shipout/backgroundshipout/foreground

CycleSafe: A Technology-Assisted Safety Jacket to Prevent Traffic Accidents

Benjamin Huang, *Electrical and Computer Engineering, Carnegie Mellon University*
Siddhanth Lathar, *Electrical and Computer Engineering, Carnegie Mellon University*
Michael You, *Electrical and Computer Engineering, Carnegie Mellon University*

***Abstract*—Although biking is convenient way to travel, cyclists are at a high risk of accidents, where thousands of cyclists die every year. One reason is that cyclists have a limited number of safety options available, such as reflective vests and flashing lights, that give them little awareness of the road. CycleSafe is a comprehensive safety system on a bike, that alerts users through a wearable jacket when the cyclist is in a dangerous situation. The jacket notifies the user through a combination of visual, auditory and sensory feedback, and also features a large LED matrix that significantly increases the visibility of the rider compared to current cyclist safety products. CycleSafe aims to provide a safer biking experience by maximizing surrounding awareness to cyclists, while also increasing their visibility to other vehicles and their drivers.**

***Index Terms*—Cyclist safety, wearable, real-time embedded system, LIDAR, haptic feedback**

I. Introduction

EVERY year, around 1,000 cyclists lose their lives and over 50,000 are injured in traffic accidents. Whereas cars are becoming safer, equipped with the latest safety features, biking technologies have remained mostly stagnant in the last few decades. This is a problem, especially because biking is becoming more popular in cities [**?**], where 71% of biking fatalities occur [**?**]. In particular, accidents are commonly caused by 5 factors - three of which are cyclists getting hit by 1) cars coming out of side streets, 2) cars turning into the side streets and 3) cars overtaking when a cyclists is moving left to avoid an obstacle [**?**]. To prevent these accidents from happening, we present the CycleSafe, a system integrated with a wearable that helps keep bikers safe from these types of accidents by:

- Helping you stay aware of your environment, giving danger warnings of lane drifting, blind spots and collision detection, and providing standard vehicle turn signals and brake lights, along with other communicative display signals.
- Navigation help, that reduces the need to look at your phone, and instead lets you focus your attention on the road.
- Gesture recognition, that reduces the need for large movements, and instead allows the user to make small movements to perform tasks, in order to reduce unsafe movement while biking.

The jacket also features many improvements to the cycling experience, such as glove warming, and on-demand music and call with your phone, features which current cyclist jackets on the market do not offer. To evaluate the effectiveness of the jacket under dangerous circumstances, we will conduct a simulation test in a closed test track. We are aiming to have $<5\%$ false negatives, and $<20\%$ false positives in 60 trial runs. We want to have a lower false negative than false positive because missing a dangerous condition is much worse than alerting the user where there are no dangers. More details about our testing can be found in **??**.

II. Design Requirements

A. Mobile App

The requirements for the mobile app are as follows:

- Send data to the Raspberry Pi as inputs to the warning system logic. Data types include:
  - Bicycle speed
  - Current location
  - Intersection data
- Give the user live-time navigation instructions
- Give the user the ability to customize their notification settings

To test the data transfer, we will write a test program on the mobile app and the Raspberry Pi that works the following way:

1) The mobile app sends a burst of pre-selected data
2) The Raspberry Pi verifies the data is in the proper format and correct

For navigation, we will test the app empirically, by choosing a series of locations to visit, and navigating to them on a bike with our mobile app giving live instructions. For customization aspect of the app, we will write UI tests to make sure all buttons are working, settings are saved, and that they are actually sent to the jacket Arduino correctly.

B. The Jacket

The requirements for the jacket are as follows:

- Give the user warnings via peripherals, which include
  - Piezo Buzzers (Auditory)
  - Vibration Motors (Sensory)
  - LEDs (Visual)
- Give the user critical and less-critical warnings in time. These requirements can be found in Section **??**
- Make sure the LEDs are bright enough based on eCFR (Electronic Code of Federal Regulations) vehicle standards

- Meet IPX4 waterproofing requirements

To test the peripherals on the jacket, we will write individual component unit testing code. Specifically,

- LED Strip: Send a series of patterns, and make sure that they appear correctly and bright enough.
- Piezo Buzzers: Send a series of beeps, and make sure they are loud enough and are happening at the correct times.
- Vibration Motors: Send a series of vibration pulses, and make sure they are felt and are happening at the correct times

In order to verify the correct functionality of these tests, we will put the jacket on a user and have them respond yes or no to the above tests. For the LED strips, we have to match the eCFR standards [?] which are

- 50 mcd for tailights
- 300 mcd for turn signal
- 300 mcd for stoplights

We can test the brightness of the LEDs using a luminosity sensor.

In order to test the waterproofing, we will follow the IPX4 specification [?], and test according to the description:

Protects from splashing water, no matter the direction (1)

Once we have our jacket assembled, we will conduct a series of splash tests, and use the unit tests described earlier for individual components to evaluate the waterproofing effectiveness.

C. *Warning Thresholds*

The requirements for the CycleSafe system will be based on a maximum cycling speed of 9 m/s (about 20 mph or 33 km/h, which is high speed for a commuting cyclist averaging 15 km/h in some places [?]). Given this speed, we can determine the lead time necessary for a cyclist to react to dangers.

1) Frontal warning: With regard to frontal collisions, the necessary reaction time is fairly straightforward, since the cyclist simply needs sufficient time (or distance) to react and stop. From the maximum cycling speed, we can determine the cyclist's braking distance. In [?], the following formula is given to determine braking distance s_{brake}, including perception and brake reaction time (the time needed to press the brakes):

$$s_{\text{brake}} = \frac{v_b^2}{254(f \pm G)} + \frac{v_b}{1.4} \tag{2}$$

where v_b is the velocity of the bicycle in km/h, f is the coefficient of friction, and G is the grade of the road (rise/run). The authors suggest using a coefficient of friction of 0.25 to account for wet weather. According to the author, the formula assumes a value of 2.5 s for perception and brake reaction time. However, the CycleSafe system is intended to reduce perception time, and hence 2.5 s would be an overestimate of total perception and brake reaction time. It would be useful to determine exactly how 2.5 s is distributed between perception and brake reaction time, since the system will not reduce brake reaction time but will reduce perception time significantly. In [?], a study is performed on human reaction and danger perception time for adult and child cyclists. The perception time for different scenarios in the study varies, but the scenario with the lowest average perception time has 0.9 s. As such, we can consider that we can likely discount about 0.9 s from the required total braking time.

At the assumed maximum cycling speed of 9 m/s, or 32.4 km/h, we have $s_{\text{brake}} = 27.3\,\text{m}$ according to the formula. This is at least 3 s at the maximum speed. Discounting the perception time, we have a lead time of at least 2.1 s at maximum speed, or 19 m, which we will set as the minimum required lead time for frontal collision warnings.

Additionally, the sensor should be able to resolve objects as small as open vehicle doors, a common cause of accidents [?]. However, vehicle door size is not a good determining factor of the resolution since we want a lower bound, not an upper bound. We can consider that the cyclist will be travelling at least half a handlebar width away from parked vehicles, which would set the minimum size of a vehicle door obstruction at that length. Handlebar width is typically around 40 cm, so we can set a minimum resolution at 20 cm. There is still significant additional allowance in real scenarios since the cyclist will most likely not be cycling with the handlebar right up against parked vehicles, or anywhere close to it, especially if they are travelling at 9 m/s.

Testing of the frontal sensors, given the 2.1 s lead time and 20 cm resolution requirement, will follow the functional testing method described below. All tests will have cyclist speeds up to 9 m/s if possible (depending on the fitness of the cyclist).

1) *Static obstacle test* (parked vehicle simulation). 10 trials in a closed course will be carried out, with varying cyclist speeds up to 9 m. The obstacle will be a large square sheet of paper 50 cm in length, held at a height between 50 cm to 100 cm off the ground. The passing condition for these tests is that a warning is active starting between 2.1 s and 3 s before the cyclist contacts the sheet of paper. The cyclist will attempt to keep a constant speed, and if the cyclists' speed fluctuates more than 1 m/s within the alert duration the test will be voided.
2) *Sudden obstacle test* (vehicle door simulation). 10 trials in a closed course will be carried out. The obstacle will be a piece of cardboard 20 cm wide and 50 cm tall, held at a height between 40 cm to 60 cm off the ground, and rotated into the cyclist's path when the cyclist is not more than 19 m away. The passing condition for these tests is that a warning is active within 0.9 s of the obstacle being placed in the way of the cyclist. The cyclist will attempt to keep a constant speed before the alert is active, and if the cyclists' speed fluctuates more than 1 m/s during that time the test will be voided. The cyclist may slow down or swerve once the alert is active.
3) *Moving obstacle test* (false positive test). 10 trials will be carried out with the cyclist pedalling on a trainer, to simulate the speed to the system. The obstacle will be a large square sheet of paper 50 cm in length, held at a height between 50 cm to 100 cm off the ground, moving towards the (now stationary) cyclist at speeds

up to 5 m/s. Again, the passing condition for these tests is that a warning is active starting between 2.1 s and 3 s before the cyclist contacts the sheet of paper. The cyclist will attempt to keep a constant speed, and if the cyclists' speed fluctuates more than 1 m/s within the alert duration the test will be voided. In particular, this test determines if the system alerts the cyclist unnecessarily when the obstacle is not moving toward the cyclist, so the upper bound of 3 s is important.

2) Blind spot warning: We determine the requirements for blind spot warnings assuming typical values. Firstly, lane width s_{lane} is approximately 3 m [**?**]. We consider that larger lane width will make cycling safer since vehicles will not pass as close to the cyclist when overtaking. We also set a minimum cycling speed threshold at which the lane-change warning system provides guaranteed reaction time, which we set at 5 m/s. We also need to determine a maximum vehicle speed at which the warning provides guaranteed reaction time, which we set at 20 m/s (approx. 45 mph, the typical maximum speed limit on non-freeway urban roads). The assumption of vehicles travelling at the speed limit is likely unrealistic, since vehicles often exceed the posted speed limit. However, in the absence of the speed limit it is difficult to place an upper bound on the maximum speed of a vehicle. While the calculations are based on the speed limit and thus CycleSafe system guarantees are given as such, the system will still function, albeit with reduced effectiveness, without such assumptions. It is inevitable that cyclists occasionally come into contact with reckless drivers, and in such scenarios we rely on a best-effort warning, instead of attempting to provide a quantitative guarantee.

To provide a blind spot warning to a cyclist changing lanes, the system must determine the time at which an oncoming vehicle will cross paths with a cyclist. Taking the minimum cycling speed of 5 m/s and vehicle speed of at most 20 m/s, an oncoming vehicle will approach at at most 15 m/s. In this case, we wish to provide the cyclist with 2 s reaction time to cancel or delay a lane change if necessary, more than enough for the cyclist to change direction (which is much quicker than braking). Note that we provided 2.1 s for braking. This will set the requirement for the sensor's effective range at 30 m.

Testing for the blind spot warning will be similar to the testing for the frontal collision warning. All tests will have cyclist speeds up to 9 m/s if possible (depending on the fitness of the cyclist), and the cyclist's left turn signal will be active throughout. The tests are outlined below.

1) *Standard lane change.* 10 trials will be carried out with the cyclist pedalling on a trainer, to simulate the speed to the system. A vehicle will come up behind the cyclist offset to the left by a distance between 2 m and 3 m, at speeds up to 15 m/s relative to the (stationary) cyclist. The passing condition for this test is that a notification is activated starting between 2.5 s and 1.5 s before the vehicle passes the cyclist, and a warning is activated between 1.5 s and 0.5 s before the vehicle passes the cyclist. Both these conditions must be true to pass the test. The cyclist will attempt to keep a constant speed, and if the cyclists' speed fluctuates more than 1 m/s after the notification the test will be voided.
2) *Standard lane change* (false positive). 10 trials will be carried out with the cyclist pedalling on a trainer, to simulate the speed to the system. A vehicle will come up behind the cyclist offset to the left by a distance between 1.5 m and 2.5 m, at speeds up to 15 m/s relative to the (stationary) cyclist but will slow down, going no quicker than 7.5 m/s when 15 m away from the back of the cyclist, and will not overtake the cyclist or come any closer than 5 m away from the back of the cyclist. The passing condition for this test is that a notification is activated but a warning is not activated. Both these conditions must be true to pass the test. The cyclist will attempt to keep a constant speed, and if the cyclists' speed fluctuates more than 1 m/s after the notification the test will be voided.

3) Right-turn cut-off warning: For right-turn cut-off warnings when a vehicle is about to make a right turn into a side street in front of a cyclist, there is a large range of possible positions the vehicle may take in relation to the cyclist. However, the vehicle will always have to come into fairly close proximity to the cyclist, which we assume to be less than one lane width (s_{lane} = 3 m). Therefore, the system should be able to detect vehicles within 3 m of the cyclist in a 180° semi-circle on the left of the cyclist. This will also set the minimum distance at which proximity warnings to cars will activate.

This will be tested using the same method as the first standard lane change test, except that the turn signal will be off. The passing condition for this test is that the proximity signals are active when the vehicle is passing such that the shortest distance to the cyclist is less than 3 m. There is no upper bound distance or time for this test since active proximity sensors do not harm the cyclist's trust in the system. 10 trials will be carried out for this test.

The testing for when vehicles cut into the cyclist's path ahead of the cyclist to turn right is subsumed under the car door simulation test. In case a car is about to directly impact the cyclist when turning right, no warning is given to the cyclist because there is no safer action the cyclist can take. Slowing down is not a safe action since it may cause problems for a driver who thinks the cyclist will be fast enough to clear a side street before the driver turns. Instead, the system protects against such occurrences using the proximity sensors and increasing the cyclist's awareness of the presence of a side street.

III. Architecture

CycleSafe will consist of 3 main components,

1) Mobile app that collects bike and location data
2) Main system mounted on bike that uses distance sensors and phone data to determine safety
3) Jacket wearable with warning feedback and traffic lights

The block diagram can be found in **??**. In this section, we will describe what smaller devices exist on each level, and how each architectural component connects with one another.

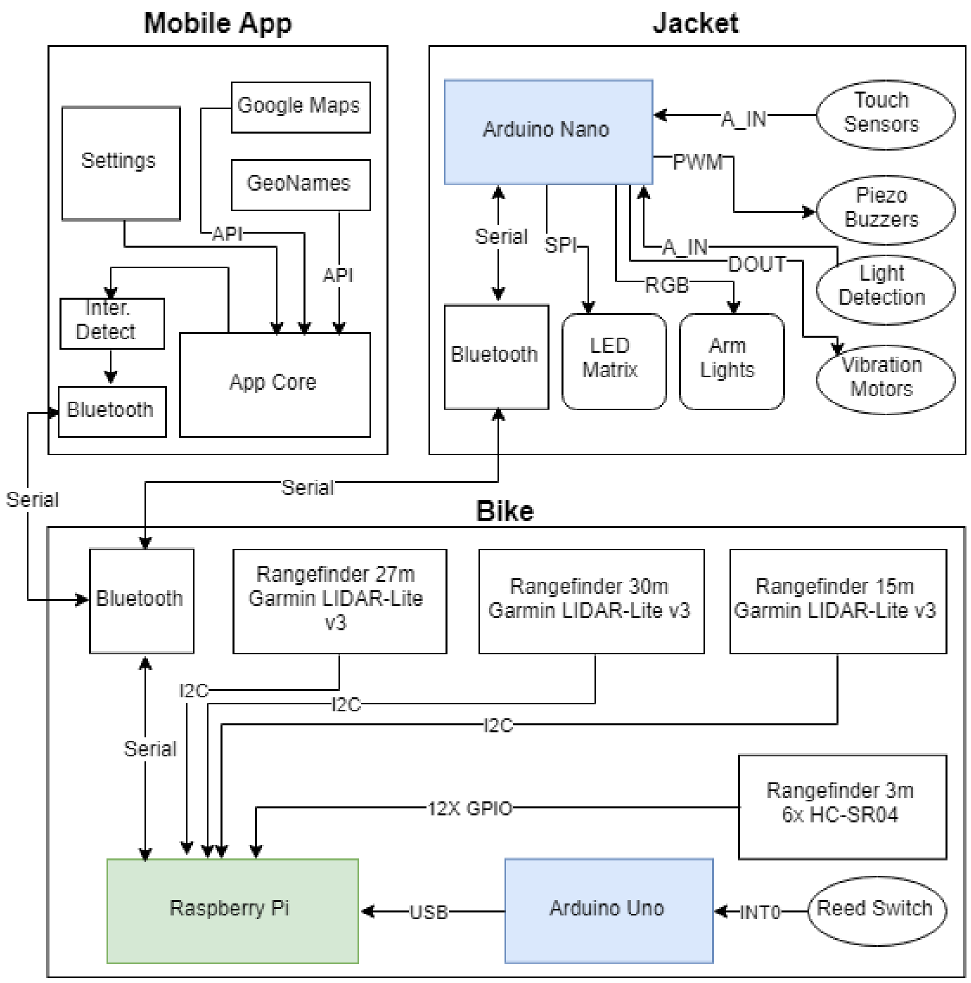

Fig. 1. Block Diagram Showing the Architecture of CycleSafe

A. *Mobile App*

The mobile app is an Android application that is responsible for giving the Raspberry Pi map and location data, as well as providing a user interface for the user to access many features of CycleSafe. The mobile app communicates with the Raspberry Pi via Bluetooth. In order for the app to give commands to the jacket, the app will have to issue a special command to the Raspberry Pi that then forwards the command to the jacket.

B. *Raspberry Pi*

The bicycle will have a mounted Raspberry Pi 3 (Pi) which will be the main processing hub and run the notification and alert system. The devices connected to the Pi are as follows:

- 3x Garmin LIDAR-Lite v3 (via I2C)
- 6x Ultrasonic Sensor HC-SR04 (via GPIO)
- Speedometer (via USB)
- Android Phone (via Bluetooth)
- Arduino Nano (via Bluetooth)

C. *Arduino Nano*

The Arduino Nano is the microcontroller that controls all features of the jacket, which has the following components:

- 16x16 LED Matrix (output via SPI)
- 2x Arm LED strip warning lights (transistor control)
- 2x Piezo Buzzer (analog output)
- 4x Cell Phone Vibration Motors (transistor control with PWM power)
- 2x Turn signal input from the gloves (digital input)

The Arduino has Bluetooth connection with the Raspberry Pi, where it will receive commands to give user warning feedback.

IV. Design Trade Studies

A. *Vehicle Detection*

The predominant monetary cost of the CycleSafe system arises from the required rangefinders. Because of the difference in speed of motor vehicles and cyclists, sensors typical for cars cannot be used for cyclists since typical lane change sensors only check the car's blind spot, which is very close to the car, and not the region visible in the car's side mirrors. Additionally, the cost of a car makes the cost of a its safety system insignificant, but this is not the case for a bicycle since bicycles are significantly cheaper than cars. The weight requirement for a safety system is similarly far more constrained on a bicycle than on a car.

A computer-vision based vehicle and obstacle detection system was considered as a potential solution. However, we noted that the power and processing requirements of such a system and the real-time nature of the safety system would make such a computer-vision based vehicle detection system impractical. Furthermore, CycleSafe requires more than to determine the location of nearby vehicles, but also to determine their speed, direction of motion, and acceleration, all of which would require processing of multiple frames at a time, increasing the processing requirements significantly. Nonetheless, a computer-vision based system would be superior in one particular situation: when detecting if vehicles are exiting side streets into the cyclist's path. This scenario is essentially impossible to address without a large number of sensors or wide-angle sensors with good resolution. However, we decided that it was worth sacrificing this scenario due to the impracticality of using such sensors or computer vision, and instead address it through signalling.

Based on these concerns, we determined that the CycleSafe system would utilize multiple single-point rangefinders that would require much less processing power to deal with. As further considerations, the CycleSafe system has to utilize longer-range sensors than in cars, and have a cost at most on the order of the cost of a bicycle. The large speed differentials between bicycles and vehicles also require higher data rates on sensors. Fortunately, there is a medium-range rangefinder commercially available. The Garmin LIDAR-Lite v3 has a 40 m range and a measurement rate of 500 Hz, while weighing only 38 g. It is designed for use in drones and similar real-time applications, making it well suited to CycleSafe. However, it costs \$130, which limits the quantity of these sensors the system will reasonably utilize. In designing the system, we took into account that only two or three of these sensors could be used in the entire system and that we would have to put them in strategic locations.

As a result, the shorter range sensors do not use the Garmin LIDAR-Lite v3, but instead use the HC-SR04, a short range ultrasound rangefinder, with an effective range up to 4 m. These sensors cost less than \$4, so despite their slow response time, which can be as low as 20 Hz and poor resolution, the CycleSafe system can utilize multiple such sensors to make up for the slow response time. In our system, we utilize 6 such sensors in order to run the proximity detection and signalling system, which only requires a range of 3 m, and alternately take measurements from each sensor in order to achieve the desired update time.

B. Component Locations

Since the CycleSafe system involves a cyclist, a bicycle and a mobile app connection, there were many decisions to make with regard to the positioning of the various components.

The rangefinder sensors could have been located on either the cyclist and the bicycle, or both. However, we decided that the potential benefits of having the sensors at a higher vantage point were outweighed by the relatively static bicycle frame as a mount, since a cyclist does not remain still when cycling.

The measurement of the cyclist's speed could have been done using either a physical measurement on the bicycle wheel or using GPS. However, due to the latency of GPS and mobile transmissions in our experience and the necessity of real-time data, it was determined that the speedometer should be physical rather than a GPS measurement.

The use of the jacket as a signalling platform rather than the bicycle was made considering the surface area and height. Since signal clarity is one of the system's primary objectives, the greater surface area of the cyclist especially when viewed from the back makes directional signals significantly clearer, as compared to the very thin profile of the bicycle viewed from behind. The added complication of needing to connect the sensor system and the signalling system wirelessly was determined to be fairly insignificant since wired communication would still be needed even if the signalling system was on the bicycle. Since the communication requirement between the sensing and the feedback system uses very little bandwidth, using Bluetooth (used in communication systems) would not increase latency significantly.

Consideration was also made as to whether the feedback system should originate from a mobile device rather than the jacket. However, we concluded that since mobile devices are actually a common cause of bicycling accidents by diverting the cyclist's attention, it would be more effective to use a platform meant solely for the safety system will less potential to distract the cyclist.

C. Battery Life

Finally, in order to power all the components of CycleSafe, we needed a battery. Unfortunately, batteries can be expensive in many ways, but tend to rise in price with

- Smaller size
- Less weight
- Longer battery life

For CycleSafe, since we want to be the least intrusive as possible to the original bike weight, we do not want to make the system too heavy; same goes for the jacket. Fortunately, the bike mounted system is very light, so most batteries are sufficient for the bike. In particular, most commuter bicycles are around 30 lb [?], and we aimed to keep our system below 10% of the bicycle weight. Our entire bike mounted system without the battery is less than 1 lb, so 2 lb of battery weight gives us plenty of options.

As for the jacket, since the cyclist is wearing it, the weight has to be significantly less. We estimated that if the average commuter cyclist is 140 lb, we wanted the jacket to be less than 5% of the cyclists' weight, or 7 lb. Since the jacket and components weighs about 5.5 lb, we are limited to about 1.5 lb for the jacket battery.

In addition to the weight tradeoffs, we also wanted our battery life to be long enough so that the cyclist did not have to charge the CycleSafe system too often. We deemed based on current products on the market [?], [?] that 1 week was a reasonable amount of time to have to charge the devices.

V. System Description

A. Android Phone

The features of CycleSafe that are controlled through the mobile app include

- *Live navigation on the road*: the app uses the Google Maps API to provide directions to the biker
- *Custom jacket lights*: If the user wishes to use a different style of turn signals or brake lights, they can customize it through the app
- *Custom jacket notifications*: The user can choose how they want to be notified in the event of a danger. This includes a combination of sound, light, and vibration feedback. The user can in addition choose to customize the type of vibration they feel.

In addition, the phone will provide information about when the biker is approaching an intersection. This is crucial, since the warning system behaves differently when the user is approaching an intersection. The phone sends intersection to the Raspberry Pi via Bluetooth.

B. Raspberry Pi 3

The Pi will be running data collection and data processing, sending commands to the Arduino Nano on the jacket. The algorithm will repeatedly run the following:

1) Send trigger signals out to two of the HC-SR04. GPIO interrupts will handle the values returned.
2) Request distance readings from all three LIDARs via I2C. Request the longer required range sensors first.
3) Check velocity via serial.
4) Check Bluetooth for incoming side-street data from Android.
5) Check Bluetooth for user-activated signals from Arduino Nano.
6) Check I2C for received LIDAR range values.
7) Determine warnings to emit.
8) Send warnings to Arduino Nano via Bluetooth.

The primary reason the ordering is as such is to reduce the amount of time the Pi spends waiting. The ultrasonic sensors are the slowest sensors to respond to a measurement request, followed by the LIDAR. The Android App and the Arduino Nano do not require time to take measurements since the data received from them is not as time-sensitive and hence they can simply supply a recently-computed value.

1) HC-SR04: The HC-SR04 require a trigger signal to be sent, and respond with data encoding using the length of a pulse. As such, the Pi is required to monitor the HC-SR04 echo pin when it expects a measurement to arrive. Signal-edge triggered interrupts will allow the Pi to obtain a timestamp when the pulse begins and a timestamp when the pulse ends, while still performing other useful work.

The processing (subtracting the time and scaling) can then be done during the warning decision stage. This keeps the interrupts short and ensures they do not interfere with each others' timing. This is also the reason for only querying two of the HC-SR04 in one cycle of the algorithm, since having all six HC-SR04 using two interrupts each could potentially interfere with their timing data.

2) Garmin LIDAR-Lite v3: The Garmin LIDAR-Lite v3 communicates with the Pi via I2C. All three rangefinders can use the same SDA/SCL lines, since they can be configured to use different I2C addresses. As it takes some time for the LIDAR to make a measurement, the Pi initiates the measurement at the beginning of the processing loop, and retrieves the measurement only right before it begins computation.

3) Warning decision algorithm: The algorithm used by the Pi to determine what kind of warning to emit to the cyclist is described here. The severe warning is given either when the cyclist is about to collide, or something is about to collide with the cyclist. As such, severe warning will be given under the following conditions:

1) The frontal sensor detects an object at a distance covered within 3 s at the cyclist's current speed. This warning tells the cyclist to stop.
2) The medium-range (15 m) blind-spot sensor detects a vehicle that is not decelerating and will collide with the cyclist within 1 s. The warning tells the cyclist to hold their lane.

The severe warnings are meant to be followed without the cyclist needing to consider other inputs. As such, they need to be extremely trustworthy when they are activated. In addition to false negatives, False positives must also be kept to a minimum.

The notifications are given to the cyclist under the following conditions:

1) Blind spot warning: when long range LIDAR detects a vehicle approaching, reaching the cyclist in in 2 s or less.
2) When an intersection is approaching.
3) When vehicles are in close proximity and may make a right turn.

C. Rangefinders

1) Frontal sensor: From Section **??**, we determined that the frontal sensor range requirement is at least 19 m. The sensor used is a Garmin LIDAR-Lite v3, which has a maximum effective range of 40 m. This will be placed on a frontal mounting point on the bicycle pointing directly forward, providing unobstructed line-of-sight in front of the bicycle. It is connected directly to the Pi 3 via the 5V line and I2C bus, which will run along the length of the bicycle.

2) Blind spot warning during lane change: From Section **??**, we determined that the effective range of the blind spot sensor must be at least 30 m. However, the blind spot sensor must face slightly diagonally in order to detect vehicles in an adjacent lane.

Given the lane width of $s_{\text{lane}} = 3\,\text{m}$ and the sensor range of $s_{\text{bs\_max}} = 30\,\text{m}$, we can determine the angle at which the sensor must be offset to detect a vehicle in the adjacent lane. Assuming the displacement from the cyclist to the closest point on the vehicle in the direction perpendicular to the lane direction is about one lane width, and the width of the vehicle w_v is about 1.8 m, we find the offset of the sensor, θ_{bs} from the axis of the direction of travel:

$$\theta_{\text{bs}} = \arctan\frac{s_{\text{lane}} + w_{\text{v}}}{s_{\text{bs\_max}}} = \arctan\frac{4.8}{30} = 9.09^\circ \qquad (3)$$

Additionally we want to provide a severe warning to cyclists if it is likely that a vehicle will collide or be forced to swerve if the cyclist performs a lane change immediately. With the

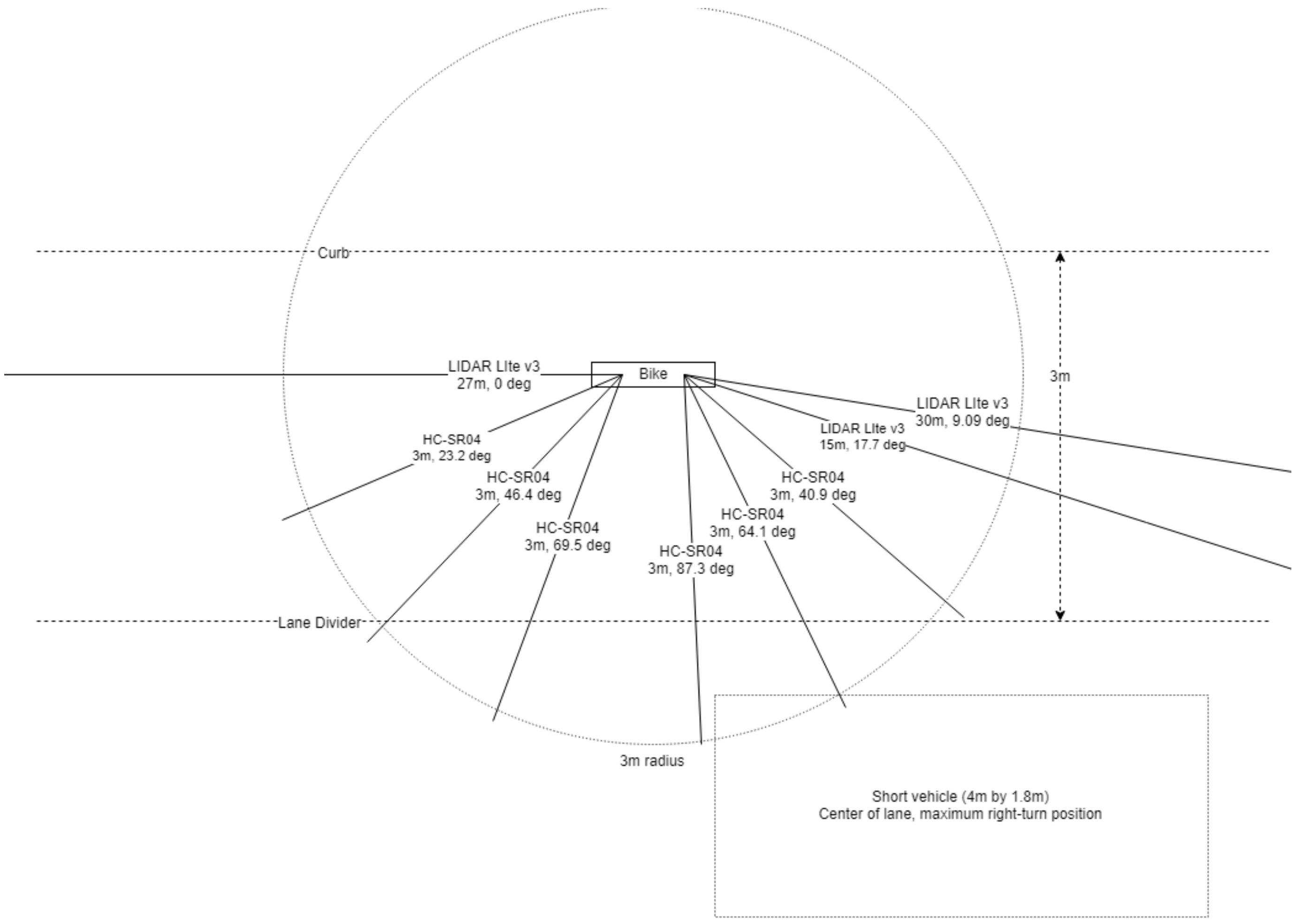

Fig. 2. Rangefinder Configuration with Bicycle Showing Ranges Covered

30 m blind spot sensor, it is well within reason to expect cars to slow down between the time they are detected 30 m away and when they overtake the cyclist. As such, the feedback given to the cyclist when the vehicle is 2 s away will only be a notification. However, in some cases the vehicle will not slow down, and possibly not notice the cyclist.

To address this, a second sensor is utilized to provide a severe warning to the cyclist when the vehicle is 1 s away. Based on the earlier assumption that an oncoming vehicle will approach at at most 15 m/s, the sensor is required to have an effective range of $s_{\text{bs\_near}} = 15\,\text{m}$. Using the same method:

$$\theta_{\text{bs\_near}} = \arctan \frac{s_{\text{lane}} + w_v}{s_{\text{bs\_near}}} = \arctan \frac{4.8}{15} = 17.74^\circ \qquad (4)$$

3) Right turn cut off warning: When a vehicle is about to make a right turn into a side street in front of a cyclist, there is a large range of possible positions the vehicle may take in relation to the cyclist. However, the vehicle will always have to come into fairly close proximity to the cyclist (less than one lane width). The CycleSafe system attempts to detect this by using an array of short range sensors. These short range sensors are positioned such that any one of them will detect a compact car within one lane width to the back-left, left and front-left of the cyclist (see Figure **??**). The coverage should start just outside the detection area of the 15 m blind spot sensor, and extend all the way to the detection area of the frontal sensor. Generally, compact cars are at least $l_{car} = 4\,\text{m}$ in length. At a distance of $s_{\text{lane}} = 3\,\text{m}$, the angle $d\theta_{close}$ subtended by such a car is:

$$\text{d}\theta_{\text{close}} = 2 \arctan \frac{l_{\text{car}}}{2 s_{\text{lane}}} = 2 \arctan \frac{4}{6} = 67.38^\circ \qquad (5)$$

Hence, to ensure that any cars passing within the minimum distance of 3 m is detected, we require a total of:

$$\left\lceil \frac{180^\circ - 17.74^\circ}{67.38^\circ} \right\rceil - 1 = 2 \qquad (6)$$

close proximity sensors.

However, as stated in Section **??** we are using three sets of sensors, enabling staggered measurements in order to offset the slow measurement response time. This results in a total of six sensors in three sets of two each, and facing angles approximately 23° apart.

4) Overall configuration: The overall configuration of the sensors is shown in the diagram below (Figure **??**). The frontal collision sensors and blind spot sensors are the Garmin LIDAR-Lite v3. The six proximity sensors are the HC-SR04 Ultrasonic Rangefinder.

5) Other considerations: It is worth noting that many laser-based range sensors typically perform more poorly in high sunlight conditions. However, this is partially compensated by the improvement in visibility and general reduction of braking distance in pleasant weather conditions.

D. Arduino Nano

The Arduino Nano is the microcontroller that is responsible for controlling all the peripherals on the jacket. More details about each individual component can be found in the following sections. Since the jacket does not send any data to the Raspberry Pi, it is a passive component in the architecture, and thus the Nano only receives commands via Bluetooth from the Raspberry Pi.

E. LED Matrix

The LED Matrix serves as the main light signalling component of the jacket wearable. Similar to how cars have turn signals and brake lights, the LED Matrix in the back can serve any of these light functions. Because the matrix is configurable to its 16×16 granularity and full RGB, it can emulate almost any signal.

For our project, the LED Matrix has the following lighting modes,

- Ambient lights: Matrix is a dim red. Especially important during nighttime, to give other vehicles on the road awareness of the rider.
- Brake lights: Matrix is a brighter red than the ambient light. This mode is triggered if the speed of the rider is slowing down.
- Turn Signals: Flashing arrows that indicate which direction the biker is trying to turn into

In order to display images on the LED Matrix, 2D data has to first encoded into 1D linear data, and then sent to the Arduino. because the Arduino has limited memory, we unfortunately cannot store all the shapes we need, because it is too expensive.

F. Piezo Buzzers and Vibration Motors

One of the goals of this project is to give the rider as much awareness about the road as possible, while also allowing the rider to focus on the road by keeping CycleSafe minimally intrusive. The Piezo Buzzers and Vibration Motors are forms of non-visual feedback to the rider, namely auditory and sensory, that use beeping sounds and vibrations to give the rider warning about dangerous situations.

This is akin to how many blind spot features in cars nowadays beep when a driver is about to turn into a lane that is not empty. This type of feedback is very important, because people can forget to check that a lane is empty before changing. In addition, for a biker, who doesn't have rear or side mirrors, they have to expend a larger motion just to check if a lane is clear. Therefore, these two forms of feedback minimize the need for the rider to check for visual warnings, and also serves as a fallback warning system in case the rider just forgets to check in the first place.

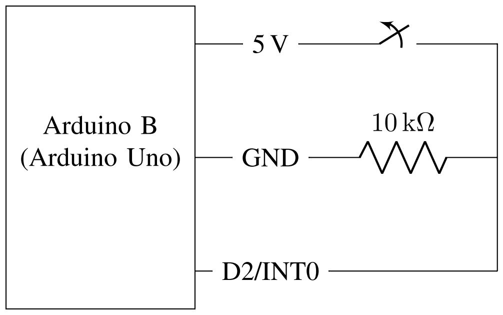

Fig. 3. Speedometer circuit configuration.

G. Speedometer

The function of the speedometer is to determine the speed of the bicycle. It consists of a single reed switch, placed on the bicycle's front fork, activated by a magnet, and connected to the interrupt pin on Arduino B. The circuit is shown in Figure **??**.

The speedometer will require the user to configure the wheel diameter or the wheel circumference using the app interface. The bicycle velocity, v_b will then be computed by Arduino B, as follows:

$$v_b = \frac{4\pi d}{t - t_4} \tag{7}$$

where t is the current timestamp, t_4 is the timestamp four interrupts prior, and d is the diameter of the wheel. Wheel diameters typically range from 507 mm to 622 mm, with 622 mm being the most common.

Waiting for four interrupts computes the average speed over the last four wheel rotations instead of merely the last rotation. This reduces speed fluctuations, which would adversely affect the accuracy of the warning algorithm. It also reduces the error in case an interrupt is missed or an additional interrupt somehow occurs. in this case, the average of four limits the error to 25% of the actual value. An average of four was used because four rotations of a 622 mm wheel on a bicycle covers a distance of 7.816 m, which is approximately one second at typical bicycle speeds of 5 m/s to 9 m/s. However, the trade-off is that the it may take up to a second for the speedometer to respond to changes in velocity. Large changes in speed are only likely when the bicycle is coming to a sudden stop, and therefore a slower response to a sudden decrease in speed will favor false positives over false negatives, which is preferable.

VI. Project Management

A. Schedule

We are using TeamGantt to organize our tasks and keep track of our progress. We have attached our Gantt Chart schedule at the end of the paper.

We are mostly on track with our project. We are anticipating some latency in integrating all aspects of the project together—once we physically construct our bicycle add-on and jacket. Fortunately, we have latency built in, so we should be able to overcome any difficulties we have in assembling our project. We plan to test in April, and give us

A link to the chart can be found here.

B. Team Member Responsibilities

Ben

- Raspberry Pi control software
- Bike distance sensor setup
- Bike speedometer
- Formal Specifications

Siddhanth

- Develop the mobile app, including Bluetooth communication, map API development, UI
- Write intersection detection algorithms
- Collect bike speed, location, map data to send as inputs to the safety logic in the Raspberry Pi

Michael

- Design haptic feedback on the jacket
- Integrate jacket peripherals with the Arduino Nano controller
- Design and create jacket lights to satisfy official vehicle requirements
- Build communication between the jacket and the Raspberry Pi controller

C. Budget

Item	Price
Garmin LIDAR-Lite v3-HP (1)	$149.99 each
Garmin LIDAR-Lite v3 (2)	$129.99 each
HC-SR04 Ultrasonic Sensor (6)	$6.99 each
EK1621x2 Reed Switch	$0.50
Rainproof Jacket	$29.99
WS2812B LED Strip	$28.95 each
Piezo Buzzer PS1240 (2)	$1.50
Cellphone Vibration Motor (4)	$1.95
Raspberry Pi Model 3	$29.99
Arduino Nano	$4.95
16750mAh Portable Charger	$34.99
Total	**$576.88**

TABLE I
PRICES FOR MATERIALS

D. Risk Management

Having parts come in late was expected, but we fortunately had many of the major parts already in hand, we could develop without some of the parts coming in. For example, we already had a Raspberry Pi, Arduino Nano, and a mobile phone at the start of the project, so we could develop on those devices while waiting for our other parts to come in.

One of the major concerns we had for our entire system was power, since we eventually wanted to have a portable power source and we had some potentially power-consuming devices, including LED strips and LIDAR sensors. To reduce risk, we did two things

1) Reduce the power consumption of these devices. For the LED strips, we made sure to only activate a subset of the 256 available lights. For the LIDAR, we only requested for distance data when we needed it.
2) Put more batteries on the bike, and plug in the jacket into the bike as a backup option for power.

VII. Related Work

There are not many projects that take a serious attempt to build a full-scale bicycle safety system. Some related work include a bicycle blind spot detection system [?], which consisted of an ultrasonic sensor on the back of the helmet and LED notification about nearby cars. Another project called Sixth Sense [?] is a wireless distance sensor that vibrates a smart device when objects, such as cars, are nearby. These projects only have very limited functionality with car detection, especially when cars are approaching at faster speeds, since the range of their distance sensors is small, and it does not seem like they have acceleration detection.

Most of the safety features of CycleSafe were inspired by current car safety features. For example Honda Sending [?] now includes the following safety features

- Collision Mitigation Braking System
- Road Departure Mitigation
- Blind Spot Information System

CycleSafe focuses on implementing collision avoidance and blind spot information.

VIII. Summary

CycleSafe provides a comprehensive safety system that makes biking safer than current biking practices. By giving the rider warnings well ahead of collisions with cars and objects, CycleSafe can prevent many of the accidents that lead to many cyclist deaths every year.

A. Future Work

Since cars have safety features, and now CycleSafe introduces safety to cyclists, a natural extension of the project is to bring safety features to motorcyclists. For a higher speed vehicle like a motorcycle, we expect to need sensors that have further ranges so we can detect oncoming traffic and objects from longer distances.

B. Lessons Learned

The biggest problem we have encountered so far is that the parts we order do not come in on time, even if they are Amazon Prime. So there should be a good amount of planning ahead to make sure that parts coming does not affect the progress of the project.

Acknowledgment

The CycleSafe team would like to thank the Electrical and Computer Engineering department of Carnegie Mellon University for their financial support in making this project possible. In addition, we would like to thank Professor Ken Mai for his advising throughout the project and Zilei Gu for her constant support and check-ins with our project.