries. Working with lithium ion batteries can potentially be dangerous so we made sure to follow all of the necessary protocols when providing power to our device. We never worked near water or other liquids and always took precautions to make sure that sensitive components like the lithium ion battery and voltage regulator were always separated from potentially hazardous materials.

The fourth ethical consideration has to do with testing our device. Since our device is meant to be used to prevent vehicle on bicycle related accidents, we had to make sure to design testing

procedures that would not endanger us or the device. For this reason, throughout the testing phase, we used scaled down versions of our alert thresholds and our own bodies to verify the functionality of the device rather than actual motor vehicles.

The final ethical consideration has to do with notifying users of uncertainties about our system. Even though our device alerts the user of potential dangers around them, it is still their responsibility to constantly monitor their surroundings since there is always a possibility of the device malfunctioning. However, being accustomed to the functionality of the device, users can often forget to do so. It is our responsibility to ensure their safety from accidents that can possibly be caused by the device, which is in line with Code of Ethics I. 1 “To accept responsibility…” [6]. When deploying our product to the market, we would strive to accomplish this consideration by emphasizing safety procedures in using the device and reminding the user that no device can completely replace their own caution.

5.4 Future Work

The future work for this project consists of three major aspects. The first aspect is that we would like to integrate our entire system onto a PCB and fit our device into a case that could be mounted on a bicycle. This would be fairly straightforward and would make our device much closer to being ready for market. The second aspect is to develop a mobile application that could communicate with the WiFi module on our radar sensor. We could then use this mobile application as an alternate method of notifying the rider of oncoming vehicles. Implementing the actual application would be fairly straightforward, but the hardware communication part with WiFi would be much more challenging. The final aspect has to do with reducing the overall cost of our device. The radar sensor was by far the most expensive component of our device. In order to reduce this cost, we would probably need to design our own radar module with only the necessary components that we need since the radar that we used for this project came equipped with many unnecessary additional functionalities. This aspect would require the most work of all three proposed future work aspects by far.

5.5 Concluding Thoughts and Societal Impacts

In terms of societal impacts, we predict that the widespread use of our product would result in two major outcomes. The first outcome would be that, in the short term, the number of motor vehicle on bicycle related accidents would be reduced due to our device’s ability to alert the cyclist and the driver of each other’s presence. The second outcome would be reflected in the long term effects of our device. Since our device provides cyclists with the ability to prove their innocence in the case of an accident, in the long term of widespread use of our device, we would probably see that drivers would take more caution around cyclists in general. This is due

to the fact that drivers would be scared of potential repercussions for reckless driving in the presence of cyclists.

Overall, we are glad to have had the opportunity to take part in this class despite complications caused by COVID. As a group, we learned a lot about teamwork, time management, design, implementation, and testing throughout the course of this semester while working on our project. We also gained valuable knowledge about how to make a product useful and marketable to consumers. We would like to thank all of our professors as well as the entire course staff for doing a great job this semester in running this class and providing us with feedback on our project at every stage. We hope that all of the experience we gained throughout the course of this class is reflected in this report.

References

[1] “Pedestrians and Bicyclists.” *IIHS-HLDI Crash Testing and Highway Safety*, 2020, www.iihs.org/topics/pedestrians-and-bicyclists.

[2] “Fatality Facts 2018: Bicyclists.” *IIHS-HLDI Crash Testing and Highway Safety*, 2020, www.iihs.org/topics/fatality-statistics/detail/bicyclists.

[3] OmniPreSense Corporation. “Home.” *OmniPreSense*, 19 Sept. 2019, omnipresense.com.

[4] Instructables. “OV7670 Arduino Camera Sensor Module Framecapture Tutorial.” *Instructables*, 23 Sept. 2017, www.instructables.com/OV7670-Arduino-Camera-Sensor-Module-Framecapture-T.

[5] “IP Rating Chart.” *DSMT.Com*, 23 Feb. 2019, www.dsmt.com/resources/ip-rating-chart.

[6] “IEEE Code of Ethics.” *IEEE Code of Ethics*, 2020, www.ieee.org/about/corporate/governance/p7-8.html.

Appendix A Requirements and Verifications Table

Table 2: Requirements and Verifications Table with Completion Status

Component	Requirements	Verification	Completion Status
Battery	Must supply power to the system for 3 hours.	Attach the battery to the system and run continuously for 3 hours.	No (though the power supply works nominally)
Voltage Regulator	Must supply appropriate voltage to all components of the system.	A. Attach the voltage regulator to the system. B. Take voltage readings for the voltage being supplied to each component.	No (soldering physically impossible due to its small size)
Microcontroller	Must be programmable to trigger the system.	Bootload and program test functions.	No (FTDI breakout has failed)
Radar	Must be able to detect an object 75 m away and provide reliable UART connection.	Attach the radar to computer/arduino to see serial input.	Yes
Microphone	Must be able to convert sound data to voltage data.	Record the sound input using arduino.	Yes
Camera	Must be able to show consecutive images.	Capture an image using arduino.	Yes
SD Card Reader	Must be able to store data into the SD card.	Store microphone sound data using arduino and retrieve on computer.	Yes
Haptic Motor	Must be strong enough to alert rider.	Connect to a power source to test.	Yes (However weak)

Component	Requirements	Verification	Completion Status
LED	Must be visible from 50 m away.	Connect to a power source to test the strength.	Yes (However weak)

Appendix B Python Code

```
import serial,timeit,time
import math
radar = serial.Serial('/dev/tty.usbmodem144301', 19200, timeout=1)
print("conncted to " + radar.portstr)


arduino = serial.Serial('/dev/tty.usbmodem144101', 19200, timeout=1)
print("conncted to " + arduino.portstr)

cur_speed, cur_distance=0,0


dist_max = 100
dist_min = 2
speed_max = 31
speed_min = 0.5


speed_deteced=timeit.default_timer()


while True:
    radar_input_raw = radar.readline()
    print(radar_input_raw)
    radar_input_decoded=radar_input_raw.decode("utf-8")
    print(radar_input_decoded)

    state, value = radar_input_decoded.split(',')

    state = state.strip('\"')
    tmp = value
    value = value.rstrip()
    print("input",state,value)
    print("current",cur_speed,cur_distance)

    # dist_range = 2 - 100m
    # speed_range = 4 - 31 mps

    v = float(value)
    if state == 'mps':      # velocity
        if v>=0 and v<= speed_max:
            cur_speed = v
            speed_deteced=timeit.default_timer()
            flag=1
```

```
elif state == 'm':
    if v>=dist_min and v<=dist_max:
        cur_distance = v
        flag=0

intensity = ((dist_max - float(cur_distance))/1000+float(cur_speed)/31)/2


# if no speed update for 2 sec, reset to 0
tt=timeit.default_timer()
#print(tt)
t=(tt-speed_deteced)
if t >= 1:
    cur_speed=0


print("intensity----------------------",intensity,)

threshold=0.07
if intensity>= threshold:
    arduino.write(bytes('r','utf-8'))
    time.sleep(0.5)
    print('trigger')
```

Figure 7: Python code. This code preprocesses the radar data to be in a usable format (speed and distance). This code also calculates the variable intensity and compares it to a threshold value that we set during the testing phase to determine if the other devices should be triggered. If the intensity meets or exceeds this threshold, we trigger our other devices and use the intensity to trigger the haptic motor at some variable level of vibration.

Appendix C Arduino Code

```
#include <SPI.h>
#include <SD.h>
#include <TimeLib.h>

int data_pin = 9;
int led=8;
const int chipSelect = 10;
int microphone_pin =A0;
int startMillis;
int triggeredTime=-10000;
int currentTime;

void setup() {

  // Open serial communications and wait for port to open:
  Serial.begin(9600);
  if (!SD.begin(chipSelect)) {
    Serial.println("Card failed, or not present");
    while (1); // don't do anything more:}
    Serial.println("card initialized.");
  } else (Serial.println("Done"));
  //End sd card 1st setup
  startMillis = millis();
}


int zing2(){
  for(int i=0;i<2 ;i++){
    analogWrite(data_pin,255);
    delay(100);
    analogWrite(data_pin,0);
    delay(250);
  }
  return 0;
}

int zing6(){
  for(int i=0;i< 2;i++){
    for(int j=0; j<6;j++){
      analogWrite(data_pin,255);
      delay(75);
      analogWrite(data_pin,0);
      delay(75);
    }
    delay(250);
  }
  return 0;
}

void zz(){
  analogWrite(data_pin,255);
  delay(300);
  analogWrite(data_pin,0);
}
```

```
void flsh(){
  while (Serial.available()){
    Serial.read();
  }
}

void flash(){
  digitalWrite(led,HIGH);
  delay(300);
  digitalWrite(led,LOW);

}

void loop() {

  if (Serial.available()>0){
    zing6();
    flsh();
    triggeredTime=millis();
  }

  currentTime=millis();
  if ((currentTime-triggeredTime)< 10000){
    writeSD();
    digitalWrite(8,HIGH);
  }else{
    digitalWrite(8,LOW);
  }


}

void writeSD(){
  String dataString = "";

  int sensor = analogRead(microphone_pin);
  dataString += String(sensor);
  Serial.println(dataString);
  File dataFile = SD.open("sound.txt", FILE_WRITE);

  // if the file is available, write to it:

  if (dataFile) {
    dataFile.println(dataString);
    dataFile.close();
  }
  // if the file isn't open, pop up an error:
  else {
    Serial.println("error opening datalog.txt");
  }
}
```

Figure 8: Arduino code. This code triggers our alert devices (LED and haptic motor) when the necessary intensity threshold is met and the Python code triggers this code to run by making serial data available. In addition to triggering the alert devices, this code also triggers the microphone to start recording and saves the recorded audio data to the SD card for later retrieval.