Augmented Reality Simulator

Purdue University

Cornell Cup 2014

Stephen Carlson

Steven Ellis

Thor Smith

Table of Contents

[1 Abstract 1](#_Toc385802324)

[2 Challenge Definition 2](#_Toc385802325)

[3 Project Entry Solution 3](#_Toc385802326)

[3.1 Customer Value Proposition 3](#_Toc385802327)

[3.2 Changes from Original Proposal 3](#_Toc385802328)

[3.3 Key Technical Elements 3](#_Toc385802329)

[4 Product Performance Evaluation 5](#_Toc385802330)

[4.1 Performance Metrics 5](#_Toc385802331)

[4.2 Failure Analysis 8](#_Toc385802332)

[5 Technical Documentation 9](#_Toc385802333)

[5.1 Component Selection Rationale 9](#_Toc385802334)

[5.1.1 Cost Requirements 9](#_Toc385802335)

[5.1.2 Microcontroller 9](#_Toc385802336)

[5.1.3 Motherboard 9](#_Toc385802337)

[5.1.4 Sensors 10](#_Toc385802338)

[5.1.5 Wireless Communication 11](#_Toc385802339)

[5.1.6 Display 11](#_Toc385802340)

[5.2 Hardware Design Considerations, Phase 1 12](#_Toc385802341)

[5.2.1 Power 12](#_Toc385802342)

[5.2.2 Packaging 13](#_Toc385802343)

[5.2.3 Printed Circuit Board 13](#_Toc385802344)

[5.2.4 Power Supply Layout 14](#_Toc385802345)

[5.2.5 Fabrication and Testing 14](#_Toc385802346)

[5.3 Hardware Design Considerations, Phase 2 15](#_Toc385802347)

[5.3.1 Power Supply 15](#_Toc385802348)

[5.3.2 Sensor Printed Circuit Board 15](#_Toc385802349)

[5.3.3 Power Supply Printed Circuit Board 16](#_Toc385802350)

[5.4 Software Design Considerations 16](#_Toc385802351)

[5.4.1 User Interface 17](#_Toc385802352)

[5.4.2 Networking 19](#_Toc385802353)

[5.4.3 Sensor Data 19](#_Toc385802354)

[5.4.4 Simulation Logic 20](#_Toc385802355)

[5.4.5 Rendering 20](#_Toc385802356)

[6 Project Execution 22](#_Toc385802357)

[6.1 Timeline 22](#_Toc385802358)

[6.1.1 Timeline Adjustments 22](#_Toc385802359)

[6.2 Budget 22](#_Toc385802360)

[6.2.1 Major Costs 22](#_Toc385802361)

[6.2.2 Funding Sources 23](#_Toc385802362)

[6.3 Mid-Review 23](#_Toc385802363)

[6.4 Process Understanding 23](#_Toc385802364)

[7 Recommendations and Next Steps 24](#_Toc385802365)

[8 Glossary 25](#_Toc385802366)

[9 References 26](#_Toc385802367)

[10 Acknowledgements 29](#_Toc385802368)

[11 Survey Responses 30](#_Toc385802369)

[12 Appendix 31](#_Toc385802370)

[12.1 Test Plans 31](#_Toc385802371)

[12.1.1 PCB Continuity and Functionality Test 31](#_Toc385802372)

[12.1.2 Power Supply Burn-In 31](#_Toc385802373)

[12.1.3 Total Runtime Test 32](#_Toc385802374)

[12.1.4 Wireless Range Test 33](#_Toc385802375)

[12.1.5 Wireless Latency Test 33](#_Toc385802376)

[12.1.6 GPS Total Accuracy Test 34](#_Toc385802377)

[12.1.7 GPS Relative Accuracy Test 36](#_Toc385802378)

[12.1.8 Brightness, Contrast, and Color Reproduction Test 36](#_Toc385802379)

[12.1.9 Maximum Simultaneous Devices Test 37](#_Toc385802380)

[12.2 Assembly and Construction 39](#_Toc385802381)

[12.2.1 Materials required 39](#_Toc385802382)

[12.2.2 Rail Assembly 39](#_Toc385802383)

[12.2.3 Display Assembly 39](#_Toc385802384)

[12.2.4 Reflector 39](#_Toc385802385)

[12.2.5 Sensor Assembly 39](#_Toc385802386)

[12.2.6 Mounting the PCB to the enclosure 39](#_Toc385802387)

[12.2.7 Mounting the enclosure to the rail 40](#_Toc385802388)

[12.3 Failure Mode Worksheet 41](#_Toc385802389)

# Abstract

Wearable computing is quickly becoming the next step of embedded systems evolution. Devices like Google Glass and the Samsung Galaxy Gear aim to make computing part of everyday life. However, existing wearable devices are not designed to work together in a collaborative real-time application with multiple users. The goal of this project is to construct a real-time collaborative augmented reality application.

To meet this challenge, the solution must have a see-through display, track the user’s position and orientation, and support multiple users, and have a strong and ergonomic mechanical design. The visual display is designed with a see through material so as to allow the user to perceive the real world behind the display and minimize peripheral vision obstruction. A Fresnel lens was used to increase the apparent distance of visible objects to help alleviate the user from re-focusing between the display and the real world.

Since the object of this project is to overlay objects onto real-world positions, instruments were installed to determine the position and orientation of the user in three-dimensional space. An inertial measurement unit tracks the orientation of the users head and a GPS device tracks the user’s geospatial location. To enable collaboration where users share the same augmented environment, WiFi was used. WiFi was chosen because of its relatively fast data speeds needed to send all of the updates about the position of a headset. At the time of this writing, multi-user functionality is not fully complete and tested. A hardhat was used in the design to provide a firm mount for the screen and to provide the user with a durable and ergonomic design.

The software to support an augmented reality application will be dependent on the eventual usage case, but the operating system which configures the headsets and handles common functionality like wireless networking and sensor processing was designed to be easy to set up and use. Tkinter was used to build a simple GUI for the user to traverse. The GUI also serves to allow non-technical users to understand the device. The application starts automatically on power up and walks the user through initial setup processes, such as wireless pairing. The project is designed to allow custom applications. These applications must maintain a consistent state across any and all participating users in a simulation and update the displayed virtual world. The device was tested with a sample application written for the device that simulated a PacMan-like maze overlaid across the user’s vision with moving ghosts. The update rate of the display was very responsive.

# Challenge Definition

Wearable computing is quickly becoming the next step of embedded systems evolution. Devices like Google Glass [1] as shown in Figure 1, the Oculus Rift, and the Epson BT-100 aim to make augmented and virtual reality part of everyday life. However, existing wearable devices do not yet allow one to work in a collaborative real-time and immersive augmented reality application, which is the goal of this project. Such a device would have broad applications in education, defense, group recreation, and artistic performances where small teams of users can work together in the same augmented reality environment.

Figure : Google Glass, an existing wearable computing product

Portability and ergonomic challenges will primarily influence project development, as competing devices are often chosen for small size and low power consumption. A good solution to this challenge must have long battery life and a low mass to promote long-term wearable usage patterns. The virtual environment will not replace the real environment, so an ideal solution must also allow the user to perceive the real world with minimal eye strain or peripheral vision obstruction. Since augmented reality systems aim to overlay objects onto real-world positions, a successful device must be able to determine the position and orientation of the user in three-dimensional space within at least a few meters of precision.

This device must be capable of wireless communication with other augmented reality devices in close proximity to enable multi-user applications, where more than one user can share the same virtual environment. From a user perspective, ergonomics are also a major concern for wearable devices; the challenge is incorporating the desired functionality while not creating an unnecessarily heavy or bulky device. A successful device must also be durable enough to withstand regular use while not requiring technically experienced users for support.

While the software to support an augmented reality application will be dependent on the eventual usage case, the operating system which configures the headsets and handles common functionality like wireless networking and sensor processing must be easy to set up and use. For a non-technical user to understand the device, the application must start automatically on power up and walk the user through any initial setup process such as wireless pairing or sensor calibration. Custom applications which are written for this device should maintain a consistent state across any and all participating users and update the displayed virtual world at timely intervals.

# Project Entry Solution

The current solution has not changed significantly from its proposal. The solution to this challenge focuses on playing a multiplayer game of PacMan. The user may walk around outdoors and see the maze, ghosts, and PacMan pellets around him or her. The simulation is complete when all PacMan pellets are successfully collected or the user runs into a ghost. The platform was designed to provide a framework for writing simulations that could allow for different games to be written besides PacMan.

The solution to the project is divided into two phases. The first phase focused on getting a preliminary prototype working, and the second phase focused on adding and integrating with the Intel development board. The block diagram for phase two can be seen in Figure 2. The key sections are broken into the software on the Intel Motherboard, the GPS and IMU, the display, and the battery and power supply.

The proposed augmented reality system took the form of a head-mounted device which overlays virtual objects onto a semi-transparent panel suspended in front of the user’s eyes. Virtual objects will be wirelessly coordinated between any collaborating devices to allow multiple users to share the same environment. This key feature addresses the limitation of most existing augmented reality solutions. Battery power is used for the whole device to permit untethered operation and satisfy the challenge requirement for portability. Sensors on the back of the helmet allow the position of the user and the user’s head orientation to be tracked.



Figure : Block diagram of the Augmented Reality Simulator system

## Headset and Display Overview

The headset packaging itself is labelled as *Headset* and *Display* in Figure 2. A hard hat provides an adjustable fit to a variety of different heads and the mounting points for the sensor board and the LCD screen. The LCD screen is mounted in a black box at the front of the helmet and the screen is reflected off of a reflective film on a slanted piece of acrylic. Aluminum rods are used to make the front and rear mountings possible.

The virtual display device is a central component. Commercial transparent displays are cost prohibitive, so the device utilizes a regular LCD screen which will project the virtual image onto a partially mirrored surface, combining the image and the environment. This design minimally affects peripheral vision and depth perception, as opposed to some existing approaches which overlay the images in software on a digital image of the environment. A Fresnel lens is used to make the display appear larger and farther away, making it easier to focus on.

## Power System Overview

The power supply board was built to accompany the DE2i-150 and make it mobile. This component is indicated by the *Power Supply*, *Fuel Gauge*, and *Battery* in Figure 2. The power supply board provides the means for tracking the battery status over USB and powering the DE2i-150, the LCD, and the sensor board.

## Sensors Overview

The sensor printed circuit board (PCB) contains the inertial measurement unit and the global positioning system sensors. This is mounted in the back of the headset and it is used to track the position of the player and the orientation of the player’s head. This is shown as the *GPS Receiver* and *IMU* in Figure 2.

## Intel Board Overview

The Intel board provides Wi-Fi capabilities for networking and it also provides the platform for key software modules. These software modules are built to perform the tasks of networking the helmets together, harvesting sensor data, displaying the user interface, rendering the 3D graphics, and running a simulation. This is indicated by the *Augmented Reality Application* and *Viewport Algorithm*.

# Product Performance Evaluation

## Performance Metrics

The experience of the end user will determine the satisfactory completion of the challenge, so frequent testing will be performed at multiple stages of the design to incorporate feedback from non-technical users. Project success will primarily be measured by the criteria shown in Table 1. Table 2 shows performance metrics specifically targeted towards software.

Because of the wearable nature of this product, one important metric is its suitability for extended usage periods. This requires both a long battery life to enable the extended usage, and hardware that avoids causing discomfort to the user during extended usage. Because the intended application is gaming, the runtime target selected was 3 hours, which is expected to be near the upper limit of how long a typical person would want to play a game continuously. While not wearable, the primary competitors in the portable gaming market are the Nintendo 3DS and PlayStation Vita, both of which have approximately 3 hours of battery life in the worst case. [2] [3] The phase one headset achieved almost exactly 3 hours of runtime, and the phase 2 headset lasts for **TODO**

The other major consideration during extended usage is the comfort of wearing the headset and backpack combination. The most easily measurable metric for long term usage comfort is the weight of the device, which can be separated into the headset and backpack. The headset weight is the most crucial, because all weight there places a load on the neck whereas a backpack can be much heavier before being burdensome to the user. The targets selected based on some informal testing were 1kg for the headset and 3kg for the backpack. In terms of the wearable computing competition, the Oculus Rift Dev Kit 2 weighs 440g (reported as 0.97lb) [4].

Aside from user comfort, the quality of the displayed image is one of the most important aspects of a wearable display. One measure of this is the field of view, or how large the display appears to the user. This is measured in degrees, with human vision being approximately 180 degrees. The calculation for this is fairly simple, as shown in the following equation:

However, the optics employed in all of these systems complicate matters slightly by making the size and distance of the display appear to be different than they actually are. Oculus states the field of view of their Development Kit 2 as 100 degrees [4], whereas Google does not directly state field of view for Glass but provides enough information to calculate it [5]. The descriptions lists the display for Glass as ‘equivalent of a 25 inch high definition screen from eight feet away’, which when plugged into the above equation gives approximately 15 degrees.

The optics system of the augmented reality headset is fairly simple with only two elements, making this calculation straightforward. First, there is the distance from the eye to the reflector, then the distance from the reflector to the magnifying lens, and finally the distance from the magnifying lens to the display. The reflector does nothing but turn the light at a 90 degree angle, so for this calculation the first two parts of the optical path can be considered together with the sum of the distances. The magnifying lens creates an image which appears to be both larger and farther away, so the distance to the display appears to be the sum of the distance from the user’s eye to the lens and the distance from the lens to the projected image of the display. The display also appears larger, increasing the field of view. The field of view calculation is shown below, in a step by step fashion.

|  |  |
| --- | --- |
| **Variable name** | **Meaning** |
| *f* | Focal length of lens |
| *o* | Distance from lens to object |
| *i* | Distance from lens to image |
| *r1* | Distance from eye to reflector |
| *r2* | Distance from reflector to lens |
| *d* | Distance from eye to image |
| *s* | Diagonal display size |
| *h* | Diagonal display image size |

As can be seen, the 61 degree field of view of the augmented reality headset lies roughly half way in between the very narrow Google Glass, which is intended as a HUD in the user’s peripheral vision, and the Oculus Rift, which is intended as a fully immersive virtual reality experience.

Because the intended application is a multiuser simulation using wireless networking, the range of this network system is an important performance metric. The usable range is one of the major factors as to what types of simulations are feasible to run on the system. Currently, there are really no other wearable computing products available designed to network with each other to which the augmented reality headset can be compared. The Oculus Rift is meant to be tethered to a desktop PC, and Google Glass tethers to a smartphone to connect to the internet. The target set for this project was 80m for line of sight usage, as that seemed like a reasonably large playfield for simulations to use. The actual measured range before packet loss became significant was TODO

There are a variety of factors which impact the responsiveness of the image displayed to the user and how accurately virtual objects are mapped to real world objects. One is the image update rate, which needs to be high enough to appear as a smoothly moving image to the user. Movies have traditionally been shot at 24 frames per second, so 20 frames per second seemed like a reasonable minimum target. The actual frame rate of the simulation running on the headset varies from roughly 15-25 frames per second depending on exactly what is on the screen at the moment. The sensor data being used as inputs to the rendering is also crucial to the responsiveness of the simulation. The location sensing determines how well virtual objects line up with real world objects, so this should occur at both a high rate and with high accuracy. The rate target was set to be the same as the frame rate target, and the accuracy target at 2m. At this level of accuracy, discrepancies are very obvious but this is approximately the best performance achievable with current GPS technology without extremely expensive receivers. This was deemed an acceptable tradeoff for GPS being able to be used outside without any sort of external hardware requirements, which most other tracking technologies (such as camera based tracking) cannot achieve easily.

Table : Augmented reality system performance measures and satisfactory design metrics

| Metric | Target | Description |
| --- | --- | --- |
| Mass | Headset: 1 kg  Backpack: 3 kg | Neither the headset nor the backpack can be too heavy to impede the performance of typical activities by the user. |
| Power | 3 hours runtime | Augmented reality applications run for long durations, so the device must operate long enough on battery power to be usable. The remaining battery capacity should be displayed to the user. |
| Location | Minimum 2 meter accuracy and precision | The accuracy of the rendered virtual world depends heavily on the accuracy and precision of the user’s geospatial position. |
| Wireless | 80 m range, line of sight | A unique feature of this device is the wireless coordination capability of multiple headsets in the same virtual environment. The current wireless signal strength should be reported to the user. |
| Comfort | No running speed or vision reduction | Ideally, the user would be able to perform actions without a noticeable burden on the user. It is difficult to quantitatively measure comfort, but the user should be able to orient their head, see the environment, and move from place to place as effectively as if the device was not worn. |
| Durability | Operate from 0 to 40 C indoors and outdoors in dry conditions | Most augmented reality applications involve users walking or running, which will subject the device to vibration and light impacts. In addition, sunlight, humidity, and dust are inherent concerns for a portable device. |
| Usability | End user can start up and use device without a technician | A user unfamiliar with the device technical details must be able to power on the device, perform start-up procedures such as wireless connection, and load the desired application. |

Table : Augmented Reality Software Performance Metrics

| Metric | Target | Description |
| --- | --- | --- |
| 3D Graphics | Minimum 5000 polygon count | The user will expect to see a good 3-D image. Phase 2 offers opportunities to increase graphical performance. |
| Graphics | Minimum 20 Hz update rate | We will need to render ghosts and the maze walls in real time according to the user’s geospatial location and their head orientation. |
| Collision Detection | Minimum 2 meter accuracy and precision | The ability of the software to detect collision detection will be affected by the accuracy of the GPS. The minimum required to be meaningful cannot exceed a couple of meters. This means ghosts could be 2 m2 to account for accuracy of GPS. |
| Network  User Position Update Rate | Minimum of 20Hz (50ms). | The position and orientation of each headset will need to be communicated wirelessly at least every 50ms to keep up with the GPS and provide meaningful feedback about collisions to each headset. |
| Network  Graphic Position Update Rate | Minimum of 20Hz  (50ms) | The position information about positions of ghosts and other virtual objects needs to be updated in a timely manner and should at least be as fast as the update rate of the player’s position. |
| User Interface Usability | End user can navigate the user interface intuitively. | A user unfamiliar with the operation of the software should be able to figure it out with minimal references to the user manual. For phase 1 this will be displayed from the control unit. In phase 2 the user interface will migrate to the headset’s display. |

## Failure Analysis

# Technical Documentation

## Component Selection Rationale

Innumerable decisions were involved when choosing parts for all project aspects, so the most important component selections for meeting project metrics are shown below.

### Cost Requirements

An initial estimate of $300 per headset and the budget shown in section 6.2.3 was used to guide parts selection. Cost is secondary to other project metrics due to the dearth of comparable devices on the market; price was only used to differentiate otherwise similar design choices. For example, Google Glass [1], a product with higher quality compact optics, is currently priced at $1500, and several other augmented reality products such as ARQuake [6] and CastAR [7] have not left development. In addition, the development expenses incurred are likely to be very different than actual production costs.

### Microcontroller

Processing power is the biggest concern for the headset microcontroller selection. Decoding and filtering the 9 degree of freedom inertial measurement unit (IMU) data into the user’s head orientation requires floating point math, and high-speed IMU filtering with low latency and timing jitter leads to better head tracking performance. Power consumption is secondary as other parts will dominate power use. The microcontroller must be able to handle each sensor connection as shown in Figure 2 and Table 3.

Table : Required on-chip microcontroller peripherals for headset

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Device | Peripheral | Special Features | Minimum Speed | Pin Count |
| Motherboard | SPI | Master Mode, Chip Select | 800 KHz | 4 |
| GPS | Serial |  | 38.4 Kbaud | 2 |
| Fuel Gauge, IMU | I2C | Open-Drain, Current Sink | 400 KHz | 2 |
| Wireless | Serial | Full Duplex | 115.2 Kbaud | 2 |
| Future Expansion | GPIO | Configurable Data Direction | *N/A* | 10 |
|  |  |  | **Minimum Pins:** | 20 |

Table 4 shows the microcontrollers which were considered, all of which met the constraints for on-chip peripherals and pin count.

Table : Decision matrix for headset microcontrollers

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Part | I/O | RAM | Flash | Clock | DMA | Special Features | Cost |
| STM32F405RGT6 | 51 | 192K | 1M | 168MHz | 16 | Floating Point Unit | $11 |
| PIC32MX695F512H | 53 | 128K | 512K | 80MHz | 8 |  | $10 |
| ATSAM4S16BA-AU | 47 | 128K | 1M | 120MHz | 22 |  | $10 |

The STM32F405 microcontroller was chosen for Phase 1 due to the availability of a low cost $15 development board [8] and its very high performance, particularly with single cycle floating point operations. This chip also features extra RAM and processing power for future expansion. For Phase 2, this microcontroller was re-used, as it was available on hand and allowed reuse of the project code base from Phase 1.

### Motherboard

The headset GPU motherboard must have a compatible SPI port to interface with the microcontroller over SPI as described above. It also must support a standard display output such as VGA, composite, or HDMI for rendering to a commercially available screen. A second available standard interface such as USB is also preferred for debugging and to allow future expansion to other user input devices for more advanced simulations.

As the graphics processing motherboard is the largest power consumer on the device, low power consumption is mandatory. Small size and light weight is also important for a portable device. By comparison, performance is much less of a concern. The motherboard should also feature a USB port and SPI interface for connection to the microcontroller.

Table : Decision matrix for motherboards

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Part | Idle Power | RAM | USB | SPI | Clock | Cost (board) |
| Raspberry Pi Model A | 2 W | 256M | 1 | 1 | 700MHz | $25 |
| BeagleBone Black | 2 W | 512M | 1 | 2 | 1GHz | $45 |
| Intel® Atom™ DE2i-150 | 12 W | 2G | 4 | 1 | 1.8GHz | $150 |

As shown in Table 5, the Raspberry Pi Model A was chosen in Phase 1 for its low power consumption, cost, and outstanding community support. Prolific examples can be found on the Internet for the Raspberry Pi, whereas the BeagleBone Black is new and difficult to acquire.

Phase 1 exposed the limitations of the Raspberry Pi Model A motherboard. Although its low power consumption led to long battery life, the graphics processing unit on board was extremely difficult to use. Tasks such as displaying text and loading models were extremely slow, impeding refresh rate metrics.

With the contribution of two Intel® Atom™ Terasic DE2i-150 boards [9] from the Cornell Cup sponsors, the motherboard selection was revisited. Due to the simplicity of coding graphics on the standard OpenGL framework for the Intel® Atom™ board and improved performance, this motherboard was chosen for Phase 2, with the higher power consumption and mass accepted as a necessary cost.

### Sensors

Due to advancements in MEMS technology, most inertial measurement units provide excellent noise performance and accuracy. Power consumption is not an issue as other parts will dominate power use. A breakout board must be available since most IMUs come in LGA packages that cannot be soldered by processes easily available to students.

Table : Decision matrix for 9-degree of freedom inertial measurement unit (IMU)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Part | Gyro | Accelerometer | Compass | Resolution | Board Cost |
| STEVAL-MKI108V2 | L3GD20 | LSM303DLHC | LSM303DLHC | 16/16/12 bits | $27 |
| MPU-9150 | MPU-9150 | MPU-9150 | MPU-9150 | 16/16/13 bits | $50 |
| SparkFun™ IMU | ITG3200 | ADXL345 | HMC5883L | 16/13/12 bits | $50 |

As seen in Table 6, the slightly increased resolution of the MPU-9150 is offset by its cost and lack of documentation, particularly for the use of its powerful internal filtering algorithms [10]. The cheaper STMicroelectronics part [11] was chosen instead for Phase 1 and Phase 2, as it offered no apparent disadvantages to the expensive SparkFun™ 9-DOF sensor stick [12].

Past senior design groups experienced problems with low resolution and slow updates from GPS. The real-time constraints of this project call for a fast and high-resolution receiver module. Power consumption is also a concern, as the GPS unit will always be tracking the user.

Table : Decision matrix for GPS receiver unit

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Part | Resolution | Update Rate | Accuracy | Antenna | Power Use | Cost |
| Venus638FLPx | ~0.5m | 20Hz | 2m | External | 29 mA | $50 |
| GP-635T | ~1m | 5Hz | 2m | Integrated | 56 mA | $40 |

The recently released Venus638FLPx vastly outperforms comparable units [13] due to its excellent resolution and astonishing 20Hz update rate as shown in Table 7. Real-world tests in section 12.1.6 confirmed the superior reception of this GPS receiver and its external antenna, leading to its selection for both the Phase 1 and Phase 2 headsets.

### Wireless Communication

Wireless communication keeps the individual headsets connected throughout the simulation. High reliability and low latency are paramount to meet the real-time requirements of the simulation. The wireless device must also transmit a minimum of 6-8 KB per second of data to meet the simultaneous device metrics, and must be able to report the signal quality and strength. A range of 150 yards line of sight will handle most simulations.

The wireless modules considered are listed in Table 8. Range tests as described in section 12.1.4 indicated that extremely large antennas are required to obtain the advertised range on many RF communications modules, so the observed range for each device is shown.

Table : Decision matrix for Phase 1 wireless communication devices

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Part | Data Rate | Range (LOS) | Latency | Cost |
| XBee® Pro 900HP | 11.5 KB/s | 300 m | 50ms | $40 |
| nRF24L01+ | 25 KB/s | 100 m | 50ms | $20 |
| Wireless-N | 6 MB/s | 200 m | 80ms | $30 |
| Bluetooth and Cell Phone Network | 200 KB/s | 2 km | 500ms | $30 |

Despite the range and data rate advantages of offloading wireless to a cell network, the test in section 12.1.5 found its latency to be unacceptable for augmented reality. The XBee® Pro was selected in Phase 1 for its superior range over the nRF24L01+ module [14], which may not reach the design metric of 80 m in real-world conditions.

With the inclusion of the Intel® Atom™ motherboard in Phase 2, the XBee® Pro was dropped in favor of the motherboard’s built-in Wireless N module. The small decrease in range and increase in latency was compensated by the ease of use of standard 802.11 networking and the extreme increase in data rate, which allows more simultaneous devices connected and more complex simulations with higher graphics quality.

### Display

The Phase 1 motherboard, the Raspberry Pi, supports output over HDMI and composite video. With the typical requirements of low power consumption and light weight in mind, a screen intended for use in a car back-up camera was obtained [15] which massed only 80 g with its enclosure removed. The low cost of just $50 also contributed to this part’s selection.

After the completion of Phase 1, user testing revealed that the resolution and low brightness of this display led to trouble using the device in bright outdoor conditions. With the switch to a more capable graphics processing motherboard, a higher resolution display was sought that could address apparent 3D graphics performance.

Two displays from the same vendor were tested; one was brighter and had a higher resolution of [16], while the other one was cheaper and used slightly less power with a resolution of [17]. Display testing as shown in section 12.1.8 showed that both displays significantly improved brightness and contrast performance. If the challenges with graphics drivers discussed in [SECTION] are addressed, the higher-resolution display is the better choice due to its superior brightness and readability; otherwise, the lower resolution of the second display will lead to better head tracking latency performance as fewer pixels have to be redrawn every frame.

## Hardware Design Considerations, Phase 1

### Power

Power is limited on the headset, as it runs on a lightweight single lithium ion cell worn by the user. The biggest power consumer is the graphics processor and display. The wireless radio and GPS also use power to send and receive data. By comparison, the current draw of the microcontroller, IMU, and other circuitry is negligible. Low-dropout linear regulators were used, since few available switch-mode regulators could efficiently convert the 3.7 V battery voltage into the required 3.3 V as shown in Table 9.

Table : Power supply constraints and supply rails in use on Phase 1 headset

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Supply Rail | Voltage | Current Required | Topology | Peripherals Powered |
| Vcore | 3.3 V | 500 mA | Linear | Graphics Processing |
| VMCU | 3.3 V | 350 mA | Linear | Microcontroller, Wireless, GPS |
| VADC | 3.3 V | 30 mA | Linear | IMU |
| VDISPLAY | 5 V | 300 mA | Switching | Display, USB |

With the estimated power requirements shown above, the battery size required to achieve the design metric run time of 3 hours can be calculated using the equation shown below. As a rule of thumb, 80% of a battery’s advertised capacity can be used. This equation was chosen for its simplicity and presentation for this application by the Senior Design advisors:

**Example**: For the Phase 1 headset data shown above, with = 3.7 V nominally for a 1-cell lithium ion battery and = 80% = 0.8, can be calculated as follows:

(η estimated from charts in [18])

Thus, a 3.7 V battery with a capacity will lead to about 3 hours of runtime. In reality, aging and temperature will reduce the actual runtime. The nearest available battery capacity was

To supply power at the right voltage, an appropriate voltage regulator was selected for each rail. Special attention was paid to the VADC regulator, as good line regulation and high ripple rejection reduce the noise seen by the IMU. A readily available voltage regulator from Micrel® [19] was found to meet all requirements to serve each of the 3.3 V supplies. The 5 V supply, being higher than the battery voltage, requires a switch-mode regulator. A low-cost power module [18] was found that met the design requirements in an easy-to-assemble package.

Two convenient choices were found for the Phase 1 battery, a SparkFun™ prismatic cell [20] and an Adafruit cylindrical cell pack [21]. Both batteries combined three smaller lithium polymer cells in parallel with a protection circuit to reduce fault current and prevent overcharging for improved reliability. While the Adafruit battery was slightly cheaper, the SparkFun™ cell had a much smaller form factor and weighed less, leading to its selection for improved portability.

### Packaging

The headset is designed to be portable and worn by the user. Of the peripherals in use, the display, inertial measurement unit, and antennas must be placed on the user’s head. An adjustable mount for the half silvered mirror and reflection shield must be provided to allow users with different head sizes to position the device for optimal viewing conditions. The device should be packaged in one unit with few exposed wires, as non-technical users should not be expected to plug in parts. Packaging must be able to withstand the vibration and impacts that are inevitable for wearable electronics, and must be able to tolerate outdoor temperatures.

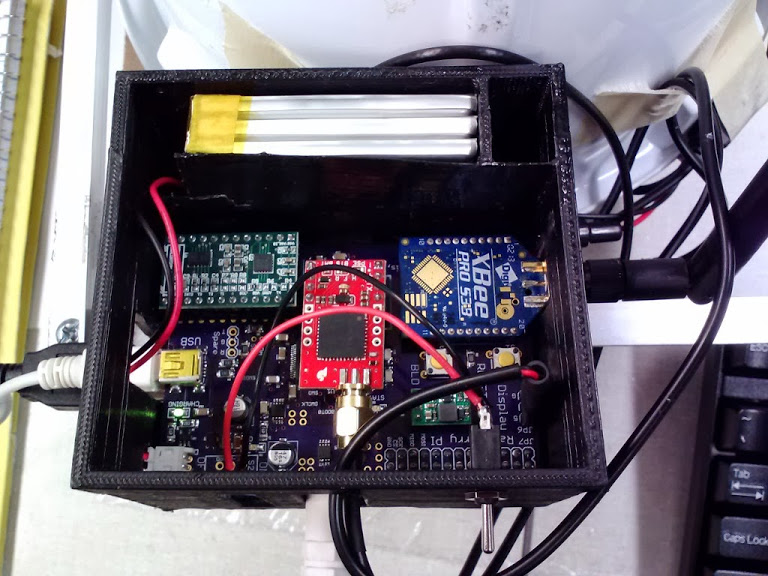
### Printed Circuit Board

Due to the extensive use of surface mount components in this project, a custom-designed printed circuit board (PCB) was chosen as the most reliable method of implementing the schematic design; the schematic and circuit board were designed using the EAGLE Layout Editor. OSH Park, the manufacturer used by Purdue Electrical Engineering for PCB production, quotes the minimum values shown in Table 10 for their standard designs. For optimal manufacturing yield and to allow tolerance for best practice, the PCB was designed with wider tracing and spacing and a larger drill size.

Table : PCB specifications for OSH Park [22], 1 mil = 0.001 inch

|  |  |  |
| --- | --- | --- |
| Metric | Given | Chosen for Manufacturability |
| Trace Thickness | 6 mil | 10 mil |
| Trace Spacing | 6 mil | 8 mil |
| Drill Diameter | 13 mil | 15 mil |
| Drill Tolerance | 5 mil | *N/A* |
| Copper Thickness | 1 | *N/A* |

To properly mate with the Raspberry Pi as shown in Figure 3, the PCB dimensions and I/O header placement had to exactly match the motherboard mechanical specifications [23]. External connectors for USB charging, the battery, and the radio antennas also had to face outwards on the PCB edges to allow connections to the final packaging.



**Battery**

**Power Supply**

**GPS**

**IMU**

**XBee**

Figure : Finished printed circuit board in its packaging with major sections highlighted; the motherboard is stacked directly below the board shown

The major radio frequency emitters on this board contribute electromagnetic interference (EMI) which could disrupt the inertial measurement unit (IMU), but the noise magnitude decreases with the square of the distance. The global positioning system (GPS) unit has an external antenna [24] which moves its noise emissions away from the PCB, so the XBee® radio and antenna [25] was placed in the lower left corner, as far away from the IMU as possible. Noise caused by signal reflections can be attenuated by matching the USB and crystal oscillator to under the 0.05 inch tolerance [26] using the EAGLE “*run length-freq-ri*” tool.

To make the PCB easier to route, pin assignments and part placements near the microcontroller were chosen to limit crossing signals. After final placement, spare I/O pins and a spare serial port were brought out to pads to enable debugging and future expansion. An image of the complete layout is attached in the [Appendix].

### Power Supply Layout

The power supply for this board must deal with a fluctuating load and battery management to maximize the device battery life while reducing the risk of device failure due to overheating. Total current could be up to 2 A during radio transmission events as evidenced by the design constraint analysis, so power supply traces were widened according to the trace heat equations shown below recommended by the industry standard ANSI IPC-2221 to reduce resistive losses [27].

**Example**: is given to be (see Table 10) and of heating was allowed. Therefore, for a power supply trace carrying of current, the calculation of the minimum trace would be (1 mil = 0.001 inch):

Therefore, this trace should be no narrower than 0.031 inches to carry the rated current with less than the specified temperature rise. In practice, traces of twice the minimum width were selected wherever possible to reduce power loss from trace resistance and increase battery life.

Power and ground planes were emphasized to simplify power routing. The bottom of the board was devoted to a ground plane with few signal traces to reduce the size of noise-inducing ground loops. Power planes also provide heat dissipation paths with low thermal resistance for the regulator circuits, keeping them cool to increase reliability.

### Fabrication and Testing

After receiving the initial circuit board from the manufacturer, the first procedure involved electrical conductivity tests as described in section 12.1.1. Importantly, this test discovered an unwanted short from the IMU power supply VADC to ground; continued conductivity measurements allowed location and repair of a hair-line copper filament shorting the target trace to a ground plane.

When the board had been tested, the power supply components were installed first to verify the operation of this important section of the board. Burn-in testing procedures as described in section 12.1.2 were applied for 24 hours at the rated design load, and the supply outputs continued to function at the desired voltages. After populating the remaining components, simple test programs were uploaded to the microcontroller to test each part of the board independently. Notable tests as mentioned in sections 12.1.7 and 12.1.4 included GPS location precision and wireless range.

## Hardware Design Considerations, Phase 2

### Power Supply

With the introduction of the Intel® Atom™ board for powering the Phase 2 graphics rendering, a substantial revision to the power supply was required. As the necessary battery to achieve the design metrics became too heavy to meet the headset design mass, the entire power supply was moved to the backpack unit with the Atom™ board. Since the sensors need to remain on the user’s head, the Phase 1 PCB was bifurcated into a dedicated sensor board and a power supply board.

Another microcontroller with USB support was now required for the split power supply board to report the battery state of charge over USB to the Intel® Atom™ board. This microcontroller had to have low stand-by power consumption, as it monitors the state of charge even when the motherboard is powered off. The STM32L100RBT6 chip was chosen due to its design and programming similarity with the sensor microcontroller and its idle mode power consumption of less than 2 µA [28].

For the power supply board, the battery size calculations were revisited with new measurements for power consumption. The Intel® Atom™ DE2i-150 board was measured as consuming 1100 mA at 12 V under normal operation with wireless active. Table 11 shows the updated power supply requirements. Due to the ease of design with the TI WEBENCH® designer and the availability of free samples, the LMZ14203H switching step-down converter [29] was selected to power the DE2i-150 board.

Table : Power supply constraints and supply rails in use on Phase 2 headset

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Supply Rail | Voltage | Current Required | Topology | Peripherals Powered |
| Vcore | 12 V | 1500 mA | Switching | Graphics Processing, Display, Wireless |
| VMCU\* | 3.3 V | 100 mA | Linear | Microcontroller, GPS |
| VADC\* | 3.3 V | 30 mA | Linear | IMU |
| VSTBY | 3.3 V | 10 mA | Linear | Fuel Gauge |

(\*) *Power sourced from DE2i-150, which is counted as part of Vcore when calculating power usage*

With the estimated power requirements shown above, the battery size required to achieve the design metric run time of 3 hours can be calculated using the equation from section 5.2.1. As large lithium-ion batteries pose a much greater risk of a safety hazard as described in section **Error! Reference source not found.**, a lithium iron phosphate battery was considered instead; the best readily available voltage that could be efficiently converted downwards to 12 V was 19.2 V. Therefore, the minimum capacity is , but the nearest affordable option was [30] from the battery supply partner recommended by the Cornell Cup sponsors. In reality, slightly longer runtimes than predicted were observed, allowing the desired total runtime metrics to still be met.

### Sensor Printed Circuit Board

Feedback from Phase 1 indicated that the head tracking system generally worked well. Phase 2 thus re-used a large fraction of the Phase 1 system board for handling the IMU and GPS sensors. The USB connector was re-purposed for communications with the Intel® board, and this opportunity was taken to reduce the PCB dimensions to lower the mass of the headset. Since the microcontroller was unchanged from Phase 1, the routing for that critical area of the PCB was also unchanged.

With the removal of the XBee®, the peak current draw of the board now remains within the 500 mA USB maximum load [26]. A single 3.3 V voltage regulator sufficient to handle 500 mA now supplies the microcontroller and GPS [19] as shown in Figure 4. The separate regulator for the IMU was also retained in its original location.



Figure : Power supply of the Phase 2 sensor PCB

After fabrication and population, testing as conducted in section 5.2.5 revealed no issues with this design. As the board was an evolution of the Phase 1 design, knowledge from that successful design proved invaluable to the rapid design and success of this PCB.

### Power Supply Printed Circuit Board

This board had a new design, needing to incorporate battery charging, monitoring, and voltage regulation for a much higher power system. Its mass was still a concern, as the backpack still had to target a wearable design mass. Connectors for USB communication, the battery, the charger, and the power output were also incorporated on the sides of the board. As this board was also fabricated by OSH Park, the specifications from section 5.2.3 were used.

Members of the previous Purdue Cornell Cup entry “The Incredible HUD” recommended the use of the DS2782 [31] fuel gauge for monitoring the remaining battery life. This board also contains the previously chosen LMZ14203 switch-mode power supply for the display and Intel® Atom™ board. The same procedures for the Phase 1 PCB from sections 5.2.3 - 5.2.5 were used to ensure a successful design process; no issues were encountered during design, fabrication, or testing. Full schematics and layouts can be found in [Appendix].

## Software Design Considerations

The Software on the DE2i-150 is divided into five modules as is shown in Figure 5. Each module communicates to the other modules using sockets. Sockets were chosen over memory mapping because data sent between modules closely resembles packets. The Simulation logic is one of the key modules because it is swappable with other logic simulations using a common application programming interface. This allows third party sources to write simulations for our hardware without needing to compile with any of our other modules or understand how each module provides its information. The *Networking* and *Sensor Data* modules are combined in the same executable.



Figure : Diagram of software modules

### User Interface

A graphical application is provided to the user that allows them to either host a simulation or join an existing simulation. Its initial screen is shown below in Figure 6.

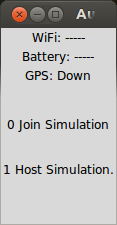


Figure : Select mode screen

The *Host Simulation* option allows the user to select headsets that may participate in the simulation and then choose a simulation to launch. The *Join Simulation* option allows the user to wait for a host as their headset is automatically configured. It displays vital information about the headset as it moves through the stages of acceptance by the host, receiving simulation information, and starting the simulation.

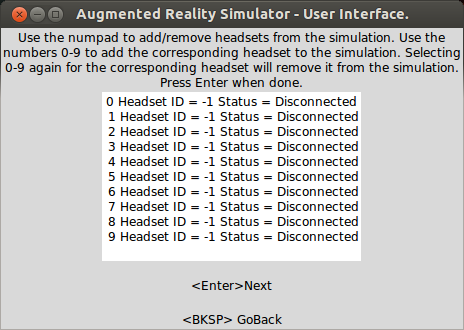


Figure : Select headset screen

The first screen shown to the simulation host is the *Select Headsets* screen shown in Figure 7. This screen allows the user to dynamically add and remove headsets that should be included in a simulation.

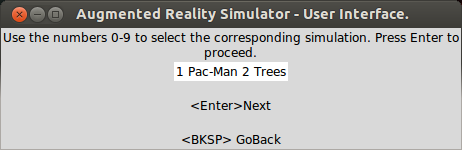


Figure : Select simulation screen

The second screen shown to the host is the *Select Simulation* screen shown in Figure 8. This screen allows the user to pick from the list of available simulations. When the host user makes a selection, the simulation will be sent and synchronized to all accepted headsets.



Figure : Start simulation screen

The next screen shown to the simulation host is the *Start Simulation* window shown in Figure 9. This allows each player to feel prepared before the actual simulation starts and after each user has received the simulation files.



Figure : Run simulation screen

The *Run Simulation* screen shown in Figure 10 may not be visible to the user since the simulation may hide it quickly, but this screen allows the user to continue monitoring status.

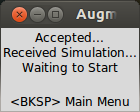
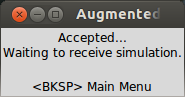
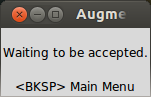


Figure : Simulation Screens for waiting to be accepted, waiting to receive simulation information, and waiting to start the simulation respectively.

If the *Join Simulation* option was selected instead, the user will then be taken through the three screens shown above in Figure 11. The transitions will happen automatically as the host proceeds through accepting and selecting the simulation options.

### Networking

The networking module abstracts away the details of sending and receiving information over the wireless network. If the headset is configured to *Join Simulation*, then this module begins by broadcasting itself as an available headset for a simulation. When accepted by a host, it waits for information about the simulation to use. After receiving simulation information, it waits for the host’s confirmation to start simulating. During the simulation, this module sends out a periodic message that informs the host of its current geospatial location and orientation.

If the headset is configured to *Host Simulation*, then it remains quiet on the network and listens for available headsets to add to a simulation, adding each headset it discovers to its list of available headsets. The user interface may choose to accept and reject headsets at this time. When the simulation starts, the networking code sends out packets to inform other headsets of updates to 3-D object positions and orientations and the state of other users.

### Sensor Data

The sensor data module is in the same executable as the networking module. This module holds the latest position and orientation information about the headset. It also performs arithmetic on the longitude and latitude coordinates received from the global positioning system (GPS) unit to produce local coordinates and as is shown below. Each scaling factor is calculated based on the arc lengths of one degree of latitude and longitude on a circular cross section of the earth at a given degree of latitude, where .

The and values are measured relative to an origin which is taken to be the first stable position after achieving a GPS lock. Since the origin is reset when starting a new simulation, this equation is an adequate approximation for latitude and longitude over the wireless range of just a few hundred meters. Performing true non-Euclidean geometry to handle the curvature of the Earth would involve prohibitively slow calculations.

### Simulation Logic

A simulation is a Python program that utilizes our interfaces to start the graphics and to collect information about the user position and the position of other players. A series of functions and objects have been provided in this API to make development as simple as possible. This section was designed to be easy to prototype with. An extension of this section would provide the ability to write simulations in C++.

### Rendering

The 3D rendering portion of the software makes use of the open source library Irrlicht [32], which is based on OpenGL and provides a relatively simple interface for accomplishing common rendering tasks. When the rendering function starts, it is passed a configuration file from which it loads the initial state of the simulation. The main rendering loop then communicates with the sensor reading code to get the user’s location and orientation, and with the simulation code to get the locations of all the virtual objects to be rendered. This data is collected is collected once per frame, and then rendered. Additionally, the battery’s current state of charge and wireless signal strength are collected from the microcontroller and displayed to the user here.

Table : List of important software classes; full software documentation can be found in [Appendix]

|  |  |
| --- | --- |
| ./simulation\_src/testApplication.py | PacMan Application |
| ./simulation\_src/level/genlevel.c | Generates a config file from an ascii art file. |
| ./simulation\_src/gpuPyInterface.py | A python sockets interface to the graphics rendering c++ code. |
| ./sensor\_micro\_src/src/main.c | The code for collecting GPS and IMU data. |
| ./gpu\_src/render/genlevel.c | Generates a config file from an ascii art file. |
| ./gpu\_src/render/main.cpp | Starts and runs all of the graphics rendering code. |
| ./gpu\_src/render/normalScreen.sh | Script to un-invert the screen. |
| ./gpu\_src/render/GameObject.h | Class that holds information about a specific 3d object being drawn on-screen. |
| ./gpu\_src/render/invertScreen.sh | Invert the screen left-right |
| ./gpu\_src/render/gpuPyThreadInterface.cpp | Serves as a sockets interface between the simulation python code and the rendering c++ code. |
| ./gpu\_src/render/InputReceiver.cpp |  |
| ./gpu\_src/render/Makefile |  |
| ./gpu\_src/render/GameObject.cpp | Class that holds information about a specific 3d object being drawn on-screen. |
| ./gpu\_src/render/SensorReader.cpp | Class that serves as a sockets interface to the networking/sensor process to collect users position. |
| ./gpu\_src/render/interface.cpp |  |
| ./gpu\_src/render/SensorReader.h | Class that serves as a sockets interface to the networking/sensor process to collect users position. |
| ./gpu\_src/render/gpuPyThreadInterface.h | Serves as a sockets interface between the simulation python code and the rendering c++ code. |
| ./gui\_src/textInfo.py | Text strings to display to the end user for instruction. |
| ./gui\_src/launch.py | The GUI used by the end user. |
| ./gui\_src/guiNetInterface.py | The sockets interface between the GUI and the networking/sensor code. |
| ./networking\_src/networking\_daemon.cpp | The code that starts the networking threads and sensor data collection threads. |
| ./networking\_src/packetStrings.py | Python script to auto-generate packets.c/h |
| ./networking\_src/packetLib.cpp | Contains all of the networking functions and associated data. |
| ./networking\_src/threadInterface.h | Serves as a sockets interface to any code module talking to the networking or sensor code. |
| ./networking\_src/Makefile |  |
| ./networking\_src/gpsIMUDataThread.cpp | This code collects GPS and IMU sensor data from the sensor board over UART. |
| ./networking\_src/gpsIMUDataThread.h | This code collects GPS and IMU sensor data from the sensor board over UART. |
| ./networking\_src/threadInterface.cpp | Serves as a sockets interface to any code module talking to the networking or sensor code. |
| ./networking\_src/packetLib.h | Contains all of the networking functions and associated data. |
| ./networking\_src/generatePackets.py | Python script to auto-generate packets.c/h |

# Project Execution

## Timeline

The timeline and milestones can be seen in the Timeline Appendix. There are two forms of the timeline available. A word document contains the graphical version and an excel spreadsheet contains the text version. An additional sheet in the excel document outlines how tasks were broken down.

The graphical timeline and the excel timeline were updated independently. The graphical timeline was used to get a feel for how much time each task took. Each task in the timeline is color coded according to what category it belongs. These categories have been chosen because they logically separate pieces of the design and separate tasks that different team members are expected to accomplish. Multiple tasks are overlapped with each other to demonstrate which parts of the project could have been completed concurrently. If the graphical timeline differs from the text based timeline on any task, the text based timeline should be considered accurate.

The excel timeline has a breakdown of all of the tasks to be accomplished on the first worksheet with the day of the accomplishment listed on the left hand side. There is a second worksheet that demonstrates how tasks were broken down from their respective categories and outlines milestones versus sub tasks.

The timeline is divided into two sections denoted as *Phase 1* and *Phase 2*. *Phase 1* refers to the first iteration of the project which was implemented for a senior design project. This iteration helped the team to realize what the final outcome would look like and to understand the challenges that must be faced while designing the second iteration.

### Timeline Adjustments

Various adjustments were made to the timeline throughout the life of the project. The biggest adjustment came about because of an unexpected death at Purdue University. Two weeks of project time were lost because the death affected all team members. As the printed circuit boards were being designed it became clear that the sensor PCB would be easier to construct than the power PCB because much of the work had been ironed out the previous semester. Thus, the power PCB construction was separated from the sensor PCB construction and pushed back. Sensor software development was able to proceed without the power PCB because the atom comes with its own power supply. The delay in the completion of the power PCB and delay in completing networking software pushed back many test items. Certain test items were removed from the calendar in the interest of completing as much work as possible.

## Budget

The entire budget and bill of materials may be found in the Bill of Materials Appendix, which is an excel document with multiple sheets. The first sheet outlines the bill of materials and expenditures for phase 1. The second and third sheets outline the bill of materials and expenditures for phase 2. The rest of this section will highlight justifications for the most expensive components and outline the sources of funding for our project.

### Major Costs

The major costs associated with phase 2 of the project include the battery, the screens, the GPS units and the inertial measurement units. The battery was costly because we needed to achieve our desired target battery life. Given the performance of our phase 1 design, it was decided that with the new Intel board a better resolution should be achieved. The screens selected were the only ones found that fit our form factor. Both were selected so that the performance of each could be evaluated. The GPS was selected as the cheapest available that maximized our resolution. The IMU was chosen to provide enough axes to be able to accurately detect a person’s head’s movement for pitch, yaw, and roll.

### Funding Sources

Table 13: Sources of funding for Phase 2 design describes the sources of funding received by the Augmented Reality Simulator team during the course of phase 1 and phase 2. All other expenses were covered by the team.

Table : Sources of funding for Phase 1 and Phase 2 design

|  |  |
| --- | --- |
| **Funding source** | **Funding amount** |
| Cornell Cup | $1500 |
| Purdue University | $300 |

## Mid-Review

## Process Understanding

# Recommendations and Next Steps

# Glossary

* **Augmented reality**: a system which overlays a computer generated image on top of real world objects so that both are simultaneously perceptible
* **Global positioning system (GPS)**: a sensor which receives signals from a global network of satellites to determine absolute position on the Earth’s surface
* **Electromagnetic interference (EMI)**: a disturbance in an electrical circuit induced by electromagnetic radiation from other circuits in proximity
* **Equivalent series resistance (ESR)**: the non-ideal resistance of real electrical components, acting as if this resistance value was placed in series with the device
* **Inertial measurement unit (IMU)**: a sensor which reports velocity, orientation, and gravitational forces
* **Inter-integrated circuit interface (I2C)**: a two-wire bidirectional synchronous serial communications system which connects multiple *master* and *slave* electronic parts through an addressable protocol
* **Jitter**: the unwanted, often random, timing deviation of a periodic waveform from its expected period
* **Printed circuit board (PCB)**: a substrate which supports and connects electrical components through an etched network of conductive traces
* **Serial peripheral interface (SPI)**: a fast four-wire bidirectional synchronous serial communications protocol used to interface two or more electronic parts
* **Turnkey startup**: the ability of a device on power-up to autonomously enter a state where a non-technical user can continue use of the device
* **Virtual reality**: a system which fills the user’s entire field of view with a computer generated image to the exclusion of the real world

# References

|  |  |
| --- | --- |
| [1] | M. I. Olsson, M. J. Heinrich, D. Kelly and J. Lapetina, "Wearable device with input and output structures". United States of America Patent 20130044042, 18 August 2011. |
| [2] | Nintendo, "How Long Will the Battery Remain Charged?," [Online]. Available: http://en-americas-support.nintendo.com/app/answers/detail/a\_id/385/p/430. |
| [3] | Sony, "Charging (PCH-1000 series)," [Online]. Available: http://manuals.playstation.net/document/en/psvita/basic/charge.html. |
| [4] | Oculus VR, "Development Kit 2," [Online]. Available: http://www.oculusvr.com/dk2/. |
| [5] | Google, "Tech Specs," [Online]. Available: https://support.google.com/glass/answer/3064128?hl=en. |
| [6] | B. Thomas, B. Close, J. Donoghue, J. Squires, P. De Bondi and W. Piekarski, "First Person Indoor/Outdoor Augmented Reality Application: ARQuake," *Personal and Ubiquitous Computing,* vol. 6, pp. 75-86, 2002. |
| [7] | M. Perenson, "Augmented-Reality Glasses: Startup’s Vision Could Change Gaming," 22 May 2013. [Online]. Available: http://www.technewsdaily.com/18143-augmented-reality-glasses-startups-vision-could-change-gaming.html. [Accessed 4 September 2013]. |
| [8] | STmicroelectronics, "STM32F4DISCOVERY Discovery kit for STM32F407/417 lines - with STM32F407VG MCU," [Online]. Available: http://www.st.com/web/catalog/tools/FM116/SC959/SS1532/PF252419. [Accessed 9 August 2013]. |
| [9] | Terasic Technologies Inc., "Terasic - News & Events - DE2i-150 FPGA Development Kit," [Online]. Available: http://www.terasic.com.tw/cgi-bin/page/archive.pl?Language=English&CategoryNo=11&No=529&PartNo=2. [Accessed 22 January 2014]. |
| [10] | SparkFun Electronics, "9 Degrees of Freedom - MPU-9150 Breakout - SEN-11486 - SparkFun Electronics," [Online]. Available: https://www.sparkfun.com/products/11486. [Accessed 11 October 2013]. |
| [11] | SGS-Thomson Microelectronics, "L3GD20 and LSM303DLHC 9-axis module for a standard DIL socket," November 2011. [Online]. Available: http://www.st.com/st-web-ui/static/active/en/resource/technical/document/data\_brief/DM00041389.pdf. [Accessed 8 October 2013]. |
| [12] | SparkFun Electronics, "9 Degrees of Freedom - Sensor Stick," [Online]. Available: https://www.sparkfun.com/products/10724. [Accessed 7 October 2013]. |
| [13] | ADH Technology Co. Ltd., "Data Sheet / GP-635T," 23 May 2012. [Online]. Available: https://dlnmh9ip6v2uc.cloudfront.net/datasheets/Sensors/GPS/GP-635T-121130.pdf. [Accessed 25 September 2013]. |
| [14] | Nordic Semiconductor, "nRF24L01 - 2.4GHz RF - Products - Nordic Semiconductor," July 2007. [Online]. Available: http://www.nordicsemi.com/eng/nordic/content\_download/2730/34105/file/nRF24L01\_Product\_Specification\_v2\_0.pdf. [Accessed 26 September 2013]. |
| [15] | Adafruit Industries, "NTSC/PAL (Television) TFT Display - 4.3 Diagonal ID: 946 - $49.95 : Adafruit Industries, Unique & fun DIY electronics and kits," [Online]. Available: https://www.adafruit.com/products/946. [Accessed 6 September 2013]. |
| [16] | Adafruit Industries, "HDMI 4 Pi - 5.6 Display 1280x800 (720p) Kit - HDMI/VGA/NTSC/PAL ID: 1666 - $139.95 : Adafruit Industries, Unique & fun DIY electronics and kits," [Online]. Available: https://www.adafruit.com/products/1666. [Accessed 8 January 2014]. |
| [17] | Adafruit Industries, "HDMI 4 Pi: 5 Display no Touchscreen 800x480 - HDMI/VGA/NTSC/PAL ID: 1678 - $64.95 : Adafruit Industries, Unique & fun DIY electronics and kits," [Online]. Available: https://www.adafruit.com/products/1678. [Accessed 5 January 2014]. |
| [18] | Pololu Robotics and Electronics, "5V Step-Up Voltage Regulator U3V12F5," Pololu Robotics and Electronics, [Online]. Available: http://www.pololu.com/catalog/product/2115. [Accessed 29 September 2013]. |
| [19] | Micrel Inc., "500 mA-Peak Output LDO Regulator," June 2009. [Online]. Available: http://www.micrel.com/\_PDF/mic5219.pdf. [Accessed 29 September 2013]. |
| [20] | SparkFun Electronics, "Polymer Lithium Ion Battery - 6Ah - PRT-08484 - SparkFun Electronics," [Online]. Available: https://www.sparkfun.com/products/8484. [Accessed 5 September 2013]. |
| [21] | Adafruit Industries, "Lithium Ion Battery Pack - 3.7V 6600mAh ID: 353 - $29.50 : Adafruit Industries, Unique & fun DIY electronics and kits," [Online]. Available: http://www.adafruit.com/products/353. [Accessed 4 September 2013]. |
| [22] | OSH Park, "Pricing and Specifications," [Online]. Available: http://oshpark.com/pricing. [Accessed 12 March 2014]. |
| [23] | Raspberry Pi Foundation, "Mechanical Profile – 3D models," [Online]. Available: http://www.raspberrypi.org/phpBB3/viewtopic.php?f=2&t=4402. [Accessed 3 October 2013]. |
| [24] | SkyTraq Technology, "Venus638FLPx GPS Receiver," Feburary 2010. [Online]. Available: http://dlnmh9ip6v2uc.cloudfront.net/datasheets/Sensors/GPS/Venus/638/doc/Venus638FLPx\_DS\_v07.pdf. [Accessed 4 October 2013]. |
| [25] | Digi Corporation, "XBee® Pro 900-HP Datasheet," August 2013. [Online]. Available: http://www.digi.com/pdf/ds\_xbeepro900hp.pdf. [Accessed 5 October 2013]. |
| [26] | Cypress Semiconductor Corporation, "AN1168: High-Speed USB PCB Layout Recommendations," November 2011. [Online]. Available: http://www.cypress.com/?docID=32407. [Accessed 19 October 2013]. |
| [27] | B. Suppanz, "PCB Trace Width Calculator," Advanced Circuits, 2007. [Online]. Available: http://www.4pcb.com/trace-width-calculator.html. [Accessed 19 September 2013]. |
| [28] | SGS-Thomson Microelectronics, "Ultra-low-power 32-bit MCU ARM®-based Cortex®-M3, 128KB Flash, 10KB SRAM, 2KB EEPROM, LCD, USB, ADC, DAC," 21 February 2013. [Online]. Available: http://www.st.com/st-web-ui/static/active/en/resource/technical/document/datasheet/DM00078075.pdf. [Accessed 25 February 2014]. |
| [29] | Texas Instruments, "3A SIMPLE SWITCHER® Power Module for High Output Voltage," October 2013. [Online]. Available: http://www.ti.com/lit/ds/symlink/lmz14203h.pdf. [Accessed 4 March 2014]. |
| [30] | AA Portable Power Corporation, "Custom LiFePO4 26650 Flat Battery Pack: 19.2V 3.3Ah ( 63.4Wh, 7A rate ) with PCB (5.9)," 2013. [Online]. Available: http://www.batteryspace.com/custom-lifepo4-26650-flat-battery-pack-19-2v-3-3ah-63-4wh-7a-rate-with-pcb-5-9.aspx. [Accessed 20 January 2014]. |
| [31] | Dallas Semiconductor, "DS2782 Stand-Alone Fuel Gauge IC," May 2009. [Online]. Available: http://datasheets.maximintegrated.com/en/ds/DS2782.pdf. [Accessed 6 February 2014]. |
| [32] | "Irrlicht Engine - A free open-source 3D engine," [Online]. Available: http://irrlicht.sourceforge.net/. [Accessed 16 March 2014]. |
| [33] | H.-K. Nienhuys, "Contrast - Lagom LCD test," 27 March 2008. [Online]. Available: http://www.lagom.nl/lcd-test/contrast.php. [Accessed 6 February 2014]. |
| [34] | Digi Corporation, "XBee® Pro 900-HP Datasheet," August 2013. [Online]. Available: http://www.digi.com/pdf/ds\_xbeepro900hp.pdf. [Accessed 5 October 2013]. |
| [35] | SGS-Thomson Microelectronics, "ARM Cortex-M4 32b MCU+FPU," September 2011. [Online]. Available: http://www.st.com/st-web-ui/static/active/en/resource/technical/document/datasheet/DM00037051.pdf. [Accessed 11 June 2013]. |

# Acknowledgements

The Cornell Cup team Augmented Reality Simulator would like to recognize the following individuals and groups for their contributions to this project.

Dr. Mark C. Johnson offered to advise the team and dedicated his time to logistics, procuring funding, and organizing status meetings to update the timeline and keep the project on schedule.

Blaine Gardner and Aditya Balasubramanian, as members of the previous Purdue Cornell Cup entry “The Incredible HUD”, advised the team about competition entry requirements, the mid-review process, and documentation.

Dr. David G. Meyer and his team of Senior Design teaching assistants, including George Hadley and George Toh, provided the team with substantial technical insight and evaluated the initial documentation for the Phase 1 senior design project.

The Purdue Electrical and Computer Engineering (ECE) department provided funding for the completion of Phase 2 and countless hours of education for all members of the team to lay groundwork for even attempting a project of this scale.

Joseph Boettcher and the staff of the Purdue ECE instrument room generously handled parts and printed circuit board ordering and gave additional technical insight. Joseph also participated in a mock Mid-Review session to evaluate improvements to the project.

Dr. Karthik Ramani provided us with the means to compare our design against Google Glass and the Oculus Rift in his lab for the purposes of the Performance Metrics in section 4.1 and the Brightness and Contrast test in section 12.1.8.

# Survey Responses

# Appendix

## Test Plans

### PCB Continuity and Functionality Test

|  |  |
| --- | --- |
| **Purpose** | Verify that a custom PCB has been correctly manufactured |
| **Results** | Are components connected to their intended destinations? (Pass/Fail)  Are there unwanted connections between components? (Pass/Fail) |
| **Materials** | Electrical resistance tester, board under test |

Manufacturing errors during fabrication can occur even on boards with generous tolerances. Each board is thus tested with an electrical resistance checker in locations where errors are most likely or most critical, particularly near the power supply and around the microcontroller. Both positive tests to ensure that all intended components are connected and negative tests to ensure that unwanted connections were not present must be conducted. One example of a failure that this test is designed to detect is shown in Figure 12.

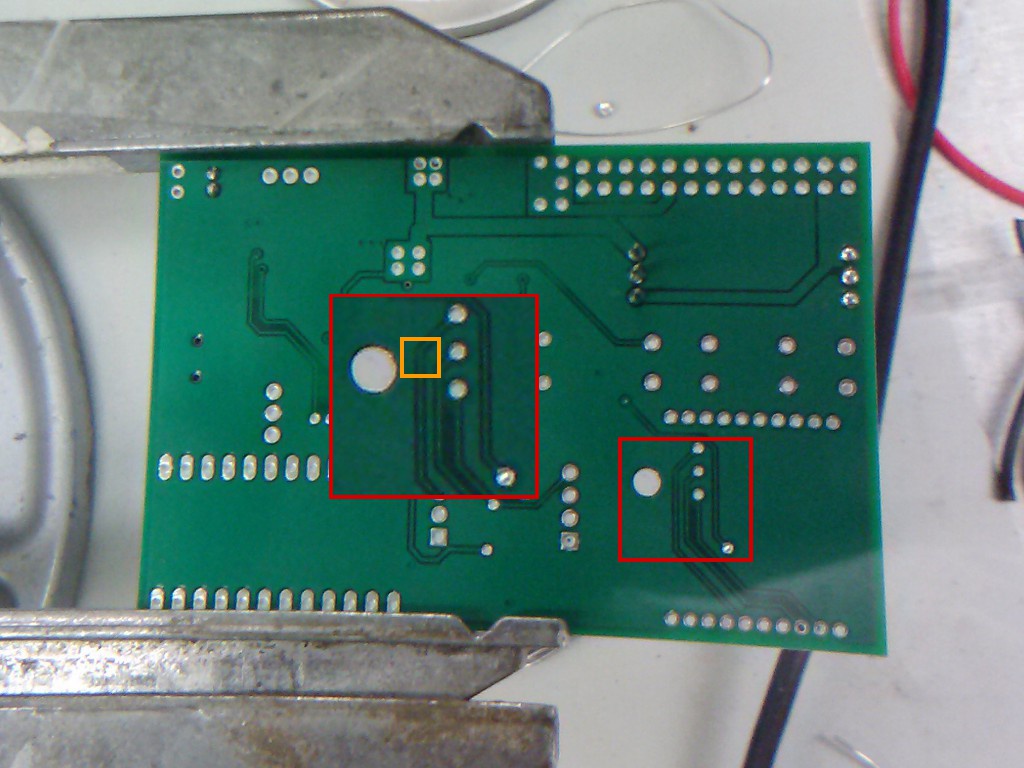


Figure : The PCB continuity and functionality check uncovered a hairline short in the highlighted location which could lead to overheating or fire if uncorrected

### Power Supply Burn-In

|  |  |
| --- | --- |
| **Purpose** | Verify that the power supply components of the design function as intended |
| **Results** | Is power supply voltage within 5% of nominal value? (Pass/Fail)  Is power supply voltage within 5% of nominal value after 24 hours of operation? (Pass/Fail) |
| **Materials** | Adjustable power supply capable of 0-20 V DC at 1 A, electrical voltage meter which can measure up to 20 V, 10 Ω 20 W power resistor |

As the power supply is involved with every other component in the design, functionality of this critical part is essential to the headset’s operation. A reliability test of the power supply can be conducted by wiring up an adjustable power supply, voltage meter, and power resistor in the configuration shown in [FIGURE].

To conduct this test, turn on the electrical voltage meter and set it on the appropriate voltage range. Then, turn on the power supply and set the output voltage to match the nominal battery voltage of the device under test. Measure the voltage displayed as shown in Figure 13 to determine the percentage error from the desired output voltage. The devices used in this project can tolerate a variation of ±5%.



Figure : Power supply burn-in test in progress with the current output voltage displayed

In addition, running the test for at least 24 hours is recommended to detect issues that may stem from overheating of the power supply. After this time period elapses, the output voltage should remain within the ±5% tolerance.

### Total Runtime Test

|  |  |
| --- | --- |
| **Purpose** | Measure headset performance on the *Power* metric in section 4.1 |
| **Results** | Average headset runtime on a full battery charge (hours) |
| **Materials** | Two headsets, battery charger, stopwatch |

Prior to conducting the test, the headset is powered OFF using its external power switch and charged for at least 12 hours using its battery charger. This ensures that the test is always starting with a consistently full battery when performing repeated trials.

The headset is then turned on, and a stopwatch is started to monitor elapsed time. The device under test is connected wirelessly to another headset which is plugged into its battery charger, and a simulation is executed as shown in section 5.4.1. A tester then uses the headset normally until the screen goes blank or the battery status indicator displays that the battery is empty. The timer is then stopped, and the time elapsed is recorded as the headset’s total runtime. Results often depend on the temperature and random variation, so this test should be run multiple times to obtain a more reliable average runtime.

### Wireless Range Test

|  |  |
| --- | --- |
| **Purpose** | Measure headset or wireless module (for purpose of component selection) performance on the *Wireless* metric in section 4.1 |
| **Results** | Average wireless communications range in line of sight conditions (meters) |
| **Materials** | Two headsets or wireless modules under test, open area with landmarks and a scale map to estimate location |

One wireless device is configured through software to reply to all packets that it receives. The second device sends a packet once per second, and should be given to a mobile tester which can walk across the open area. This test is limited to testing line of sight (LOS) range, as it is difficult to estimate the distance traveled across an area with many obstacles.

To perform the test, the mobile tester should slowly walk in a straight line away from the starting point, while the headset at the starting area is replying to packets received from the moving headset. When replies are no longer displayed on the mobile headset, note the location where the last successful reply was received using landmarks at the edge of the test site. After conducting tests in several different directions, each point is plotted on a scale map as is shown in Figure 14 and the distance to the starting point measured and averaged to determine the wireless range.

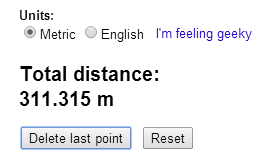
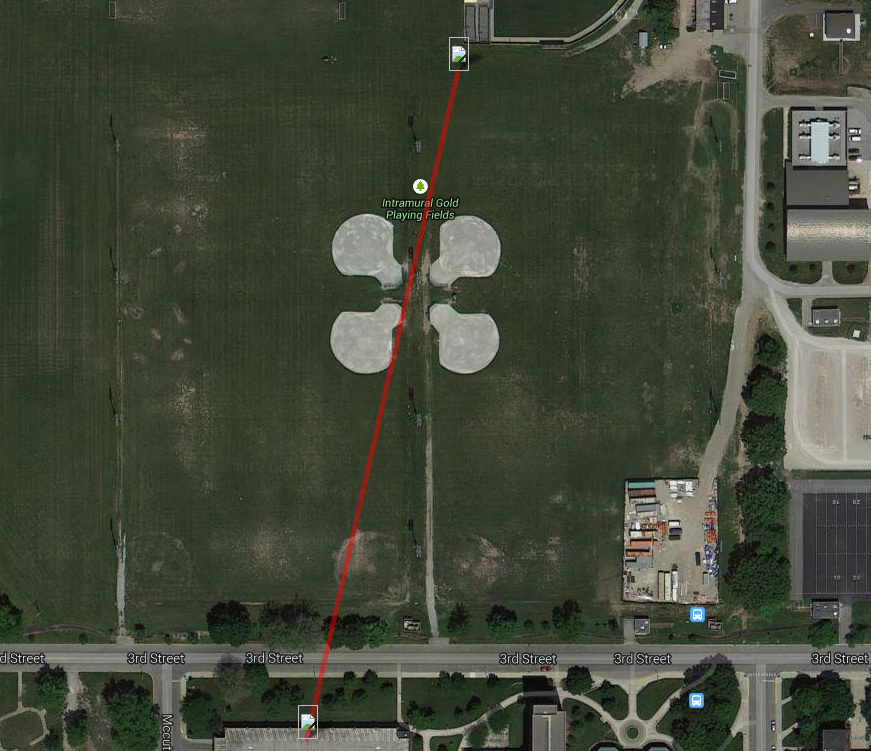


Figure : Wireless range test results, plotted on a map which indicates a range of 311.3 m

### Wireless Latency Test

|  |  |
| --- | --- |
| **Purpose** | Measure wireless communications latency for purposes of component selection |
| **Results** | Average wireless round-trip latency (milliseconds) |
| **Materials** | Two wireless modules under test, open area |

This test is fairly simple but factors into the important metric of network update rate. As each headset moves, its data must be sent to the host headset, and the reply containing other headset positions received. A long latency would make the virtual simulation appear slow or blocky, degrading the user experience.

To perform this test, the software on one wireless module was configured to send a packet to the other module, while the other module was configured to reply to all packets received just like in the range test in section 12.1.4. The first module was assigned to calculate the local time difference between the packet being sent and the reply being received, giving an estimate of the wireless latency. On the software designed for this device, the Linux ping utility could be used to perform this test automatically. A minimum of 25 packets need to be tested, as random variance can affect the values of each trial. In addition, the test must be performed at several different ranges, as the latency can be affected by the distance between the wireless modules.

### GPS Total Accuracy Test

|  |  |
| --- | --- |
| **Purpose** | Measure headset or GPS module (for purposes of component selection) performance on the *Location Accuracy* metric of section 4.1 |
| **Results** | Estimated geospatial location accuracy (meters) |
| **Materials** | Headset or GPS unit under test, reference route with scale map |

The software on the headset under test must be configured to log each location data point received from the GPS module into a file. The headset is then powered on at the starting location. Once the GPS module obtains a position fix, the tester carries the headset while walking along a reference route at a consistent speed. The reference route should be chosen so that it will be easy to visibly plot on an overhead view scale map, ideally along sidewalks, streets, or buildings. In addition, the reference route should include open areas and locations with trees or structures to test GPS performance in a variety of conditions, but must not be a simple straight line and should include a variety of directions.

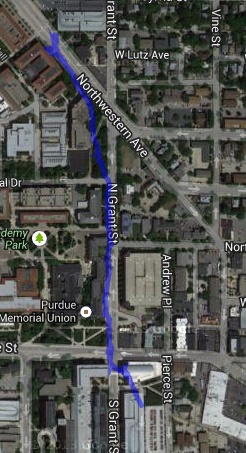


Figure : Total accuracy test data plotted as a blue line on a satellite map for the purposes of comparison to the known reference route

Figure 15 shows an example route plotted from the results of this test, using the headset and the chosen Venus 638FLPx receiver. This route spanned about 1.2 km on Purdue University’s main campus.



Figure : A location where the GPS unit maintained excellent performance despite interfering trees; the reference path is shown by the red line, while the blue line shows the computed path

For the majority of this test, the GPS accuracy was within 2 meters of the reference route. One example is shown in Figure 16, where the trees did not prevent the GPS from accurately tracking the user’s position.

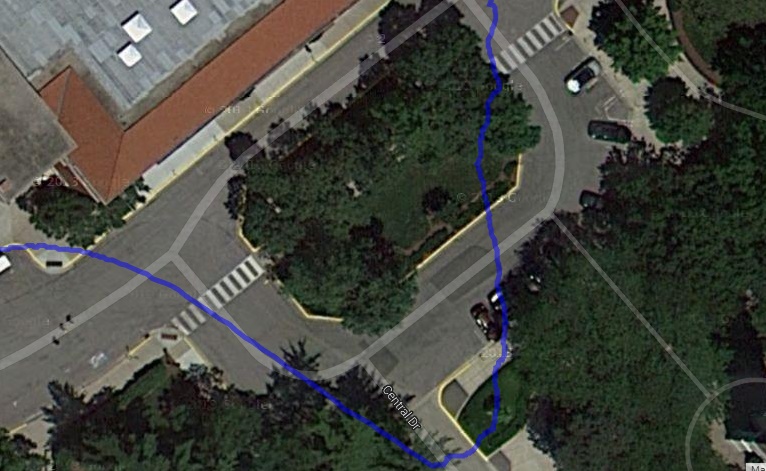


Figure : A location where the GPS unit displayed worse than its typical accuracy; the reference path is shown by the red line, while the blue line shows the computed path

However, in a few locations, the GPS sensor accuracy was worse than 2 meters, such as in the area shown in Figure 17. While calculating the actual average distance to the reference route is very difficult, the estimated GPS accuracy was determined to meet the 2 meter metric.

### GPS Relative Accuracy Test

|  |  |
| --- | --- |
| **Purpose** | Measure headset or GPS module (for purposes of component selection) performance on the *Location Precision* metric of section 4.1 |
| **Results** | Estimated geospatial location precision (meters) |
| **Materials** | Headset or GPS unit under test, small open area |

This test measures the *precision* of the GPS, or its repeatability in small areas. This test can be more important for many augmented reality applications than GPS *accuracy*, as most augmented reality environments are limited to just a few hundred square meters.

To calculate GPS precision, turn on the device under test in the area allocated for this test and allow the GPS sensor to obtain a position fix. Note the starting latitude and longitude coordinates of the device while it is stationary. Then, travel in a loop around the open area with the headset, being careful to return the headset to the exact position that it began. Note the ending latitude and longitude, and convert these coordinates to an X and Y position using the equation in section 5.4.3. The location precision can then be calculated by determining the distance from the original position:

As random variation plays a significant role in the result of this test, the average location precision should be calculated as the average of at least five trials.

### Brightness, Contrast, and Color Reproduction Test

|  |  |
| --- | --- |
| **Purpose** | Measure the quality of display devices for component selection purposes |
| **Results** | [Red, Green, Blue, Cyan, Magenta, Yellow] color reproduction (# distinct shades)  Brightness (# bars visible)  Contrast (# distinct bars visible) |
| **Materials** | Devices under test, headset, outdoor sunny area |

In order to perceive a realistic 3-D image, the brightness and contrast of the display when viewed in the context of the augmented reality device must be clearly visible to the user. Each display should be tested by displaying the web page available at [33] on the device under test while it is mounted on a headset. The tester should look at the display through the headset’s optics as shown in Figure 18 to evaluate the appearance of the display in its intended application.

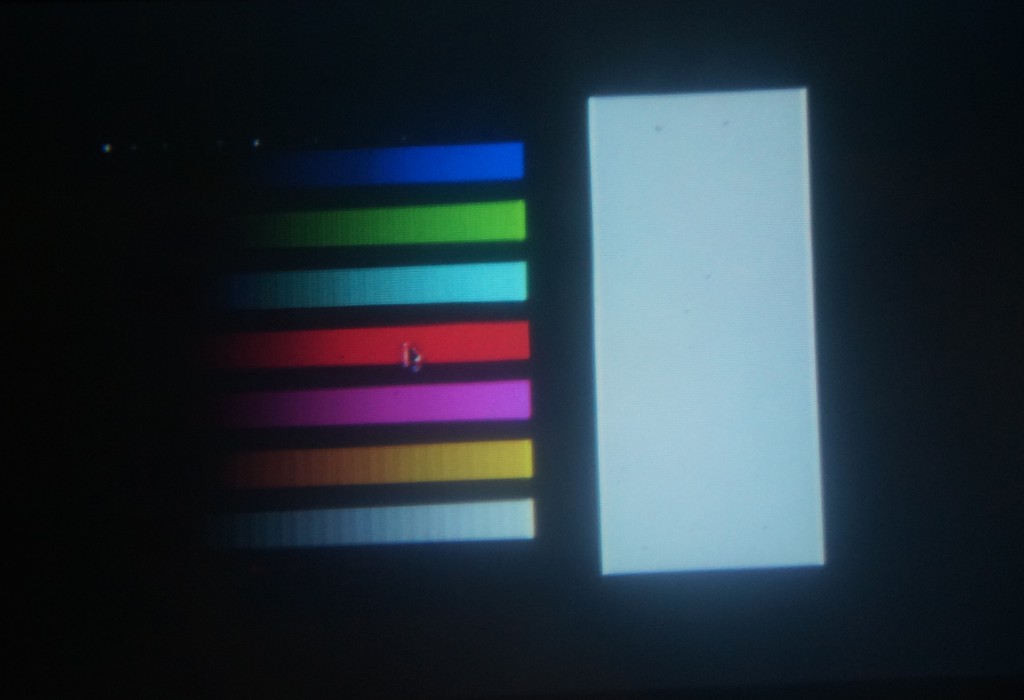


Figure : Contrast image displayed on a headset under test

When viewing the image, the display’s brightness is evaluated by counting the number of bars shown on the lowest grey strip that are distinct from the background. More visible bars indicate a brighter display which makes the virtual world easier to see for the user and decreases eye strain. Contrast is calculated as the number of distinct shades visible in the same grey strip, where more visible shades indicates a better difference between bright and dark. Individual colors can also be examined using this test using the other color strips; the number of bars visible, and their appearance compared to other colors on the screen, indicates the faithfulness of the display’s color reproduction.

### Maximum Simultaneous Devices Test

|  |  |
| --- | --- |
| **Purpose** | Extrapolate the value of the *Simultaneous Devices* metric of section 4.1 from a limited number of constructed headsets |
| **Results** | Calculated number of simultaneous devices supported by the simulation (#) |
| **Materials** | Two headsets under test, stopwatch |

It is extremely impractical to build enough headsets to actually determine the maximum number of simultaneous devices that can participate in one simulation. As the traffic caused by each headset is fairly constant, the value can be estimated by determining the traffic caused by just two headsets with knowledge of the maximum sustained wireless bandwidth.

This test is performed by first configuring a simulation as described in section 5.4.1, but stopping at the screen prompting to start the simulation. On the host device, a command must be executed which prints out the total number of bytes sent over the wireless link; for the software developed for this device, the Linux command ifconfig should be executed in a terminal. After noting this value, a stopwatch is then started when the simulation begins with the host selecting the *Start Simulation* option.

After one minute has elapsed, the host should end the simulation, and the total number of bytes is again displayed. The maximum number of simultaneous devices can be calculated using the following equation, where is the wireless bandwidth found in the manufacturer’s datasheet:

**Example**: For the Phase 1 headset, the XBee® Pro 900 datasheet [34] states a maximum bandwidth of 200 Kbps or 25 KB/s. The Phase 1 *Pac-Man* simulation was tested, and the test found and , to the nearest kilobyte. Therefore, the calculation proceeds as follows:

This test is also subject to random variance, but the one minute duration helps compute a more reliable average value. In reality, the theoretical wireless bandwidth may not be attainable, so this test may overestimate the maximum number of simultaneous devices metric.

## Assembly and Construction

Constructing a headset for this project is a fairly simple but quite laborious process. At a high level, the basic steps are to mount a screen and reflector to a suitable enclosure, construct and mount to the helmet a set of aluminum mounting rails, mounting the display assembly to the mounting rails, and then constructing and mounting a sensor enclosure to the rails.

### Materials required

Table 15 shows the materials required to assemble the headset packaging.

Table : Materials required for assembly of headset packaging

|  |  |
| --- | --- |
| Material | Quantity |
| Hard Hat | 1 |
| Angle aluminum, 1/8” thickness, ½” leg | 3 feet |
| Project Box, 4”x6”x2” | 1 |
| Project Box, 5”x2.5”x2” | 1 |
| Zip ties | 25 |
| Screws | 10 |

### Rail Assembly

Due to the difficult of mounting hardware directly to a hard hat, a system of rails must first be mounted to facilitate the rest of construction. The material used will be aluminum angle bar with 1/8” thickness and ½” length.

### Display Assembly

This part will vary quite a lot depending on the type of display being used and how it will need to be mounted. We have created headsets using two different displays and the procedures for each were drastically different.

### Reflector

To achieve the desired augmented reality effect, a sheet of semi-transparent, semi-reflective material is required. In this case it will be a sheet of polycarbonate with a reflective film applied on one side, and a tint film applied to the other side to mitigate brightness issues. A sheet of approximately 6”x4”x1/8” must be cut out. Several holes must be drilled for mounting purposes. Two will be near the bottom corners, which will be used for wire loops to hold the reflector at the correct angle. Four more will be along the top edge, for hinges which secure the reflector to the front of the headset. After the sheet has been cut to the correct size and the holes drilled, a sheet of reflective film the same size as the plastic needs to be cut out and applied as per the directions supplied with film. Then the tint film is to be applied to the other side in the same way. Be mindful that the reflective side needs to be facing the user once everything is assembled.

### Sensor Assembly

The sensors used to measure the user’s location and head orientation reside on their own circuit board which must be mounted to the headset. This is accomplished by mounting the PCB inside of its own enclosure and mounting that enclosure to the rear portion of the rail assembly on the headset. The enclosure used will be a 5”x2.5”x2” ABS project box.

### Mounting the PCB to the enclosure

In order to mount the PCB to the enclosure, several holes will first need to be drilled.

### Mounting the enclosure to the rail

## Failure Mode Worksheet

A failure modes, effects, and criticality analysis (FMECA) worksheet examines the possible failure modes of the device given a failure of one of its parts. For each potential failure, the effects and criticality is stated. A high-criticality failure is likely to injure the user, while a failure which causes the device to simply cease operating is low criticality. Medium criticality failures may involve a display that appears functional but distracts the user with flashing images or intermittent failures which may not be obvious to a non-technical user. Failure modes examined for this device are shown in Table 16 and Table 17.

Table : FMECA worksheet for processing units and sensors

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Failure Mode** | **Possible Causes** | **Effects** | **Detection Method** | **Criticality** |
| Controller crashes | Failure of C1-C13, R2, Y1, U1 | Microcontroller computes invalid data | Corrupt display | Medium |
| Graphics processor crashes | Failure of JP7, U5, components on motherboard | Motherboard produces invalid image | Corrupt display | Medium |
| Controller fails to start | Failure of C1-C13, R2, Y1, U1 | Microcontroller sends no data to motherboard | Blank or static display | Low |
| Graphics processor fails to start | Failure of JP7, U5, components on motherboard | Motherboard does not produce an image | Blank display | Low |
| Display does not function | Failure of display driver board or cables | Image is not displayed to the user | Blank display | Low |
| IMU data invalid or missing | Failure of U6, U7, C12-C13, C19 | Calculated user head orientation is incorrect | Invalid display, virtual heading is wrong | Low |
| GPS data invalid or missing | Failure of JP4, JP5, U4, GPS unit | Calculated user geospatial position is incorrect | Invalid display, virtual position is wrong | Low |

Table : FMECA worksheet for power supply and battery charging

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Failure Mode** | **Possible Causes** | **Effects** | **Detection Method** | | **Criticality** |
| VIN shorted to GND | Failure of U6, U5, U4, battery protection circuit | Device overheats; fire or explosion may occur | Observation | | High |
| VUSB shorted to VBAT | Failure of U3 and battery protection circuit | Battery is overcharged; fire or explosion may occur | Observation | | High |
| VADC > 3.3 V | Failure of U6, VIN shorted to VADC | IMU reports invalid values or is damaged | User head rotation is ignored | | Low |
| VCORE > 3.3 V | Failure of U5, VIN shorted to VCORE | Motherboard crashes or is damaged | Blank or static display | | Low |
| VMCU > 3.3 V | Failure of U6, VIN shorted to VMCU | Microcontroller crashes or is damaged | Blank or static display | | Low |
| Battery monitor fails | Failure of U2 | Reported battery charge is incorrect | Battery charge indicator invalid | Low | |