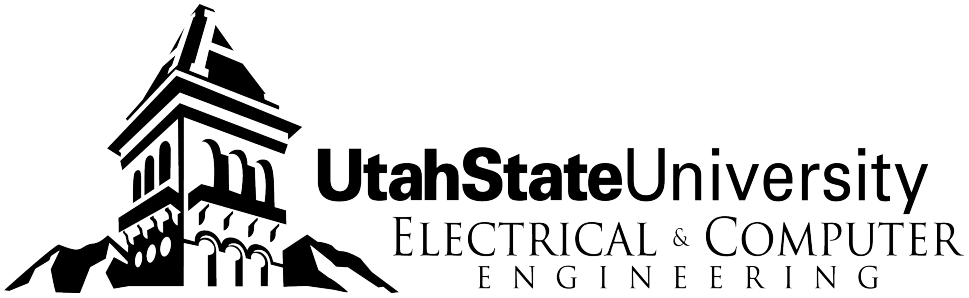


S.M.A.R.T Helmet

Benjamin Zeller
Joshua Perkins

May 4, 2017

Signature Page

Signature	*Signature*	*Signature*
Date and email	*Date and email*	*Date and email*

Revision History

Revision	Description	Author	Date	Approval
1				
2				
3				
4				
5				
6				
7				
8				
9				
10				

Contents

Specifications

1 Introduction

1.1 Purpose of the System

The Purpose of this system is to design a motorcycle helmet that provides: a safe riding experience, GPS navigation, and video recording. In recent years the commercial vehicle has surpassed normal motorcycle technology in nearly every regard. Many modern vehicles now come equipped with GPS technology. Motorcycle innovation, however, has not been able to keep up with this level of technological advancement. This gap in technology leaves many motorcycle enthusiasts forced to choose between their passion and convenience.

As early as the 1980's GPS navigation has been available, although at the time the U.S. military was engaged in scrambling the GPS signal decreasing the general accuracy of any civilian components. In the year 2000 President Clinton ordered the military to cease these operations allowing civilian devices a much greater degree of accuracy bringing with it the advent of GPS navigation in a typical sedan.

Primarily due to size and weight constraints this technology has been slow to adapt to the world of motorcycles. Until the last few years many of these GPS devices were often nearly the full size of a typical motorcycle helmet visor. The S.M.A.R.T. Helmet however utilizes advanced technology in projection and signal processing allowing motorcycle riders the experience that a commuter has been able to enjoy for nearly the last two decades.

The overall purpose behind this system is to allow motorcycle riders the ability to navigate safely and effectively while maintaining their attention on the road.

2 METHODS

2.1 Overview

The S.M.A.R.T. Helmet employs the use of two microprocessors, a pico-projector, cameras, a battery pack, and the user's smart-phone to calculate and display GPS navigational directions to the front visor of the helmet. The helmet itself is outfitted with a reflective film specifically designed for HUD applications and a battery pack.

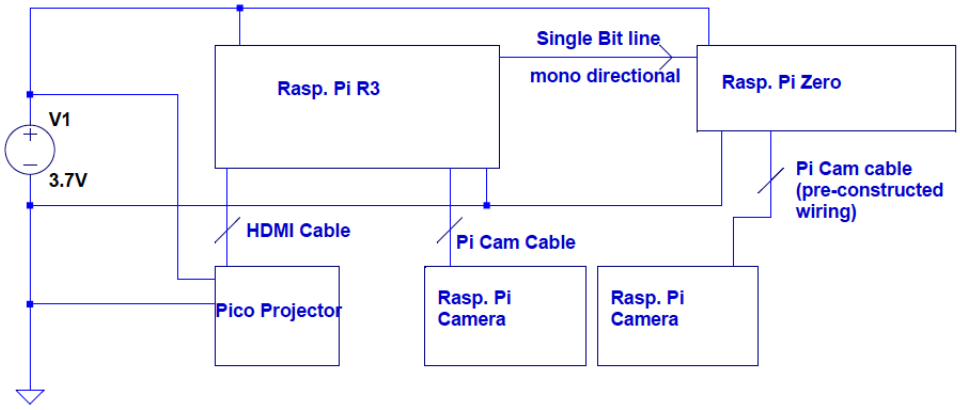
Each microprocessor controls an individual camera to record the rider's experience both in front and behind the rider. The secondary or slave microprocessor controls the primary camera pointing forward. This was done so that as the primary microprocessor receives and updates GPS navigation information the primary video would not be lost.

2.2 System Level Design

When designing the system we were tasked with some hard decisions. Those decisions were: what components to use, how to display, and how to control the device. The selection criteria for each section was based upon, relative ease of use, size, low cost, and included features.

2.2.1 System Diagram

The basic system can be seen in the simple system diagram below. The battery pack supplies the power required for each of the microprocessors which in turn power their own cameras. The projector is connected to the voltage source as a method to recharge it's own individual battery pack. The mono directional line between the microprocessors allows the main processor control over the secondary to initiate recording.



System Diagram

2.3 Chosen Components

2.3.1 Microprocessors

The Primary microprocessor was chosen to be a Raspberry Pi R3 B+ due to its simple interfacing, blue-tooth and WIFI connections, relative size and speed, and abilities to record high definition video. The Raspberry Pi was chosen over the Arduino platform, despite open-source schematics, due to the powerful CPU, GPU, included peripherals and low cost.

The Secondary Microprocessor is a Raspberry Pi Zero. The choice to use the Zero was a simple one, after the main processor was decided. We found that being able to use the same or nearly the same platform encouraged collaboration and made for ease of debugging.

2.3.2 Display

The display always posed a problem. Initially we had high hopes to work with displays like Google Glass, or even a transparent Organic Light Emitting Diode screen. After researching and window shopping for months, these technologies proved fruitless. We decided to approach from display from a commonly used method. We decided that we could use a projector with a transparent reflective screen.

The projector is a Brookestone Pico-projector. Because this is a projection type HUD there are a few complications. Firstly HUD visual quality is reduced by both ambient light and the quality of the projector. Secondly, as projectors are only capable of projecting in a single direction we were forced to either angle the projector towards the visor or to include an angled mirror within the helmet. Angling the projector was the most acceptable decisions due to weight constraints and affordability.

2.3.3 Controls

When we first considered the issue of controling the device, we always had one solution in the mind: an Android App. We decided on this immediately because it is free to develop, and Apps can be free. With the Android SDK a programmer has acess to a descrete set of sensors and communication protocols. These protocols include Bluetooth, GPS, and Internet. The built in GPS, and Bluetooth allowed us to cut extra costs at no expense to the end user. Not only is it

a affordable solution but it also negated any extra weight, or space required inside of the helmet.

2.4 Programming

In order to control the helmet we had to program various features to do what we wanted them to do. The programming languages that we used and why we chose those languages, and our choice of Source Version Control are detailed below.

2.4.1 Source Version Control

To control the development flow of each individual element we utilized a Source Version Control (SVC) software. Our choice of software was GitHub. Both students knew of, and how to use GitHub which made for seamless integration with the project. Utilizing this software allowed the us build each program while maintaining an unedited copy of each significant update of the Helmet software.

2.4.2 Python

The primary language controlling the helmet is the Python language. This language was chosen due to its relative ease and abundance of libraries available. The use of these libraries allowed for a much simpler integration of our commands and controls into the helm than trying to build up new functions throughout the process. While Python is not a language regularly taught to USU students the various tutorials and forum groups found through Internet searching allowed us to learn enough of the language to perform basic signal and image processing needed for this project.

2.4.3 Android

Programming in Android was a requirement of the project. There are many methods to do this. One method was to make use of MITs Inventor. This provided with little control and made no sense to us. We ultimately decided that one of us should learn to use Java and XML to build the App the normal way. We were able to make use of example apps, and programs from Google.

We used Android Studio to manage and design the App. This allowed us to control configuration and the User Interface (UI) elements. Although it is possible to use other

development environments, Android Studio was chosen due to its lack of cost, and support by Google.

2.5 Debugging

In order to verify requirements and track errors many methods were used. Described below are the methods of debugging and bug tracking that we used.

2.5.1 GitHub

GitHub being our choice of SVC, also allowed us to track bugs in each version of the software. We decided that this was an appropriate choice to maintain the code and develop a strong release candidate. Bugs can be reported by anyone with access to the GitHub repository that contains all the source code for the project.

2.5.2 Python

IDLE was used as a primary interpreter for the python programming and debugging. While the traditional debugging tools associated with debugging are not available on the free version that was used, a somewhat unconventional type of debugging was utilized. Our method for debugging was to program small snippets of code and view their individual output. Once these small snippets were in working order they were added to the overall base code used in the helmet. After the code snippets were added together for the final code it was tested for final verification and appropriate operation.

2.5.3 Android

In Android Studio, a developer has options to integrate not only an SVC but also use a Android Virtual Device (AVD). An AVD emulates a users phone, on a developers computer. This allows those without an Android device, or an older Android device to develop for said system. The AVD was used until it was noted that our phones, both Android, could be tethered to the computer for debugging.

This revelation allowed us to speed up compilation time in the Android Studio environment. A faster compilation time allowed for speedy development. We ended up using a Nexus 6P as our primary development platform. We were able to test all building blocks of the Android App on the phone successfully and find any bugs in the software.

2.5.4 GPS

Being able to test the Application controlling the directions was a matter of issue for the final product. It was not reasonable for every round of debugging for either of us to run down the street with a Pi, Screen, Battery, and Phone. We elected to make use of the on-board Developer Options native to Android Phones. We also made use of GPS spoofing Applications to simulate driving on a specified route. The Application we used is called GPS Joystick. This App was chosen over several others due to it functionality. This App provided not only user defined routes, single point GPS, but also speed spoofing.

This allowed our phones to provide a set speed, by the GPS spoofing Application, and follow a given path. This is important in developing a navigation Application because an end user will be following roads to reach their destination.

2.6 Construction

The completed helmet is constructed from a DOT certified helmet with our components secured to the outer shell. Each component has been secured to the shell of the helmet in a somewhat temporary manner as a means to establish an exact location base prior to permanent mounting. An epoxy mold was created and mounted permanently to the interior of the helmet as a means to angle the projector towards the helmet visor.

3 RESULTS

3.1 Table of Results

The testing results are summarized in the table below. All testing requirements, and definitions can be found in the initial proposal. Methods of verification can also be found in the proposal.

Table 1: Results

Number	Req. Title	Verification type	Verified	comments
4.1.1.1	No Component may extend past 0.25 incehse of exterior body of helmet.	visual	yes	
4.1.1.2	No Physical component may inhibit user vision while in use	visual	yes	
4.1.2.1	The device shall have a switch which will allow the user to activate/deactivate the HUD	visual	yes	
4.1.2.2	The GPS navigation and video recording shall be controlled directly by the user's smartphone	visual	yes	
4.1.3.1	The smartphone application shall conform to all Google Play Policies and Agreements.	unk	unk	
4.2.1.1	The device shall conform to all DOT requirements for motorcycle helmets	varied	conditional	helmet was purchased as a DOT acceptable device and modified. No modifications significantly degrade from overall quality of helmet.

Table 2: Results

Number	Req. Title	Verification type	Verified	comments
4.2.2.1	The device shall connect via blue-tooth to the user's smartphone.	visual verification that data transfer has occured	yes	
4.2.2.2	GPS navigation directions shall be displayed upon the HUD withough obscuring driver's vision.	visual	yes	
4.2.2.3	The device shall not draw more than 500 mA at 5 volts to comply with USB 2.0 specification.	current testing	no	all components used are in compliance with USB specifications at purchase. Assumed all components maintain this specification as no component was significantly altered at any point through design
4.3.2.1	The video recording shall be recorded on a removable device such as an SD card.	removal of SD card and read on separate device	yes	
4.3.2.2	The device shall maintain a batter pack not exceeding a weightof 0.5 lbs.	weight verification	yes	
4.3.2.3	The device shall not exceed an overall weight of 5.0 lbs	weight verification		

4 DISCUSSION

Our project primarily focused on the application of the concepts we have learned throughout our education. While scratch design was not heavily focused upon, implementation of off the shelf components and integration into a fully functional device was.

Throughout the process the main area of difficulty that was encountered was the processing of the GPS navigation data. in our original design we wanted to utilize the GPS directions of Google Maps, unfortunately this violates their terms and conditions so we were forced to find an alternate method. In the end we were able to find a simple city to city GPS direction calculator that will display directions to and from various cities throughout the continental United States.

We ran into another problem with the shell of the helmet when we found that the USU 3D printing office notified us that they were incapable of printing a shell the full size that we required. This left us with the option of printing out a shell that would be incapable of maintaining DOT standards or to modify an existing helmet.

Overall this project has been a success while there have been setbacks we have been able to build the helmet and get nearly what we wanted to out of it. Nearly every requirement was met with only a few aspects being left out of our testing.

Each of us has been able to learn a significant amount about the interaction of components in a complex system. How these individual components build up into something that would be otherwise unachievable except by from scratch design that would otherwise take several months if not years to complete. We have also been able to gain a significant amount of experience in learning to program in the python programming language, a common language used in the engineering industry.

Future generations should they desire to continue with this project may seek out a better implementation of the projection and HUD displays. Another topic to look into may be video stabilization during riding.