UNIVERSITY OF MIAMI

MULTIDIMENSIONAL WAVE FIELD SYNTHESIS CONTROL THROUGH WIRELESS HARDWARE

By

Robert Charles Rehrig

A THESIS PROJECT

Submitted to the Faculty of the University of Miami in partial fulfillment of the requirements for the degree of Master of Science in Music Engineering Technology

Coral Gables, Florida

May 2014
UNIVERSITY OF MIAMI

A Thesis Project submitted in partial fulfillment of
the requirements for the degree of
Master of Science in Music Engineering Technology

MULTIDIMENSIONAL WAVE FIELD SYNTHESIS CONTROL THROUGH
WIRELESS HARDWARE

Robert Charles Rehrig

Approved:

Dr. Christopher Bennett  Dr. Shannon K. de l’Etoile
Associate Professor of Music Engineering  Associate Dean of Graduate Studies

Dr. Colby N. Leider  Will Pirkle
Professor of Music Engineering  Associate Professor of Music Engineering
Using the Huygens-Fresnel principle, it is possible to synthesize a wavefront for a virtual sound source via an array of loudspeakers. This technique is known as wave field synthesis and it is a popular audio strategy used in virtual and augmented reality. Unfortunately, most applications require high costs devices to track a user for targeting. A low-budget method for controlling the audio source is proposed, incorporating a low-cost, low-power 9-degrees of freedom sensor board that wirelessly relays information to a remote system via Bluetooth Low Energy. The data can then be filtered and used to steer the virtual sound source in three dimensions to better target the user.
# TABLE OF CONTENTS

LIST OF TABLES ................................. vi
LIST OF FIGURES ............................... vii

CHAPTER

1 Introduction .................................... 1
  1.1 Focus and Aims ............................. 1
  1.2 Motivation .................................. 2
  1.3 Antipated Outcomes ........................ 2

2 Background .................................... 4
  2.1 Previous Work .............................. 4
    2.1.1 Similar IMU Designs .................. 4
    2.1.2 Audio Panning ......................... 6
    2.1.3 WFSDesigner ........................... 8
  2.2 Wave Field Synthesis ..................... 9

3 Proposed System ............................. 14
  3.1 Wireless Communication .................. 15
    3.1.1 Available Systems .................... 15
    3.1.2 Bluetooth Low Energy ................ 22
  3.2 Sensor Selection ........................... 26
  3.3 Other Necessary Parts .................... 28

4 Development and Implementation .......... 30
  4.1 Sensor Development ....................... 30
6 Conclusion

6.1 Future Work

6.1.1 Sensor Board

6.1.2 Wave Field Synthesis Control

6.2 Other Applications

6.3 Conclusion

LIST OF REFERENCES
LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Power and Efficiency requirements of several popular wireless protocols</td>
<td>21</td>
</tr>
<tr>
<td>2</td>
<td>Operation requirements of several popular wireless protocols</td>
<td>21</td>
</tr>
<tr>
<td>3</td>
<td>Sensor properties</td>
<td>28</td>
</tr>
<tr>
<td>4</td>
<td>Port mapping for version 3.1</td>
<td>50</td>
</tr>
<tr>
<td>5</td>
<td>Noise testing results</td>
<td>58</td>
</tr>
<tr>
<td>6</td>
<td>Connectivity testing results</td>
<td>60</td>
</tr>
<tr>
<td>7</td>
<td>Battery life testing results</td>
<td>63</td>
</tr>
<tr>
<td>8</td>
<td>Usability testing results</td>
<td>64</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>1</td>
<td>BEASTdome speaker configuration</td>
<td>7</td>
</tr>
<tr>
<td>2</td>
<td>The original acoustic curtain</td>
<td>11</td>
</tr>
<tr>
<td>3</td>
<td>Acoustic curtain testing</td>
<td>12</td>
</tr>
<tr>
<td>4</td>
<td>Wave field synthesis derivation diagram</td>
<td>12</td>
</tr>
<tr>
<td>5</td>
<td>Proposed system diagram</td>
<td>15</td>
</tr>
<tr>
<td>6</td>
<td>Network Topologies</td>
<td>17</td>
</tr>
<tr>
<td>7</td>
<td>Bluetooth Low Energy channel locations</td>
<td>23</td>
</tr>
<tr>
<td>8</td>
<td>Sensors that make up the 9-degrees of freedom</td>
<td>26</td>
</tr>
<tr>
<td>9</td>
<td>Proposed system properties</td>
<td>29</td>
</tr>
<tr>
<td>10</td>
<td>Schematic of version 1 board design</td>
<td>33</td>
</tr>
<tr>
<td>11</td>
<td>Layout of version 1 board design</td>
<td>34</td>
</tr>
<tr>
<td>12</td>
<td>Schematic of version 2 board design</td>
<td>39</td>
</tr>
<tr>
<td>13</td>
<td>Layout of version 2 board design</td>
<td>40</td>
</tr>
<tr>
<td>14</td>
<td>iPad app used for IMU debugging</td>
<td>42</td>
</tr>
<tr>
<td>15</td>
<td>Schematic of version 3 board design</td>
<td>44</td>
</tr>
<tr>
<td>16</td>
<td>Layout of version 3 board design</td>
<td>45</td>
</tr>
<tr>
<td>17</td>
<td>Charging current and voltage cycle</td>
<td>45</td>
</tr>
<tr>
<td>18</td>
<td>Schematic of version 3.1 board design</td>
<td>47</td>
</tr>
<tr>
<td>19</td>
<td>Layout of version 3.1 board design</td>
<td>48</td>
</tr>
<tr>
<td>20</td>
<td>Sensor board size comparison</td>
<td>48</td>
</tr>
<tr>
<td>21</td>
<td>BLEConnect screenshot</td>
<td>53</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>22</td>
<td>WFS Controller screenshot</td>
<td>54</td>
</tr>
<tr>
<td>23</td>
<td>Sensor board charging curves</td>
<td>62</td>
</tr>
<tr>
<td>24</td>
<td>Copy of usability survey</td>
<td>65</td>
</tr>
</tbody>
</table>
1

Introduction

With the exponential growth and evolution in technology and music, human-computer interfacing has become more necessary as a means to interact within our environment. The technology also brings about the possibilities of submersing ourselves in new virtual worlds that we can interact with as if in a physical environment. This is possible by manipulating and synthesizing information to the human body’s senses. One aspect of this that is growing in popularity in this field is audio, specifically multidimensional audio. There are a few techniques for doing this, such as surround sound, stereo headphones, or wave field synthesis. As these strategies are implemented into new virtual reality environments, it becomes apparent that a way to interact with the world around the user is necessary, specifically in the form of an advanced and intuitive audio pan controller.

1.1 Focus and Aims

A long-term goal of this field is to develop a new way to interface with multichannel speaker environment set up with wave field synthesis. This study’s objective was to develop a new and unique multidimensional sensor with the capability of relaying information wirelessly to a machine that performs wave field synthesis processing. It would be possible to use the proposed controller in other applications, but for the purpose of this project, decisions were made based
on the constraints of a wave field synthesis system. The aims for this thesis can be broken down into three needs:

- Design and develop a low-power hardware sensor capable of tracking movement in multiple dimensions and relaying the data wirelessly.

- Develop software for wirelessly connecting and communicating with the custom sensor.

- Pass the movement profile data to a wave field synthesis system so that the data can be used to control the virtual position of the source.

### 1.2 Motivation

Spatial audio is growing in popularity, and low-power, consumer priced interfacing controllers are left to be desired, especially for applications such as gaming and music composition. In addition, a device that is small in size and weight, as to not interfere with natural movement would a bonus. I intend to combine my interest in both music and human-computer interfacing to develop a device to meet these needs.

### 1.3 Antipated Outcomes

As outlined in the focus section, the expected result of this project is a wave field synthesis system capable of being controlled by a wireless IMU. The IMU should be able to dictate the coordinates of the wave field synthesis virtual source in three dimensions (x, y, and z). This means the final hardware must have at least three recordable axes, but preferably up to nine for better precision.
Lastly, there will be a sleek user interface for selecting the control and configuring the wave field synthesis output.

This is significant because there is not a low-cost, low-power controller currently available for the purpose of interfacing with a wave field synthesis system or audio panning. Currently available sensors use a more complicated wireless network structures that often require a middle-man to receive and interpret data before that information is available to the system. The sensor proposed in this thesis not only offers 9-degrees of freedom, but it also relays the data directly to the receiver using a new wireless technology, Bluetooth Low Energy.
2

Background

2.1 Previous Work

This thesis builds on previous work and it is important to understand those pieces before understanding the strategies outlined in this paper that add and improve upon the past work. First, it is important to understand how sensors are beginning to evolve in both accuracy and methods of communication. Technology is growing at an exponential rate, so technology new as of last year, is outdated today. Second, it is necessary to understand how wave field synthesis works in order to see how the proposed sensor board can act as a controller for the system.

2.1.1 Similar IMU Designs

Applications for IMUs range from aircraft monitoring to gait analysis to human-computer interaction. Because these sensors have been around for a while, it is important to understand where the technology stands before trying to develop and a new feature. IMUs are nothing new in the media world since they allow for new ways of interfacing with information. A common application is gesture recognition. Since these sensors are capable of returning detailed movement data, machine-learning algorithms can be applied to detect very precise patterns and interpret those movements as specific triggers. This can even be applied to beat detection, which is a common audio application for IMUs. Something as simple as a 1-dimensional accelerometer can be used to detect up
and down motions to derive a rhythm or tempo as explained in [Lee et al., 2007].

More accurate and detailed calculations are possible with multiple dimensions and sensors. Research groups and companies have begun taking advantage of this by developing IMUs with multidimensional accelerometers, and even adding gyroscopes for more precision [Höfer et al., 2009, Nam et al., 2013]. Though similar to the proposed project, these products all lacked the wireless capabilities.

More and more devices that utilize multidimensional sensors and wireless communication are being developed, but when compared to the sensor outlined in this thesis they either lack dimensional accuracy, they use unnecessarily complicated wireless technologies, or they are larger in form factor [Bevilacqua et al., 2007, Jacobs et al., 2008, Jacobs, 2007, Kuyken et al., 2008]. A few similar and recent available IMUs were the MARG (Magnet, Angular Rate, and Gravity) sensors [Todoroff, 2011], which requires a wired network on the body and a single node for wireless transmission, the Wireless Interactive Sensor (WIS) platform [Lu et al., 2012], which simply uses a commercial board design larger that the sensor described in this thesis, and APDM’s IMU sensors [APDM Inc., 2014], which tracks 9-degrees of movement and relays that data from the individual sensors, but it operates over wifi and requires data to pass through a wireless hub before it can be accessed. All three sensors offered 9-degrees of freedom, but their communication and wireless methods could be improved upon as explained in chapter 3.
Previously developed IMUs tend to focus movement measurement. The simpler sensors with only an accelerometer or gyroscope were designed for beat detection, while the sensors with more degrees of freedom were also used to capture movement and gestures. Only the WIS platform was designed as an interactive device for interfacing with information. The other two sensors with 9-degrees of freedom were only used for movement profiling.

2.1.2 Audio Panning

Controllers for audio panning have existed since the 1950’s. Also known as “pan pots”, the name is a direct reference to the process, panning the position of a sound source across multiple speakers in a single or multiple dimensions. Pan pots come in a variety of packages, from free-space tracking to moveable joysticks.

Pierre Schaeffer and Jacques Poullin developed one of the first panning devices in 1951 called the “relief desk” or “potentiomètre d’espace”. This controller was first demonstrated in Pierre Henry and Schaeffer’s *Symphonie pour un homme seul*. Four speakers were positioned around the audience, two in front to the left and right, one at the rear in the middle, and the last in the rear, but positioned in the middle of the ceiling. The sounds could then be moved or panned across the speakers to create a spatial audio effect. On stage, the performer was surrounded by four inductive coils positioned similarly to that of the speakers (left, right, in front, and above) and a transmitter coil could be moved around within the area to control the intensities from each speaker at which the sounds were played [Teruggi, 2007, Manning, 2013]. Other audio
panning devices include the stirrer, developed by Lowell Cross in the 1960’s, a joystick used to control four sound sources simultaneous in space; and the elBo and footpad controllers, developed by Colby Leider, a joystick for audio manipulation and control as well as a pressure sensing board that can be used to navigate two directions with a user’s feet [Cross, 2014, Leider et al., 2009].

Figure 1: The 30+ speaker configuration inside of the University of Birmingham BEASTdome [University of Birmingham, 2014].

In addition to controllers, there are also a variety of venues with large multichannel speaker configurations designed for spacial audio and panning. One of the earlier theaters was the Acoustmonium designed by Francois Bayle in 1974. The Acoustmonium consists of 80 loudspeakers of different sizes and shapes for spatial tape playback [Prager, 2012]. The BEAST, Birmingham ElectroAcoustic Sound Theatre, is a theater that contains over 100 discretely
accessible loudspeakers at the University of Birmingham. University of Birmingham also holds the BEASTdome, a smaller theater in the shape of the dome with over 30 speakers positioned throughout as shown in figure–1 [University of Birmingham, 2014]. Larger than the BEAST is the AlloSphere, a three story building at the University of California, Santa Barbara, that is houses a 140 speaker sphere setup [University of California Santa Barbara, 2010].

2.1.3 WFSDesigner

This thesis builds on previous work by Matthew Montag and his development of the WFSDesigner Software as an application to show the functionality of this project. WFSDesigner is an open source Windows program for visualizing and controlling a wave field synthesis environment. The program relies on the following libraries and frameworks: QT for the GUI framework, FFTW for fast convolutions to apply to filters, Libsndfile for audio file interaction, and PortAudio for audio interfacing [Montag, 2011a].

WFSDesigner allows a user to select the type of loudspeaker array configuration, set the location of the virtual sources through either the 3D visualization map or by setting specific coordinates, and the synthesis mode. Once properly configured, the program handles the source derivations and outputs to the loudspeaker array. The wave field synthesis process is explained in greater detail in the next section.
2.2 Wave Field Synthesis

To show the versatility of the sensor board and its application to wave field synthesis, a system was developed to show how this IMU could be used as a control for wave field synthesis source positioning. Wave field synthesis is a technique that requires a large array of speakers used to create an artificial audio wavefront for a virtual sound source. The greatest advantage wave field synthesis offers over techniques like stereo and surround sound is the audio does not depend on the listener’s position since it is a fully constructed wavefront.

The intended larger wavefront, or primary source, is created by the combination of the smaller individual wavefronts produced by the speaker array, referred to as secondary sources. This is the basis of the Huygens-Fresnel principle; a primary source wave field at time $t + \delta t$ can be synthesized by an array of infinite secondary sources at an infinitely small distance from one another. In practice, the speaker array is positioned as a line, plane, or circle around the user, and the output of each speaker is timed in order to construct the primary source. Since wave field synthesis uses a technique to form a full field of sound, and it does not rely on psychoacoustic tricks, wave field synthesis can be used to create environments where the listener is able to walk around without experiencing any change to the location of the sound source or destabilization of the apparent source location. It would be the same situation as if the source was present at the coordinates of the virtual source because the wavefront for that non-existent source has been reconstructed.
The first idea of wave field synthesis appeared in 1934 by Steinberg and Snow called the acoustic curtain. The theory of the acoustic curtain was to use an array of microphones to record an audio source and replay the source in a remote listening location via an array of loud speakers as shown in figure–2. Though it was not possible to fully develop this idea at the time, similar testing took place during the 1930's at Bell labs using three microphones (left, center, and right) in front of a sound source and retransmitting the audio at a distant location through three speakers (left, center, and right) as shown in figure–3. This is not classified as wave field synthesis since it does not recreate the original wavefront, but the technique does retain relative audio amplitudes and delays producing a perceptual spatial audio effect known as the precedence effect. This work would pave the way to other spatial audio techniques such as surround sound and ambiophonics.

It wasn’t until the late 1980’s that wave field synthesis finally became a reality through the work of Berkhout at the Delft University of Technology [Berkhout, 1988]. Berkhout outlined a technique for holographically reconstructing of a wavefront, which is now known as wave field synthesis. This theory was first explained in his 1988 paper titled, “A Holographic Approach to Acoustic Control”, in which he stated the spatial reconstruction of direct and reflected waves would be ideal for acoustic control because it would be indistinguishable from true sound fields. He later further explained the idea in a 1993 paper titled “Acoustic Control by Wave Field Synthesis”
Figure 2: Diagram of the original acoustic curtain that featured a large number of microphones in front of a sound source and a speaker to be paired with each microphone [Steinberg and Snow, 1934a]. [Berkhout et al., 1993].

Berkhout’s proposal was derived from both the Huygens-Fresnel Principle and the Kirchhoff-Helmholtz integral. These formulas are important in reconstructing a primary source from a one- or two-dimensional array of speakers. The most basic concept of wave field synthesis includes a source, a receiver, and a hypothetical surface, illustrated as the sphere on the left in figure–4, within a 3-dimensional medium. The Kirchhoff-Helmholtz integral shows that it is possible to calculate the pressure at any location within the sphere if the pressure and particle velocity at any point, $\partial V$, along the outside of the sphere is known.

The Kirchhoff-Helmholtz integral is defined as:

$$ P(x, \omega) = \int_{\partial V} \left( G(x|x_0, \omega) \frac{\partial}{\partial n} P(x_0, \omega) - P(x_0, \omega) \frac{\partial}{\partial n} G(x|x_0, \omega) \right) dS_0 $$

Where $P$ is the pressure at the position $x$ of the receiver, $x_0$ is the position on the
Figure 3: Diagram of arrangement for sound localization tests using acoustic curtain and only a set of three microphone and speaker pairs (left, center, right) [Steinberg and Snow, 1934b].

Figure 4: Steps of the wave field synthesis derivation from a continuous 3-dimensional primary source to a discrete line of secondary sources [Montag, 2011b].

boundary surface, $\omega$ is the frequency, $n$ is the normal vector for the source at $x_0$, $S_0$ is the surface area of the volume, and $\frac{\partial}{\partial n} P(x_0, \omega)$ is the directional gradient of the primary source in the same direction as $n$. $G(x|x_0, \omega)$ represents Green’s function, which can be interpreted as the virtual source field generated by the second sources. Green’s function is dependent on two things, the dimensionality of the reproduction (2-dimensions vs 3-dimensions) and whether the sources are monopole or dipole. In the case of wave field synthesis, the sources are more usually modeled as monopole, since the secondary sources are generated from
loudspeakers.

As shown in figure–4, Rayleigh integrals also play a part in the wave field synthesis derivation. Since the Kirchhoff-Helmholtz integral states that the volume surface can be represented by monopole and dipole sources, the equation can be reduced to a Rayleigh integral for either case. The Rayleigh I integral states that the pressure at \( x \) can be synthesized by monopole sources distributed about the surface, while the Rayleigh II integral says the same except for dipole sources. Both cases are for 3-dimensions, but 2-dimensional equations can be derived from them since most wave field synthesis setups are made up of line arrays of loudspeakers instead of plane arrays. This is a problem for waves that are non-cylindrical since by reducing the equations to 2-dimensions, the synthesized primary source is attenuated with \( 1/\sqrt{r} \) the further wave is from the array, therefore making it impossible to produce a plane wave. To account for this, the Rayleigh 2.5D integral was proposed by Vogel in 1993 [Vogel, 1993]. This equation corrects the amplitude error for a specified reference line [Schuitman, 2005, van Zon, 2003].
Proposed System

The proposed system is a multidimensional sensor that is capable of wirelessly relaying data to a remote system running wave field synthesis software. The kinematics given from the sensor are used to determine the position for a 3-dimensional virtual phantom source in the wave field synthesis program as shown in figure–5. A previous Music Engineering Masters student, Matthew Montag, developed the software selected to perform the wave field synthesis in 2011 as part of his thesis [Montag, 2011a]. The program was written for the Windows Operating System, so the first restraint is that the program dealing with communication between the sensor and the remote system must be written for Windows.

With knowledge of the application, it is important to first figure out some of the basics needed for the sensor hardware. The main components necessary are 1) a way to communicate with a remote computer that handles the wave field synthesis side, so a wireless transceiver; 2) sub-sensors that measure multiple dimensions of movement, preferably 9-degrees of freedom; 3) a way to manage power to the board; and 4) a microcontroller to handle all of the onboard management. It is important to note that there needed to be a way to program the microcontroller as well. With some basic design requirements specified, initial part selection and testing could begin.
Figure 5: Diagram of the proposed system. Starting in the bottom right, the user’s hand movement is recorded in multiple directions by the attached sensor. The sensor wirelessly relays kinematics to the computer. The computer then controls loudspeaker array outputs for virtual source at controlled by user’s hand position.

3.1 Wireless Communication

One of the first things to consider when creating hardware that needs to communicate wirelessly with other systems is the wireless protocol. With a wide variety of wireless communication methods available to use, it is important to choose the one that is best suited for the hardware. There are considerations like power consumption, network topology, transmission bitrate, and form factor since the antenna will need to be located on the hardware. Without an appropriate communication method, the usefulness of the circuit board would be degraded.

3.1.1 Available Systems

In order to pick the best wireless technology, it was important to explore all available protocols available. To limit search results due to the overwhelming amount of wireless options available, the first limitation for a wireless technology
was that it needed to have low-power options. Because the system would need to be able to run off a battery, it was decided that lower power consumption was one of the biggest requirements. By placing an emphasis on battery power and battery life, the results were limited to Bluetooth Low Energy, ANT, Zigbee, Zigbee RF4CE, IrDA, and Wi-Fi.

Wireless-fidelity IEEE Standard 802.11, more commonly known as Wi-Fi, was the first protocol to be considered due to its long history and wide availability. Wi-Fi is quickly becoming a common communication method not just in computers, phones, and tablets; it is becoming incorporated into household devices, cars, and videogames. Unfortunately, Wi-Fi would be overkill for this system. Wi-Fi is designed and optimized for high-speed high bitrate communication. So, though it is an efficient protocol, it is excessive for low data transfers and it would lead to unnecessary power consumption [Smith, 2011b, IEEE Computer Society, 1997].

The next wireless technology to be considered was infrared (IR) communication, which is managed by the Infrared Data Association (IrDA). Unlike the other low-power protocols, IR operates in a much smaller wavelength spectrum, the infrared spectrum as its name indicates. In 2009, IrDA announced technology to transfer data at high speeds (1 Gbps) while running on low power, but this is only possible from very small distances (about 10cm for 1 Gbps) [Infrared Data Association, 2009]. While this is a much higher bitrate than is necessary for this project, in general infrared communication requires shorter
distances, more power than the other wireless methods mentioned, it is more susceptible to interference, and it needs to have line of sight with its receiver. Therefore, it would not be practical to use IR in this project.

(a) P2P

(b) Star

(c) Tree

(d) Mesh

Figure 6: Network Topologies [Frenzel, 2012]

The other three technologies, ANT, BLE, and Zigbee, are more practical for low-power applications. In order to compare and outline the three, it is necessary to understand the different types of network topologies first.

Point-to-point (P2P), shown in Figure 6a, displays a situation where there is only one master node and one slave node. The master listens for data from the slave so that it can either process it or pass it along to another network. Neither node ever changes roles. A star network, also known as a multipoint-to-point
(M2P), shown in Figure–6b, is similar to the P2P network since none of the nodes change their role and the slaves always send data to the master for processing or relaying, but never to one another. This network allows for multiple slaves to connect to a single master and this is type of network is most desired for this project. The tree structure, shown in Figure–6c, builds onto the star structure so that multiple star networks can communication through a connection between the master nodes and one of the master nodes can be connected to an outside network in order to pass data from the tree along. The last common setup is a mesh network, shown in Figure–6d. A mesh network allows all nodes to talk to each other in order to pass data along to the master node, which also makes a mesh network more reliable. Nodes, on the far side from the master node can send data to closer nodes to have those closer nodes relay the information to master. The benefit of this is if one node gets disconnected from the rest of the network, an alternative path can be formed to the master with the other available nodes as long as there exists a path to the master node. Though more reliable, this comes at the cost of the network being more complicated.

Unlike the above wireless protocols, the last three were designed for low-power network communication, and though similar, they all have their individual strengths and weaknesses. ZigBee was released in 2002 by a group of 16 companies known as the ZigBee Alliance. The protocol was based on the 802.15.4 IEEE Standard [Zigbee Alliance, 2011], and the strength that it offered was low-power mesh networking. This is great for creating far-reaching Local
Area Connections (LANs), but it comes at the cost of extra power to the router nodes. ZigBee is capable of creating star networks as well, though because of its mesh network capabilities, the connection setup is more complicated than necessary for this configuration.

Zigbee’s network operates on multiple channels starting at 2.4 GHz. Each channel is 2 MHz wide with 5 MHz separation in-between. Before setting up a network on one of the channels, Zigbee must conduct a thorough frequency scan to find a channel with minimal interference because once the connection is set up, the channel cannot be changed. The complexity of the network setup, which is what makes it great for a mesh set up, causes it to be difficult to make quick connections between slave and master nodes.

A common application for Zigbee is a network that connects different sections and components of a house. This way, information can be quickly relayed throughout the building via Zigbee’s mesh network. In 2009, Radio Frequency for Consumer Electronics (RF4CE) was standardized to add to this idea by adding remote control support. RF4CE was designed to replace and improve upon infrared technology, while remaining a part of the ZigBee protocol [ZigBee Alliance, 2013].

Next to be standardized was the ANT/ANT+ wireless network by Dynastream Innovations Inc. and the ANT+ Alliance in 2004. ANT+ has eight available frequencies for communication, starting at 2.4 GHz and 1 MHz wide, but in most instances, only one channel is used since it utilizes Time Division
Multiplexing (TDM). Multiple connections are able to use the same channel because of this, since the transmissions are set up into time slots. Ant+ is capable of all the network configurations mentioned previously, but the mesh style network is most common. Each sensor is capable of master and slave roles, and depending on the connection, it can act as either one, even if it has established different role for a separate connection. ANT+ was designed as a Personal Area Network (PAN), so that individual components could communicate with one another. Common application for ANT+ is sports equipment such as stationary cycles where multiple components of the machine can communicate and track the user’s movement on the equipment. One of the downfalls of the protocol is that it is proprietary, so in order to advertise that the hardware utilizes ANT+, it must be certified, which is a subscription cost per year.

Bluetooth Low Energy (BLE), originally developed in the Nokia Research Center under the name Wibree, was picked up by the Bluetooth Special Interest Group in 2007. It is also marketed as Bluetooth Smart and Bluetooth 4.0. BLE works best under the P2P and star network designs. It operates in the 2.4 GHz spectrum, with 40 channels and each being 2 MHz wide. One major advantage of Bluetooth Low Energy is its ‘frequency hopping’ ability. This means that after a connection has been established on a channel and there happens to be interference on that channel, the two nodes can decide to switch to a new, clearer channel. If the channel ends up being so bad that a message containing a new channel to switch to cannot be picked up, BLE is light weight enough that the
connection can just be dropped and then quickly (3ms connection time) reestablished on a new channel.

Much like ANT/ANT+, Bluetooth Low Energy is ultra low-power and it is designed to be a Personal Area Network. BLE is used for connecting sensors local to the body, and chipsets are becoming more and more common in devices local to the body such as cellphones and tablets [Bluetooth SIG Inc., 2013]. Also, the Bluetooth Low Energy is designed for quick proximity based connections. Though not as fast as ANT/ANT+’s connection time, it is still basically negligible.

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Peak Current</th>
<th>Efficiency</th>
<th>Throughput</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLE</td>
<td>12 – 16 mA</td>
<td>0.15 µW/bit</td>
<td>300 kbps</td>
</tr>
<tr>
<td>ANT</td>
<td>17 mA</td>
<td>0.71 µW/bit</td>
<td>20 kbps</td>
</tr>
<tr>
<td>ZigBee</td>
<td>30 – 40 mA</td>
<td>186 µW/bit</td>
<td>100 – 250 kbps</td>
</tr>
<tr>
<td>Radio Frequency</td>
<td>30 mA</td>
<td>variable</td>
<td>100 – 250 kbps</td>
</tr>
<tr>
<td>Infrared</td>
<td>10.2 mA</td>
<td>11.7 µW/bit</td>
<td>1 Gbps</td>
</tr>
<tr>
<td>Wi-Fi</td>
<td>116 mA</td>
<td>0.005 µW/bit</td>
<td>11 Mbps</td>
</tr>
<tr>
<td>Bluetooth Classic</td>
<td>185 mA</td>
<td>15 µW/bit</td>
<td>3 Mbps</td>
</tr>
</tbody>
</table>

Table 1: Power and Efficiency requirements of several popular wireless protocols [Zigbee Alliance, 2011, Bennett and Coggin, 2013, Smith, 2011a, Smith, 2011b].

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Spectrum</th>
<th>Range</th>
<th>Latency</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLE</td>
<td>2.4 GHz</td>
<td>100 m</td>
<td>3 ms</td>
</tr>
<tr>
<td>ANT</td>
<td>2.4 GHz</td>
<td>30 m</td>
<td>0 ms</td>
</tr>
<tr>
<td>ZigBee</td>
<td>2.4 GHz</td>
<td>100 m</td>
<td>20 ms</td>
</tr>
<tr>
<td>Radio Frequency</td>
<td>2.4 GHz</td>
<td>100 m</td>
<td>20 ms</td>
</tr>
<tr>
<td>Infrared</td>
<td>330 – 350 THz</td>
<td>0.01 m</td>
<td>25 ms</td>
</tr>
<tr>
<td>Wi-Fi</td>
<td>2.4 &amp; 5 GHz</td>
<td>150 m</td>
<td>1.5 ms</td>
</tr>
<tr>
<td>Bluetooth Classic</td>
<td>2.4 GHz</td>
<td>100 m</td>
<td>100 ms</td>
</tr>
</tbody>
</table>

Table 2: Operation requirements of several popular wireless protocols [Zigbee Alliance, 2011, Bennett and Coggin, 2013, Smith, 2011a, Smith, 2011b, Bluegiga Technologies, 2011].

Table–1 and Table–2 show summaries of the different wireless protocols for easier analysis. It is clear that BLE and ANT are the most similar, and most practical for a PAN sensor. Bluetooth Low Energy was selected in the end because it is slightly more power efficient, specifically designed for a star network,
and the larger range allows for more flexibility with reading/recording data from the sensor.

3.1.2 Bluetooth Low Energy

Bluetooth has been around for quite some time; invented in 1994 and IEEE standardized in 2002. Its aim was to act similarly to Wi-Fi, but remain as a Personal Area Network. A common application for Bluetooth is wireless headsets, since only a local connection is necessary to transfer audio data. Over the years, Bluetooth has been revised and improved since its introduction, often in terms of data transmission rates. Originally Bluetooth v1.0 transmitted at 1Mbit/s, but now it is up to 3Mbit/s in Bluetooth v3.0, also known as Bluetooth Classic. Bluetooth v4.0, Bluetooth Low Energy, targeted another aspect of the protocol, power consumption, and it reduces power consumption by about a factor of 100, as shown in Table–1.

Though Bluetooth Low Energy was originally developed separately from the Bluetooth family, it has been made to follow similarly to the rest of the Bluetooth specifications. Firstly, BLE utilizes Gaussian Frequency-Shift Keying (GFSK) modulation for transmission in the 2.4 GHz Industrial, Scientific and Medical (ISM) radio band. As mentioned previously, BLE is made up of 40 channels, each 2 MHz wide. As shown in Figure–7, three of the channels (marked in green) are used for advertising. They are positioned at 2402, 2426, and 2480 MHz locations as to not experience interference from Wi-Fi, since these are the three most important channels. They are important because they are used for
initial communication that is necessary before setting up a more permanent connection on another channel. The other 37 channels are used for long-term communication, and the channel that is used for a specific connection based on a channel analyze to find frequency with the least amount of interference before the long-term connection is established. BLE differs slightly from the rest of the Bluetooth family here since it is more relaxed in timeout time. This means, there must be a longer gap in communication before the connection is regarded as lost, but this means the slave node can save on power by transmitting less frequently.

(a) Bluetooth Low Energy 40 channels

(b) In relationship to Wi-Fi channels

Figure 7: Bluetooth Low Energy channel locations [Nilsson, 2011]
There are only a handful of states in the BLE specification [Bluetooth SIG Inc., 2010]; advertising, scanning, master, and slave. Advertising mode means the device is sending information about itself in hopes of being picked up by a scanning device for connection. Within advertising mode, the device can send more information about itself if a scanning device makes a request for more information. As mentioned, the companion to advertising is scanning, which listens for the advertising packets. A device in scanning mode can either strictly listen for advertising packets, or it can act on those packets by requesting either more information from the advertiser or a connection with the advertising device. If a connection is established, the scanning device becomes the master, and the advertising device becomes the slave. According to the BLE specification, the connections utilize 32 bit addresses, so theoretically, the master device can connect up to $2^{32}$ slave devices. Slave devices can only connect to one device, the master device, since the most common network topology for BLE is the star formation.

The packets that are transmitted are made up of multiple parts, the preamble, the access address, the Packet Data Unit (PDU), and the CRC. The preamble is the first byte of the packet and it is used for timing and frequency syncing as well as a way to way to represent the beginning of a packet. It can either be 0x55 or 0xAA, which both represent a series of alternating 1’s and 0’s. The next 4 bytes is the access address. It specifies the address of the device that the packet is intended for; the hex address 0x08E89BED6 is reserved for
advertisement, otherwise this would be a specific device address once a connection is established. Starting at byte six is the PDU. It can range from 2 to 39 bytes and size and information depends on the type of packet. The first 2 bytes of the PDU always represent a header, and the following 0 to 37 bytes are known as the payload. If an advertisement packet is being sent, the payload includes either advertisement information about a device, scanner acknowledgement/requests, or a connection request. The other type of payload is for a data channel, which includes the lower level communication for transferring data between a slave and master. The last 3 bytes of the packet is the Cyclic Redundancy Check (CRC), it is used for checking for errors in the transferred PDU. Packet size is directly related to the time it takes to transfer the data. A smallest packet, 10 bytes, would take $80\mu s$, while the largest packet, 47 bytes, would take around $300\mu s$ to transfer.

With the protocol selected, the next step was to choose a wireless chip capable of BLE communication. The chip chosen was BlueRadio’s nBlue BR-LE4.0-S2A [BlueRadios Inc, 2012b, BlueRadios Inc, 2012a] because it was low power (31.4mA TX/22.1 mA RX), utilizes the Texas Instrument CC2540 System on a Chip (SoC), capable of Universal Asynchronous Receiver/Transmitter (UART), SPI, and USB data interfaces, included 12-bit Analog-to-Digital Converters (ADC) on eight of the pins, and there was a great online forum/community set up for it. From this chip, the rest of the board could be designed matching similar specifications such as voltage and data
3.2 Sensor Selection

Another aspect of this project that required some research was sensor selection. Though not as in-depth as the wireless selection, it was still a key feature to the board, so to ensure the best output, initial reading and testing was necessary to make sure the sensors would fit the aims of the project. Since the sensor board was set to have 9-degrees of freedom, an accelerometer, gyroscope, and magnetometer were needed. An accelerometer is used to measure acceleration, and in the case of a 3-dimensional accelerometer, it can measure acceleration in the x, y, and z planes as shown in figure-8a. A gyroscope measures rotation, so for 3-dimensions it would measure rotation around the x, y, and z planes, as shown in figure-8b. Lastly, the magnetometer measures the geomagnetic field felt in 3-dimensions (x, y, and z orientations), which is useful for determining orientation.

![Diagram of sensors](image)

(a) Accelerometer  
(b) Gyroscope movement, where α denotes yaw, β denotes pitch, and γ denotes roll

Figure 8: Sensors that make up the 9-degrees of freedom
The sensors would need to meet the following specifications:

- Relatively small in size (length, width, and height), since the board would need to be as small as possible to make it comfortable for a user to wear.

- Operate on 3.3V since the BLE chip operated on 3.3V and it would be most convenient for the entire circuit to run off the same voltage line.

- Consume as little current as possible (sleep modes were a plus) since the voltage would already be set, a lower current would mean lower power consumption.

- High precision output.

- Utilize Serial Peripheral Interfacing (SPI) for onboard communication since the microcontroller with the BLE chip uses SPI for transferring data locally.

Conveniently, the accelerometer chosen is coupled with a magnetometer on the same chip that takes up minimal space on the sensor board (3mm x 3mm x 0.95mm). The sensor is the Bosch Sensortec BMC050 eCompass with 6-degrees of freedom [Bosch Sensortec, 2011]. The chip is capable of SPI communication, can operate on 3.3V, and draws only 540 µA in regular mode. The accelerometer offers 10-bit resolution and an array of programmable g-ranges (±2g, ±4g, ±8g, ±16g). Acceleration is measured in units of ‘g’ (hence g-range), and 1g is equal to Earth’s gravity at sea level. For this application, ±2g should be plenty. The
magnetometer’s range is $\pm 1000\mu T$ in the x and y direction and $\pm 2500\mu T$ in the z direction, which is large enough to capture the Earth’s magnetic poles.

The gyroscope chosen was STMicroelectronics’ L3G4200DTR [STMicroelectronics, 2010]. This model was picked because it was SPI compatible, could run on 3.3V, drew relatively little current (6.1mA), and it was capable of 16-bit resolution along with an 8-bit temperature data output in order to calculate temperature offset. The gyroscope has a selectable range of $\pm 250\text{dps}$, $\pm 500\text{dps}$, or $\pm 2000\text{dps}$, where the range refers to the angular velocity and ‘dps’ stands for degrees per second.

<table>
<thead>
<tr>
<th>Property</th>
<th>Accelerometer</th>
<th>Gyroscope</th>
<th>Magnetometer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Axes</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Range</td>
<td>$\pm 2g$, $\pm 4g$, $\pm 8g$, or $\pm 16g$</td>
<td>$\pm 250\text{dps}$, $\pm 500\text{dps}$, or $\pm 2000\text{dps}$</td>
<td>$x/y$: $\pm 1000\mu T$, $z$: $\pm 2500\mu T$</td>
</tr>
<tr>
<td>Noise Density</td>
<td>0.8mg$/\sqrt{Hz}$</td>
<td>0.03dps$/\sqrt{Hz}$</td>
<td>$x/y$: $1\mu T$/$/\sqrt{Hz}$, $z$: $1.4\mu T$/$/\sqrt{Hz}$</td>
</tr>
<tr>
<td>Supply Voltage</td>
<td>1.2 – 3.6V</td>
<td>2.4 – 3.6V</td>
<td>1.2 – 3.6V</td>
</tr>
<tr>
<td>Normal Current</td>
<td>139$\mu$A</td>
<td>6.1mA</td>
<td>0.5mA</td>
</tr>
<tr>
<td>Sleep Current</td>
<td>0.5$\mu$A</td>
<td>5$\mu$A</td>
<td>1$\mu$A</td>
</tr>
<tr>
<td>Output Rate</td>
<td>10 – 20Hz</td>
<td>100 – 800Hz</td>
<td>10 – 20Hz</td>
</tr>
<tr>
<td>Bandwidth</td>
<td>8 – 1000Hz</td>
<td>50Hz</td>
<td>8 – 1000Hz</td>
</tr>
<tr>
<td>Resolution</td>
<td>10 bits</td>
<td>16 bits</td>
<td>$x/y$: 13 bits, $z$: 15 bits</td>
</tr>
</tbody>
</table>

Table 3: Properties of each sensor.

3.3 Other Necessary Parts

The last two necessary parts before designing the circuit were a way to program the microcontroller embedded in the BLE chip and a chip to manage battery power and charging. The goal was to find a programming interface that could connect via USB micro-B; this way, the same connection could be used to charge the battery. The Future Technology Device International Ltd (more commonly known as FTDI) FT230X chip was selected because it supported 4 pin
UART at 3M baud, could operate on 3.3V, handled USB 2.0 interfacing, and pulled minimal current (8mA), not that that would be a problem because when it was powered on for programming, the board would likely be running off USB power.

The Texas Instruments LM3658 Dual Source USB/AC Li Chemistry Charger was chosen as the power management chip [Texas Instruments, 2007]. Besides being made my the same company as the microcontroller embedded in the BLE chip, the LM3658 can charge off USB or an AC wall adapter, supports a wide range of current options, and it offers two pins for monitoring the status of the charger.

![Figure 9: Diagram showing the individual properties of each piece in the wave field synthesis final system. Wireless IMU uses BLE to send kinematic data to the wave field synthesis system, made up of BLEConnect and WFS Designer, which calculates new coordinates for a virtual source and sends that information to the loudspeaker array.](image)
4

Development and Implementation

There were multiple steps in the development of the full human interfacing system. The first step was prototyping and debugging the multidimensional sensor. Once that was working properly, software to interact with the sensor was written and then integrated into WFS Designer.

4.1 Sensor Development

Multiple revisions were planned for this board design. Over a one and a half year period, the sensor went through extensive testing, debugging, and redesign to improve functionality, remove redundant and debugging components, and reduce size.

4.1.1 Version 1 Design

The first version was two-pronged; to integrate components and evaluate interfaces. Testing of individual components, and limited interaction between some of the parts was possible with the development kit, but this initial version combined and ran each peripheral and the microcontroller together. The needs and requirements of the board were delineated in the previous chapter, but there were other required components in order achieve intra-functionality. One of those parts was the Analog Devices ADP2108 step-down DC-to-DC converter [Analog Devices, 2012]. This chip was necessary because it would convert the battery voltage (3.7 – 4.2V) to the necessary 3.3V. It also served as a voltage
control so even when the battery voltage would drop the longer it ran, the converter would make sure the voltage was always at a steady 3.3V.

One of the largest parts on this board was the debug header. Because it was the first version, a debug header was necessary to be able to monitor points throughout the circuit such as the SPI lines, analog pins on the microcontroller, and the voltage points which included the USB power line, battery power line, and the converted voltage line. In addition to debugging with the strip header, LEDs were attached to some of the output pins on the Texas Instruments LM3658SD charging chip, the FTDI FT230XS programming chip, two of the microcontroller ADC outputs, and to VDD to see power was going to the board when it was turned on. Without the debugging header and LEDs, the debugging process would have not been possible.

Another important part of this revision was the onboard memory, Cypress Semiconductor FM25L04B 4-Kbit Serial F-RAM [Cypress Semiconductor Corporation, 2014]. This chip was used for storing data before broadcasting it. If another board was connected via the output SPI lines, that board would also be able to access the information in the memory. This would allow for two or more boards to share data without using wireless communication if they were set up properly to avoid data access interference. The chip was appealing because it used SPI, it can operate on 3.3V, and only draws 200µA. The 4Kb of space was also more than enough for temporarily storing sensor data, since output from all the sensors combined was only 18 bytes
Lastly, this design of the sensor board included three switches. The first switch was a push button connected to the Bluetooth Module’s reset line. When pressed, the button would send a logical high (VDD) to the pin, resetting the microcontroller. This was helpful for quickly cycling power for putting the chip in firmware upgrade mode. In order to enable firmware upgrade mode, there was a single pole, double throw switch with its output connected to PIO_14 (pin 19) on the Bluetooth Module, and the switch could either set PIO_14 to logical high for firmware upgrade mode, or set in to logical low (ground) for the chip to operate in normal mode. Lastly, the biggest switch on the board was a three position switch used for setting the functional mode of the board and for turning the BMC050 eCompass sensor on or off. The board had three modes so that it could either send data from the IMUs, send stored digital data from the memory chip, or send analog input data from the ADC breakout pins. The reason this switch also controls power to the eCompass is because the chip select line connected to the eCompass that controls the accelerometer is also used to select the memory chip since the parts are used in separate modes.

The layout for the first design utilized a non-compact form factor to facilitate simple debugging. It was a two layer board with only the top side populated with parts. The bottom layer was used for routing, and the unrouted area was used to create a ground plane. The ground plane was recommended for the BlueRadio chip to help reduce electromagnetic interference around the
33

antenna.

Figure 10: Schematic for the first version of the sensor board. The design is separated into sections: USB programming (top-left), battery management (top-right), onboard memory (middle-left), voltage regulator (middle), debug header (middle-right), BlueRadio connections (bottom-left), eCompass connections (bottom-middle), and gyroscope connections (bottom-right)

4.1.2 Version 1 Testing and Debugging

After fabrication and assembly, the board measured 2.13 x 2.04 in\(^2\). The first two problems related to the layout and were the result of incorrect landing pattern designs. First, the micro USB header was missing pads to attach it to the board. The pins were connected correctly, but the wings had nowhere to connect to in order to hold the part down and the pins were too small and fragile to keep the part attached to the board. The other problem was the diameter of the holes for the programming switch were too small. A temporary fix for these two problems was to secure the parts with hot glue until they could be fixed in
Figure 11: Layout for the first version of the sensor board. Top layer routing is shown in teal and bottom layer routing is shown in yellow. Parts are outlined in grey and vias are colored purple.
the next revision.

The next problems encountered were part of the board’s functionality. The testing process involved using firmware code written for the TI CC2540 development board and changing ports to match the IMU sensor, since both boards used the same BLE chipset. A logic analyzer was used reading and deciphering the SPI communication and ADC output and input through the debug header pins to ensure proper signals were being sent or received. A Voltage meter was also used to checking voltage outputs at different stages of the circuit to ensure correct voltages, especially around the battery and voltage conversion chips. The CC2540 development board was also used for debugging during the wireless evaluation as a way to detect any BLE packets being transmitted.

The first problem encountered was that the BlueRadio chip was not advertising, or more specifically broadcasting wirelessly. As mentioned, the development board came with firmware to use it as a BLE packet sniffer, but unfortunately, no packets were detected when the sensor board should have been advertising, and this proved that the first version board was not transmitting. This problem was most likely related to the signal routes running under the antenna. These routes were a problem because they could either disrupt the electromagnetic properties of the antenna by grounding or carrying the signal along the routes. The traces could also unintentionally extend the antenna length, which would affect the transmission frequency.

Though the wireless functionality was affected, the rest of the BlueRadio
chip was fully functional, including the microcontroller. Between the microcontroller, the memory, and the debug header, it was possible to continue testing and debugging. The next problem to be encountered was with the shared chip select line between the onboard memory and the accelerometer. It was discovered that when the sensor board was in a mode that powered down the accelerometer by connecting the VDD inputs to GND, the chip select lines were pulled low as well, causing the memory to be inaccessible. By disconnecting the line connecting the accelerometer chip select from the rest of the circuit, the onboard memory became functional.

The last problem was discovered during the extended wireless evaluation. It was discovered that the ADP2108 voltage converter was outputting a much higher voltage (3.7V) than it should (3.3V). The first thing was to take a look at the part number and double check that it was the correct voltage converter. After confirming the part was correct, other boards were analyzed. At first, all of the boards functioned correctly, but after extended use, the boards would no longer power on when charged, and they all had the problem with the voltage converter outputting 3.7V instead of 3.3V. After double checking the datasheet and not being able to find any problems with the supply voltage and current it required, it was decided that the chip should be replaced anyway for a converter that was less noisy and more reliable.
4.1.3 Version 2 Design

The second version of the sensor board was slightly easier because most of the schematic and many of the parts stayed the same, but because a few of the parts were changed and the previous layout design had some critical faults, such as the routing under the antenna, a complete layout redesign was necessary. This redesign also created an opportunity to improve the sensor board by using four metal layers instead of two. What this means is the first revision only had metal layers on the top and bottom of the PCB for routing, but four layers allows for two addition layers of routing in the middle of the circuit board, or those layers can be used as metal planes instead, which can be helpful when a circuit requires many parts to connect to the same source. GND and VDD are two sources used frequently throughout the sensor board’s circuit, so it made most sense to create metal planes for those in order to reduce noise, reduce resistance, reduce heat, make routing easier, and help protect parts like the BlueRadio antenna and the eCompass from the magnet field interference. The top and bottom layers were still used for most of the routing, but in a few instances, it was necessary to jump down to one of the middle layers. The top middle layer was used as the ground plane in order to be closest to the antennas and magnetometer in order to ground as much interference as possible and isolate the parts for the supply plane.

One source of the magnetic field interference could come from ADP2108 step-down voltage converter because it is a switching power supply. Between the noise and previous problems with the chip burning out, it was decided to replace
the voltage converter with a low noise, low dropout regulator. This would also help to diagnose the incorrect voltage problem by testing if it was the converter that was bad and burning out or if it was the fault another part of the board. The selected replacement chip was the Maxim MAX8887 linear regulator [Maxim Integrated, 2009]. At low currents, 100mA, the voltage regulator is low noise and has a very low dropout voltage of about 40mV, which matches the needs of the circuit.

In addition to the new voltage control, a power switch was added to the design so that the board wasn’t always power on. The new switch would be a single pole, double throw connecting the battery voltage output to either the regulator or no connection. No connection was chosen over connecting to ground in order to reduce power loss when the board was turned off. The previous single pole, double throw switch used to set the board to firmware upgrade mode would be replaced by a jumper on the debug header to connect PIO_14 to VDD or leave it open.

A smaller double pole, triple throw switch that only selected the mode, and no longer affected the eCompass, was selected to replace the three-position mode selection switch. The chip select problem between the eCompass’ accelerometer and the onboard memory was solved by sharing the memory’s chip select with one of the ADC lines. This was possible because the memory is only used during digital read and broadcast mode, when the ADC lines are not used. If the memory chip select were pulled low, there would be no SPI activity, so the
memory would not be affected.

Lastly, the micro USB header was replaced by the flipped version of itself so that the pins coming off the part could line up better with the FTDI FT230XS programming chip. The layout of the parts could be mirrored and the routing would not need to leave the same side of the board in an effort to reduce noise and maintain equalization with the clock.

![Schematic for the second version of the sensor board](image)

Figure 12: Schematic for the second version of the sensor board with changes from the previous version highlighted in blue rectangles. The design is separated into sections: USB programming (top-left), battery management (top-right), onboard memory (middle-left), voltage regulator (middle), debug header (middle-right), BlueRadio connections (bottom-left), eCompass connections (bottom-middle), and gyroscope connections (bottom-right)

### 4.1.4 Version 2 Testing and Debugging

The second revision yielded positive results, ensuring the revised schematic and layout were correct. The board measured 1.65 x 1.69 in², resulting
Figure 13: Layout for the second version of the sensor board. Top layer routing is shown in teal, second layer is shown in striped teal, third layer is shown in green, and bottom layer routing is shown in yellow. Parts are outlined in grey and vias are colored purple.
in a reduction of about 35% in size.

The new voltage regulator was operating without fault, and it also wasn’t affecting the magnetometer or BLE antenna with any measurable artifacts. The BlueRadio board was able to broadcast and transmit BLE packets successfully, making debugging and testing much easier. In order to test the functionality of board, including wireless communication, an iOS application was developed using Apple’s Core Bluetooth Framework to check the BLE communication since many iOS devices have a BLE chip built-in and the application was able to be developed along side the sensor board development using the BlueRadio development board [Apple Inc., 2013, Apple Inc., 2014]. Figure–14

4.1.5 Version 3 Design

With the board design functioning properly, this revision was used to reduce the footprint of the layout. In order to do so, most of the debugging elements were removed and both sides of the PCB were used for part placement instead of just the top. This move allowed for separation of the power and serial components from the Bluetooth and IMU parts, which meant less chance for interference between noisy components and sensitive signal components due to the ground plane between the bottom and top layers. Everything on the top layer only needs to pull 3.3V, which was available from the VDD plane. The battery and USB headers were both attached to the bottom layer, so the higher voltages and currents would not even make it up to the top layer with the other more sensitive parts.
Figure 14: iPad app with the current values and plot of the nine degrees of freedom on the sensor board.
The large debug header was removed because most of the pins were no longer necessary for debugging since the schematic remained primarily unchanged. The ADC lines were given their own four-pin header in order to read in external analog signals to broadcast. The digital input mode was removed in an effort to reduce space by getting rid of the onboard memory, but also because it was not being used. Partly for this reason, the three-position switch was removed and replaced by a spot for a zero ohm resistor that would be used to set the mode, IMU broadcast or analog input broadcast. The reset switch was also removed and replaced by a jumper, in an attempt to reduce space. Most of the LEDs were removed and replaced by a single two-color LED that would be used to show battery and Bluetooth status; green for connected to a BLE device, orange for disconnected, red for charging, and blinking for low battery. In order to get the battery states, two status lines from the TI LM3658 charging chip were routed to two input pins on the microcontroller. This was possible because two pins were now open due to LEDs and the onboard memory being removed.

4.1.6 Version 3 Evaluation and Debugging

This revision ended up being less than half the size of version 2 (~57% smaller) and over 70% smaller than version 1, measuring in a 1.11 x 1.09 in\(^2\). Similar to version 2, this revision also used four layers, the two middle layers functioning as GND and VDD planes with limiting routing on those layers.

While evaluating this version, an issue with the charging setup was discovered. After fully charging, the battery charging chip would not enter state
Figure 15: Schematic for the third version of the sensor board with changes from the previous version highlighted in blue rectangles. The design is separated into sections: USB programming (top-left), battery management (top-right), onboard memory (middle-left), voltage regulator (middle), debug header (middle-right), BlueRadio connections (bottom-left), eCompass connections (bottom-middle), and gyroscope connections (bottom-right)
Figure 16: Layout for the third version of the sensor board. Top layer routing is shown in teal, second layer is shown in striped teal, third layer is shown in green, and bottom layer routing is shown in yellow. Parts are outlined in grey and vias are colored purple.

Figure 17: Charging current and voltage cycle. Expected voltage plots are shown in black and the observed voltage and current are shown overlaid in blue and red, respectively [Texas Instruments, 2007].
to maintain charge if powered on. Instead, it would eventually timeout and turn off, which would result in the board going back into normal operation. As the power drained while the board was still on, the battery would not recharge due to the time-out, and the LM3658SD chip remained off until it was power cycled. Figure–17 shows the different charging states of the LM3658SD chip and the current and voltage characteristics overlaid in blue (current) and red (voltage). In order to debug this, the stat lines were monitored in firmware and then used to control the LEDs to indicate the state of the charging chip. The LEDs were used specifically to check for the bad battery state.

In order to fully evaluate this issue, three fully drained batteries were recorded during their charging, and they all exhibited the same response. The charger would successfully enter and exit the prequalification stage since the timeout for that stage is 30 minutes and the stat lines did not change in that time. The next stage, constant current constant voltage (CCCV), is where the bulk of the charging took place, and it is most likely where the problem was because the timeout timer for this stage was 10 hours long and the bad battery event would happen at the end of this time. The battery would never make it into maintenance mode, which would mean either the voltage of the battery was never greater or equal to 3.0V or the end-of-charge current, 9mA, was never reached. The battery voltage was monitored during charging, and it exceeded 3.0V, so it would seem the problem was the current not getting low enough.

Various charging tests were performed to determine the reason for the
high current draw. It wasn’t until the rest of the board was powered down by holding the microcontroller in reset mode that the battery charged properly. This was determined by attaching meters to monitor the battery voltage and current during charging. While reset was held on the microcontroller, the battery reported values inline with what was expected from the datasheet. The meters showed the battery make it to fully charged, then properly navigate maintenance period by periodically kicking on to boost the voltage and current again and cycle through the states, while avoiding bad battery mode.

Figure 18: Schematic for version 3.1 of the sensor board with changes from the previous version highlighted in blue rectangles. The design is separated into sections: USB programming (top-left), battery management (top-right), onboard memory (middle-left), voltage regulator (middle), debug header (middle-right), BlueRadio connections (bottom-left), eCompass connections (bottom-middle), and gyroscope connections (bottom-right)
Figure 19: Layout for version 3.1 of the sensor board. Top layer routing is shown in teal, second layer is shown in striped teal, third layer is shown in green, and bottom layer routing is shown in yellow. Parts are outlined in grey and vias are colored purple.

Figure 20: Relative size comparison between sensor board versions. From left to right, version 1, version 2, version 3.1, and a quarter for scale.
4.1.7 Version 3.1 Design

Version 3.1 is the newest board design, and the reason it is not a new complete version is because this board does not offer much change besides a fix to the battery charging problem discovered in version 3. To fix this problem, the STAT1 output of the TI charging chip is tied to reset, which will pull the pin to ground and disable power to the rest of the board. This fixes the charging problem, but it unfortunately disables use of the LEDs during charging. In order to update the microcontroller firmware while USB is plugged in, which would cause the battery to start charging, a jumper was added between STAT1 and the microcontroller reset pin. As new techniques are explored for reducing current pull during charge, it is likely the charging solution will change, but this acts as a temporary fix.

Another small change was for the board to always be on when it was not connected to a charger. The removal of the power switch freed up space in all directions, and a smaller push button switch for resetting the microcontroller was re-added in order to easily cycle the power. The battery header was also removed and replaced by two pads that the battery leads could connect to further reduce the thickness of the board. By reducing the thickness, the IMU could be closer mapped to the user and there would be less chance of it interfering with any movement during wave field synthesis interaction.
<table>
<thead>
<tr>
<th>PIO</th>
<th>Pin</th>
<th>Port</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>21</td>
<td>P0_0</td>
<td>IN</td>
<td>ADC</td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>P0_1</td>
<td>IN</td>
<td>ADC</td>
</tr>
<tr>
<td>2</td>
<td>23</td>
<td>P1_0</td>
<td>OUT</td>
<td>Green LED</td>
</tr>
<tr>
<td>3</td>
<td>26</td>
<td>P0_6</td>
<td>IN</td>
<td>STAT2</td>
</tr>
<tr>
<td>4</td>
<td>28</td>
<td>P2_1</td>
<td>IN</td>
<td>Mode Select</td>
</tr>
<tr>
<td>5</td>
<td>24</td>
<td>P1_1</td>
<td>OUT</td>
<td>Red LED</td>
</tr>
<tr>
<td>6</td>
<td>25</td>
<td>P0_7</td>
<td>IN</td>
<td>ADC</td>
</tr>
<tr>
<td>7</td>
<td>29</td>
<td>P2_2</td>
<td>IN</td>
<td>STAT1</td>
</tr>
<tr>
<td>8</td>
<td>27</td>
<td>P1_2</td>
<td>OUT</td>
<td>CS1</td>
</tr>
<tr>
<td>9</td>
<td>22</td>
<td>P1_3</td>
<td>OUT</td>
<td>CS2</td>
</tr>
<tr>
<td>10</td>
<td>5</td>
<td>P0_2</td>
<td>IN</td>
<td>SPI_MISO</td>
</tr>
<tr>
<td>11</td>
<td>6</td>
<td>P0_4</td>
<td>OUT</td>
<td>SPI_CSB</td>
</tr>
<tr>
<td>12</td>
<td>7</td>
<td>P0_5</td>
<td>OUT</td>
<td>SPI_CLK</td>
</tr>
<tr>
<td>13</td>
<td>8</td>
<td>P0_3</td>
<td>OUT</td>
<td>SPI_MOSI</td>
</tr>
<tr>
<td>14</td>
<td>19</td>
<td>P2_0</td>
<td>IN</td>
<td>FW Upgrade EN</td>
</tr>
</tbody>
</table>

Table 4: Port mapping for version 3.1

4.1.8 Version 3.1 Evaluation and Debugging

This revision is the same size as the previous version since only a couple connections and parts were changed. The charging issue has been tested and it has been fixed, but as mentioned previously, new designs are being tested for a more permanent fix in the next revision.

4.2 Software Development

After version 2 of the board design was fabricated and assembled, it was possible to start software development since BLE communication was possible. The firmware was written to transmit the data via BLE, so without a program to receive that information, the hardware would serve little purpose.

4.2.1 Wireless Software Development

Two of the first things considered before starting the coding process were which operating system would the software be developed for and were there any existing libraries written for C or C++ for BLE stack management. OS X was appealing because code had already been developed to communicate and read
data from the TI CC2540 development board and most of that could be reused to write a Mac application, also as part of that, Apple already has a well documented objective-C library for BLE communication. Unfortunately, WFS Designer was written for Windows, and though in theory it could be ported to OS X, there was no way to know how long it would take to fully port things. For that reason, Windows was selected instead.

With the operating system selected, the next step was to find a freely available BLE library written for C or C++. Fortunately, this problem was solved during the selection of the BLE dongle. Since the computer used for software development did not have a Bluetooth Low Energy chipset built-in, a wireless USB dongle was necessary. The Bluegiga BLED112 was ideal because Bluegiga also offered a free Software Development Kit (SDK), and the SDK included example code. The sample code would serve as a good starting place for the communication program.

4.2.2 BLEConnect

The BLEConnect program was written entirely in the C programming language to match the language used for Bluegiga SDK. As mentioned, the program was developed for Windows and it is used to connect to and grab data from one of the wireless IMU boards. BLEConnect can either function automatically and connect to the first BLE IMU it receives an advertisement packet from, or it can list available sensor boards to connect to and the user can select the one they wish to connect to. The former design is used when combined
with the WFS Designer program because one-way communication is only set up. WFS Designer can only receive the standard output stream from BLEConnect and it is not set up to send information through BLEConnect’s standard input stream to the program. When BLEConnect is ran on its own, the user can display and select a BLE sensor to connect to by passing the argument “list”. Without any arguments, BLEConnect will go into autonomous mode.

BLEConnect first checks that the communication (COM) port is connected to a BLE transceiver and then opens a connection with that port for transmitting and receiving data. Next the program checks for the “list” argument. If it is given, the program sets a flag to enter user-controlled mode, otherwise the program continues autonomously.

After the connection with the BLE dongle has been established and the mode determined, BLEConnect will beginning reading in BLE packets. It will try 15 times to read in any advertisement packets and if it is in user-control mode, the different sensor board address, names, and signal strengths will be listed in order to select one to connect to, as shown in figure 21. If the program is in auto mode, it will connect to the first sensor it finds, as mentioned previously.

Once a sensor has been selected, the program establishes a connection with the sensor, which is mostly handled by the SDK. After the connection is set up, packets can are read in normally and sorted through for data. The first thing to do is check that the correct service UUID, since that is how data is indexed within BLE. Within the service UUID location, a characteristic UUID can be
Figure 21: Screenshot of BLEConnect running and connecting to a sensor device. It first displays all BLE devices in range, and then it connects to one with the proper service UUID for the sensor board (0xffe0). Next, it looks for the correct characteristic UUID (0xffe1), and once that is found, it begins pulling data from that location in incoming packets.

found that is the location of the sensor data to be read. These UUID addresses are set in the firmware of the TI C2540 microcontroller, so since they are known ahead of time, they can be set in the BLEConnect code as well.

The data is then read in the same order it is set in the firmware; the first six bytes are accelerometer data, the next six bytes are gyroscope data, and the last six bytes are magnetometer data. Before being displayed or passed along to a delegate function, the values are converted from hex to decimal to make it easier for the user to read.

4.2.3 WFS Controller

WFS Controller is the name for the combination of WFS Designer by Matt Montag and BLEConnect. WFS Designer performs all of the wave field synthesis output control, while BLEConnect handles input. As previously mentioned, WFS Controller is a fork of WFS Designer, which has been modified
in a few ways. First, since WFS Designer starts out by creating a sub-process of BLEConnect so that BLEConnect can continue to run in the background while WFS Controller is running. This allows a user wearing the sensor board to be able to interact without any affect to the wave field synthesis part of the code, except for the parameter controlling source location.

![Screenshot of WFS Controller running with a single source loaded and two output channels.](image)

Another addition to the program is the ability to turn sensor board interaction on and off. If the sensor input mode is turned off, source location can be selected manually, given that a source has been added. If the sensor input is turned on, the program begins to listen for messages from the BLEConnect code running in the background. A socket connection is set up to listen for strings
outputted by the BLEConnect program using stdout and when a new string is available, a function is called within WFS Controller to update the source position with the values from the new string.

Currently, only the accelerometer values are being used to change the positions, but ideally a filter such as a Kalman filter could be used that would combine all three sensor inputs to create a much more exact and higher resolution movement profile. This filter output could then be used to control the source coordinates in three dimensions.
5

Evaluation

The focus of the testing and evaluation was the sensor board, not the wave field synthesis system since wave field synthesis was only an application to demonstrate the capabilities of the controller.

The focus of the testing and evaluation was primarily on capabilities and usability of the sensor board. The three focus points for evaluation were accuracy, connectivity, and power, and a small usability survey was conducted at the end. In order to test the accuracy of the device, it was assumed that the accuracy of the individual sensors was correct as stated in their respective data sheets. The focus was of the accuracy test was on the noise environment of the board, and whether or not it significantly affected the output signals. The range has been previously stated in Bluetooth Low Energy specs as 30 to 100m, so the focus of this test was to gauge the range of connectivity in environments with different amounts of interference in the 2.4 GHz spectrum. Lastly, power consumption was monitored in order to assess battery life and the efficiency of the design.

5.1 Sensor Noise

Noise can become a big issue when observing high resolution IMU in situations such as the hand movement and gesture recognition. In order to evaluate the presence of the noise in the sensor data, the sensor board was placed in different environments with varying amounts of dynamic magnetic interference
(magnetic fields not attributed to the Earth’s static magnetic field). This activity was measured with an electromagnetic interference (EMI) detector, and though this was mostly to see the affect on the magnetometer, all sensors were still evaluated this way.

5.1.1 Testing Setup

In order to get the raw sensor data, the iOS application that was used to monitor BLE activity and sensor data was modified to save 100 samples of each sensor at given time. For each location, 10 groups of samples were taken. The min and max values, range, and noise floor were calculated for each group of samples for the individual sensor dimensions. Since the measurements were taken when the board was at rest, meaning it was placed on a motionless surface and not disturbed, any change in value could be attributed to noise. By taking the range, activity such as gravitational and magnetic pull would not be a factor because those values would be constant offsets, so they would not affect the variance attributed to noise. The equation used to calculate the noise floor is

\[ NF_{dB} = 20 \times \log_{10}(\text{range}_{\text{noise}}/\text{range}_{\text{total}}) \]

where \( \text{range}_{\text{total}} \) is the total range outputted by the sensor based on the dimensional resolution \( 2^{\text{number of bits}} \) - accelerometer resolution is 10bit, gyroscope resolution is 16bit, magnetometer x and y resolution is 13bit, and magnetometer z resolution is 15bit), \( \text{range}_{\text{noise}} \) is the previously calculated peak to peak range, and the output is in decibels (dB). It is important to note that this would be the noise floor for the worst case scenario. Often it is the case that the noise would follow more of a Gaussian
distribution instead of a even distribution calculated using peak to peak.  

To study the effect of magnetic field interference on the sensors, the sensor was evaluated in locations with high, medium, and low magnetic field levels. All three areas were outside and the detected fields were within the 16 to 2000 Hz bandwidth. The low field was 10nT of interference, the medium was 110nT of interference, and the high was 900nT of interference, each with an order of magnitude difference between them.

5.1.2 Results

<table>
<thead>
<tr>
<th>Sensor and Axis</th>
<th>Magnetic Field Strength (nT)</th>
<th>Average Noise Range</th>
<th>Standard Deviation of Noise</th>
<th>Average Noise Floor (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accelerometer x</td>
<td>10</td>
<td>44.8</td>
<td>8.87</td>
<td>-27.26</td>
</tr>
<tr>
<td></td>
<td>110</td>
<td>44.5</td>
<td>8.68</td>
<td>-27.26</td>
</tr>
<tr>
<td></td>
<td>900</td>
<td>43.7</td>
<td>8.62</td>
<td>-27.42</td>
</tr>
<tr>
<td>Accelerometer y</td>
<td>10</td>
<td>40.4</td>
<td>8.10</td>
<td>-28.15</td>
</tr>
<tr>
<td></td>
<td>110</td>
<td>38</td>
<td>7.77</td>
<td>-28.68</td>
</tr>
<tr>
<td></td>
<td>900</td>
<td>40.9</td>
<td>8.04</td>
<td>-28.03</td>
</tr>
<tr>
<td>Accelerometer z</td>
<td>10</td>
<td>40.5</td>
<td>7.91</td>
<td>-28.09</td>
</tr>
<tr>
<td></td>
<td>110</td>
<td>37.4</td>
<td>7.81</td>
<td>-28.76</td>
</tr>
<tr>
<td></td>
<td>900</td>
<td>40.7</td>
<td>7.64</td>
<td>-28.18</td>
</tr>
<tr>
<td>Gyroscope x</td>
<td>10</td>
<td>204.9</td>
<td>22.71</td>
<td>-53.82</td>
</tr>
<tr>
<td></td>
<td>110</td>
<td>303.5</td>
<td>55.96</td>
<td>-48.14</td>
</tr>
<tr>
<td></td>
<td>900</td>
<td>70.8</td>
<td>11.61</td>
<td>-60.71</td>
</tr>
<tr>
<td>Gyroscope y</td>
<td>10</td>
<td>1211.5</td>
<td>213.68</td>
<td>-35.25</td>
</tr>
<tr>
<td></td>
<td>110</td>
<td>328.4</td>
<td>63.69</td>
<td>-46.55</td>
</tr>
<tr>
<td></td>
<td>900</td>
<td>151.9</td>
<td>16.67</td>
<td>-56.56</td>
</tr>
<tr>
<td>Gyroscope z</td>
<td>10</td>
<td>92.6</td>
<td>12.92</td>
<td>-60.12</td>
</tr>
<tr>
<td></td>
<td>110</td>
<td>166.8</td>
<td>23.64</td>
<td>-54.38</td>
</tr>
<tr>
<td></td>
<td>900</td>
<td>60</td>
<td>9.73</td>
<td>-63.04</td>
</tr>
<tr>
<td>Magnetometer x</td>
<td>10</td>
<td>15.9</td>
<td>4.30</td>
<td>-54.29</td>
</tr>
<tr>
<td></td>
<td>110</td>
<td>16.6</td>
<td>4.24</td>
<td>-53.92</td>
</tr>
<tr>
<td></td>
<td>900</td>
<td>17.5</td>
<td>4.49</td>
<td>-53.48</td>
</tr>
<tr>
<td>Magnetometer y</td>
<td>10</td>
<td>15.5</td>
<td>4.02</td>
<td>-54.68</td>
</tr>
<tr>
<td></td>
<td>110</td>
<td>15.6</td>
<td>4.00</td>
<td>-54.65</td>
</tr>
<tr>
<td></td>
<td>900</td>
<td>17.4</td>
<td>4.85</td>
<td>-53.53</td>
</tr>
<tr>
<td>Magnetometer z</td>
<td>10</td>
<td>20.1</td>
<td>5.48</td>
<td>-64.37</td>
</tr>
<tr>
<td></td>
<td>110</td>
<td>21.8</td>
<td>5.61</td>
<td>-63.84</td>
</tr>
<tr>
<td></td>
<td>900</td>
<td>25.3</td>
<td>6.17</td>
<td>-62.42</td>
</tr>
</tbody>
</table>

Table 5: Results of noise evaluation in areas of varying magnetic field strengths.

Table 5 shows the average range and noise floor for each sensor dimension in the various magnetic fields. Surprisingly, the magnet field interference does not
appear to have a negative affect on any of the sensors. The gyroscope noise floor even drops in the higher interference environments. This could be attributed to the affect of magnetic fields outside the measured bandwidth. Lastly, one explanation that the magnetometer noise was unaffected could be that local magnetic forces could possibly distort or bias the readings, instead of affecting the noise level. The range of noise is only relative, and an overall affect to the manometer reading would not be noticed when only looking at the spread of noise values.

5.2 Range and Connectivity

The sensor board is only useful when it’s connected to a receiving node to interpret the data, so evaluating the connectivity range of the device was important for understanding the limitations of the board. To analyze this, the device connectivity was examined in multiple environments with varying amounts of activity in the 2.4GHz spectrum to see how far away the board could get before disconnecting to the master node.

5.2.1 Testing Setup

Connectivity range was tested by connecting the sensor board to an iOS device via Bluetooth Low Energy and then moving the board away from the device and recording the distance of disconnect. First, the testing area was scanned for wireless activity and the decibel level of the strongest source was recorded. Next, the sensor board was connected to the iOS device (an iPod Touch) and placed on the ground. The sensor remained stationary and the iPod
Touch was ambulated increasingly further away from the sensor. The iPod Touch could show both whether data was being received from the IMU and display the received signal strength indication (RSSI) value, the measured power of the received BLE signal. The connectivity distance was measured with a surveyor’s wheel. This was done three times in three separate locations with varying amounts of activity in the same band as BLE (2.4GHz). The first location was the one with the least amount of activity, only a couple wireless sources and the highest level was -85dB. The next location had between five and ten wireless nodes broadcasting and a peak of -75dB. The last testing location had close to 20 wireless sources and the most powerful source was recorded at -65dB, also each an order of magnitude increase.

5.2.2 Results

<table>
<thead>
<tr>
<th>Peak Noise Level (dB)</th>
<th>Connectivity Distance (ft)</th>
<th>Average Distance (ft)</th>
<th>Standard Deviation (ft)</th>
</tr>
</thead>
<tbody>
<tr>
<td>-85</td>
<td>113.67</td>
<td>104.97</td>
<td>7.70</td>
</tr>
<tr>
<td></td>
<td>99</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>102.25</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-75</td>
<td>91.67</td>
<td>90.83</td>
<td>1.10</td>
</tr>
<tr>
<td></td>
<td>89.58</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>91.25</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-65</td>
<td>83.33</td>
<td>84.83</td>
<td>3.42</td>
</tr>
<tr>
<td></td>
<td>82.42</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>88.75</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 6: Results of connectivity evaluation in areas of varying wireless activity.

Table 6 shows the results of testing the BLE connectivity in areas with various levels of wireless activity. From the results, it can be concluded that it is harder to maintain connectivity over further distances when there are stronger wireless nodes in the area. It is important to add that these distances are from when the connection was completely dropped. In most cases, the data-rate began
to suffer around the halfway point or when the RSSI value dropped below -90dB. When this happened, the connection was still maintained, but not all BLE packets were received, so some of the data was lost and it only got worse as the sensor board got further away.

5.3 Battery Evaluation

The battery evaluation came in two parts, 1) testing current and voltage draw during charging to ensure that the curves matched the datasheet, and 2) testing battery length on a full charge in both connected and disconnected situations. The voltage and current statistics were especially helpful for solving the charging problem on sensor board version 3.0. As previously mentioned, the problem was the battery would fully charge, but once charged, the charging chip would fail to pass one of its check points to go on to the state that maintained the charge. This is suspected to be due to a high current draw while the board is powered on. Instead of going into the maintenance state, the charging chip would go into a bad battery state, and the board would drain the battery since it was still on. The charging chip would need to be power cycled in this case before it could begin charging again.

5.3.1 Testing Setup

To study the current draw during charging, an ammeter was placed in series with the battery connection to the board. A voltmeter was connected in parallel with the battery connection for evaluating the voltage. Before the start of the test, the battery was fully drained to get the full charging curves. Voltage
and current values displayed on the meters were then recorded every few minutes throughout the charging.

The battery life was tested for both connected and disconnected sensors. For the connected test, four sensors were wireless connected to an iOS application, and then sent data to the master node as with normal operation. The disconnected sensors were not connected to anything, so they were stuck in advertisement mode. Both testing setups were measured with a stopwatch and a camera to record the process.

5.3.2 Results

The graph displayed in figure–23 shows the voltage and current curves during charging. This graph matches the curves displayed in the datasheet for the charging chip. The current starts out high and slowly drops off as the battery full charge, and the voltage subtly rises during the process.

![Charging Voltage and Current vs Time](image)

Figure 23: The voltage and current curves recorded while charging the sensor from completely drained to fully charged.
Table 7: Results of 110 mAh battery life evaluation for both connected and disconnected sensors.

Table 7 shows the battery life times of the four sensor boards for both the connected and disconnected tests. Surprisingly, the battery life is about 33% shorter for the disconnected sensors. One reason this is short could be because the board is sending larger packets when it is advertising than when it is sending sensor data, but it is also still gather all the sensor data even though it is not sending it anywhere. Another reason could be because of the LEDs. The two LEDs draw about 20mA of current each, which is a significant amount of the overall current draw. This is less likely the case for causing the battery life difference between the connected and disconnected states, but it is likely to be reducing the battery life significantly overall.

5.4 Usability Survey

In order to test the effectiveness and usability of the sensor board as a controller for the wave field synthesis system, a small survey was conducted with 14 participants. Each participant was asked to complete the survey shown in figure–24. The participant was given the sensor board and then instructed to turn the board in specific directions and note where they heard the sound source as a result. The wave field synthesis environment involved an array of 16 4” loudspeakers in a line array spaced 5” apart (center to center). Table–8 shows
the percentage of participants that answered each question correctly, as well as the percentage greater than chance. For this test, chance for each question would be 33.33% since each question has three answers to choose from, left, center, and right.

5.4.1 Results

<table>
<thead>
<tr>
<th>Question</th>
<th>Answered Correctly</th>
<th>Greater Than Chance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Question 1</td>
<td>92.86%</td>
<td>178.57%</td>
</tr>
<tr>
<td>Question 2</td>
<td>92.86%</td>
<td>178.57%</td>
</tr>
<tr>
<td>Question 3</td>
<td>71.43%</td>
<td>114.29%</td>
</tr>
</tbody>
</table>

Table 8: Results of testing wave field synthesis control usability with the wireless sensor board. Questions refer to the questions asking for participant to note where they hear a sound source and are in order appearance as shown in figure–24.

It is clear from the table–8 that the sensor board works as a viable wave field synthesis controller since each question generated a response more than two times greater than chance. I believe the uncertainty of some of the questions came from the wave field synthesis system instead of the controller since only a small speaker array was used. The other part of that is the sensor board had too wide of a range of control allowing the source to get so far away in both the left and right directions that it became difficult to distinguish due to the soft sound. This could be fixed by either shortening the control distance in the code, or by making it a user option. The user would be able to set resolution or the total area they would like the controller to cover.
Wave Field Synthesis Usability Test

How many years of music experience do you have?
Do you have any problems hearing?
If yes, please explain:

Please hold the board so that the tape side is facing the speaker array with the tape side facing up:

Which direction do you hear the sound source? (circle one)

Left  Center  Right

Please turn the board 90° clockwise as show to the right.

Which direction do you hear the sound source? (circle one)

Left  Center  Right

Please turn the board 180°counter-clockwise as show to the right
Which direction do you hear the sound source? (circle one)

Left  Center  Right

Thank you for your participation!

Figure 24: A copy of the survey given to participants to evaluate the usability of the system. Participants would move the controller as noted in the survey and circle where they heard the sound source coming from.
6

Conclusion

The objectives of this thesis were to design a low-power wireless hardware sensor for use as a controller for wave field synthesis source positioning. As a result of this project, a controller capable of monitoring 9-degrees of movement and relaying the data wirelessly was created. That controller was measured to meet and exceed necessary range of effective use while maintaining wireless connection. The controller passed power evaluations by demonstrating full functionality for over three and a half hours. Software was written to receive the IMU data from the sensor wirelessly and pass that information to the wave field synthesis control system, and the full system was tested for baseline usability. There are still improvements and additions that could be made to both the sensor and the wave field synthesis control system, but the goals initially set were successfully achieved.

6.1 Future Work

There are multiple areas of this project that could be improved upon and added to. These areas can be split up into two main focuses, the sensor board (hardware) and wave field synthesis control (software). As outlined in the previous testing section, there are a few parts of the sensor board that could use improvement, and the software is only at a basic functioning model.
6.1.1 Sensor Board

It is clear from the connectivity testing that the sensor board needs to become more power efficient. The main area of focus is the inactive mode since it drains the most power. Because the individual sensors are not necessary during this period, they could be put into a sleep mode during advertisement. Advertising could also become less frequent, which would reduce some power consumption and affect connection very little, especially since the connecting device could be configured to search longer for devices because the master node’s power consumption is less of a concern. Lastly, if the board detects inactivity over a long period of time (maybe 20 minutes), it could power everything down except for the accelerometer, and the accelerometer could issue an interrupt to the microcontroller as soon as the board is disturbed. This would most likely conserve the most power, but it would also require the user to move the IMU before connecting to it again.

As mentioned in the evaluation chapter, the LEDs are drawing a significant amount of current. The board can also reduce power consumption overall by pulse width modulating (PWM) the LEDs. Pulse width modulation means instead of leaving the LED on all the time, it would be on for a certain amount of time and then off. Normally this is measured as a percentage, so a PWM of 40% could mean the LED was on for four cycles and then off for six cycles. This would reduce the amount of current the LED was drawing because it wouldn’t be drawing current while it was off, so theoretically, a PWM of 40%
could reduce the LED current consumption by 60%.

The sensor board can continue to be improved in other ways as well. Removing unnecessary analog input/output components, especially those that require headers, can reduce the overall size. The BlueRadio BLE chip could be replaced with individual components, and the antenna could be built into the board with a metal layer to reduce size.

6.1.2 Wave Field Synthesis Control

The source position control can be improved greatly by using all nine degrees of freedom from the wireless IMU. Currently, the system only uses the accelerometer data to control the synthesized source, but by using filtering techniques, such as a Kalman filter, it would be possible to incorporate all the sensor outputs in a way that reduces noise and improves accuracy.

The ability to choose a BLE device to connect to from WFS Controller interface, instead of automatically connecting to the first sensor board found would be a great addition. In order to do this, a two-way communication must be set up, which currently doesn’t exist. WFS Controller is able to listen to the outputs of BLEConnect, but BLEConnect cannot receive anything from WFS Controller. A way to correct this would be to combine both programs into one, and then set up a messaging queue for communication.

Lastly, Bluetooth Low Energy is very robust, as outlined in the proposed system section. Because of this, it is possible to connect multiple sensors to a single master node, and this strategy could be used to control multiple virtual
sound sources. The ability to control multiple sources with multiple sensors in the same system would be a new idea, and possible with additional code to BLEConnect and WFS Designer to support reading from multiple slave nodes.

6.2 Other Applications

Though this sensor board was designed for wave field synthesis interfacing, this was only one application for how this controller could be applied to computer and audio interfacing. Other possibilities include gesture recognition and movement profiling. One way this sensor is already being applied to a completely different application is in research project that focuses on motion analysis of human ambulation. The sensor is used in the project to gather movement data from both the patient’s prosthetic leg and normal leg in multiple body locations.

6.3 Conclusion

In conclusion, this sensor was designed with wave field synthesis control in mind, but the areas of application are endless. With the above improvements to the hardware and software, resolution and control can be greatly improved, and the system can be expanded to new dimensions.
LIST OF REFERENCES


[ZigBee Alliance, 2013] ZigBee Alliance (2013). Understanding ZigBee RF4CE.