Smartphone Application to Detect Texting While Driving

Detecting Texting while Driving

Project Plan

Group:
sdmay19-16
"The Face Mates"

Client:
Daji Qiao

Faculty Advisor:
Daji Qiao

Team Members:
Ryan Baker - Lead Architect
Derek Clayton - Report Manager
Lucas Golinghorst - Test Engineer
Andrew Knaack - Lead Designer
Sara Mace - Meeting Scribe
Kristina Robinson - Project Lead

Team Website:
sdmay19-16.sd.ece.iastate.edu

Version:
2.0

Version Date:
9/28/2018

TABLE OF CONTENTS

List of Figures

List of Tables

List of Symbols

List of Definitions

- False positive: In this case, a false positive is our application incorrectly identifying a passenger as a driver and blocking their texting capabilities.
- False negative: In this case, a false positive is our application incorrectly identifying a driver as a passenger and not blocking their texting capabilities.
- TwD: An acronym which stands for Texting while Driving

1 Introductory Material

1.1 Acknowledgement

We want to acknowledge our client and advisor, Daji Qiao, for thinking of this project. We also want to acknowledge the department of Electrical and Computer Engineering at Iowa State University for encouraging us to work on professional projects and providing us with extra phones for development purposes.

1.2 Problem Statement

Our project will address the issue of texting while driving. According to teensafe.com, just 3 years ago, 391,000 injuries were caused by distracted driving accidents. Even more serious than that, 9 people are killed everyday from distracted driving accidents [1]. Answering texts while driving might seem harmless, but replying to a text message takes the driver's eyes off of the road for at least a few seconds. It takes only three seconds of the driver having their eyes off of the road to be in a car crash. While texting and driving is not the only cause of distracted driving, approximately 660,000 drivers use a cell phone during the daytime [1]. That is a significant amount of people texting while on the road which will make it more likely for them to get in a car crash. In order to find a solution to this problem, we will build an android application to detect if someone is texting while driving. Accurately detecting whether someone is in the driver's seat and that they are texting while driving is our biggest challenge. In order to do that without simply locking out everyone from their phones, our solution will have to include many different measures to ensure accurate detection.

The solution to this problem will be to develop an android application to detect whether someone is TwD. Once TwD has been detected, the user will not be able to send text messages until they stop driving. We will ensure that we can accurately detect TwD by checking multiple different categories. The first thing we will check is if the driving is moving faster than 10 miles per hour. This will tell us if they are in a moving vehicle, regardless of direction (including reverse). The second category we will look at is determining where the person is sitting in the car. We will use the phone camera to determining what is in front of the person as well as seeing which way the seatbelt is going across the users chest. Our last category of verification is learning how the user normally uses their phone so we will be able to find differences when they are TwD.

To keep the scope manageable, we will only focus on interaction with the default Android Messages app. Once this has been accomplished, we will explore the solution's compatibility with any arbitrary texting application.

1.3 Operating Environment

The operating environment will be the user's vehicle. We expect that this environment will not be subject to any extreme conditions, but rather a controlled environment. If an extreme condition does occur (such as a car accident), then the app is no longer of use in that environment anyway. It is also worth noting that the application is intended to be active even when the user is not in a vehicle, since it is vital for the program to understand a user's regular texting habits when not driving. Our application will be downloaded onto the user's phone, so withstanding hazards in the environment it is used in is not much of a concern.

1.4 Intended Users and Intended Uses

Our intended users are anyone who drives a vehicle and has an Android smartphone. As our app is meant to detect texting while driving, if someone is not a driver then it doesn't make sense for them to have an app that prevents them from texting and driving. Our user also needs to have an android smartphone as the app we are making is an android app, so therefore they need an android phone to run the app.

The intended use of our app is for anyone who wants to make sure that they cannot text and drive. For example, this could be a parent who has their child install it because they want to make sure the child stays safe; or, someone who acknowledges that they have a problem with texting and driving and want to take measures against their habit.

1.5 Assumptions and Limitations

Our list of assumptions is as follows:

1. The phone used during driving is being consistently used by the driver. They will not use someone else's phone while in the driver's seat. This assumption needs to be made since we will use machine learning to track the phone owner's tendencies. If the driver uses someone else's phone then machine learning will not help us with detection.

2. The user has the app enabled consistently, even when not driving. This will allow us to learn the owner of the cell phone's texting tendencies when they are not driving so we can more accurately detect texting while driving.
3. The app should be relatively simple to activate. Therefore, the user does not have to go through many steps themselves to make it functional. The user will be unwilling to use the application if it is too complicated to activate.
4. The driver, if texting, will only ever be texting with one hand. This is the assumed normal behavior. Detecting texting while driving will is simplified by this assumption if we can figure out a way to determine how many hands they are using to type on their phone.
5. Our application will only be used by drivers in Iowa. Only letting our application be used in Iowa will simplify the data privacy laws that we need to follow.
6. The user will not be riding a bike and texting. Although it might be possible to text while riding a bike, to simplify our solution we will assume that the user is not riding a bike.
7. This application will not cover the various messaging apps available. There are far too many to account for, but this could be a next step to our project if time allows.

The project limitation are as follows:

1. The application must not take up unreasonable amounts of memory or battery life to run. This could cause problems or irritation for the user if violated.
2. The application will support English and not any additional languages or translations.
3. The application will not make use of any external hardware such as external cameras. Only sensors existing on the phone will be available to use.

1.6 Expected End Product and Other Deliverables

The end product is a standalone Android application which detects texting while driving. This will be done with machine learning techniques and readings from the phone sensors. This application will not require any additional hardware to be run.

Since this project is split across two semesters, there will be a preliminary end product by the end of the first semester. This will be a working prototype with all intended functions of the final product in use while the application is running, but not necessarily perfected or well-developed by that point.

2 - Proposed Approach and Statement of Work

2.1 Objective of the Task

The objective of our project is to make a reliable mobile application for detecting and preventing texting while driving. To accomplish this, we will create an application for Android phones which will use built-in sensors and learned user texting habits to reliably know when to lock texting functionality on the phone. The application will be a standalone application, thus not requiring any additional hardware. The end product will have a low false positive and false negative rate so drivers cannot accidentally be allowed texting anyway, and passengers will still have full functionality of their phones.

The primary goal of this solution is to detect when a driver is texting and driving. The secondary goal is to implement a lock out process that is triggered when texting and driving is detected. This in turn will create a safer driving experience for the user and the other drivers on the road.

2.2 Functional Requirements

1. The application prevents drivers from sending texts if certain dangerous conditions are met. This is the most important functionality and the main purpose of the project.
2. The false positive rate and false negative rate should not be greater than 10%. It would be very easy to block 100% of drivers, but that would also block passengers as well. We want to be able to block as close to 100% of drivers while blocking as few passengers as possible.
3. The application should run on Android OS 6.0 and newer. We want to access as many phones as possible without being limited to an unnecessarily basic version of Android. We would like our application to be available to as many as possible, and with Android, older versions of software are actually very common. That is why we would like to support the oldest software possible.
4. The app does not use an internet connection (it is a standalone app). This is necessary if service is poor and to avoid breaking any applicable data handling laws with outside data storage.
5. Optional: Stop the phone from viewing texts. This would of course allow the texts to be received, but would not allow the user to access them.

6. Optional: Response time is fast enough to block the reading of any incoming texts. We want the program to be fast enough to block these incoming texts. If it is not fast enough to do this, there is not much point in having it.

2.3 Constraint Considerations

Our non-functional requirements are as follows:

1. The app itself will be easy to set up and run. We want the app to be as easy as possible for the user to set up. This does not count hardware, as hardware will probably be a tedious setup and can't be helped.
2. The app must be reliable at least 95% of the time, meaning that it runs consistently without crashing. This is important since the app is active most hours of the day to monitor user habits; also, obviously it would defeat the app's purpose entirely if it crashed while the user was driving.
3. The app must comply with applicable sensitive data laws (presumably resolved by making it a standalone app). Personal information is potentially at risk, so it is important that our app does not break the law when dealing with that sensitive data.

The constraints are as follows:

1. The application must be compatible with Android 6.0 and onwards. This will allow more phones to be able to use our application without compromising functionality.
2. We can't (or at least shouldn't) simulate actual texting while driving because it is dangerous behavior, so there may be some variation in the data between testing and when trying to approximate when someone is *actually* texting while driving.
3. Testing the app most accurately requires at least two people with an actual car driving around. This is primarily a time constraint, since this type of testing is more time-consuming than testing that can be done from a computer, and both participants must be available.
4. Our team lacks machine-learning experience, which is vital to the project. Part of the detection process will be the ability to recognize a difference in the normal texting pattern of the driver so we will need to learn about machine learning so we are able to incorporate that into our project.
5. The project is intended for research, not marketability; therefore, a fully functioning product may not result.

2.4 Previous Work And Literature

There are several existing applications and research projects on this subject. Not all are fully developed to the extent of being able to stop texting while driving but only detecting the likelihood of a user being the driver. Their areas of research include the driver voluntarily locking him/herself out of the phone, monitoring changes in typing habits, using external peripherals (outside the scope of this project), and using a phone's built-in sensors to detect the specific location of someone in a car.

Another application that exists on the market currently is called Safe Drive Zone. This application works by having two modes. According to safedrivezone.com, the first mode is the parent mode and the second mode is a driver mode [5]. This driver mode has limitations that are set by the parent mode. Some such limitations include redirecting calls to a pre set up text letting the caller know that the person is driving. One advantage is that it has multiple modes and can communicate directly with another person through the parent and driver app. On the market, some insurance companies have apps that measure your driving skills and if you have good skills it can give you discounts on your car insurance.

One such app is Allstate's DriveWise app. According to allstate.com, the app tracks your speed and how hard you brake, and a customer receives discounts based on not driving at recklace speeds and not braking hard [4]. While these apps don't specifically monitor texting and driving, they do track speed which is something we are interested in monitoring. One advantage of app's such as these is that they measure based off of different sensors the phone has, so we know that these measurements can be done. One disadvantage is that it doesn't measure texting while driving, so it does not solve what we are trying to solve.

2.5 Proposed Design

There following are six technologies which make up this project.

Speedometer: The first and most fundamental is the speedometer, a sensor Android phones have built-in. This will determine if the system is trying to detect texting while driving at all. If the speedometer is below a safe speed, the system will switch to its learning mode. If it is above safe speed, the system will switch into detection mode and begin calculating the probability of texting while driving.

Texting Speed: When not in detection mode, key input will be captured while the texting application is in use so that the system can learn the user's typical texting speed. During detection mode, the system will take this norm and compare it to how fast the user is currently typing. If the texting speed is significantly slower, it implies the user may be distracted (possibly driving).

Spell-Checking: When not in detection mode, the user's captured key inputs will be analyzed for spelling errors so that the system learns how many errors are in the average sentence the user types. During detection mode, the system will take this norm and compare to the number of errors in what the user is currently typing. If the ratio of misspelled words is significantly higher, it implies the user may be distracted (possibly driving).

Centripetal Acceleration: Using GPS functionality, it can be determined what side of the vehicle the user is in when it makes turns. If the GPS determines the user is one the right side of the vehicle, then the user cannot be in the driver's seat and detection mode will be disabled. If the car encounters a bump in the road, GPS functionality can also determine if the user is in the front or back of the vehicle. If GPS determines the user is in the back, then detection mode will also be disabled. If the user is determined to likely be the driver, spelling and texting speed indications are much more likely to disable the phone's texting functions.

Phone Camera: While in detection mode, the front and back cameras of the phone can be used by the system to try to figure out where the phone is in the car. If the system can identify key features of the car like the wheel or the seatbelt, it can provide information about where the user is likely sitting (i.e. the direction of the seatbelt across the user's body can tell which side of the car the user is in).

Phone Handling: While in learning mode, the system will try to become accustomed to the user's phone-handling habit, such as the user's dominant hand, how often they use the phone with one hand, etc. In detection mode, this information will be used to detect if the user is handling the phone significantly differently than normal. For example, if the user almost always texts with two hands, but is trying to text with only one hand while the system is already in detection mode, it is more likely that the other hand is being occupied steering the wheel (indicating texting while driving).

A combination of these technologies should be able to work together to consistently learn when users are texting while driving and when they are not. Table a. shown below is the dependency diagram for these technologies. The speedometer comes first because it determines the application's state. On the left side of the speedometer are the

technologies which determine user position in the vehicle; on the right are the technologies which analyze user habits. All of the latter have distinct learning and detection modes. At least one technology from both sides is necessary to have the minimal functionality of detecting TwD, leading to a phone lock.

“Phone lock” means that the user will be blocked from using texting applications, possibly disabling the keyboard altogether. The user should ideally not be blocked from making phone calls (in case of emergency), and blocking the use of other distracting applications are beyond the scope of this project. Once triggered, the phone lock will remain active until the user has dropped back down to acceptable speeds (below 10 mph) for at least five minutes. This will prevent texting from being enabled again every time the driver slows down at a stop sign or a stoplight. If the driver only texts while stopped at these places, the phone lock will not activate, but if they continue to text as they speed up from these points the lock will work normally.

The greatest advantage of this setup is its modularity. Since multiple parts serve the same end, these will cover each other’s weaknesses and inaccuracies. Also, parts that are later realized to be infeasible can be stripped from the design without completely condemning the design. The only serious problem for the system comes when multiple parts fail.

Some disadvantages of the design flow from the same place. Because there are many parts to this project, we have to use our time efficiently or else not all modules will be fully implemented before the final deadline. Also, with multiple modules relying on specific hardware components on the phone, the problem arises where tests can be rendered useless in the event that those components are damaged. For instance, the camera test portion of our solution relies on a working phone camera. Without this component, the camera test will not work.

Our design also implements machine learning, which adds considerable processing time for detection that will need to be considered during implementation. The algorithm will be working with hundreds of images per user, requiring the processing of megabytes of data. This could cost considerable battery life and processor time, and could lead to a potential phone heating problem (further testing will confirm these concerns).

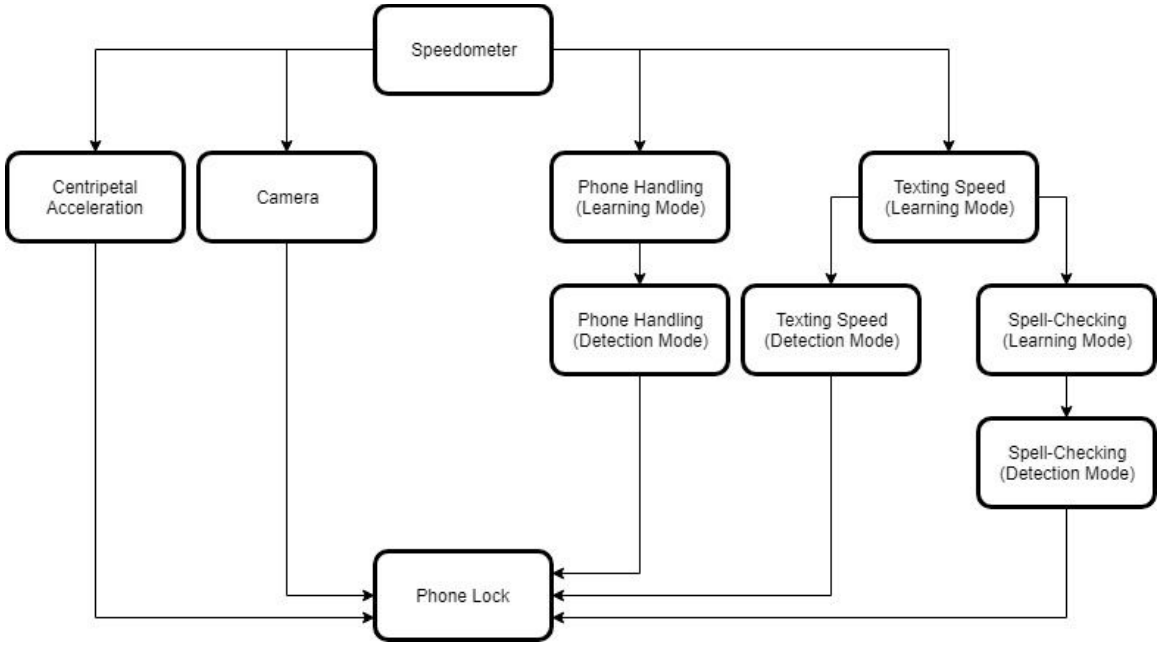

Figure a: Dependency Diagram

2.6 Technology Considerations

Our solution will be developed with usage of Android devices running the 6.0 OS and newer. Most of our data to determine texting while driving will be provided by the phone's built-in sensors. Android phones have a wide assortment of sensors including: accelerometer, gravity, gyroscope, linear acceleration, proximity, and rotation vector. The accelerometer and linear acceleration sensor measures the force of acceleration on the phone on the x, y, and z access. It is capable of detecting when the phone shakes, tilts, spins, or moves. The android gravity sensor monitors acceleration on all axes, but is primarily used to detect when the phone is falling. The gyroscope sensor measures the the phone's rate of rotation on the x, y, and z axes. It's primary function is to monitor when the phone turns or spins. The proximity report the distance of objects from the phone's screen. It is capable of measuring the phone's position during using. The rotation vector sensor monitors the phone's orientation per each axis.

2.7 Safety Considerations

Our primary safety concern is that we should not simulate actual texting while driving. As this is considered dangerous behavior we do not want to actually attempt this so we could have some slight differences from our tests with what we expect to happen.

2.8 Task Approach

The first stages of our projective included heavy research per the client's request. As texting while driving is a well known problem, there have been many studies on the subject as well as attempts at a solution. These attempts include apps with various means of detection and functionality. Our client was interested in looking at what the industry and researchers have already produced before moving forward.

From the requirements of the project, a non-external texting while driving detection mobile application, the direction of our application development is clear. As our client has requested we produce an android application, we will be developing a mobile application using the android developer kit. The kit has built in functions that interface with the sensors present on the phone, while newer versions increase the amount of sensors available to us. Further research narrowed down the sensors we use to create a viable detection scheme.

The sensors we will be interacting with on the phone include its accelerometer, gyroscope, and GPS. As we reach further stages in our project, we may expand our use of sensors, as there are many available to us. Table 1 contains some of the key sensors and functions associated with them.

Sensor	Type	Description	Common Uses
TYPE_ACCELEROMETER	Hardware	Measures the acceleration force in m/s^2 that is applied to a device on all three physical axes (x, y, and z), including the force of gravity.	Motion detection (shake, tilt, etc.).
TYPE_AMBIENT_TEMPERATURE	Hardware	Measures the ambient room temperature in degrees Celsius (°C). See note below.	Monitoring air temperatures.
TYPE_GRAVITY	Software or Hardware	Measures the force of gravity in m/s^2 that is applied to a device on all three physical axes (x, y, z).	Motion detection (shake, tilt, etc.).
TYPE_GYROSCOPE	Hardware	Measures a device's rate of rotation in rad/s around each of the three physical axes (x, y, and z).	Rotation detection (spin, turn, etc.).
TYPE_LIGHT	Hardware	Measures the ambient light level (illumination) in lx.	Controlling screen brightness.
TYPE_LINEAR_ACCELERATION	Software or Hardware	Measures the acceleration force in m/s^2 that is applied to a device on all three physical axes (x, y, and z), excluding the force of gravity.	Monitoring acceleration along a single axis.
TYPE_MAGNETIC_FIELD	Hardware	Measures the ambient geomagnetic field for all three physical axes (x, y, z) in μT.	Creating a compass.
TYPE_ORIENTATION	Software	Measures degrees of rotation that a device makes around all three physical axes (x, y, z). As of API level 3 you can obtain the inclination matrix and rotation matrix for a device by using the gravity sensor and the geomagnetic field sensor in conjunction with the getRotationMatrix() method.	Determining device position.

Table 1: Sensor types supported by the Android platform

Each sensor will aid in the overall detection scheme, each having its own test. The tests include a speed test, position test, centripetal acceleration test with the gyroscope, and vertical acceleration test. The speed test determines if the phone is exceeding speeds beyond what is possible for a human. The position test will determine where the user is in the car. The centripetal acceleration test will serve as another indicator of position. And the vertical acceleration will serve the same purpose. With these combined tests, our mobile application should achieve both high detection rates and accuracy.

Our strategy for accomplishing our project goals can be found in Figure b. provided below.

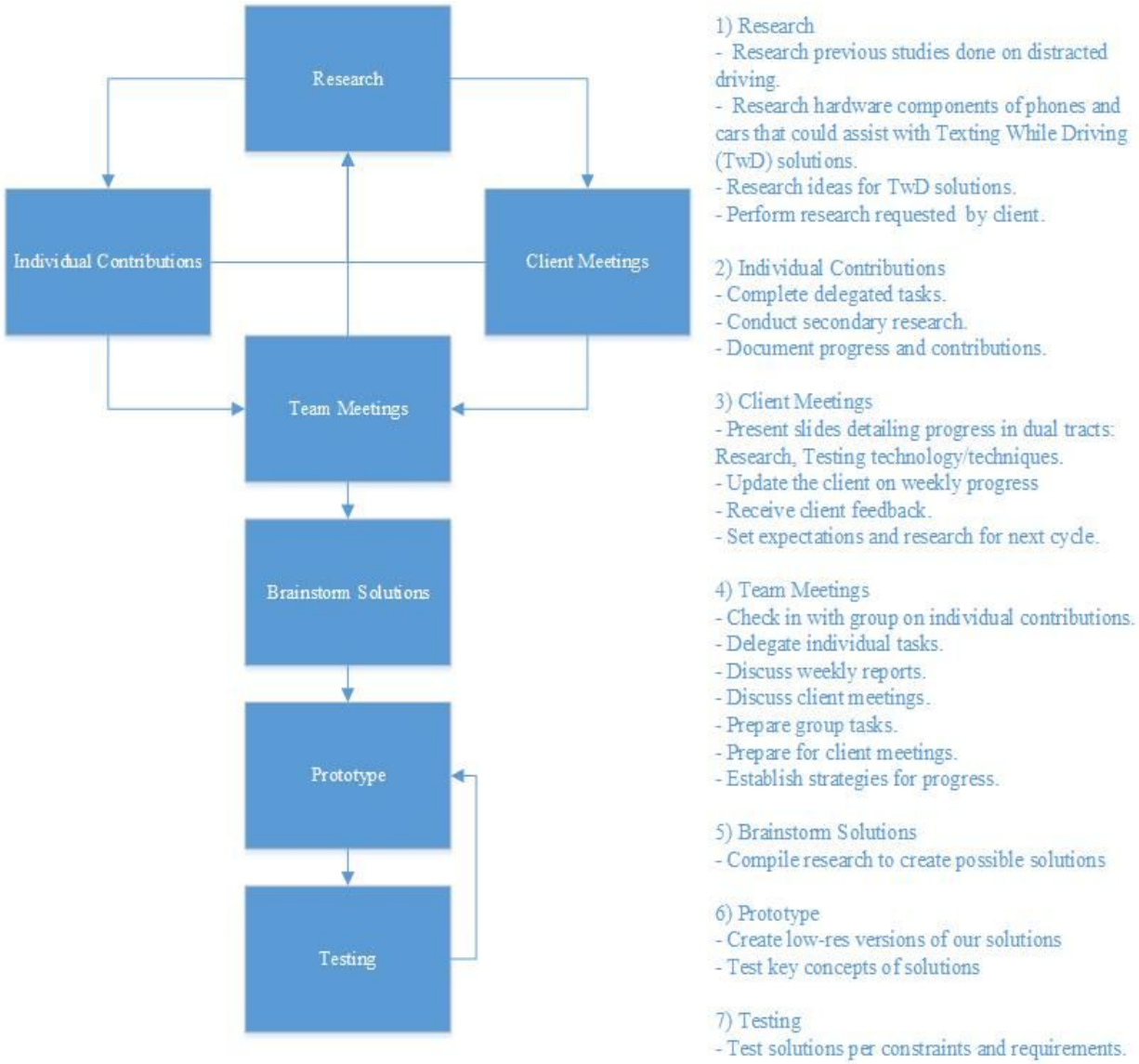

Figure b: Weekly Working Process Diagram

2.9 Possible Risks And Risk Management

1. (Physical health risk) Testing our application will not be safe unless we are in a controlled environment. More realistic testing will require use of a personal vehicle. Although the driver will not be the one performing the tests, it is easy to be distracted by the passenger working nearby and make driving less safe.
2. (HR / Scheduling risk) It is natural that team members will become sick during the course of two semesters. This can seriously hamper their ability to work efficiently for the duration of the illness, and it is almost certain to occur for every person in the group at some point, perhaps multiple times. Anything from minor to major injuries can also occur, purely by accident and unrelated to the project.
3. Testing will require a car having sufficient gas in it, which is a cost.

2.10 Project Proposed Milestones & Evaluation Criteria

The first clear milestone of the project is activating the accelerometer functionality, which enables Mode Switching to occur. This will probably be the easiest part of the phone to use, and it will be the first check for danger in the program. It decides whether the program will be in learning mode or in risk analysis, so it makes the most sense to accomplish first. This can be easily tested by riding as a passenger in someone's car or on the bus, comparing the accelerometer readings directly to the speedometer.

The second clear milestone is the completion of a prototype Learning Mode of the application. It will be difficult to see how effective the Learning Mode is without also creating the Detection Mode, which uses information gathered by Learning Mode. Because of that, it should obviously be developed first. This can be tested by making sure data is actually being recorded just through regular extended use of the phone.

The third clear milestone is the completion of a prototype Detection Mode. At this point, the detection may not be very accurate, but it should at least be working correctly in tandem with the other states of the system (i.e. everything communicates properly, no glaring bugs, etc.). Full testing is now possible.

The fourth milestone would occur after developing an enhanced version of the modes working together after seeing how our initial efforts turned out. By this point the false positive and false negative rates should be anywhere below 50% (in other words, the program is correct more often that it is wrong).

The fifth and final milestone will be guaranteeing the false positive and false negative rates are anywhere at or below 10%. Heavy testing will be required to reach this milestone, and there is a possibility it will not be reached. If it is reached, however, the project can be considered fully successful.

2.11 Project Tracking Procedures

Weekly progress reports are required by the instructor for this project. Since this is already a mandatory assignment, this is an easy way to keep track of our progress over time. Since GitLab is also in use, we will have a history of every version of the program code as well.

2.12 Expected Results

The outcome of this project will ideally be a stable, self-contained, standalone Android application which can learn its user's texting habits as well as detect and prevent texting while driving at least 90% reliably and false positive rate no greater than 3%. It will require no external hardware at all and provide the expected results.

The android development kit includes a wide range of debugging processes which will serve as preliminary testing with unit tests. It will be possible to run simulations using the android development kit before we go out and test in the field. From these tests will be get a preliminary detection rate, accuracy, and false positive rate. Comparing these results, after hundreds of simulations, will give us an idea of how effective the application could be.

Field testing will be done after simulations. Field testing will include the use of vehicle and the test android phones given to us by the client. In a controlled environment, it should be safe to test the functionality of the mobile application.

The six individual technologies will require some calibration to make sure they are accurate. To accomplish this, every team member will collect a small sample size of rough data to be compared to what the software implementation is providing. For example, we can take typing speed tests for texting speed that give us results in characters per minute. We can use this data and check it against what the application thinks our typing speed is and adjust accordingly.

Additionality, each technology implemented in our detection algorithm will require individual testing to validate its effectiveness in the overall detecting scheme. Using a

combination of simulations and field testing focused solely on that particular test will give us insight as to which tests should have greater weight in the system. If one technology is found to be more reliable than another it will begin greater focus, while the other technology is reserved (though not abandoned) as a secondary detection method.

2.13 Validation

Functional Requirements validation

1 & 2. Through emulation of driving conditions, we will see 50 times if the app accurately blocks texting and driving so that the false positives and negatives are at satisfactory levels stated in our requirements

3. Ensure that the application is able to run on test Android phones that have an OS of 6.0 by running test for requirements 1 & 2 on all named updates 6.0 and newer (Ex. 6.0 and 6.1 are both Marshmallow so we would not test both 6.0 and 6.1)

4. We can ensure the application doesn't require internet connection by turning off the service and Wi-Fi manually and making sure that the application still runs like the test for requirements 1 & 2.

Optional validation

5. Run test for requirements 1 & 2, with the addition of having others text that phone and trying to see if the user can read those texts.

6. Run test for requirement 5, and make sure that anything that does pop up is suppressed fast enough that it is not possible to read. This would be under a second, such as 0.2 seconds

2.14 Test Plan

We are going to need at least 2 different ways to test. For the passengers side, we can simply have someone be a passenger while someone is driving. From there we can measure how many false positives we get. The driver's side is a little bit more complicated, since we do not want to actually text and drive. A couple ways we could go about this are first, in the unlikely scenario we get our hands on a european car with a steering wheel on the passenger's side or a drivers ed car of some sort, we could just have the person that would normally be driving text. We could also just have a distraction for

the texter that would be a similar level to driving so that there is no danger. Both of these scenarios would pose little risk and would yield results.

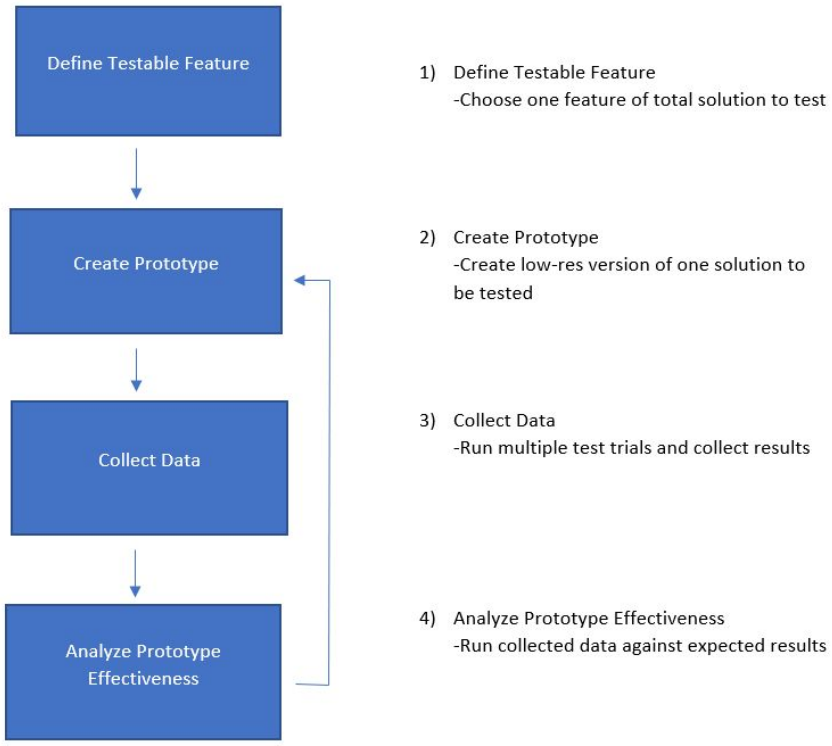

Figure c: Test plan for each feature

3 - Project Timeline, Estimated Resources, and Challenges

3.1 Project Timeline

Deliverables	Due Date
Project Plan Version 1	September 28th
Design Document Version 1	October 12th
Project Plan Version 2	October 26th
Speedometer Prototype	November 23rd
Project Plan Version 3	November 30th

Design Document Version 2	December 7th
Learning Mode Prototype	February 8th
Detection Mode Protype	March 8th
Application with false positive and false negative rates of 50%	April 5th
Final Project: Application with false positive and false negative rates at or below 10%	April 26th

Table 2: Deliverables for project timeline

Our project timeline is summarized in Table 2 (listed above), and reflects the team's current understanding of deliverables and project dates. Since we are in the beginning stages of our project, we are focusing heavily on research and testing techniques for our project. We have to learn about the current technology and approaches that have been used before so we can make our application into an improved version to help with the current research out there for detecting TwD. Research and the beginning stages of development along with some testing will be our main focus for this first semester of senior design.

Once we get into the second semester of senior design, the team's main focus will be development and testing of our application. This will be where we finish prototypes of different elements of the project and start integrating them together. We will also have constant testing going on to ensure that each component of our application is working properly. As shown in Figure b in the appendices, this will allow us to have a sufficient amount of time at the end of the project to test and make refinements to our project.

This timeline will allow the team to work on the components of the project consistently to be constantly making forward progress. As we get further into the project, we will have a greater understanding of how much time different components will take to develop. For now our gantt chart allows for approximately 2 weeks of development on each component followed by a 2 week period of testing for each component. Some components will take longer which will then be accommodated for as the project progresses. This project timeline gives the team some flexibility and will help keep the project moving forward through the year.

3.2 Feasibility Assessment

We plan to have around 3-4 tests to determine whether a texting person is both driving and texting. We hope to have false positives be under or around 20% and a false negative rate around 10%. These are just goals right now, and it is really not feasible to accurately predict them.

We foresee several challenges, both personal and technical, ahead of us, including getting to this goal. Here we will discuss personal challenges. One challenge we see is coming up with learning algorithms to see how a user typically behaves when they are not driving. None of us have any experience here, so there will most likely be a learning curve. Another challenge we see is having the test be able to aggressively detect drivers, but at the same time not give false positives. This will theoretically become more difficult the more tests we do. Another challenge we could have to work through is working with all the hardware in the phone. It again is unfamiliar territory to us. Lastly, all the papers we have looked at are from people much more qualified for this work than us. We would like to at least match their results, but this will be difficult for us given the time constraint and our levels of experience.

We have many personal challenges as this is a complex problem to solve using technologies many of us our not familiar with, but we also have technical challenges. Before we have a working prototype, a potential challenge we might encounter is that the feasible components of our solution might not be adequate for telling where the user is sitting in a car. If this is the case, then the other components we implement would be useless if we can't figure out if the user is driving. Assuming that our application will be able to determine if the user is driving, another challenge we will most likely face once we have a prototype that includes all of the components we are going to use is that the processing power it will take to run our machine learning algorithms and do image processing will make our standalone application run slow and drain the user's battery because all of the processing that will need to be done on the phone. This could also cause the phone to heat up quickly which could cause the user's phone to shut off. We will no doubt come across many more roadblocks as we develop our project, but these are the personal and technical challenges that we could encounter.

3.3 Personnel Effort Requirements

Listed below is the table for Personal Effort Requirements, divided by significant tasks in each milestone. The number of estimated group members required per task and the number of hours required from each of those members are listed.

Task	# Workers	Hours / Worker
1.1) Creation of project frame in Android Studio	1	2
1.2) Recording speed with the accelerometer	3	8
1.3) Creation of Mode Switching	3	13
2.1) Learning to get data from keyboard and text messaging application	3	10
2.2) Backend: storing data learned by the Learning Mode	2	10
2.3) Determining normal/abnormal behaviors	2	15
3.1) Getting user behavior during Detection Mode	2	15
3.2) Locking out texting during Detection Mode	2	15
4) Improving Learning/Detection Mode after initial tests	4	15
5) Final improvements to Learning/Detection Mode after extensive testing	6	5

Table 3: Personal Effort Limitations by Task

3.4 Other Resource Requirements

The materials required for the project are readily available. The only ones needed are Android phones, computers, and cars, none of which need to be purchased and all of which are accessible already. Laptops are being used by every group member for mobile development, and the university has many desktop computers available if more computers are needed.

3.5 Financial Requirements

The financial costs of the project are quite low. Android Studio is a free development tool, and every member of the group has access to an Android phone for programming and test, either because they already owned an eligible device or because they are using

one of the two provided for us by Iowa State University's College of Engineering at no charge.

The only cost worth considering is any fuel which must be consumed for proper testing inside a car. However, even this should be minimal and could be easily divided up among team members for a low cost per person if we alternate testing vehicles.

4 Closure Materials

4.1 Standards

Discussion of necessary standards that apply to our application will take place in the future. That said, we expect the application to abide by all necessary user data privacy laws.

4.2 Conclusion

Our project aims to provide a solution that can reliably detect texting while driving. To accomplish this, we will develop a standalone android application using android developer software. Each week we will iteratively develop a more functional version of the solution and present to the client for feedback. This project will be heavy on research and prototype testing as we experiment the feasibility and accuracy of different solutions. A number of solutions of varying complexity will be explored. Our ultimate goal is to create an application that will utilize the sensors embedded in the phone to successfully detect texting while driving. Once detected, a protection system will activate that makes the user unable to send text messages until they stop driving. The delivery of this solution will provide a means for safer driving through text prevention.

4.2 References

[1] “100 Distracted Driving Facts & Statistics for 2018.” *TeenSafe*, 5 July 2018, www.teensafe.com/distracted-driving/100-distracted-driving-facts-and-statistics-2018/.

[2] “Android Developers.” *Android Developers*, developer.android.com/.

[3] Cheng, Bo, Jian Xuesi, Li Xiang-Yang, and et al. “You’re Driving and Texting: Detecting Drivers using Personal Smart Phones by Leveraging Inertial Sensors.” *Mobicom*. ACM, 2013.

[4] “Drivewise® From Allstate GET REWARDED FOR SAFE DRIVING.” *Allstate*, https://www.allstate.com/drive-wise.aspx

[5] “Safe Drive Zone.” *Safe Drive Zone,* www.safedrivezone.com/.

[6] Wang, Yan, Jie Yang, Hongbo Liu, and et al. “Sensing Vehicle Dynamics for Determining Driver Phone Use.” *MobiSys.* ACM, 2013.

[7] WRAL, “'Oh my gosh!' Raleigh woman's snow photo goes viral,” *WRAL.com*, 13-Feb-2014. [Online]. Available: https://www.wral.com/-oh-my-gosh-raleigh-woman-s-snow-photo-goes-viral/13390109/. [Accessed: 24-Oct-2018].

4.3 Appendices

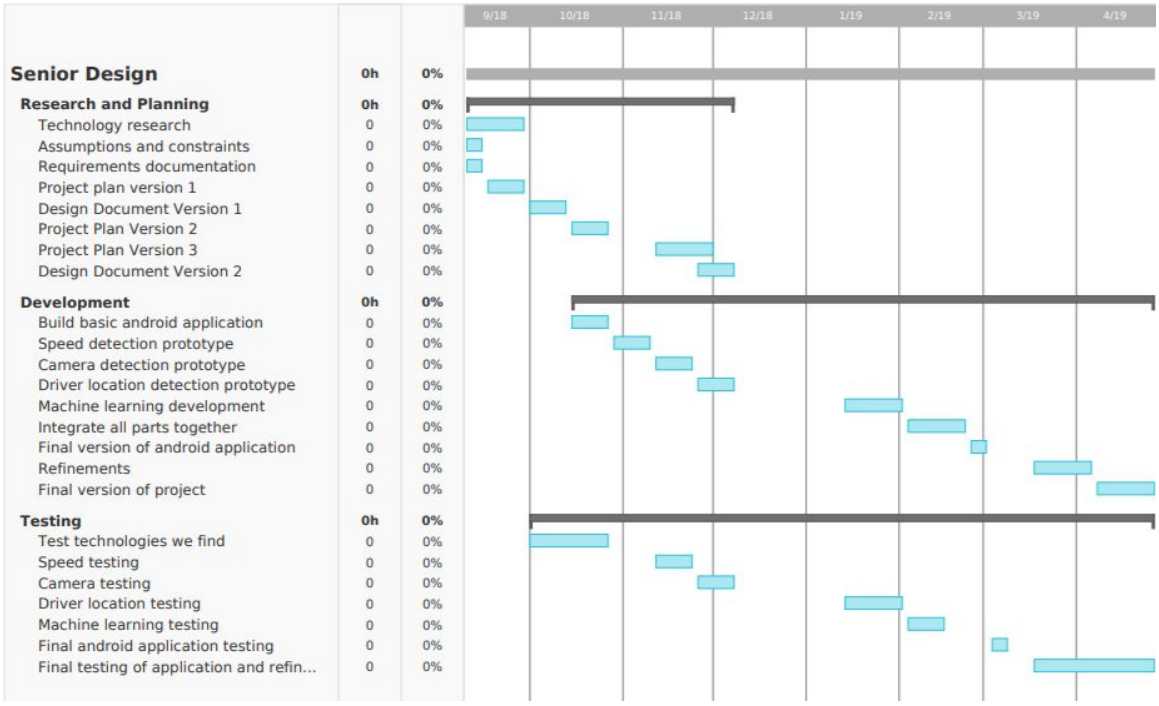

Figure d: Gantt chart for entire project timeline

Figure e: Fiery destruction possibly caused by texting and driving