[[1]](#footnote-1)

AlarmBand

Thomas E. Baim, EE, Vincent M. DiBlasio, CSE, George W. Puliafico, CSE, and Christopher Bartoli, CSE

*Abstract*—Since many individuals feel unsafe while alone, particularly in public, AlarmBand is being designed to provide users who worry about safety an automatic silent-alarm that will allow them to make emergency alerts and calls without having to touch their phone. A bracelet system will pair with a running smartphone application to allow users to use specific discreet motions to send for help or alert others. Ideally, the phone/bracelet combination will be a cheap, effective, and single-purpose alternative to pre-existing silent alarms and alert software for devices such as the Fitbit or Apple Watch.

# INTRODUCTION

M

any individuals worry about personal safety in uncomfortable situations. Particularly, individuals often feel most vulnerable when they are walking alone, or in an unfamiliar environment. This paranoia stems from the permanent existence of real-life threats, either to an individual's life or their belongings. The latest FBI report on crime data (2016) illustrates that crime is rare, but non-negligible [1]. For example, about 400 instances of violent crime were reported out of 100,00 people, as can be seen in Table I. This can make it difficult to feel truly safe in public, and often becomes a convincing factor for individuals to avoid ever navigating alone under certain conditions.

TABLE I: 2016 FBI Crime Data, collected from [1]

A good representation of public opinion can be viewed by consulting the Gallup News annual crime survey. According to their 2016 data, 35% of adults say that there is an area within a mile of where they live that they would be afraid to walk through alone at night, 47% of people worry about crime and violence a “great deal”, and 70% of people think that there is more crime in the U.S. than there was a year ago [2]. Evidently, individuals in the U.S.. feel concerned about safety in public, particularly when alone. Providing tools to make these individuals feel safer would thus be beneficial towards improving these opinions.

Since personal safety is important to many people, calling 911 has been established as the default way to handle an emergency situation. Unfortunately, there are certain instances where manually making a phone call is inconvenient or impossible. These include scenarios where you do not want to alert an attacker or potential attacker to your actions, or if you are physically unable to interact with your phone at the given time. In order to provide alternative options, many attempts have been made at fabricating devices and software to help an individual feel safe without having to make the phone call themselves.

A well-known example of technology that helps users in situations where they are physically impaired is LifeAlert. Consisting of a button worn as a necklace, users can push the button whenever they fall and are unable to stand, or are otherwise impaired and unable to reach a phone, in order to trigger a home communication system that puts them in vocal contact with the LifeAlert company, who can get help on your behalf [3]. While this does represent technology that removes a large amount of the user’s physical interface interaction in signaling for help, the requirement of vocal communication means that this technology is not subtle enough to translate into a threat-protection device.

Other developed solutions include a simple push-button apparatus that emits a high-pitched tone in order to alert onlookers to danger and ward off attackers. Although these devices are cheap and numerous due to their simplicity, they are intrinsically not subtle, and thus do not function well as a model for a silent alarm.

In order to provide more subtle solutions for alarms, applications for smartphones have been developed. Typically, these consist of a button on the phone screen that the user pushes in order to signal for help. This is done without any vocal communication. Unfortunately, this requires you to have your phone out and the application open, presenting a large amount of direct user interaction in order to signal for help.

AlarmBand strives to combine low levels of user interaction with non-vocal alert communication in order to provide an effective silent alarm to its users. Our proposed solution is a bracelet that will pair with a user’s mobile device to call for help when activated. The bracelet will track the user’s arm movement, listening for a specific motion that will trigger an alarm. This alarm will be wirelessly related to the mobile device, which will communicate the user’s distress to an emergency contact.

# Design

## Overview

To best achieve our goal of making a low-interaction silent alarm, AlarmBand will consist of bracelet system paired through Bluetooth with a smart phone application. Motion from the bracelet will be tracked, and when a determined signal is detected, a command is sent to the application, where the application makes the necessary response based on the given command. The full block diagram for the bracelet can be seen in Figure 1.

TABLE II

Specifications

|  |  |
| --- | --- |
| Specification | Value |
| MCC score for Signal Detection | 0.9 |
| Size | Fit comfortably on wrist |
| Wireless Communication Distance | <10 m |
| Battery Life | >8 hours |
|  |  |

The large lower block consists of the subsystems that will be contained in the bracelet. The power system is expected to power each of the other components, and provide a battery life of at least 8 hours. The sensors will consist of a chip with an integrated gyroscope and accelerometer of at least 6 degrees of freedom in order to provide enough data for us to accurately interpret motion. The microprocessor will take data from the sensors as it is collected and synthesize it so we can determine if patterns in the data match our desired movement patterns. Consisting of an Arduino Micro, the algorithms written for the Microprocessor will be done in C/C++ for compatibility. To determine the accuracy of the algorithms, we desire to reach an MCC score of 0.9 in correctly recognizing desired signals and not activating false positives. Finally, the Bluetooth module will take the resulting command generated by the interpretation algorithms and send it wirelessly to the paired smartphone application. The wireless communication is expected to work at up to 10 meters. Overall, the bracelet should be compact enough to comfortably fit around one’s arm so that the system can be properly worn as a bracelet.

The upper block simply consists of the phone application, which will be designed for Android and will handle commands given to it by the microprocessor.



Fig. 1: Block Diagram for our system

## Sensors

The sensor block will serve to generate the signals that will indicate when a user is or is not attempting to activate the device. The primary module is a MPU-6050 accelerometer and gyroscope sensor, but there will also be a simple push button to generate a secondary signal.

The MPU-6050 will produce data based on the positioning and movement of the user’s arm. Six different signals are generated by this device: linear and rotational acceleration readings for each of the x-,y-, and z-axises. To assess the accuracy of the accelerometer, we will display the data in real time as we rotate and reposition the chip such that each axis in turn is affected by the full force of gravity. We will know the device is correct if we see the acceleration data for each axis rise to *+/- g* and fall back to zero as the axis is pulled by gravity and then returned to rest. We will similarly test the gyroscope readings by observing data changes in real time as we rotate the device on one axis at a time, each time with the same rotational force. If the data shows one-directional rotational data in accordance with the axis that is being rotated and each axis shows the same magnitude of rotational acceleration, then the device is giving accurate readings.

The push button will generate a high signal when pressed. We will test this by monitoring the voltage difference across the button. When the button is activated, the voltage difference should be about 0V.

To build this block, we will use our knowledge from Circuit Analysis I and II to correctly plan the wiring of the devices. We will need to learn from the MPU-6050 datasheets what the exact conversion is from output data to acceleration units.

## Microprocessor

The microprocessor will be used to process the data coming back from the MPU-6050 chip. This is what we are using for gesture recognition. We are taking the 3 axis accelerometer data and using it in conjunction with the gyroscope data to find the changes in movement while holding or wearing the bracelet. Using 50 past values we are tracking how the user is moving and whether or not that movement matches the values that are generated from a specified activation gesture. We are going to be using the same chip on the Arduino Micro but we will need to implement it on our custom PCB. We are going to need to learn how to use reflow ovens and how to program directly to the microprocessor instead of using the Arduino IDE. We will use microcontroller programming knowledge from Computer Systems Lab I and II to do this part of the project.

## Bluetooth Module

We will utilize Bluetooth to handle wireless communication between the microprocessor and mobile device. It must be capable of establishing a communication channel with the mobile application so that the microprocessor can transmit signal an alert activation to the mobile device.

This communication will be tested by establishing a connection using a mobile application provided by Adafruit. Once a connection is established (as confirmed on the mobile device) we will attempt to print strings of data from the microprocessor to the Bluetooth module, which should then communicate them to the application where the data will finally be displayed.

## Phone Application

This is the Android application we are developing. It is being programmed in Java and will be how the user interacts with Alarmband. We still need to learn about how Android handles permissions and how to deal with an outside source making calls in the background. The majority of the background knowledge we are using to create this app is coming from the comp-sci courses we took and specifically, Software Intensive Engineering. We will be testing the app by seeing if it handles the signals from the bracelet correctly.

## Power System

Design of the power subsystem involved consulting the datasheets or manufacturer’s listing for each of the individual components in our prototype. The Arduino Micro requires 5V regulated input, and can be provided either through USB or pin. The Adafruit Bluefruit BLE module also requires 5V input. Finally, the Sparkfun MPU-6050 gyroscope/ accelerometer requires only 2.3-3.4V input. Since the Arduino Micro provides both a 5V power pin and a 3.3V power pin, we can power the Arduino through USB or pin connection with a 5V, and then hook up the Bluetooth module to the 5V pin on the Arduino and the MPU-6050 to the 3.3V pin, thus allowing us to power the whole system by powering the Arduino specifically.

Since our device is intended to be used over and over again, it is necessary that we implement a rechargeable battery system. Ideally, the battery would be housed inside the bracelet, and when battery gets low, the user could simply plug in the bracelet to recharge it. We therefore chose to implement a Lithium-Polymer (LiPo) battery with charging circuitry for our project. However, since rechargeable batteries can be highly volatile when mishandled, we decided to instead incorporate an existing recharger board instead of endeavoring to recharge the battery using a circuit we designed ourselves. Fortunately, existing circuitry for this purpose exists: the Adafruit Powerboost 500 charger is designed to regulate a compatible 3.7V LiPo battery and output about 5V consistently. This board is rechargeable via micro USB, which we would expose to the user as the port from which to charge the bracelet. By connecting the 5V output to the 5V input of the Arduino Micro, this battery/charger combo should be sufficient in recharging our system.

When considering compatible batteries for our system, the charging circuitry limited us to battery of at least 500mAh. Adafruit provides a 500mAh LiPo battery, and the next smallest available size was 1200mAh. For our prototype, we chose to buy the 1200mAh battery due to available stock through the manufacturer, with a long term goal of potentially switching to the smaller battery if desired and when stock updated. The 1200Ah battery has a 0.5C rating, which means that at any given moment, the battery should only provide a maximum of 600mA output current. This means that the current draw for our total system must not exceed 600mA at maximum performance. The battery is intended to work for 400 recharging cycles total before it should be replaced/discarded.

In order to measure the total current draw from our system, we acquired an in-line current/voltage meter designed to plug in via USB between the charging circuitry and the rest of our project. When hooked up, as can be seen in Figure 2, the meter measured that our project would draw anywhere between 10 and 30mA at once, averaging out at about 20mA (with the expected 5V output) when the Arduino, sensors, and Bluetooth module were all operating maximally. If we then divide this out of the 1200mAh battery capacity, we can expect that our system should be powered by this battery for 60 hours before needing to be recharged. Additionally, 20mA guarantees that we are always below the 600mA hard limit for the battery.

In considering whether or not to switch to the smaller 500mAh battery, there are two important factors: size and battery life. The smaller battery is about half the size of the current battery, thus aiding us in minimizing the size of the bracelet. The negative impact is that the lower battery capacity means we get a little less than half the current battery life, meaning the bracelet would have to be recharged more frequently.



Fig. 2: Picture of our system with current/voltage meter

# Project Management

TABLE III: MDR Goals with completion status

|  |  |
| --- | --- |
| MDR Goal | Status |
| Prototype that can collect/display data | Done |
| Establish Bluetooth connection from device to phone | Done |
| Gesture Recognition Proof of Concept | Done |
| Numerical Analysis for Power | Mostly Done |

Of the MDR goals set at PDR, most of them are accomplished. We have a working prototype of the components that will be in the band that can collect data from the accelerometer, and displays it in the Arduino IDE. We wanted to be able to connect to a laptop via Bluetooth, but that was slightly changed since the Bluetooth module we’re using is Bluetooth LE. The device connected to a phone, which was corrected goal. The final goal was to have a more thorough analysis of power data. We have an average use case for what the Arduino should be pulling, but more testing needs to be done to determine a worst case scenario so we can have a more accurate estimate of battery life. Additionally, data is being transferred between the device prototype and a phone using the Adafruit BLE app, and we have a button implemented that will be the low alert level signal for the bracelet.

Looking forward, we need to complete the worst case power analysis to come up with a better battery life estimate. We also need to pick the final motion that will trigger the alarm, and create an algorithm that will accurately detect that motion. The rest of the android application needs to be finished, and the low alert level signal needs to be determined.

The members of Alarmband work together well and all have unique areas of expertise, which creates an environment that is conducive to solving problems quickly and efficiently. Tom, being the only EE in the group is spearheading the hardware and will be designing the PCB. He is also helping the development of the android application, along with Vinni. Chris and George are both working on gesture recognition and algorithm development. We communicate both in person and online, using a group chat app.



Fig. 3: Gantt Chart showing progress goals

# Conclusion

In our current prototype, we have all of our modules connected and communicating. The MPU-6050 is connected to the microprocessor via a repurposed ethernet cable. We chose this method because the length of the cable allows for a large degree of movement of the accelerometer, while avoiding coupling concerns through the use of twisted-pair wires. The microprocessor is collecting and processing data the six movement signals from the MPU-6050, examining the latest fifty readings to determine the movement of the device. We chose a specific movement based on our ability to successfully detect it with a relatively rudimentary algorithm. Through trial and error, we decided that our motion for MDR would be a quick z-axis rotation in combination with a large x-axis acceleration. This looks like a violent forward-and-inward snap of the wrist. We examine the z-axis gyroscope data and x-axis acceleration data, and if these signals are over a certain threshold then the movement is considered activated. The microprocessor also graphs the input data in real time to a serial monitor graph. Currently, activation of the push button causes the displayed data to cycle between all six readings, the three accelerometer readings, and the three gyroscope readings. Detection of a specified gesture is represented by the generation and printing of a seventh signal to the serial monitor graph from the microprocessor. We are able to successfully establish a Bluetooth connection to a mobile device and transmit data from the microprocessor. The application is unfinished, and we are still relying on a Bluetooth testing mobile application provided by Adafruit.

For FPR, we will want to have a finished bracelet that is in its own housing and looks like something people could actually wear. We will also have a finished application that will be used to pair with the bracelet and provide feedback and user options for the bracelet. From now until FPR will bring a lot of challenges, the biggest of which will be perfecting the gesture recognition algorithm. Other notable difficulties will include figuring out how permissions work in the android manifest file. We will get there by learning using resources both from classes and from online.

References

1. “Crime in the U.S. 2016,” *FBI*, 08-May-2017. [Online]. Available: https://ucr.fbi.gov/crime-in-the-u.s/2016/crime-in-the-u.s.-2016/.
2. I. Gallup, “Crime,” *Gallup.com*. [Online]. Available: http://news.gallup.com/poll/1603/crime.aspx
3. *LIFE ALERT® - HELP! I've fallen and I can't get up!®*. [Online]. Available: http://lifealerthelp.com/default.aspx. B. Smith, “An approach to graphs of linear forms (Unpublished work style),” unpublished.
1. T. E. Baim from Natick, Ma (e-mail: tbaim@umass.edu).

V. M. DiBlasio from Springfield, Ma (e-mail: vdiblasio@umass.edu).

G. W. Puliafico from Ashland, Ma (e-mail: gpuliafico@umass.edu).

C. Bartoli from Ashland, Ma (e-mail: cbartoli@umass.edu). [↑](#footnote-ref-1)