2012

SSVEP-based brain computer interface using the Emotiv EPOC

Brian J. Zier
Eastern Washington University

SSVEP-BASED BRAIN COMPUTER INTERFACE USING THE EMOTIV EPOC

By

Brian J. Zier

In partial fulfillment of the

requirements for the degree

Master of Science

A thesis presented to

Eastern Washington University

Cheney, Washington

Summer 2012

THESIS OF BRIAN J ZIER APPROVED BY

_____ DATE: _____
PAUL SCHIMPF, PHD. - GRADUATE STUDY COMMITTEE CHAIR

_____ DATE: _____
BOJIAN XU, PHD. - GRADUATE STUDY COMMITTEE MEMBER

_____ DATE: _____
DORIS MUNSON - GRADUATE STUDY COMMITTEE MEMBER

MASTER'S THESIS

In presenting this thesis in partial fulfillment of the requirements for a master's degree at Eastern Washington University, I agree that the JFK Library shall make copies freely available for inspection. I further agree that copying of this project in whole or in part is allowable only for scholarly purposes. It is understood, however, that any copying or publication of this thesis for commercial purposes, or for financial gain, shall not be allowed without my written permission.

Signature: _____

Date: _____

TABLE OF CONTENTS

I. Introduction..... 1

1.1 Related Work..... 3

1.2 Goals..... 5

II. Methods & Design..... 7

2.1 Overview..... 7

2.2 Tools..... 7

2.2.1 Headset..... 8

2.3 User Interface..... 9

2.4 Data Processing..... 11

2.4.1 Data Acquisition..... 11

2.4.2 Preprocessing..... 12

2.4.3 Processing..... 13

2.4.4 Detection..... 16

2.4.5 Screen Drawing..... 17

2.5 Task Scheduling..... 18

2.6 Data Logging & Playback..... 21

III. Results & Discussion..... 24

3.1 SSVEP Verification..... 24

3.2 Display Timing Verification..... 25

3.2.1 Refresh Rate Verification..... 25

3.2.2 Missed Flips..... 27

3.3 Baseline Removal..... 28

3.4 Threshold Determination..... 29

3.5 Response Latency Testing..... 30

3.6 BCI Testing..... 31

IV. Conclusion..... 35

References..... 37

Vita..... 39

I. INTRODUCTION

This research project aims to develop a Brain-Computer Interface (BCI) system which utilizes Steady-State Visual Evoked Potentials (SSVEP) to allow a user to control objects on-screen simply by looking at flickering controls. "Brain activity produces electrical signals detectable on the scalp, on the cortical surface, or within the brain. [BCIs] translate these signals into outputs that allow users to communicate without participation of peripheral nerves and muscles" [1]. Electroencephalography (EEG) is the measurement and recording of those electrical signals on the surface of the scalp. Event Related Potentials (ERP) "are those EEGs that directly measure the electrical response of the cortex to sensory, affective, or cognitive events" [2]. As cited by Beverina, SSVEP are a form of ERP and

> are natural responses for visual stimulations at specific frequencies. When the retina is excited by a visual stimulus ranging from 3.5 Hz to 75 Hz, the brain generates an electrical activity at the same (or multiples of the) frequency of the visual stimulus. They are used for understanding which stimulus the subject is looking at in case of stimuli with different flashing frequency [3].

Primarily these responses occur in the occipital region of the brain. The occipital lobe is located in the posterior region of the brain and contains the visual cortex. When a person focuses his/her attention on a visual stimulus presented at a particular frequency, the electrical potentials on the scalp near the occipital lobe are modulated by the given frequency. SSVEP-based BCI uses EEG to analyze the activity in the brain and identify those potentials.

With SSVEP-based BCIs, the user is presented with multiple visual stimuli to act as controls for the BCI system. This has been done in two significantly different ways.

First, flashing lights, such as LEDs, have been used [4]. This method is fairly simple, because the lights can be pulsed at any given frequency. This gives a great amount of control over which frequencies are chosen for the system and can be picked based on the strongest elicited response through experimentation. One limitation is that SSVEP is known to produce a somewhat less powerful response at harmonics (i.e. integer multiples) of the stimulus frequency. This means that the frequencies chosen for two separate controls should not be harmonics of each other because this will significantly complicate detection. If the user directs attention at one control, a response may be seen at one of its harmonic frequencies. This may appear as if the response is from one of the other controls which does not have the user's attention.

The second method for displaying the stimuli is to present the flickering controls on the computer screen itself [4]. This provides the advantage of the user being able to focus on-screen for both the stimulus/control as well as the feedback. This also allows for the controls to be something other than a simple light. For example, in the demonstration presented in this thesis, arrows are used as controls to indicate each control's specific functionality. There are at least two primary types of on-screen controls: single graphics stimuli and pattern reversal stimuli [4]. The demonstration presented in this thesis uses pattern reversal stimuli. The challenge with on-screen controls is that the frequency of the presented stimulus is restricted by the refresh rate of the monitor on which the controls are displayed. Available frequencies are limited to "subharmonics of the screen refresh rate" [4]. Because of this limitation and the aforementioned limitation of using stimulus frequencies that are not harmonics of each other, the choice of frequency for on-screen

controls is very small. This presents a challenge for introducing more than a small number of controls. Because there are so few available frequencies, the frequencies cannot necessarily be chosen exclusively based upon the ones which generate the greatest response.

1.1 Related Work

SSVEP-based BCI research is rapidly growing. The IEEE database was searched for "Steady State Visual Evoked Potential". Of all the SSVEP papers published through the IEEE since 1990, half of them have been published within the last 4 years. The results available from the 14 year span from 1990 to 2004 include 36 papers. From 2005 to 2012, 146 papers were published, with almost half published in the last 3 years. Almost 80% of the papers were published from 2005 onwards. Likewise, a search for the same terms in the ACM Digital Library revealed a similar pattern. In fact, over 92% of the 51 papers were published from 2005 onwards. It is clear that research related to SSVEP is expanding quickly (see Figure 1). While much research has been done in the field of SSVEP, very few of these papers are directly relevant to the research presented in this thesis. The following two papers are directly relevant to this research.

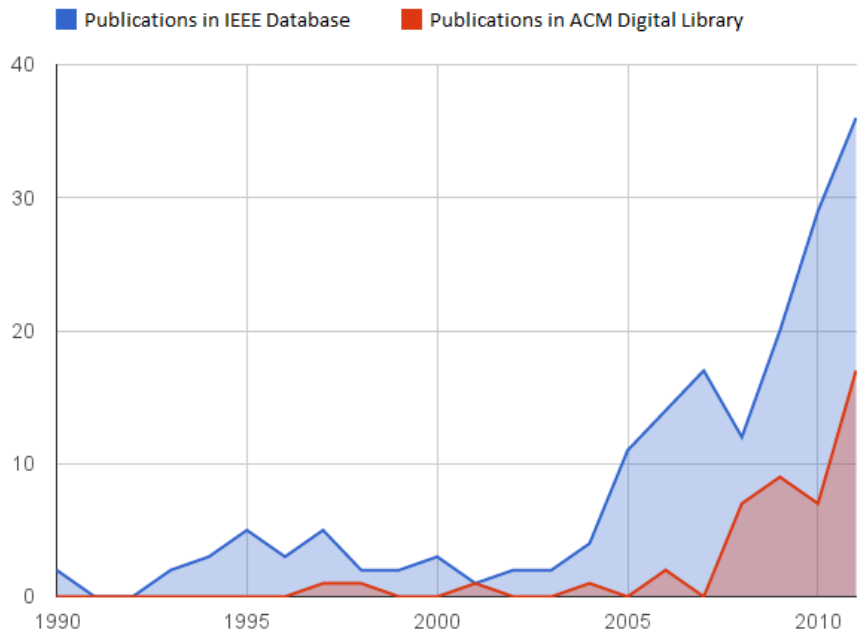

Figure 1: SSVEP Publications from 1990 to 2011

[5] successfully implemented a SSVEP-based BCI demonstration utilizing the Emotiv EPOC EEG headset (EPOC). Their publication gives an overview of several projects related to BCI, including various studies involving invasive BCI and non-invasive BCI such as the P300, SSVEP, and some research into Error-Related Potentials. P300 is another form of ERP and is described as "a positive ERP component, which occurs with a latency of about 300 ms after novel stimuli, or task-relevant stimuli, which requires an effortful response on the part of the individual under test" [2]. The research team in [5] performed at least two studies with SSVEP, the second of which was using the EPOC. Because the paper consists primarily of an overview of several techniques, the authors do not discuss details of their implementation. The only detail mentioned about the system was that they "did not use [their] conventional EEG system, but utilized a commercial EEG gaming headset EPOC from Emotiv" and reversed the headset in order

to get a specific location of the electrodes "needed to access brain regions other than the ones the EPOC was designed for" [5].

Hoffmann's study on the practicality of using the Emotiv EPOC for BCI research in [6] was one of the driving inspirations for this thesis. His research focused on assessing the EPOC for the P300 signal. After doing much work with filtering the signals and removing artifacts, he was unable to reliably use the EPOC for P300-based BCI. However, these studies were limited due to time constraints and some further research was mentioned as a possibility. In his conclusion, Hoffmann states, "the results therefore should be considered as a first step, but does not justify a final verdict about the possible uses of the Epoc in the study of ERPs" and "the fact that the experiments were not successful therefore only means that additional options need to be evaluated and more time has to be invested to make them work than what was possible in this thesis" [6]. This work suggested that the EPOC may be unusable for P300-based BCI, but that left the question of investigating its use for SSVEP-based BCI. This thesis attempts to show its feasibility for such an application.

1.2 Goals

The primary goal of this thesis is to show that it is possible to use SSVEP on a relatively inexpensive, consumer-grade EEG device, the Emotiv EPOC (see Figure 2). Although much research has been done in the field of SSVEP-based BCIs, it is still a young and developing area and practical devices have yet to appear on the market. There has been some exploration of the use of the Emotiv EPOC headset for BCI applications,

but few have examined its applicability to SSVEP. Generally, research-grade EEG equipment has been used in previous studies (see Figure 3). Not only is research-grade equipment expensive, it also involves significant effort for setting up experiments. In contrast, the Emotiv EPOC is a simple, single-unit headset designed for consumer use. It is a computer peripheral and uses a wireless universal serial bus (USB) dongle for communication with standard home computers. This thesis attempts to show its viability for SSVEP and relative simplicity in comparison to other research-grade EEG equipment. Another goal is to have the controls appear on-screen, as opposed to using a separate control display as in other SSVEP studies.

Figure 2: Emotiv EPOC (borrowed from emotiv.com)

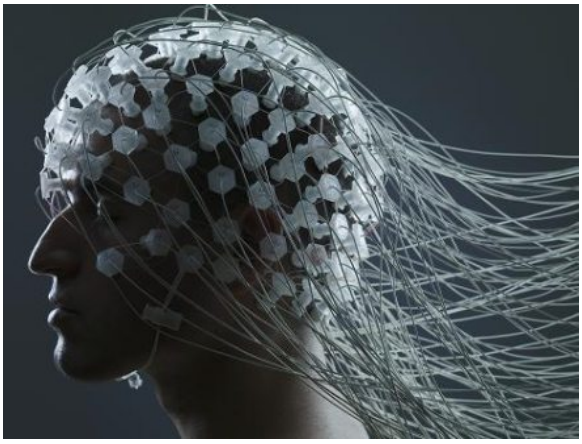

Figure 3: Research-Grade EEG Device (borrowed from [7])

II. METHODS & DESIGN

2.1 Overview

In order to achieve the goals of this thesis, a simple BCI demonstration using the EPOC headset was implemented. The demonstration presented a screen to the user with a ball in the center of the screen. The user was then able to move the ball around the screen by directing his/her attention at four directional arrows. These arrows were the controls for the BCI. Each arrow flickered at a different frequency.

2.2 Tools

The MATLAB programming language and environment was chosen to implement the demonstration because of its power and ease of use. It is very easy to quickly test various implementation details. MATLAB also has toolboxes which allow additional functionality. Two toolboxes in particular were used for this demonstration. The signal processing toolbox was used extensively for analyzing the EEG data from the headset. Additionally, Psychtoolbox (PTB), a third-party toolbox, was used to generate the display elements and visual stimuli. Also, the Emotiv Software Development Kit (EDK) was used for interfacing with the EPOC. It is accessible from a variety of programming languages. It is primarily written in C, but the company also provides wrappers and/or example code for accessing the Application Programming Interface (API) in C++, C#, Java, and MATLAB. MATLAB provides methods for calling functions in C code which allows for straightforward access to the EDK's API.

2.2.1 Headset

Emotiv provides several different versions of their product. Generally the EPOC is designed to be used by programmers to access the headset's pre-processed data. The API allows the programmer to utilize state information about the user's brain based upon the information Emotiv puts together. The EPOC identifies the user's emotional state, cognitive state, and facial expressions. The headset is designed as a video game accessory where the programmers are usually interested in using the device as a controller. In such cases, the programmer is not interested in the raw EEG data, but rather, the user's intent in order to perform the associated actions. The Consumer Edition of the headset allows for this basic use, but the Research Edition is necessary for anything more advanced. The product chosen for this project was the Research Edition. This provided both the EDK for programming with the headset and access to raw EEG data from the headset. There are two different physical models of the headset and two different versions of the EDK. The headset model and EDK provided with the Research Edition allow access to the EEG data, whereas the Consumer Edition does not. Because this project involves analyzing the EEG signals, the Research Edition was necessary.

All editions of the Emotiv EPOC contain 14 electrodes plus two reference electrodes. The headset has an internal sampling rate of 2048 Hz, but the hardware does some bandpass filtering to remove the 50 and 60 Hz power components and other forms of preprocessing to reduce noise. The data is then downsampled to 128 Hz before becoming available to the system for capturing the EEG signals. The captured data contains values for each of the 14 electrodes on the EPOC headset.

2.3 User Interface

The user interface of the demonstration program created to support this thesis was simple (see Figure 4). The user was presented with a full screen window with a solid black background. The display size was set to 1280 by 1024 pixels. Along each of the screen's edges were four black and white checkerboard patterned isosceles triangles (referred to simply as the arrows or the controls). The arrows were 300 pixels wide (from point to point) and 100 pixels tall (from base to point). Each of the black and white squares which comprised the checkerboard pattern was 32 pixels wide and tall. In the center of the screen was a dark blue, filled circle representing a ball. This circle had a 50 pixel diameter. For testing purposes, each of the arrows also included an adjacent text notation of the current threshold value for that control. The thresholds are discussed in more detail later. The interface also allowed for keyboard input. The demonstration program allowed the detection thresholds to be individually adjusted on the fly. Additionally, there were keyboard inputs that allowed the user to indicate which control had his/her focus for post-run analysis (also discussed later). The demonstration continued until the user pressed the escape key to quit.

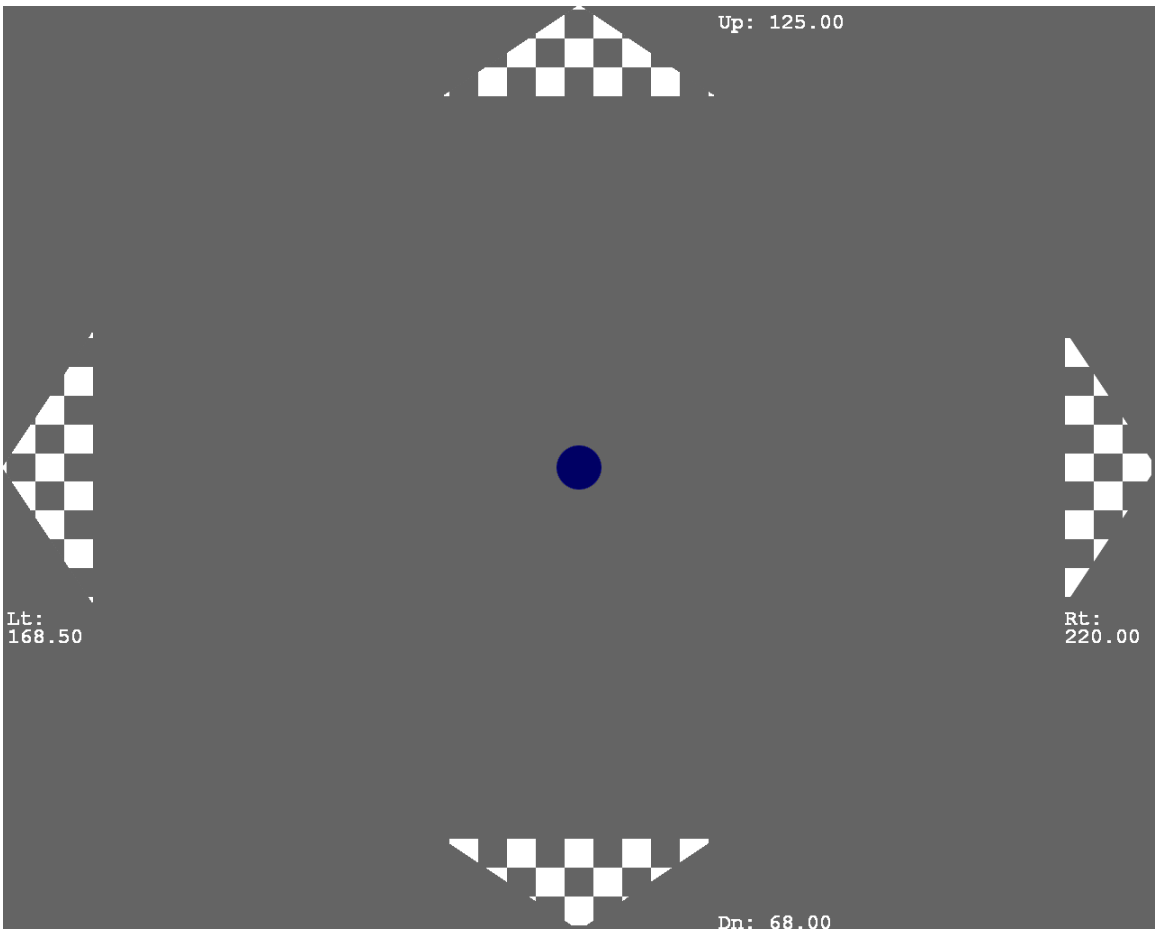

Figure 4: BCI Demonstration User Interface (gray background shown for contrast)

The monitor refresh interval is how much time occurs between two subsequent refreshes or redraws of the screen. This value was the basis for the frequencies of the control stimuli. It also determined how much time is available for the data collection and analysis which is explained in more detail in section 2.5. The monitor refresh rate used was 85 Hz, meaning the refresh interval was 11.76 milliseconds. There were four controls in the BCI demonstration, which meant that there were also four control frequencies. The frequencies chosen were 8.5 Hz, 9.444 Hz, 10.625 Hz, and 14.167 Hz. These frequencies corresponded to alternating the checkerboard pattern every 6, 8, 9, and 10 frames. How these pattern changes were scheduled is also discussed section 2.5.

2.4 Data Processing

Figure 5 shows the data flow diagram for the demonstration application. The following subsections describe the elements presented in the diagram.

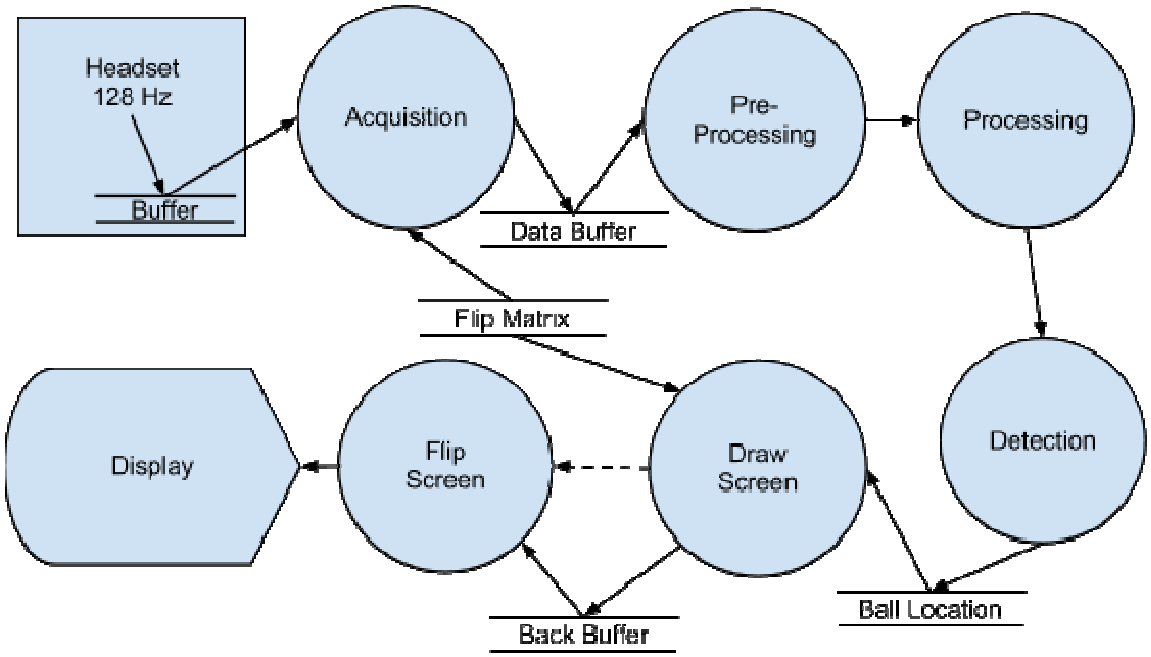

Figure 5: Data Processing Data Flow Diagram

2.4.1 Data Acquisition

A key component in the development process was to capture data from the EPOC EEG headset. Initial development was done using the C# .NET wrapper API provided by the EDK for this purpose. Utilizing and modifying some example code provided in the API and user manual, an EEG logging application was written. This application connected to the headset, read the data from the device every 100 milliseconds, ran for a specified number of seconds (passed as an argument to the program), and then wrote the captured data to a comma separated value (CSV) file. This allowed for later static analysis of the recorded EEG signals. This program demonstrated the relative simplicity of connecting to and retrieving data from the EPOC.

The concepts from the above program were used in the demonstration to collect the available data from the headset and put it into a circular buffer. According to the EDK user manual, the software will collect the EEG data, storing it in the internal sample buffer; so the programmer's application is required to pull that data often enough not to overrun the internal sample buffer [8]. In the initialization code, the internal sample buffer of the headset was set to a size of one second. The EPOC headset has a sampling rate of 128 Hz, or, 128 samples of data for each second of data captured, so the internal buffer will store a maximum 128 samples. However, the main loop of the program extracts the data at least every 141.2 milliseconds (18 samples), so there is no risk of overrunning the internal buffer. The circular buffer used by the demonstration program holds three seconds of data (384 data samples).

It is important to note, that when collecting data from the Emotiv EPOC, the connection to the headset must be closed at the end of execution. Failing to close the connection leads to corrupted data in subsequent uses of the headset regardless of the program accessing it. This was discovered during the testing phase of the BCI demonstration.

2.4.2 Preprocessing

After the data was pulled from the headset, it was processed to identify the SSVEP response. However, before processing it, there were several things done to preprocess the data to isolate the occipital channels and sanitize the signal. First, the two occipital channels were averaged together. The SSVEP response should appear in both

and averaging helped eliminate some of the noise or non-important differences between the two. Also, a common average reference was utilized. The common average reference is an average value for all 14 electrodes of the headset over the course of the time signal. This, again, helped to reduce or eliminate the unimportant signals from the target signal by expressing the occipital lobe signal as variations from overall EEG activity. The DC offset was also removed from the resulting signal, although this could also have been accomplished by using a high-pass filter later in the processing stream.

2.4.3 Processing

Since SSVEP is based on a particular frequency being present in the EEG data, signal processing was done to analyze the various frequency components in the current window of data (i.e. the circular buffer). In order to accomplish this, the time signals were converted into the frequency domain using a Fourier transform. "The Fourier transform is a method of representing mathematical models of signals and systems in the frequency domain" [89]. A discrete Fourier transform (DFT) is

> an approximation of the Fourier transform that can be calculated from a finite set of discrete-time samples of an analog signal and which produces a finite set of discrete-frequency spectrum values. This Fourier transform approximation is well suited for calculation by a digital computer [9].

In this case, the analog signals were the EEG signals from the headset device. The 128 Hz sampling rate produced the discrete-time samples of those signals. "Efficient computer algorithms for calculating discrete Fourier transforms ... fall under the general classification of fast Fourier transforms (FFTs)" [98]. A FFT is a computationally efficient way to approximate the frequency domain from a given time signal. Power Spectral Density (PSD) "is the power of each frequency component in the signal. ... The

DFS gives the amplitude of the sine and cosine component at each frequency. Power is the square of amplitude" [10]. In this demonstration, a PSD was used for the signal processing.

MATLAB's Signal Processing Toolbox includes an implementation of the Welch method for estimating the PSD. This function accepts the time signal, a window size for the subwindows for its internal FFT, the number of samples to overlap those subwindows, and the number of FFT points to use. The accuracy of a PSD depends upon the sampling rate of the signal, the number of FFT points used, the size and boundaries of the window of the time signal, the windowing function used, and other variables. The window of data used in the demonstration was 3 seconds, which resulted in 384 samples. Because the sample rate of the data is 128 Hz, a DFT will result in a sampling of the frequency domain with a bin width of .33 Hz. This means that the resulting frequency spectrum will be sampled at frequency values that are .33 Hz apart (e.g. 0 Hz, .33 Hz, .66 Hz, 1 Hz, etc). However, in this demonstration, 512 FFT points were used, so the 384 sample window was zero-padded to 512 samples. This artificially increased the bin resolution to .25 Hz. Although this appeared to improve the resolution of the frequency spectrum, it was a resampling of the same spectral information, which was the spectrum of the EEG convolved with the spectrum of a 3 second Hamming window. Zero-padding increased the density of samples taken on the resulting convolution, but did not change the width of the windowing function, which was the limiting factor on obtaining spectral information about the underlying EEG signal. The spectrum of the windowing function can be interpreted as a "smearing" function on the underlying EEG spectrum. The

number of FFT points was chosen to be 512 rather than 384 based upon a visual comparison of plots of the resulting spectra. This appeared to allow for better distinction between two relatively close frequencies as well as produce a stronger response (see Figure 6).

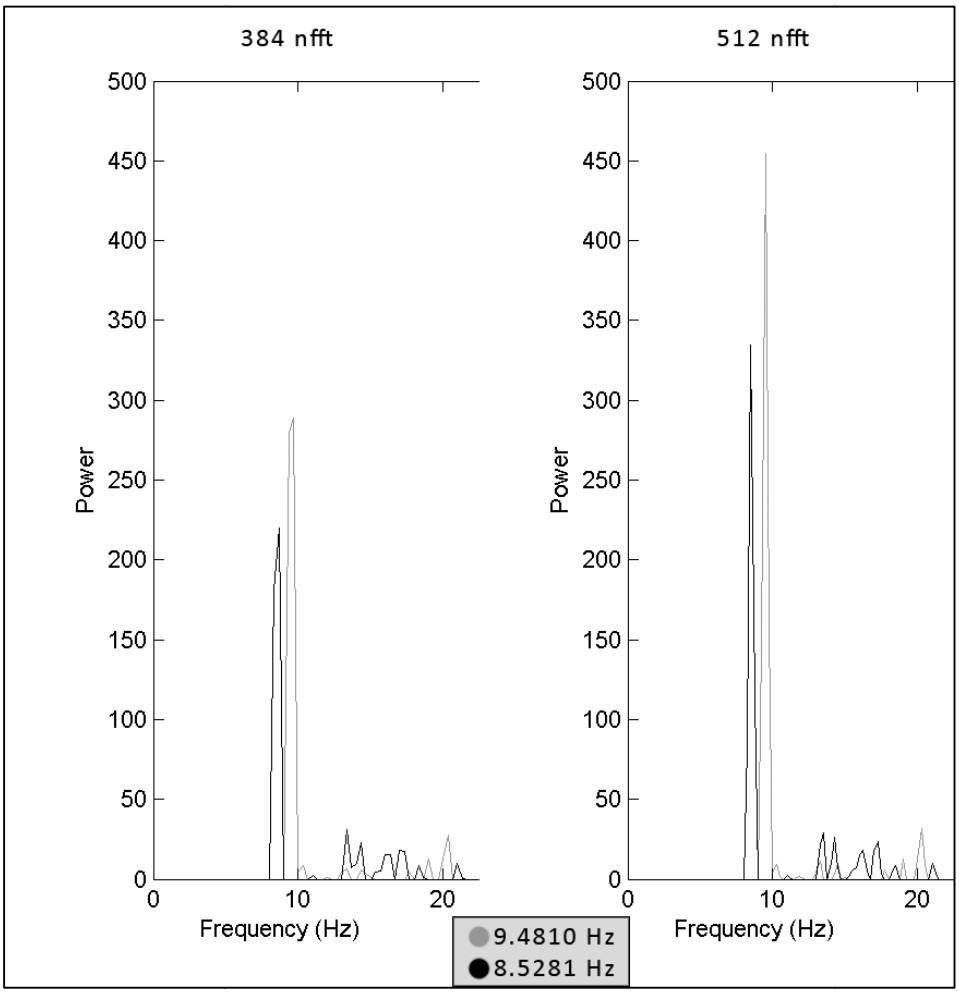

Figure 6: EEG PSD Plots Using 384 and 512 FFT Points

In order to further improve the results of signal isolation for the identification of SSVEP responses, the option of baseline removal was investigated and used in the demonstration application. A baseline was recorded while the user was viewing a solid 50% gray screen. This established a "standard" in terms of the brain activity that was present without any visual stimulation. The baseline signal was then processed in similar

fashion to the active EEG signals. However, the parameters to the PSD function were modified in order to smooth the spectrum through the use of averaging. This was useful as the baseline was intended to contain a measure of the "normal" or "non-target" powers in each frequency bin. Averaging and smoothing the spectrum helped to reduce any noise that may have been in the baseline recording. The smoothed baseline spectrum was subtracted from the live signal spectrum. The baseline is discussed further in the Results and Discussion section.

2.4.4 Detection

If one of the frequencies of the arrow stimuli was present in the data, with high enough power, the ball object was moved in the appropriate direction. Not only was this the object of the BCI demonstration, it also provided visual feedback to the user to allow them to see that the control was recognized. There are various ways in which this detection can be done. As mentioned in the previous section, the bin resolution of the frequency spectrum was .25 Hz. Most of the power at any given frequency should appear in the two nearest bins in the spectrum estimation. The demonstration program used the sum of the two nearest frequency bins to determine whether or not a control has been activated. If the sum of the two adjacent frequency bins surpassed the threshold value for the given control, that control was activated. If more than one control frequency was detected, the one which was a greater percentage above its threshold was triggered. The triggering of a control adjusted the ball's location in the direction of that control. This location was then later used for drawing the ball in the proper place on screen.

2.4.5 Screen Drawing

The display was developed using PTB for drawing on the screen. PTB provided access to various tools generally used for psychology experiments, such as presenting visual or auditory stimuli to users and gathering their feedback, with keyboard input for example. PTB gave access to low level graphics capabilities on the system for low-latency display, which was necessary for displaying the visual stimuli at precise frequencies. To control the exact frequencies displayed to the user, a visual display system capable of high-precision control over the graphics card was used. To create the display system, PTB was initially used to generate a checkerboard pattern on the screen and alternate the pattern at regular intervals. This interval determined the frequency of the pattern change, which could be altered between runs of the checkerboard program.

The above code was then modified to produce the display in the demonstration. To begin, the screen was initialized by PTB. This created a full-screen window and collected information about the monitor, such as the resolution and black and white color indices and, most importantly, the monitor refresh interval. Also, the pattern images for each of the arrows were loaded and converted to textures. According to the PTB documentation, textures are the fastest method of drawing on the screen. Because the program needed to execute with strict timing constraints, the drawing had to happen instantly. Once the textures had been loaded, they were drawn on the screen. The code did an initial flip of the screen to collect the first vertical blank (VBL) timestamp, which is discussed in the next section.

PTB uses a back buffer for drawing. This means that while an image is displayed on-screen, drawing commands are issued to generate images off-screen. Once all drawing commands have been issued, the flip command can be called. This switches the back buffer and the on-screen image. This way the display change can be synchronized with the vertical blank of the monitor. The vertical blank (VBL) on a CRT monitor refers to the time between when the electron beam moves from the bottom of the screen back to the top. To draw images on screen without a tearing effect, the flip of the back buffer with the on-screen images needs to happen during the VBL. PTB's flip command allows for this synchronization. Additionally, it returns a timestamp of when the VBL occurred. This timestamp can then be used in combination with the monitor refresh interval to schedule future flips to occur at particular times in sync with the VBL.

2.5 Task Scheduling

MATLAB is a single-threaded programming language that can only execute one command at a time. In order to ensure that each of the control patterns was flipped at the proper rate, a static cyclic executive was used.

> The cyclic executive executes an application which is divided into a sequence of non-preemptible tasks, invoking each task in a fixed order throughout the execution history of the program. The cyclic executive repeats its task list at a specific rate called its cycle, or major cycle in the common situation in which all tasks do not execute at the same frequency. When the frequencies are not identical, the task list defines a sequence such that each task is repeated sufficiently often that its frequency requirement is met. In this case, the execution of each individual task is called a minor cycle, and the frequency of the major cycle will be set to the least common multiple of the frequencies of each task [11].

The "non-preemptible tasks" scheduled in the cyclic executive for this demonstration application were the flipping of the checkerboard patterns. The least common multiple

(LCM) of these events (based upon the frequencies mentioned previously) was 360. Rather than code this many minor cycles, this project used a cyclic executive that repetitively drew the patterns from a sorted list of events. The pseudocode is shown in Listing 1.

Listing 1: Cyclic Executive Pseudocode

```
while(true)
{
   check keyboard input
   if (enough time available)
   {
      collect data
      analyze data
      if (control detected)
         move ball
   }
   if (up scheduled) - swap up arrow pattern and draw
   if (down scheduled) - swap down arrow pattern and draw
   if (left scheduled) - swap left arrow pattern and draw
   if (right scheduled) - swap right arrow pattern and draw
   draw ball
   flip screen with back buffer
}
```

Within the main loop of the program (the major cycle of the cyclic executive), keyboard input checks, data collection and analysis, and control detection must also occur. The cyclic executive uses a timeline in which events are scheduled. The program is able to use the timeline to determine when there will be enough time to perform these tasks without interfering with the schedule of control flips. Figure 7 shows a portion of the timeline. Each color represents one of the four controls changing pattern. As can be seen from the timeline, the schedule of flips appears irregular.

Figure 7: Cyclic Executive Timeline Segment

Because the four directional control arrows were changing pattern at different frequencies, the gap of time between pattern flips varied. To account for the time variation within which the program must collect and analyze data from the headset, a check was made to ensure that there was enough time in which to complete the data work before the next scheduled flip. This was based on the flip matrix generated at the beginning of execution. The flip matrix contained the schedule of when each control flipped from one pattern to the next. It also contained the information about the gap between flips and determined when there was enough time to complete the data work while still changing the control patterns on schedule. If the data work took any longer than the allotted time, the next screen flip was delayed. For this reason, the flip matrix was consulted before doing any data acquisition or processing. If not enough time was available, the data work was skipped.

During the program's initialization, the flip matrix is generated, as described in Listing 2.

Listing 2: Flip Matrix Generation

- get the LCM of all four control flip intervals
- generate a matrix of binary values for each of the frames (from 1 to the LCM) and for each control indicating whether or not that control should flip on that frame based upon each control's flip interval
- count the non-flip frames (gaps) between flips of any control
- combine the gap list with the flip list

2.6 Data Logging & Playback

In order to evaluate detection algorithms, it was important to implement capability for analyzing data from runs of the demonstration application. The demonstration program captured the most recent 120 seconds of data and dumped it out to a CSV file. Not only did it output the EEG data that was captured during the run of the application, it also included the values of the thresholds, the number of data samples that were pulled from the headset each time data was collected, and the indicators of which arrow had the user's focus at any given time. This data allowed for post-run analysis offline to evaluate detection methods and thresholds.

In addition to dumping the last two minutes of data from the demonstration to a CSV file, it was necessary to develop a tool for analyzing the data. The tool is capable of playing back the data that was captured during a demonstration run. The playback utility shows several plots of the data, such as a rolling plot of the current window of EEG data being analyzed, and a plot of the frequency components of each of the four controls with the threshold values being applied at the time (see Figure 8). On the last plot, there is also a visual indication of which arrow had the user's focus during the current window of data. By default, this tool analyzes the data in the same way that the demonstration application does. For example, if thresholds were adjusted during the run of the demonstration, the adjusted thresholds were also used for detection during playback.

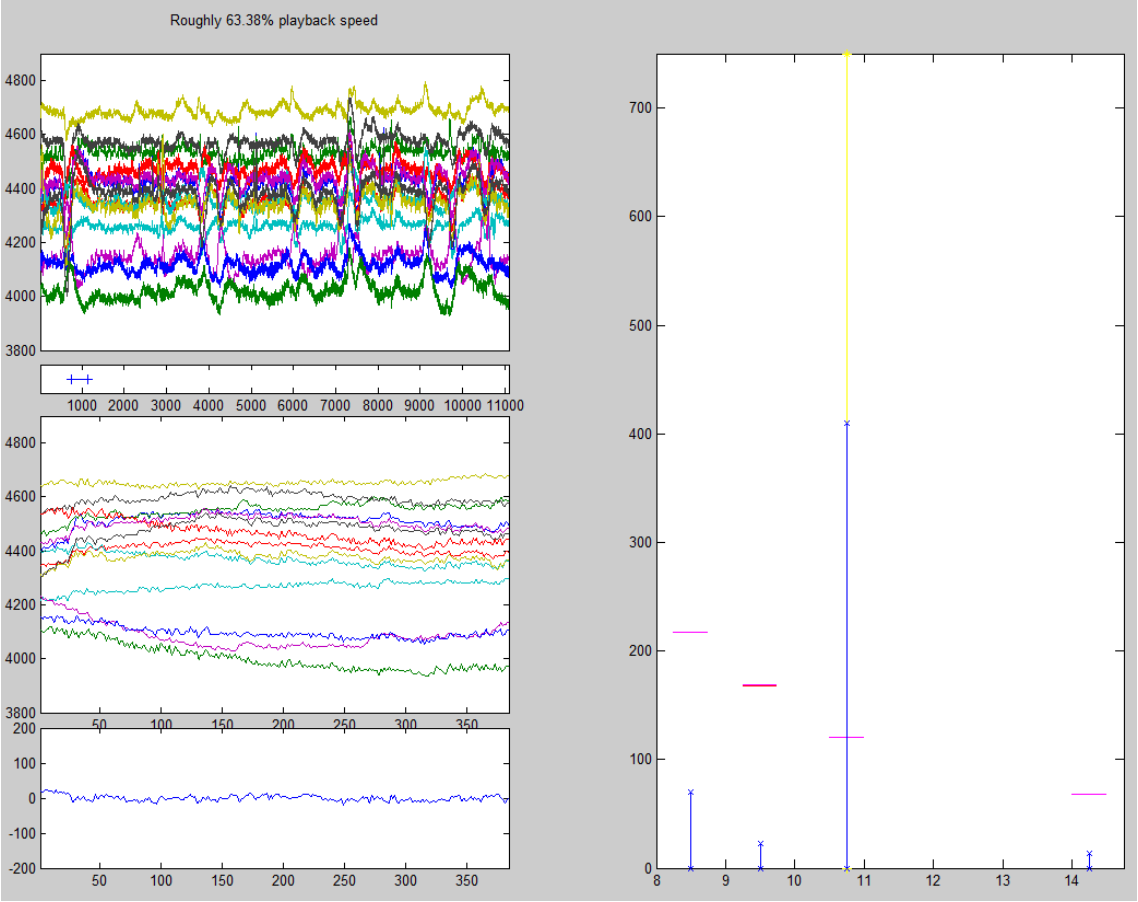

Figure 8: Data Playback Tool Screenshot

The playback tool allows for visual analysis of what happened during the last two minutes of the demonstration. It also collects statistics about the detections and analysis. Because keyboard keys were used to indicate the user's focus during the run of the demonstration, it could be determined if detections made during analysis of any window were correct or false. Additionally, the power of each control frequency in the signal can be collected for each window. Using this information, the average power of each control frequency could be determined while the user directed attention at that control, focused on a different control, or was not focused on any control. The goal of determining these values was to identify the proper values for thresholds which maximize the true

detections and minimize false detections. Besides tuning the thresholds, the playback tool allowed the detection algorithm itself to be modified and re-run on the same set of data. This allowed for direct comparison of multiple detection methods.

III. RESULTS & DISCUSSION

3.1 SSVEP Verification

Initial verification of the presence of an SSVEP response was obtained by recording the EEG signals using the aforementioned C# program while simultaneously running the checkerboard display program. This allowed EEG data to be captured and saved to a CSV file while the user watched a flashing checkerboard. Static analysis was then performed on the CSV file.

After isolating the important signal data as much as possible using the preprocessing steps, the resulting signal was passed through MATLAB's PSD function. This returned a spectrum of the signal, a function of the power per hertz, which allowed the various frequency components of the signal to be analyzed. The spectrum was then plotted to produce a visual representation of the power or energy of each frequency component in the signal. A large spike could be seen at the frequency of interest. Because the user was viewing a checkerboard which was alternating its pattern at a specific rate, this frequency had a very strong power in the spectrum. Figure 9 shows the spectrum obtained from a two second window, using the Welch PSD method with a single window (resulting in no overlap) and 512 fft points. A large spike is seen at 9.5 Hz, which is the nearest bin frequency to the stimulus.

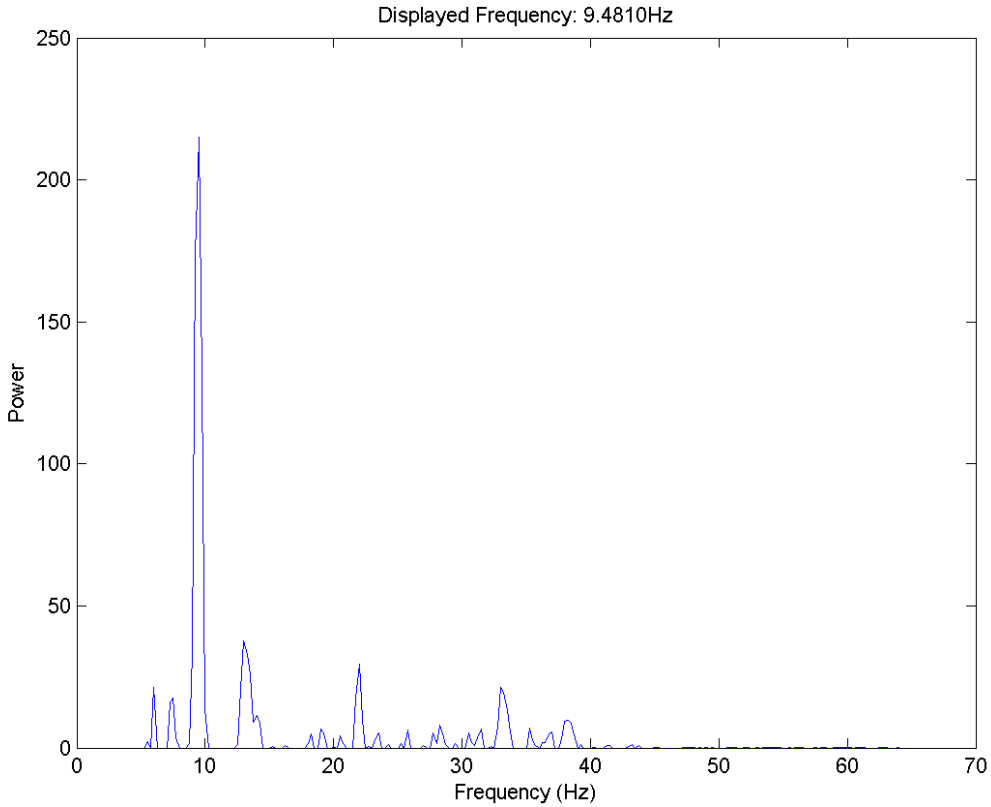

Figure 9: Plot Showing SSVEP Response at 9.5 Hz

3.2 Display Timing Verification

3.2.1 Refresh Rate Confirmation

Early attempts to verify an SSVEP response in the EEG signals gave peaks at frequencies in unexpected bins. For example, the checkerboard was scheduled to flip every 5 frames with the monitor refresh rate set to 85 Hz, which should have produced an SSVEP response at 17 Hz. However, the results of plotting the EEG spectrum were showing a spike around 13 Hz. This generated suspicion that the display was refreshing at 60 Hz rather than the reported 85 Hz. If the checkerboard was changing its pattern every 5 frames on a 60 Hz display, the expected response would be around 12 Hz, which was much closer to the 13 Hz that was seen. In order to verify the accuracy of the display

system at this point in the process, a simple experiment was performed. The experiment utilized a Fluke Scopemeter, a 5V power supply, a breadboard, a 10 kOhm resistor and a photoresistor. Both the resistor and the photoresistor were wired to the breadboard on the power supply. These were also wired into the Scopemeter for measuring the current through the system (Figure 10).

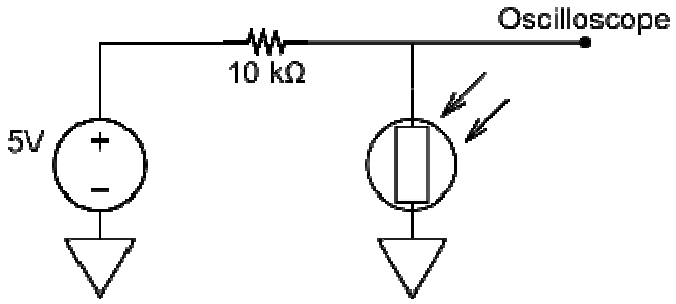

Figure 10: Experiment Wiring Diagram

The photoresistor was positioned so it was directly against the monitor over a single square of the checkerboard. When the checkerboard pattern was being displayed, the Scopemeter measured the oscillations in current through the photo-resistor. The cycle period showed that the displayed frequency was indeed accurate. This experiment confirmed that PTB was correctly measuring the refresh rate of the monitor and that the display equipment was properly working. The frequency mismatch was later discovered to be caused by a bug in the code analyzing the EEG signals. Rather than evaluating the two occipital channels, two EEG sensors from further forward and on the right side of the user's head were being used by mistake.

3.2.2 Missed Flips

In addition to the VBL timestamp that was returned from PTB's flip call, several other values were also returned. One of these values indicated whether or not the flip missed the stimulus onset time requested. A call to flip generally included a requested time for PTB to make the switch of the front and back buffers. The flip attempted to synchronize with the first VBL after the requested time. If PTB was able to make the switch at the requested time, it returned a negative missed value; otherwise it returned a positive value. This allowed the demonstration to determine if a significant number of flips were missed.

One of the biggest challenges with creating the basic display system was encountering a large number of missed frames. While a few sparse missed frames would not be a problem, many missed frames caused the displayed frequency to be altered significantly. In order for SSVEP to be effective, the pattern displayed on the screen had to alternate at a consistent frequency. PTB documentation recommended running the software on a realtime operating system for the best performance. However, it also provided methods for reducing the impact of a non-realtime operating system. For example, this project was built on a Windows 7 machine. Windows is not a realtime operating system. It does, however, provide process priority. PTB has a function which will request elevation to a higher priority for the MATLAB process. In order to elevate to "realtime priority", MATLAB had to be started with administrative privileges. Otherwise, the highest priority PTB could achieve is "high priority". Running the

program with “realtime priority” significantly reduced the number of missed flips seen during a run of the checkerboard display program.

3.3 Baseline Removal

One other method used to improve the signal was to utilize a baseline. A baseline was established by recording the EEG data while the user was not being presented with any stimulus. This gathered data about the normal signals present in the user’s brain which was then used to further isolate the components of the signal affected by the stimulus. Figure 11 shows an example baseline spectrum.

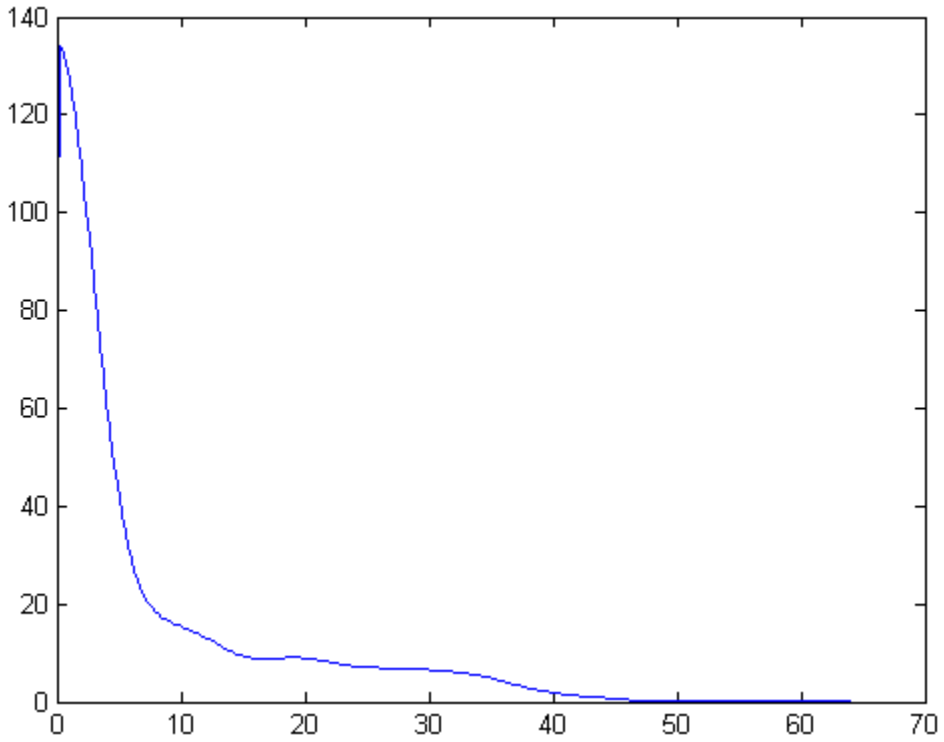

Figure 11: Smoothed Baseline Spectrum

In the future, the baseline measurement may not be necessary for the demonstration program. As seen in Figure 11, the smoothed baseline could be

approximated by a low-order polynomial and could possibly be estimated from live data rather than requiring an extra calibration step. As will be discussed in the next section, it was determined that each control frequency requires a unique threshold with or without baseline removal, so the baseline could possibly be folded into threshold determination. Rather than subtracting the baseline, the value of the baseline at any control value could perhaps also be added to the threshold value. This would reduce the number of necessary calibration steps by one. Whether the baseline and threshold calibrations need to be performed for each unique user is a matter for further study. Additionally, future study should investigate whether a baseline must be established once or before each run of the demonstration application, or if it can be continuously estimated during the application's execution.

3.4 Threshold Determination

The thresholds set the level of power required for a particular control frequency to trigger the control action. Initially, the project hoped to be able to use a single, flat threshold for detection, allowing one value to be chosen for the required power to activate any of the four controls. However, it was discovered that the SSVEP response generated at any given frequency was significantly different. "The amplitude of a typical EEG signal decreases as 1/f in the spectral domain" [5]. Because of this, it was determined that a single threshold would not be sufficient. It may be possible to use a function as a threshold, setting one value for a particular control frequency with the others calculated based on this value. Currently, the demonstration uses a second order

polynomial, although each threshold may be individually tuned using keyboard input while the demonstration is running.

The polynomial threshold used in the demonstration was generated based upon average responses seen at the control frequencies. EEG data was recorded while the user was focused on a checkerboard flashing at a single control frequency. The data was then analyzed to determine the average power of the target frequency in the EEG signal for a rolling 2 second window. This was done for all four control frequencies. A polynomial was then fit to the average response from those tests. As will be discussed in the next section, these thresholds did not produce usable results. Further analysis and study of additional EEG data is necessary to determine a proper method for setting the thresholds.

3.5 Response Latency Testing

An experiment was performed in which the user was presented with the checkerboard pattern flashing at 8.5 Hz for several seconds before the pattern changed to 9.444 Hz. The data was then analyzed to determine how much time passed between the change in the stimulus frequency and the change in the SSVEP response in the EEG data. This experiment was run five times. The average response latency was 1.73 seconds. The maximum latency was 2.00 seconds with a minimum of 1.53 seconds. This implies that with the current signal processing methods, detection latency is primarily determined by the time required for the SSVEP signal to dominate the content of the PSD window (just over half the window width). This indicates that the BCI demonstration should be able to

detect a control activation within two seconds of the user shifting focus from one control to another with the current window size of three seconds.

3.6 BCI Testing

After implementing the basic setup of the BCI demonstration application, testing was done. The performance of the SSVEP BCI demonstration application was highly dependent upon the threshold values set for detection. When the thresholds were set to a very low level, the user exhibited a good amount of control over the ball when focused on a particular control. In other words, the ball moved in the correct direction indicated by the control on which the user was focused. However, when the user was not focused on any control, the ball moved somewhat sporadically around the screen. Because the thresholds were low enough, "normal" brain activity caused the control frequencies to have enough power to trigger movement of the ball.

If the thresholds were set high enough, false detections were reduced. Unfortunately, when the thresholds were set too high, the controls became difficult to trigger. Both of these phenomenon were observed during testing, showing that a simple ad-hoc threshold setting would not be satisfactory.

Based upon these results, a receiver operating characteristic (ROC) analysis is necessary to improve the performance of this demonstration. As cited by Fawcett, "a [ROC] graph is a technique for visualizing, organizing and selecting classifiers based on their performance. ROC graphs have long been used in signal detection theory to depict

the tradeoff between hit rates and false alarm rates of classifiers" [12]. Utilizing a ROC analysis will help to tune the thresholds to a point that minimizes false detections while maximizing true detections. An ROC graph is shown in Figure 12.

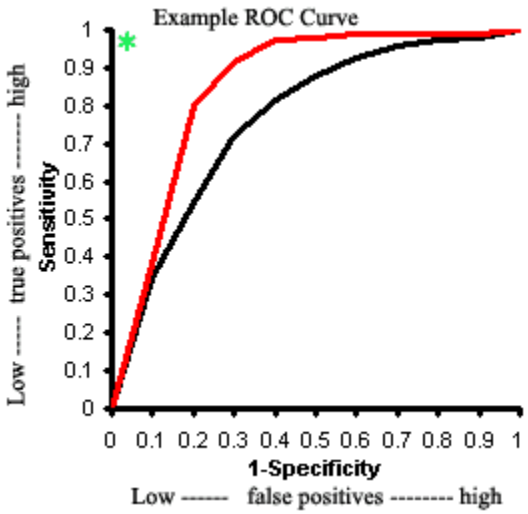

Figure 12: Example ROC Curve (borrowed from [14])

Another way to further tune the thresholds would be to incorporate better baseline characterization into the analysis. A single baseline measurement is subtracted from the live signals in the demonstration. It is possible that the baseline is not sufficiently stationary to allow a single characterization prior to beginning a session. Continuous estimation of the baseline from all spectral frequencies could improve the performance.

One other topic of interest is artifact removal. Artifacts are natural or external events which generate disturbances in the EEG signal. For example, eye blinks, eye movement, and facial muscle movements are examples of natural artifacts that impact the EEG. Besides the natural artifacts, external artifacts can impact the signal, such as bumping the headset or adjusting the position of the electrodes. External artifacts are

usually easy to reduce or eliminate. However, natural artifacts such as eye blinks are very hard to prevent. All of these artifacts can have various effects on the processing of the EEG signals. To reduce these effects, techniques can be implemented to account for and eliminate the artifacts in the original signals. This is another area of future research that could potentially improve the performance and/or usability of the demonstration application.

Based on the strong SSVEP response seen during the SSVEP Verification phase, as well as the poor performance and weak responses seen in the BCI demonstration itself, the influence of distractors should be investigated. During initial testing, all data was recorded while the user viewed a single checkerboard on the screen flashing at a particular frequency. The primary difference between initial testing and the BCI demonstration was the BCI demonstration included four different controls pulsating at different frequencies, all on the same screen. The controls were also much smaller than the initial checkerboard. Because SSVEP is attention-based, the presence of additional controls may be interfering with the response (Figures 13 and 14).

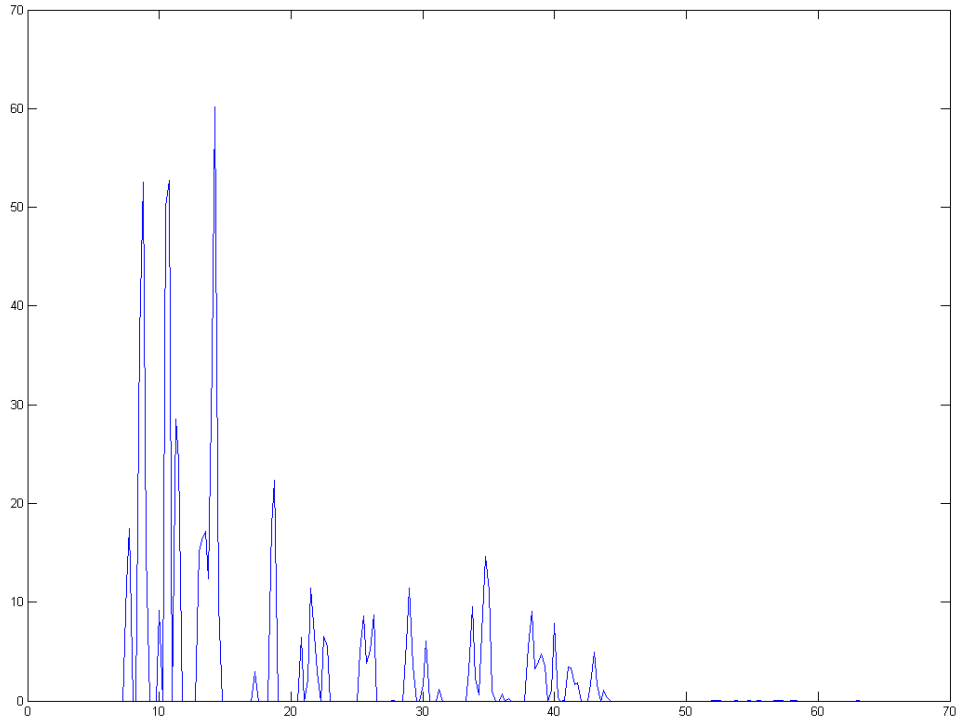

Figure 13: Responses Seen at 8.5 Hz, 10.75 Hz, and 14.25 Hz While The User Was Focused on The Control Flashing at 14.167 Hz During the BCI Demonstration

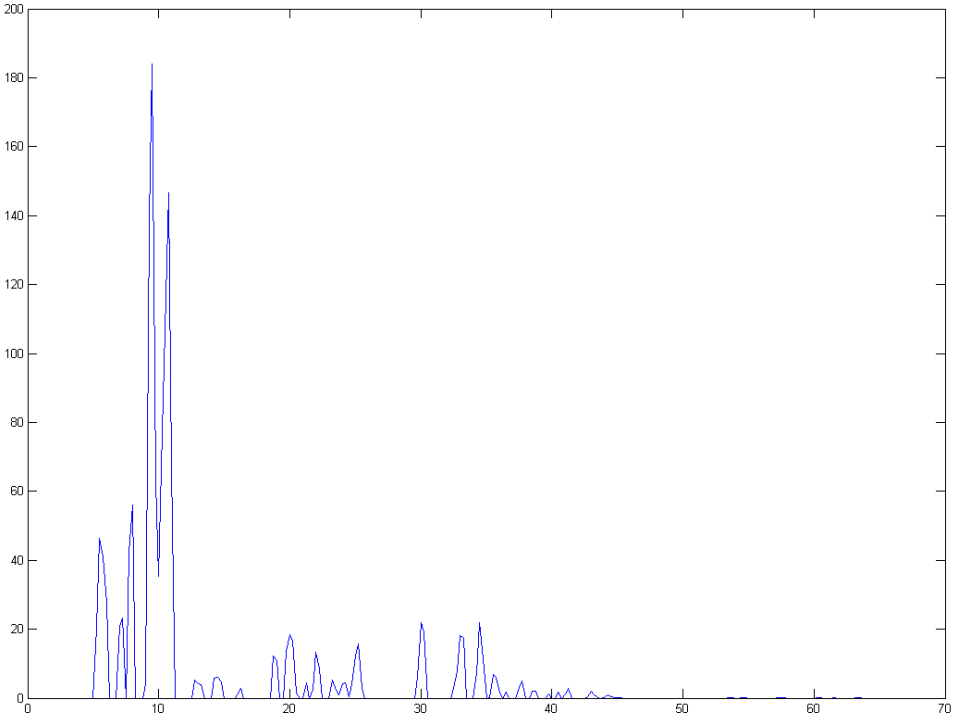

Figure 14: Responses Seen at 9.5 Hz and 10.75 Hz While the User Was Focused on The Control Flashing at 9.444 Hz During The BCI Demonstration

IV. CONCLUSION

Based upon the results of the demonstration application, it appears likely that SSVEP-based BCI is feasible using the Emotiv EPOC EEG headset. Although the program in its current form is not fully functional, further tuning and adjustment would produce a usable application. The results of the SSVEP verification phase of this study showed a definite ability to measure the SSVEP response from the EEG signals coming from the EPOC. Although, static analysis and realtime analysis are very different, the BCI demonstration shows potential and appears feasible.

One area which was not explored is the impact of the visual stimulus itself. Some of the initial testing was done using a large rectangular checkerboard pattern, whereas the demonstration application used smaller triangular checkerboards. This may have had an impact on the SSVEP response in the brain. In addition to the size and shape, the color of the stimulus could also have an impact. Zhu et. al. conducted a literature review studying the various stimulation methods used in SSVEP studies and determined that "improvements to stimuli can enhance the SSVEP [signal to noise ratio], simplify signal processing, enable the use of more targets, [and] prevent loss of attention" [4].

Additionally, the determination of the detection thresholds should be studied in more detail using an ROC analysis. Further study of the baseline and thresholds would be beneficial. The following are several questions for further research. Can a baseline be estimated dynamically versus statically? Is it feasible to fold the baseline into the thresholds? Can a single threshold be obtained as a required ratio over baseline, as

implied in [14]? How much variation in baseline and optimal thresholds occurs across subjects? What is the impact on the SSVEP response of having distractors in the user's field of view?

REFERENCES

[1] D. J. Mcfarland and J. R. Wolpaw, "Brain-Computer Interfaces for Communication and Control," *Communications of the ACM*, vol. 54, no. 5, pp. 60-66, May 2011.

[2] S. Sanei and J. Chambers, *EEG Signal Processing*, West Sussex, England: John Wiley & Sons Ltd, 2007.

[3] F. Beverina *et al*., "User adaptive BCIs: SSVEP and P300 based interfaces," *PsychNology Journal*, vol. 1, no. 4, pp. 331-354, 2003.

[4] D. Zhu *et al*., "A Survey of Stimulation Methods Used in SSVEP-Based BCIs," *Computational Intelligence and Neuroscience*, 2010. doi: 10.1155/2010/702357

[5] N. Manyakov *et al*., "Brain-Computer Interface Research at Katholieke Universiteit Leuven," *Proc. of the 4th Int. Symp. on Appl. Sci. in Biomedical and Commun. Technologies*, ACM, Oct. 2011. doi: 10.1145/2093698.2093765.

[6] A. Hoffmann, "EEG Signal Processing and Emotiv's Neuro Headset," B.S. Thesis, TU Darmstadt, Darmstadt, Germany, 2010.

[7] G. Fish, "the curious case of a skeptical bait and switch," unpublished, 2009. http://worldofweirdthings.com/2009/12/14/

[8] *Emotiv SDK User Manual*, Emotiv, San Francisco, CA, 2011.

[9] C. L. Phillips *et al*., *Signals, Systems, and Transforms*, 4th ed. Upper Saddle River: Pearson Education Inc., 2008.

[10] P. H. Schimpf, "Module 11a: Discrete Fourier Analysis," unpublished, 2011.

[11] C. D. Locke, "Software Architecture for Hard Real-Time Applications: Cyclic Executives vs. Fixed Priority Executives," *The Journal of Real-Time Systems*, vol. 4, pp. 37-53, 1992.

[12] T. Fawcett, “An introduction to ROC analysis,” *Pattern Recognition Letters*, vol. 27, iss. 8, 2005. doi: 10.1016/j.patrec.2005.10.010.

[13] S. B. Halls, “Sensitivity and Specificity issues regarding BMI thresholds,” unpublished, 2008. http://www.halls.md/bmi/specificity.htm.

[14] ringoring, “Brain-computer interface based on SSVEP,” Almaty, Kazakhstan, 2009. http://www.slideshare.net/ringoring/braincomputer-interface-based-on-ssvep.

VITA

Author: Brian J. Zier

Place of Birth: Puyallup, Washington

Undergraduate Schools Attended: Eastern Washington University

Degrees Awarded: Bachelor of Science, 2010, Eastern Washington University

Honors and Awards: Graduate Assistantship, Computer Science Department, Eastern Washington University, 2010-2012

Best Student Paper - Honorable Mention, Midwest Artificial Intelligence and Cognitive Science Conference, 2011

Outstanding Student in Computer Science, Eastern Washington University, 2010

Graduated Magna Cum Laude, Eastern Washington University, 2010

Graduated Honors Ad Majorem, Eastern Washington University, 2010

Publications: B. Zier and A. Inoue, "Fuzzy Relational Visualization for Decision Support," *Proc. of The 22nd Midwest Artificial Intell. and Cognitive Sci. Conf. 2011*, pp. 8-15, 2011.

Professional Experience: National Science Foundation Grant-Funded Living Laboratory Internship, Eastern Washington University, 2009