Project AUGI

Jonathan A. Pedrosa, Adam Ilter, Phillip Lee and Zulkafil Ahamed

Department of Electrical Engineering and Computer Science, University of Central Florida, Orlando, Florida, 32816-2450

***Abstract* — This paper illustrates the different design elements needed to create an augmented reality navigation experience. These elements include (1) Sensor support for determining position and orientation of the navigation device; (2) Integration of sensors into a single hardware device and their communication protocols with the host device; (3) Augmented Reality Navigation Application to provide the user with a new means of navigating to their destination with simple visual cues. This paper will also highlight the design decisions made and the motivations for making them.**

***Index Terms* — Computer applications, Computer interfaces, Message passing, Navigation, Portable computers, System-level design.**

I. Introduction

The goal of this project is to develop an enhanced navigation experience that goes beyond a simple two dimensional (2D) map, or turn by turn directions. To do this augmented reality is used to add visual aids to the user experience in order to more easily display directly to the user, the location of desired points of interest, and a visual path of how to get there. Augmented reality will be used to overlay visual cues on top of what the user is seeing on their device. This method will ensure that the user can easily confirm that what they see on the map directly conforms to the environment they see around them.

Navigation applications have often required that the user be able to understand and interpret a 2D map, understand their location on a map, and be able to identify significant landmarks identified on the map, such as cross streets, buildings or terrain. Even after the user has accomplished this task, assistance provided by the application to inform the user of where they need to go next on the path to their destination have also been limited to a line on a 2D map, or verbal step-by-step instructions. This is suitable for those who have a natural sense of direction, or an aptitude for map reading, but can be difficult for lay users. It can also be difficult for them to identify specified landmarks that are often labeled with vague outlines.

The inherent difficulties of navigating with a 2D map that displays locations with at best outlines of significant landmarks, or at worst a single point with a title are easy to see. Users can have a difficult time identifying locations on a map because they have no idea what the target destination is supposed to look like. GPS navigation can be very helpful in informing the user of their current location and direction of travel, but it can't point out to the user what they are looking for or how it looks like.

Even turn by turn instructions can be of limited usefulness because the world we live in does not lend itself to fit cleanly in a 2D X,Y space. Anyone who has traveled in an urban environment, especially one that is built up from an older city can understand the frustration of receiving an instruction to "Turn Right", and there be two or three right turn options because the intersection has five or six streets coming off it. Perhaps the intersection it multilevel, with one right turn passing under an elevated roadway that also has an off ramp to the right to get onto the elevated roadway.

These ambiguities can lead to lost time and frustration as the user is forced to guess what the correct path is and determine through process of elimination which one was correct. Signs can be unreliable as well, as they can be placed in hard to notice locations, or be damaged or missing, thus making it difficult to know if the user is at the correct cross street, or at the specified landmark that designates their next turn to their destination.

If a device could be designed and built that not only generated a path to a user's destination, but could actually highlight on the screen precisely what the user is looking for, these issues could be resolved. Imagine a device that when the user holds it up to the landscape they are looking at, the device draws virtual information to its screen that directly super imposes onto what the use is seeing in their landscape. This can enable the user to directly follow visual cues their destination, and be able to precisely identify what it is they are looking for.

With a device like this (See Figure 1), even the most novice of user would be able to follow the virtual image of say a line or arrow, without having to worry if they passed the street or sidewalk that they were supposed to turn on. They would not need to worry about identifying intermediate land marks along their path that mark the location of their next turn. They also wouldn't need to determine if they are heading the right direction every time they change heading.

To implement this augmented reality setup, an application will be developed that will integrate a 2D map interface, augmented reality navigation, Global Positioning System (GPS) location tracking, and customization of

TABLE I
SUMMARY OF SENSOR SPECIFICATIONS

Sensor	Sensor Specifications		Sensor Description		
	Input Type	Accuracy	Model	Manufacturer	Cost
GPS	GPS Satellite	2.5m	Venus634LPx	SkyTraq	$49.95
Magnetometer	Magnetic Fields	5 milli-gauss	HMC5883L	Honeywell	$14.95
Barometer	Air Pressure	1 Pascal	BMP085	BOSCH	$19.95
IMU	Gravity/Acceleration	250dpi/2g	MPU-6050	InvenSense	$39.95
Photocell	Ambient Light	1k Ohm	GL5528	100y	$1.50

points of interest options. The user will enter a destination and the application will plot a route from the user's current position to the desired destination.

Once a route is plotted, the user will be presented with a new screen that shows a live camera feed of the devices view, with a virtual route overlaid onto the feed. This way, when the user holds up the device, they will be able to see a representation of what they are seeing, with a virtual line drawn over it. The user now only needs to follow the virtual line to the desired destination, without the need to deal with the drawbacks described above.

Figure 1: Concept Image showing virtual overlay.

This application will be integrated into an electronic device that is portable, uses a common hardware platform, and a common operating system (OS) foundation. By building for an electronic device that is uses common components and software foundations, the developed navigation application should be eminently portable to other electronic devices such as smart phones or tablets.

To design this electronic device, we will begin with a hardware development kit as a foundation, and add any additional hardware functionality we may need in order to achieve the goals of this project. The hardware kit for our project revolves around the PandaBoard ES.

The PandaBoard ES is based on a system on chip (SoC), the OMAP4460 by Texas Instruments. It is designed to be low-power consumption, cost effective single-board computer for development. The OMAP4460 is a duel-core A9 CPU with a 384 MHz GPU, and has 1 GiB of DDR2 SDRAM. It will be the platform which the operating system, version 4.0.3 of Android, will run on. This board is mounted on the Panda expansion board which has a 7.0 inch LCD screen. A resolution of (800 by 480 pixels); with five point capacitive touch, and five user keys/buttons.

By starting with a hardware and software platform that is common to the industry, the sourcing of parts, design of the software application and the ability to customize the design will be made eminently easier. This includes integrating a microcontroller and sensors to the design. This also includes ensuring that the software application that is written can operate on any similar Android platform.

Figure 2: Pandaboard ES with expansion board

The design team for this project is composed of two senior electrical engineering students, and two senior computer engineering students. The electrical engineers will focus on the hardware aspects of this project, which will include designing an expansion board that adds hardware functionality to the base hardware, and providing a battery power supply to make the system portable. The computer engineers will focus on the software aspects of this project, including application development, setting up any low level communication between the hardware in the expansion board and the hardware in the base device and integration into the OS environment. They will also ensure proper OS to hardware operation.

II. Project Design Areas

A. Sensors

When deciding what sensors were needed for this project, the primary consideration centered on what would be needed to align a virtual image layer onto a live camera feed of what the user was seeing. For this purpose, not only would the devices position need to be known, but also it orientation heading, and altitude. It was also decided that it would be important to maintain a proper image brightness level in order for the user to be able to see the virtual overlay clearly. Therefore five sensors were chose for incorporation into a single add-on circuit board for this project. Their performance specifications are shown in Table I.

The first sensor will be a GPS sensor; which will be used to track the user's current position. GPS sensors typically are limited in their accuracy, especially when conditions are not ideal, such as being indoors. The sensor we chose for this project does have an accuracy rating of 2.5 meters, which should be sufficient for the needs of this project. [1]

The default settings for the GPS unit will be adjusted to enable support for the FAA's Wide Area Augmentation System (WAAS), a faster refresh rate (8 Hz), and hot start settings. A super capacitor of .2 F will be used to keep positional data in the GPS memory for up to seven hours after system shutdown. This will allow the GPS to quickly re-establish its connection to GPS satellites so long as power is restored before the capacitor runs out.

The GPS antenna chosen for this project is a patch antenna with a 2.0 db Passive Gain at zenith. It operates on the 1575.42 MHz band, and with an impedance of 50 ohm's. It will be attached to the top of the tablet device in order to ensure it points skyward as much as possible.

The next sensor we chose was a 3-Axis Magnetometer, which is needed to determine the user's directional orientation. Normally compasses need to be held level in order to operate normally. Even though the magnetometer we chose is 3-Axis, it still requires post processing on the host to generate a heading regardless of how the device is being held. This is an important factor to consider as the device can be expected to be held at a wide variety of angles. Creating what is known as a "tilt compensated" compass will be a complicated task, but necessary in order to ensure the virtual and camera images are aligned. Figure 3 illustrates this.

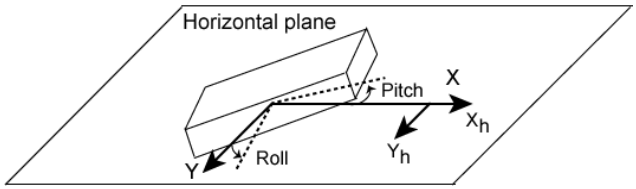


Figure 3. Tilt Compensated Angular inputs [2]

The next sensor we chose is known as an Inertial Measurement Unit (IMU). This sensor contains both a gyroscope and an accelerometer to determine device orientation and its motions. The outputs from this sensor will be post processed in order to generate Euler Angles which will illustrate the devices orientation in a spherical manner. This will be another vital component in aligning the virtual layers with the camera layers and will also be used in the calculations for the tilt compensated compass.

Another sensor that was included is a barometric sensor. This sensor measures air pressure and is used to determine the altitude that the device is at. In this way it becomes possible to tell how far from the ground (to 1ft accuracy) the device is, and will also allow the device to know what floor of a building it is on based on the local air pressure.

This will require a lot of post processing logic in order to account for local weather conditions, which can cause changes in air pressure. By collecting data over time, and judging the rate of air pressure changes, it should be possible to distinguish influences from things like the weather, from actual device movement up a flight of stairs.

Finally, the last sensor to be integrated will be a photocell that will allow us to adjust the screen brightness and the opacity of the virtual image that we will be overlaying on the camera feed, in order to make it as easy as possible for the user to view the images. It will be connected to the microcontroller's integrated voltage meter.

B. Microcontroller and Communications

All of these sensors will be connecting to a central microcontroller that will provide a single point of access for the application to go for the data it needs. The microcontroller chosen for this project is the IOIO Microcontroller. This microcontroller offers a lot of advantages for this project. There are a plethora of connection options, including multiple I2C protocol ports, Universal asynchronous receiver/transmitter (UART) communication ports, integrated analog to digital conversion ports, power supply ports for both 3.3V and 5V, and programmable boot loader.

This microcontroller also provides programming libraries in the Java programming language, which is ideal for integration to our Android Programming environment. This will allow us to program all of the communication protocols in Java, and provide the easiest path for the application to interface with the microcontroller. A final benefit is that this device comes with an Android driver, allowing it to manage all of the USB communications between the microcontroller and the host machine, relieving the project of this chore.

The IMU, Magnetometer and Barometer communicate to the microcontroller via the I2C interface. The IOIO libraries provide a simplified means of opening an I2C communication port and then transmitting and receiving data to the required sensor. All three sensors share the I2C port, with each sensor having a unique id associated with it. A data request message is then broadcasted on the I2C bus and received by all of the sensors. However only the sensor with the matching address contained in the broadcast will respond to it with the requested data.

This allows the implementation of a polling algorithm that will cycle among the sensors connected to the I2C bus, to collect the latest data from these sensors and pass them to the application. This requires a careful study of the datasheets for each sensor to learn the specific byte formats required to request data, and what data can be expected back in reply.

The GPS sensor is connected to two uni-directional UART ports, where it transmits a constant stream of ASCII characters over one, and can receive data over the other. The microcontroller passes the received characters to the application for parsing out of the relevant data. The application will then be able to use the data to determine the user's global position and allow the user to plot a destination from this position. The sensor can have its default settings adjusted by sending it strings of ASCII characters over the other UART line.

The final sensor is the photocell, which is hooked up to a port designed to measure the voltage of the line and report it to the application. IOIO has 16 pins, commonly known as "Analog Inputs", which are capable of measuring voltage levels between 0V-3.3V, with a precision of about 3mV. Since the photocell varies its resistance in relation to the light it receives, the voltage in the line will vary in a similar fashion, allowing the application to determine the brightness settings appropriate for a given amount of light.

All of these sensors and the microcontroller will be integrated onto a single circuit board which the project team will design. This circuit board will be responsible for receiving power from the batteries integrated into the device, and distributing it to the microcontroller, sensors, and the Pandaboard.

Figure 4 illustrates how the various sensors are connected to the microcontroller. The microcontroller will be connected to the Pandaboard ES via a USB cable, the communication protocols of which are handled via the IOIO Android driver. This will allow the application to interface easily with the IOIO microcontroller.

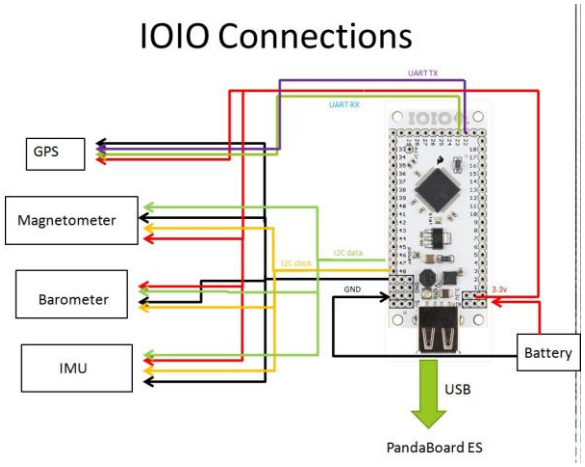


Figure 4. Sensor connections to IOIO Microcontroller

C. Circuit Board Design and Manufacture

Once the sensors and microcontroller are integrated together and fully prototyped, the design can be drawn up in electrical schematics. This will be done using the Easily Applicable Graphical Layout Editor (EAGLE) program which is available to students through UCF. This version of Eagle allows up to 999 schematic sheets, up to 16 signal layers in design, and can be 4m by 4m (routing

area). This is more than enough to fit our microcontroller and sensors.

Fortunately all of the purchased sensors for this project came with EAGLE files of their designs and the microcontroller did as well. This provided an excellent starting point for design the circuit board that was to integrate them. There was still a lot of rework to do however.

One big element to keep in mind was that getting the printed circuit board made is one thing, soldering all of the electronic circuits to the board is another. The design that came with the sensors and microcontroller was made with very small electrical components (402 surface mounts) which would be very difficult for the group to solder. All of these electrical components needed to be changed out for equivalent, larger versions that would be easier for the project team to solder (1206 surface mounts or through hole). Special assistance was needed to solder the integrated circuits of the sensors to the board which were of a ball grid array type.

A careful accounting of the power needs for the sensors, microcontroller and Pandaboard ES is needed also. All of the sensors combined with the microcontroller can draw up to 500mA and the Pandaboard with the LCD screen can draw up to 1 A. This comes too close to the 1.5 A max of the power regulator that came on the microcontroller board. As such a larger 3 A power regulator was integrated into the design, in order to ensure there are no issues with the power draw of the components.

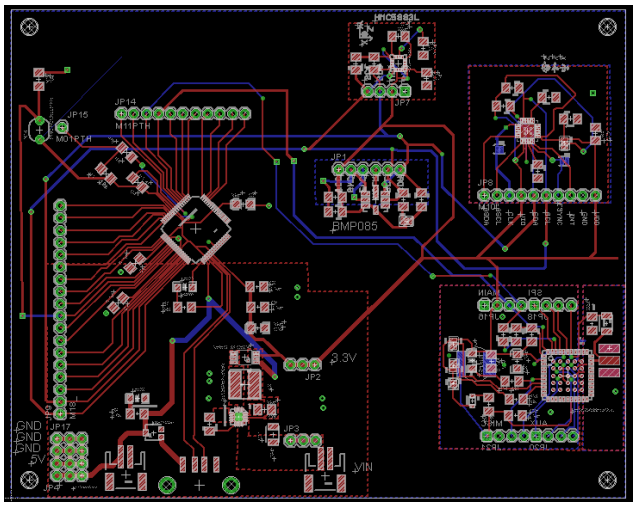


Figure 5. Circuit Board Design

D. Software Application

The application will seek to implement an augmented reality experience to help the user arrive at their destination. A virtual overlay will be displayed on top of a live camera feed in order to feed the user directional data on how to get to their destination. This overlay will seek to draw a virtual line over the desired path that will take the user to their destination. This means that the application will need to constantly pull in data from the onboard hardware sensors, and calculate its current orientation and heading, so that it can line up the virtual image with the camera feed. This way the virtual line will follow a road or sidewalk that the user sees in the camera.

This application will have several threads running in order to provide this experience. The first will be an IOIO thread, which will be implemented as a service in the Android OS. This service will run in the background, and constantly poll the sensors for updated data, and feed them to the main navigation and augmented reality activities which will make use of this data. Since Android only allows one activity at a time, implementing the IOIO protocols as a service was necessary, so that the other program activities have access to the sensors as needed. [5]

The next thread will be the Map Activity, which will run the 2d map that the user will use to enter a desired destination. This activity will need a constant stream of GPS data and a compass heading, in order to correctly display the user's current position on the map. Any destination entered by the user will have a path computed from the user's current position to the destination.

Google Maps is a built in mapping and routing application that is delivered with every Android device. It also has many other features such as: detailed 3D maps, positional based turn-by-turn navigation, public transit, real-time traffic updates, access to a large database filled with information on local businesses and establishments, and many more. This feature is going to be the heart of our Augmented Navigation Application. [4]

Once this path is plotted, a new activity will run that will start the live camera feed and take the plotted data from the map activity, and generate the virtual cues on top of the camera feed in order to guide the user. This will involve generating the 3d images as well. Since this will be a new activity, the Map Activity will be on pause while this activity is running. The IOIO service will continue to run however, which this activity will need a constant stream of data from. Figure 6 illustrates the software structure of this application.

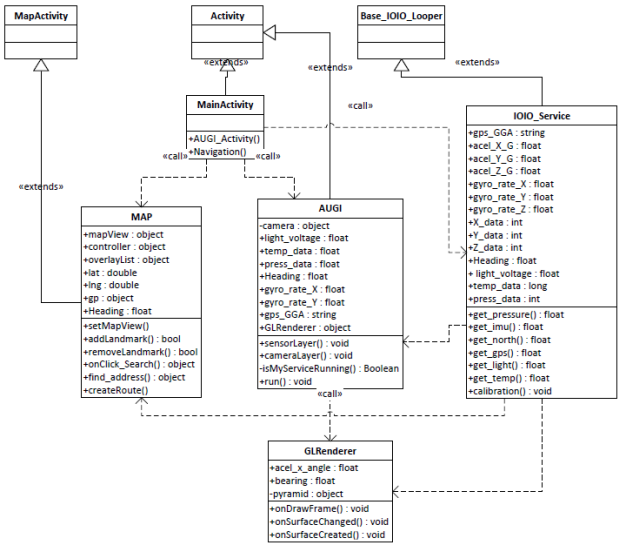


Figure 6: Application Class Diagram

All development work for this application was performed used Eclipse IDE with the Android SDK integrated in. This provided a familiar programming environment to generate all of the Java code this program required. It also was useful in enabling advanced debugging and testing options, either with the virtual Android device simulator built into the SDK, or by connecting a real Android device via USB and running the application live on the device, while debugging information passed back to the main console.

III. Project Prototyping

While the software application was being researched and developed, the hardware sensors and microcontroller needed to be prototyped in order to ensure they would work properly when integrated into a single board. For this purpose the five sensors were combined with a microcontroller onto a breadboard in order to build an initial representation of the board we intend to ultimately produce. Since each sensor came on its own mini circuit board with all the onboard circuits they needed, it made their installation to a breadboard a straightforward affair.

A breadboard is commonly used as a base for prototyping electronics. What is great about breadboards is that is does not require you to make soldering connections. This allows for a fast and convenient way to interchange parts and easy to isolate for debugging your circuits. Breadboard can also be reused after you finish your prototyping.

With this prototype board, all of the I2C communication protocols, the UART protocols, and the application interface with the IOIO microcontroller could be written and tested long before we have an actual, manufactured board. This required a careful study of all of the datasheets that came with each sensor in order to determine the structure of the data streams used to communicate with the sensors.

This also will allow the team to run early versions of the application on the board to gauge its development, and ensure that it is operating as intended in all stages of development. This will prove a critically important feature, as we don't want to do all of the project development on one device, only to not have it work on the main host device. Figure 7 shows an image of the prototype board used by the project during development.

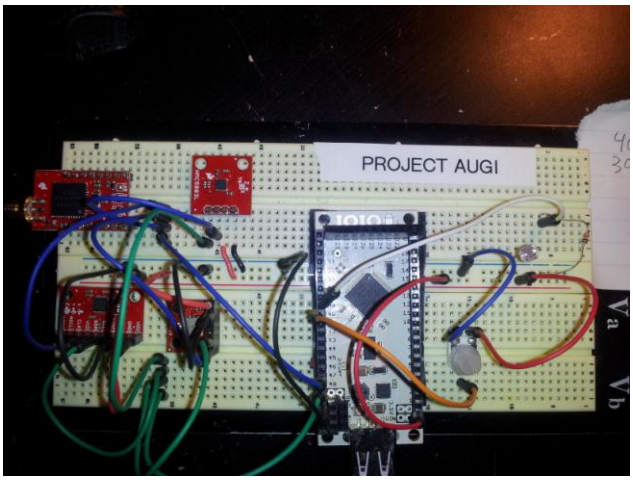


Figure 7: Prototyping breadboard of Custom Sensor Board

IV. Project Tablet Housing

The focus of this project is on PCB Construction and application programming. As such only a very basic and simple housing is anticipated at this time; basically something that can contain all of the electrical components, including the screen and battery and still be portable. Portable is a relative term however; this device isn't going to be an iPad! The dimensions will be 8.5in x 11in x 2in. Ventilation holes will be created to enable heat dissipation from the device. A device fan should be unnecessary, as the Pandaboard is composed of low power and low heat processing units that are normally used in fan-less tablets anyways.

In this design, the circuit components are mounted in a stacked manner. This is done in order minimize the square area needed to house the components in the X and Y plane. This does lead to a perceptible increase in thickness in the tablet, but this is not considered as a serious issue. The design does allow for

integrating all of the buttons that are normally seen on an android device, namely Home, Settings, Return, Volume Up, Volume Down, and a power reset switch.

In total the housing will contain three circuit boards at 4x5 inches, a 7 inch LCD touch screen, a 2200 mAh battery, the GPS antenna, and 6 buttons. The chosen construction material is Plexiglas, which will allow the interior parts to be easily seen and shown, without the need to disassemble the device. All Plexiglas joints will be chemically bonded together as is the normal practice when building with Plexiglas.

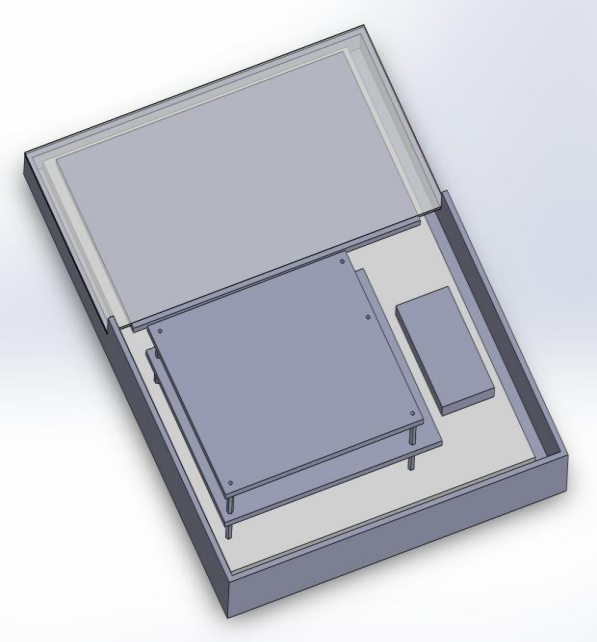


FIGURE 8: TABLET HOUSING MOCKUP

V. CONCLUSION

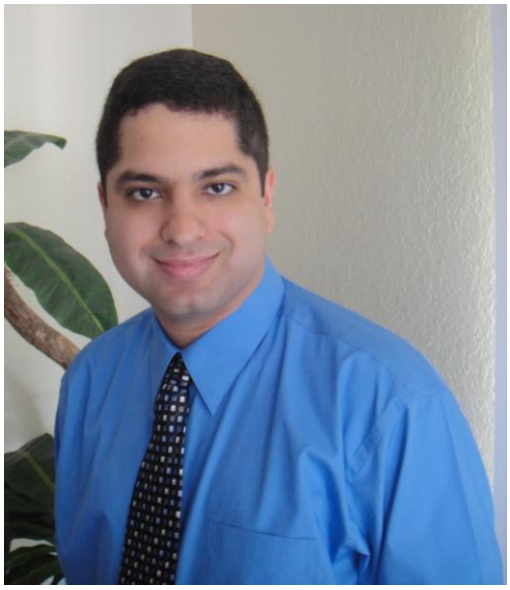
This project has proven to be a very ambitious undertaking; especially when one considers that the design team had never made an Android application before hand, not had ever designed something nearly as large or complicated at this design has proven to be. Nonetheless one by one, most of the technical challenges have been overcome. The device is able to determine its position and orientation, calculate a path to the desired destination, and display visual cue to the user on how to get to their destination.

This project proved to be very complete in that it operates on all levels of computer design. From the moment the bits are generated in the sensors, this project had to organize them, transport them to the host CPU, post process them, and feed them to a high level application for display to the user. It thus proved insightful to the lifecycle of computer data, and the sub-mechanisms of computing technology. The project group is proud to have been a part of this undertaking, and certainly learned a lot in the process!

VI. ACKNOWLEDGEMENT

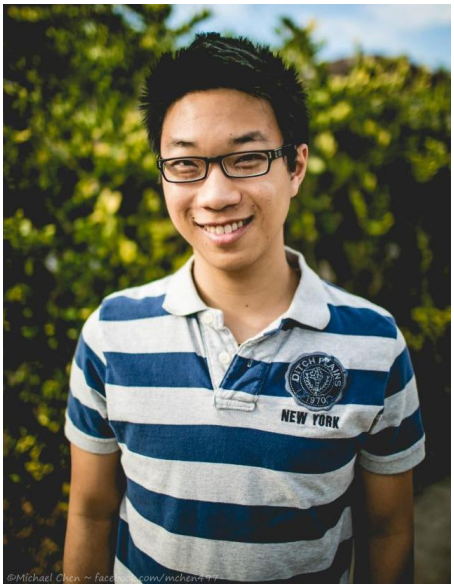
The authors wish to acknowledge the assistance and support of the United States Army Simulation and Training Technology Center for generously sponsoring this project to the amount of $2,000. Also the project group would like to thank Tesseract Sensors for allowing us the use of their facilities for soldering those very difficult integrated circuits. In addition the authors wish to thank their families for tolerating the very long hours put into this project to make it happen, which would not have been possible without their help and support.

VII. BIOGRAPHY



Jonathan A. Pedrosa: will graduate with a Bachelor's Degree in Computer Engineering. He has always had a deep interest in computing technology, and helped disassemble his first computer at age 10. He currently has a paid internship at the UCF Institute for Simulation and Training's - Simulation and Training Technology Center, where he has worked for over two years in the DARPA Cell providing research assistance to DARPA Projects . His post graduation plans include working as a Systems Engineer while seeking his Masters Degree in Computer Engineering and enjoying life with his wonderful family.

Adam Ilter: started at the University of Central Florida in 2008 after graduating from Cypress Bay High School in Weston, FL. His interest in the field of computers began in his middle and high school years where he programmed and built his own computers for fellow students and teachers. He always had an interest in computers, so there were no doubts when it came to picking his ideal major in college. As a college student, he became more involved with the software aspects of Computer Engineering and as a result, went on to intern with WAVE Corporation in Maitland, FL in 2012. There he was able to indulge himself with real-world application and software methodology. Adam's ambition is to one day own his own computer software/engineering firm where he can build his product from the ground up.



Phillip Lee: is a graduating senior with a bachelor's degree in Electrical Engineering. His focus is on software programming and embedded systems. In his collegiate year Phillip has interned with Hover Fly Technology and Tesseract Sensors. He is always making time for his passion to dance and the robotics club. In the years to come Phillip would like to pursue a Master in Computer Engineering.

Zulkafil U. Ahamed: is a senior at the University of Central Florida. He will be graduating with a B.S. in Electrical Engineering in December 2013. His interests include circuit analysis, power generation and wireless communication. He plans on pursuing a master's degree in Electrical Engineering in Spring 2014 with a focus in studying Power Electronics.

VIII. References

[1] https://www.sparkfun.com/products/11058
[2] https://www.sparkfun.com/products/10530
[3] Pfleeger, Shari. "Software Engineering" pg. 223-290. Prentice Hall 2010
[4] Deitel, Paul. "Android for Programmers" pg. 35-67. Pearson 2012
[5] Mednieks, Zigurd. "Programming Android" pg. 111-163. O'Reilly 2011