MS Project: Final Report

Demonstrating Realistic Avatar Control in a Virtual Environment
Through the Use of a Neural Impulse Actuator

Author: Emmett Coakley

Committee:

Advisor: Joe Geigel

Observer: Hans-Peter Bischof

Reader: Reynold Bailey

Department: Computer Science

Cluster: Computer Graphics

Quarter: 20101

Date: August 19th, 2010

Abstract

This project focused on the development of techniques to improve realism within the field of three-dimensional avatar control. This was accomplished by replacing a traditional hand-based peripheral controller with a Neural Impulse Actuator headset, a device which read and reported a user's brainwaves in real-time. The avatar's Virtual Environment was designed to make use of the headset's output features. A series of headset-based trigger events were implemented, each of these allowing the user to alter the environment based upon a series of preconditions. These requirements were most often met via control of the Neural Impulse Actuator device (NIA). The project's success rate was based upon how well a new user was able to interact within the environment, with regards to adapting to the system, influencing the virtual world, and performing faster with neural-based commands than with keyboard-based commands.

1. Introduction

Recent technological innovations on both the software and hardware levels allow for users to assume a high level of realism and authenticity in the realm of avatar control within a virtual environment. The majority of avatar-interfaces center on some form of peripheral device, such as a keyboard or game controller. While these devices allow for a user to assume interactive immersion into their avatar's environment, they are limited to being able to use only their hands for avatar control.

There have been several recent technological advances in the creation of alternate forms of avatar interaction such as motion capture, virtual reality goggles, power gloves, etc. One advance, which has seen much use in the realm of videogame control, is the development of commercial electroencephalographs (EEG) for user interaction. An EEG device is placed upon a user's scalp, and is able to read the electrochemical impulses that flow across a person's forehead. These signals are then translated by the device's software into a series of human-readable brainwave values. This device allows for user-avatar immersion as it enables near-instant avatar response based upon a user's mental activity. Again, however, this technology is still in its infancy, and there exists only prototypes of games that are run purely on EEG devices [1].

The current use of EEG devices with avatar control today is primarily placed into one of two categories: Videogame design or neurological analysis. The biofeedback generated through an EEG by a user can be made to further the study of how the mind works in conjunction with the body, and vice versa. Specific studies have utilized this type of EEG-Avatar control, one such using a tetraplegic to simulate walking in a virtual world using only an electroencephalograph [2]. The purpose of this study was to allow a user with extremely limited physical control, to be able to move freely in a virtual environment by using only the device. The EEG was designed to be able to interpret signals asynchronously to allow for complete real-time immersion into the system with minimal latency while obtaining high amounts of raw EEG data and translating it into human-readable wavelengths. The purpose of the study was twofold: to greater understand the parts of the mind which control muscle movement function in a subject that could not physically move, and to be able to use the results to allow for improvement in physical rehabilitation techniques. Initially, the EEG was custom designed to work with the user's mind,

and the software required time to be able to interpret his unique signals, as all users will emit slightly different values. Once the software was calibrated, the user was effectively able to move throughout the virtual world using only the EEG. They were able to move, interact, and communicate with the virtual avatars 90% of the time. However, like other EEG-avatar experiments [3], they did so mainly for the purpose of understanding how to interpret the biofeedback, instead of focusing on improving user/avatar interaction within the environment.

While this system allows for seamless interaction within a virtual environment, it does so using a highly expensive and high complex EEG-device. As it is proven that virtual interaction is possible using an EEG, our project does not going to focus on trying to design such a virtual system. The focus of the project was to design software that allows for an EEG to allow for extra control in the virtual environment, through the use of an inexpensive commercial EEG model. Since electroencephalographs are precise medical tools, their pricing ranges between one and two-hundred thousand dollars apiece. We planned to be able to simulate such realism through the use of an over-the counter device.

Currently, there exist two major companies that produce inexpensive EEG devices: Emotiv [4] and OCZ [5]. Both devices offer the interpretation of raw EEG data and are able to translate it into understandable wavelengths. These are currently marketed as videogame peripheral devices. The difference between the two is that while the OCZ's Neural Impulse Actuator (NIA) device is used to mask over currently developed games, the Emotiv's headset EPOC is designed to run with specifically created programs that make use of the unique hardware. The EPOC model comes equipped with wireless communication, 14 unique saline sensors and a gyroscope feature. It is currently priced at $300 for purchase and an extra $500 for the developers SDK. The EPOC is the most complex commercial videogame EEG in existence

today; however, as of 2010, only demonstrations and tutorials of games have been designed for interaction. The NIA, by contrast, has 3 saline sensors and allows for the recognition of facial muscle feedback alongside bioelectrical feedback. This is priced at $120 and the SDK is free for download. Unlike the EPOC, there are no games currently designed or in development to use the NIA's biofeedback. Instead, the NIA allows for a user to mask over a currently running executable. The mask acts as a buffer between the NIA hardware and the executing program, as whenever the user issues a physical response to an onscreen action, the software will interpret the impulse and send a signal to the executing program faster than the user could physically respond.

While the EPOC is the current forerunner in mind-based peripheral controllers, our project focused on determining whether or not it would be possible to use the NIA to demonstrate a similar level of interaction within a virtual environment. The reasons for choosing the NIA were similar to the reasoning behind not using a medical-based EEG: complexity and time. Our task was to use a simple inexpensive over-the-counter EEG device and attempt to simulate avatar control within the virtual environment. While this has already been proven to be possible, our task was to be able simulate effective avatar control at a severely reduced cost using OCZ's Neural Impulse Actuator device alongside the Luster-based Virtual Theatre Environment.

Over the course of the project, we extended the development of this type of user interface, allowing for a greater level of avatar interaction within the virtual world through the use of an over the counter electroencephalograph known as a Neural Impulse Actuator (NIA). This device masks over any form of executable software and has neural impulse commands sent directly to the executable program quicker than a user would be able to physically do so. This was accomplished by having a piece of software known as a mask act as a liaison between the

NIA hardware and the executable program. This meant that current executable programs could not directly understand the impulse signals being sent to them. The mask simply converts the impulses into basic keyboard commands and afterwards transfers them to the executable.

In order to enable a sense of realism, the executable software had to be able to do more than react to basic input commands. The software needed to be able to understand the impulse signals themselves and, in doing so, allow for the avatar to be able to properly interact in an environment based upon a user's thoughts, and not simply upon keyboard-based signals. Once the link had been established between the NIA and the Virtual Environment, a series of events and event triggers were constructed to simulate the increased avatar control. These events are the essential link between neural impulse commands and the virtual environment interaction. They are both environment and avatar altering actions which occur whenever both an impulse trigger is activated and a set of precondition requirements are fulfilled. Events are described in further detail in Section 2.2.

2. System Specifications

Essentially, there were two major phases to the design of the project: Communication between the Neural Impulse Actuator (NIA)/Virtual Environment Software, and the building/user testing of events. The first phase required the use of the Virtual Theatre Software system. This system was a luster-based virtual environment which allowed for distributed connectivity, real-time networking and interaction, and motion capture technology. For the project, we only dealt with the Virtual Environment provided by the software. The Virtual

Environment itself was designed to run using the Luster engine, which is modified using the lua language. The Luster environment also allowed for the incorporation of meshes, c++ plugins, xml, flash, audio, and particle files in order to recreate a digital environment. The majority of the event code was implemented in a series of .lua files, which interpreted the impulse commands being sent from the NIA client. Depending upon a user's brainwaves at a particular moment, the .lua code allowed for the triggering of unique graphical events, incorporating the mesh, particle, and other graphic file types, to occur when those requirements are met.

2.1. Device Functionality

The following is list of the core systems, and modifications made to them for the project:

Element:

- Neural Impulse Actuator

Basic Information:

- The Neural Impulse Actuator's commercial purpose was to allow its software to mask over an executable program (videogames, specifically) and allow for neural impulses to be read, translated, and then sent to the executable as a command. For instance, in terms of a videogame, rapid eye movement might be interpreted by the NIA mask as incoming danger to the player's avatar. The mask would then issue a basic command to the executable, which would be interpreted by the game itself (ie, the mask might

send the command “F5” which could shield the character from damage faster than the user could press the F5 button).

- The NIA is comprised of several components: headset, box, and client.
    - Headset: Interprets the bioelectrical feedback across a user’s scalp via three saline sensors.
    - Box: Grounding and interpreting peripheral hardware which connected the headset to the PC’s USB drive. Dissipates white noise when grounded to the user’s body.
    - NIA Client: Software which relays the binary impulse signals sent from the box and passes them onto an executable program based upon a mask file’s specifications.
- Designed for use using the NIA 4.0.1 beta client on Windows 7 (32bit)

Modifications:

- Designed a mask which would allow for the NIA Client to use the Virtual Theatre Software as an executable program.
- The mask (NIAClient.nia) will send a variety of neural impulse commands to the Virtual Theatre Software.
    - Impulses Sent: Alpha, Beta, and Tension.

Element:

- Virtual Theatre Software

Basic Information:

-A 3-dimensional Virtual Environment that allows for avatar and object generation.

-Designed for use by the Luster runtime engine.

-Avatars can be controlled (via keyboard control) in the 3D Virtual Environment.

-The 3D World is designed via meshes, avatars, skeletons, particles, C++ plugins, particles, audio, and lua files.

-Environment is modified via Client and Server calls. Multiple clients can modify a single server.

-Designed for use with the Luster runtime version 0.19.5f on Windows 7 (32bit).

Modifications:

-Event triggers, which added, modified, or removed existing objects within the virtual space based upon when certain neural impulse requirements are met.

-NIAClient.lua: A demonstration program which allowed for a standalone version of the virtual environment to exist, ie. the server is also the client. Incorporated the newly designed events, as well as received input data from both keyboard and NIA impulse commands.

-Avatars.lua: Objects were classified as avatars, and were made controllable by user via the NIAClient.lua and event code.

-Cues.xml: Individual cues which allowed for the creation/destruction of avatars, lighting, and particle effects. Additional cues were added for new events.

2.2. Events

In any system, a user's avatar is only able to interact with an environment to the extent of what the system permits. A system may allow the user to move their avatar through the use of keyboard commands, joystick toggles, mouse clicks, etc. For the purpose out the project, these actions were referred to as events. Events were comprised of three distinct portions: Triggers, Requirements and Actions. Triggers were the input commands which signaled that an event should be activated. Requirements were the preconditions which needed to be satisfied before that event could be performed. Actions were the consequential events which occurred once the preconditions were met.

For instance, a common avatar-control event could be defined as enabling movement control. To trigger the start of the event, a user would need to input a key command, for instance the "UP" arrow key. This would signal to the system that the user wishes to move the avatar forward. Before this can be completed, a set of requirements must be fulfilled. These are usually a set of basic questions such as "Will this move the character into a restricted area (ie, wall)?", "Is the character already moving?", etc. If the requirements are met, then the action phase begins. The action phase modifies the environment based upon the event's purpose. In this case, the

avatar and camera will move forward relative to their current positions. The above example can be interpreted as the following:

Event Name:

Movement – North

Triggers:

[Keyboard] Arrow Key “UP” is held down.

Requirements:

Movement of the avatar will not place the avatar into a restricted area.

Action:

Move the avatar north.

Events were the key to allowing effective avatar control within the environment. Our experiment took the above style of events one step further and incorporated another type of trigger: Impulse Trigger. The NIA Client and VTMSK.nia file allow for neural impulses to be directly streamed into the Virtual Theatre environment once certain criteria are met. For instance, if we required forward movement to be mapped to a specific impulse rather than a key command, we could have tuned the event to work as such:

Event Name:

Movement – North

Triggers:

[Impulse] – Alpha Wave Spike of 50% or greater.

Requirements:

Movement of the avatar will not place the avatar into a restricted area.

Action:

Move the avatar north.

This would have provided the same functionality as the previous event, except that the user no longer required a keyboard for interaction. While our experiment did still require the use of a keyboard for avatar movement, many of the key portions of the experiment required the use of neural impulse events. The impulse version of the “Movement – North” command was in fact not used for avatar movement, but for an object’s movement (See: Push/Pull). The following events and their triggers were designed and implemented for the project’s demo:

Event Name: Floating Sphere

Trigger:

[Impulse] - Alpha Spike must be between 70% and 100%.

Requirements:

A ball cannot already exist within the current space.

Alpha Waves must be dominant.

Action:

Generate a controllable 3D sphere which can be pushed or pulled through space.

Event Name: Aura

Trigger:

[Impulse] - The wave must be between 70% and 100% impulse level.

Requirements:

One wave (Alpha, Beta, or Tension) is dominant.

Action:

Depending upon which impulse wave is dominant, the avatar will emit a wave of orbs. The color of these orbs is dependent upon the current dominant wave. Red for Tension, Yellow for Alpha, and Blue for Beta.

Event Name: Levitation

Trigger:

[Impulse] - Beta Spike must be between 35% and 100%.

Requirements:

Beta Waves must be dominant.

Action:

The avatar will steadily rise for a few seconds before stopping, hovering slightly over the ground. After 10 seconds, the avatar will slowly return to its original position.

Event Name: Push

Trigger:

[Impulse] - Alpha Spike must be between 50% and 100%.

Requirements:

After movement, the object will not be placed within a restricted area.

Alpha Waves must be dominant.

Action:

An object moves further away from the avatar at a specified speed for 7 seconds.

Event Name: Pull

Trigger:

[Impulse] - Alpha Spike must be between 50% and 100%.

Requirements:

After movement, the object will not be placed within a restricted area.

The object cannot be closer than one space in front of the avatar.

Alpha Waves must be dominant.

Action:

An object moves closer to the avatar at a specified speed for 7 seconds.

Event Name: Fire

Trigger:

[Impulse] - Tension must be between 50% and 100%.

Requirements:

A ball must exist within the 3D space.

Action:

The floating sphere bursts into flames for 10 seconds.

Event Name: Inferno

Trigger:

The floating sphere must exist within a specified portion of the 3D space.

Requirements:

The floating sphere must be on fire.

Action:

A massive firestorm erupts at a given location.

2.3. Software and Event Design

The first coding task was to allow for the NIA headset to be able to communicate with the Luster environment. As mentioned previously, the NIA itself is comprised of three portions: the headset, box, and client. The headset is the device which attaches to the user's scalp and obtains the raw bioelectric impulses from that user's forehead. This information is then sent to the NIA's peripheral box. The box itself interprets the raw EEG data and transforms it into human readable data. The data is then transferred to the user's PC, in which the NIA client is able to relay the user's current brainwaves in a readable format.

The task at hand was to be able to send the final interpretation of the impulse data from the NIA Client into the Luster environment. The reasoning behind this would be that this data is essential in allowing for events to be triggered. This was done by developing a .nia mask file. This file worked with the end-level impulse readings from the NIA interpreter, in which depending upon the current impulse levels, a variety of keyboard commands could be issued

from the NIA client directly to another executable. The file “VT-Nia.nia” informs the NIA client to send a variety of key commands in response to a spike of brainwave activity. For instance, the Levitation Event had an Impulse Trigger of 35% Beta Spike. This would translate to whenever the NIA client interpreted that Beta waves were acting between 35-100%, the button command “55” (Spacebar) would be issued to the Virtual Theatre executable. The Virtual Theatre program would then factor the key command into whether or not an event could be triggered at that exact moment. For a complete list of the exact inputs for each impulse, refer to Section 2.2: Events.

Once the NIA Client was able to stream input commands directly to the executable, the next step was to allow these to be interpreted as impulse triggers. In order for an event to become active, the event needed to have its impulse and environment requirements met. For example, the Levitation’s Impulse Trigger was that single beta waves needed to be over 35% and that no other impulse (tension, or alpha wave) could be over 35%. The system itself was threaded with event listeners, each of which monitored for any of the possible impulse commands. If any were given, a notification was sent to each of the possible events which could utilize the trigger, ie whenever “button 55” was issued, an immediate command was sent to the Levitation event informing it that one of the requirements was met. In this manner, the Virtual Theatre software was able to keep track of the impulse commands being read via the NIA headset as well as those specifically being sent out as ordered from the VT-Nia.nia file as well.

Specifically, the mask was also be to allow for both the informing of the Virtual Environment whenever a spike would trigger, as well as whether or not the spike was still triggering. Event Listeners were designed to be mapped to each possible trigger, and to keep track of both when they were activated as well as when they were deactivated. Neural activity works in such that when a certain method of thinking is applied, a greater ratio of wave activity

increases. This does not mean that when a user is in a state of active thinking will constantly generate high levels of alpha waves. Instead, the user generates a high frequency of alpha wave spikes. The same works with beta waves as well; a person meditating will not have high levels of beta wave activity, as much as a high frequency of high beta wave spikes. The Event Listeners enable the VT client to be able to interpret exactly when a spike begins as well as when it ends. This allowed for the Virtual Theatre environment to be able to track which waves are currently spiking, as well as a greater level of control over the impulse triggers on the side of the events themselves.

The final phase to occur in event generation was the action phase. This occurred whenever an event has its triggering requirements fulfilled, ie when both the environment and the impulses are within the range specified by the event's parameters. The action itself is the modifications made to the virtual environment, such as the movement of the avatar or the generation of an object. The scripting of events dealt entirely within the realm of the Virtual Theatre software. However, most of these events came prepared with the Virtual Theatre Client, and only a handful [Levitation, Push, Pull, and Fire] added anything more than a cue call to the system.

3. Results/Analysis

As stated previously, in order to simulate an effective form of avatar control, the system was not designed solely around the specifications of one user, but generically so that system could be able to adapt to any host. However, in order to allow for a universal system, the

software would initially need to be designed around a single user. In short, there were two phases to testing; the first was to allow for the generation of events and to have those events be triggered by one initial user, and the second phase was to ensure that the NIA headset and software was able to work for any user. The degree of success was determined by the quantity of users who were able to successfully trigger the events within the virtual world, to what level of ease/difficulty they were able to do so, and the speed at which they were able to complete a tutorial game with regards to both neural and keyboard based interactions.

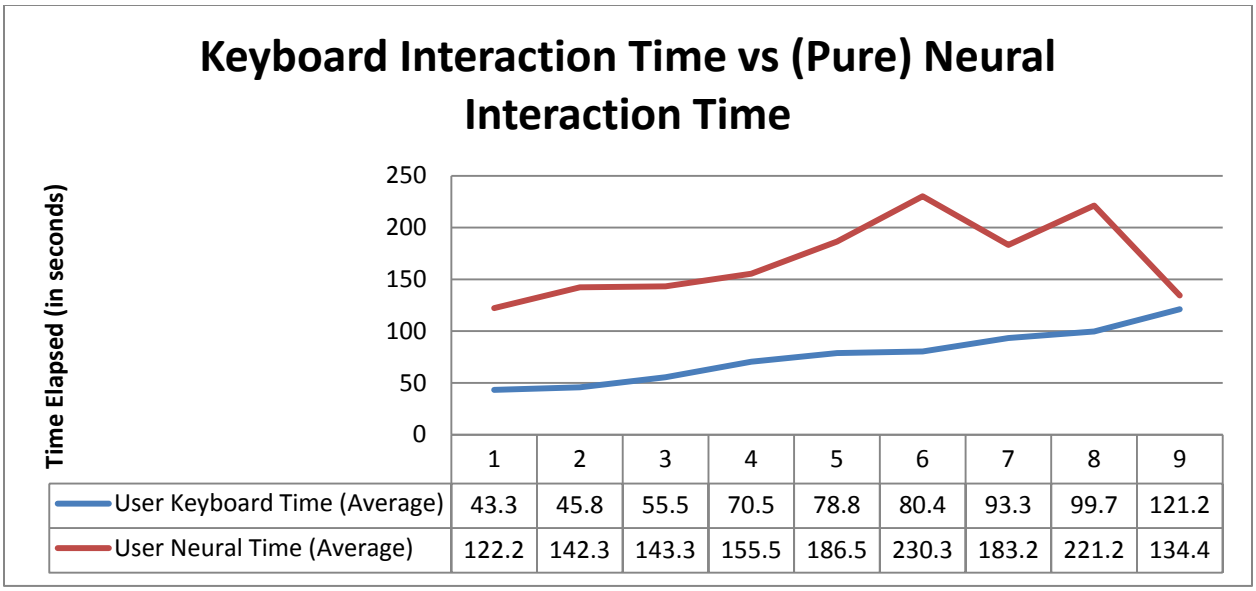
Initial testing was done in-house without the use of outside user specifications. To test the effectiveness of these events, an interactive demonstration was developed. The demo enabled a user to have their avatar move freely within the environment using the W-A-S-D keys and camera movement controlled by the mouse. A short game was developed around the current Virtual Theatre environment, requiring the user to be able to activate all of the events in a given specified order (some in rapid succession).

The demo was designed so that the user was placed within an audience, standing directly in front of an elevated stage. The right of the stage stood a podium which contained a water bottle. The user was able to control the avatar and move around within the environment by using the WASD keys, as well as by moving the camera by holding down the right mouse button and moving the mouse itself. The user was then tasked with moving the avatar and the camera so that they are positioned in front of the podium looking just above the top of the water bottle. The user was then required to activate the “floating sphere” event (Alpha waves dominant and at 50% or higher), in order to generate the orb. Once the orb was generated, the user was able to pull the orb to the avatar using their alpha wave spikes, and (if CAPS LOCK is toggled) push the orb away from the avatar using the same criteria. The user was also able to set the orb on fire through

the use of the tension impulse. The user's goal was to be able to push the orb onto the stage from the audience. However, as the stage was elevated, the ball will always stop short of the stage whenever the avatar was positioned on the ground. The user needed to activate the Levitation Event (beta dominant and spiking 35% or above), pull the orb to the avatar, and then quickly push the ball away in the direction of the stage. If this was all done in sequence, the orb just needed to be set aflame in order for the Inferno Event to be triggered, which will make the stage erupt into a roaring wildfire. The demo was then concluded.

Testing was completed in multiple phases, with several chances for iterative development. Initially, the subject was be tasked with completing the demo using only the neural events. The second time this was completed was by making use of keyboard commands for the event triggers. This included a button press for each possible event. For instance, in order to trigger fire, the NUMPAD 4 key would need to be held down. Also, the instant the button was depressed the false impulse would stop signaling, and the event would stop.

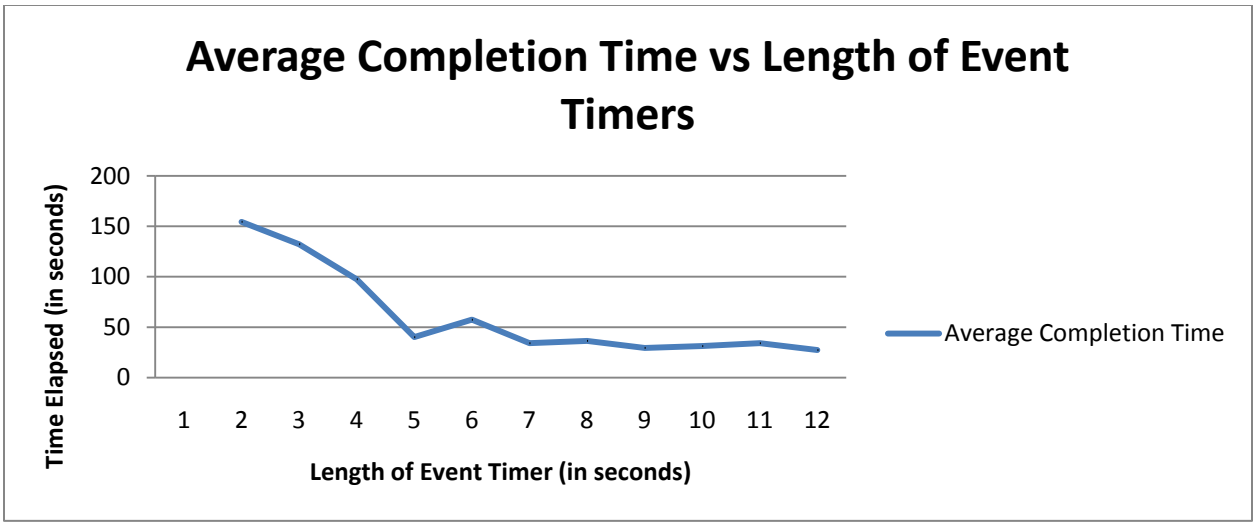
It is to note, that most users complained about the complexity of the key commands. In order for a person to position the ball on the stage, it would require at least four or five keys being pressed at the same time, in addition to having one finger in use with the right mouse button. This came in stark contrast to the neural impulse commands being reduced to solely controlling the WASD keys and the mouse.



	1	2	3	4	5	6	7	8	9
User Keyboard Time (Average)	43.3	45.8	55.5	70.5	78.8	80.4	93.3	99.7	121.2
User Neural Time (Average)	122.2	142.3	143.3	155.5	186.5	230.3	183.2	221.2	134.4

(Figure 1)

The results of the initial testing can be seen in Figure 1. Despite the additional benefits of the headset, the keyboard events still constantly outperformed the neural events by a margin of 30% - 150%. From this point it was realized that subjects had difficulty continuously pumping out a certain type of brainwave in order to activate an event. This was due to the fact that waves are only generated at higher frequencies when forced to (ie, a meditating person would cause an increase in beta waves spikes, and not overall beta waves). In order to aid the effectiveness of the neural commands, timers were added to each event. The timer function allowed for events to continue for a certain period of time once triggered. This enabled for users to trigger an event for a longer period of time, and allowed them to focus on generating the next command while still utilizing the previous event. Testing was then completed to determine how long each event should continue for. The experiment performed was to have moderately experienced users test the system with incrementing levels of event times (Figure 2).

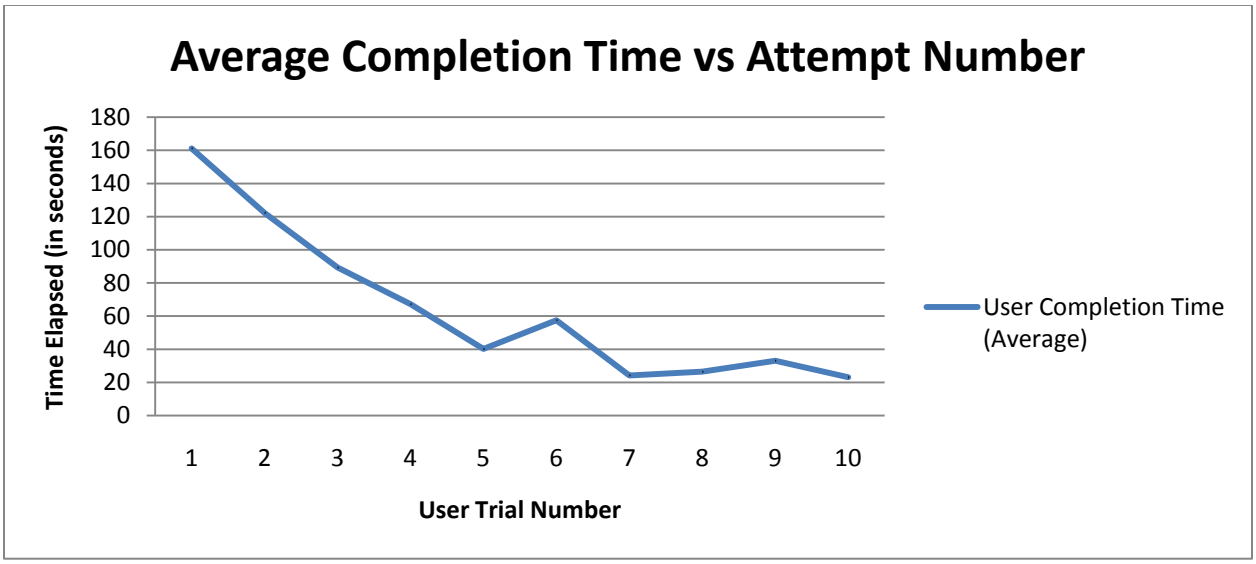


(Figure 2)

The results from the experienced user test (figure 2) was that overall, an increase in the amount of time an event lasted, the quicker the demo could be completed. However, completion time became stable after the five second mark, indicating that once a user had mastery over summoning an event, it didn't matter whether or not the event was held for five seconds or more. This could be attributed that the subject has obtained a mastery of event triggering and didn't need a window greater than five or so seconds to be able to recreate an event. Thus, for the remainder of testing, events were held at a timer of five seconds. Any less and the user experienced a level of difficultly multitasking events, and any longer and the user began to rely on the system to complete itself.

In conjunction with the "five second timer" we tested to what degree the users were able to adapt with the system (Figure 3). We tested how efficient users were with completing the demo over a given amount of attempts. The results proved that like most systems, time yields proficiency and experience. Users were able to harness their brainwaves as well as an understanding of the environment and avatar control at efficient ratios. Experienced users were

able to master the system and the event controls within a few attempts. After seven or eight attempts, the average user was able to complete the tutorial within half a minute, give or take ten seconds.



(Figure 3)

From this testing we were able to also receive a large amount of user feedback. One of the complexities of using an EEG device for interactive media is that each user contains a unique set of brainwaves. Unlike keyboard and joystick interactions, it is much more difficult for a person to force concentration than it is to push a button. In this respect, each test user demonstrated a unique interaction with the device based upon predetermined patterns. For instance, users who described themselves as calm and relaxed would be able to adapt to the beta wave events much quicker than those who described themselves as focused or thinkers.

Likewise, many users were able to produce alpha waves at relatively higher frequencies, and had difficulty recreating beta waves.

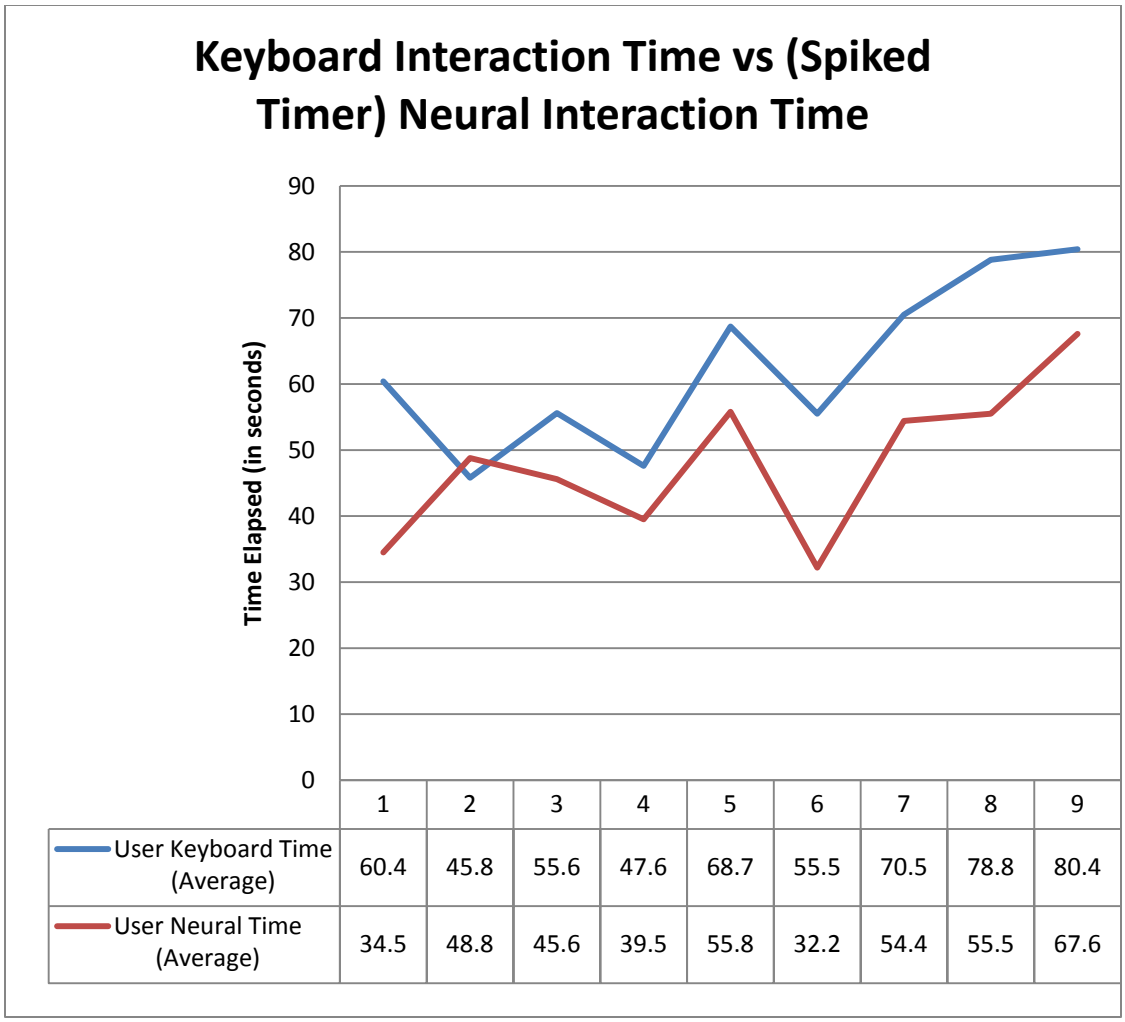
The majority of problems which arose from this project stemmed not from the events themselves, but from the neural headset. The NIA, while able to accurately obtain impulse readings, is a severely volatile piece of equipment. The headset is limited in its ability to obtain more than two wave types. While it is able to interpret both EEG (alpha and beta) as well as Muscle (tension, glancing) data, it is limited in its intake. Most EEGs are able to translate both gamma and theta waves, which require a high level of precision and decryption by the neural interpreter. Event triggers for the NIA device were only able to be coded from either alpha, beta or tension waves (glancing was not used), which inevitably restricted the level of complexity that events could be designed.

Furthermore, the NIA box required constant grounding to the user. Outside devices such as pagers, cell phones, iPods, etc interfered with the biofeedback being read from the headset, and would often return false positives. In testing, the worst case scenario was that the NIA would either return 100% or 0% for all impulse waves, and nothing in between. This was eventually solved by having the NIA interpreting box be strapped to the user's arm during testing. This essentially grounded the device, with only a mild irritation to the user.

Another common issue discovered was the problem of interfering data. A variety of activities could adversely affect the results of the NIA headset, and this often returned high to maximum values for each possible wave. This was commonly attributed to whenever the headset was either at rest or being interfered with by some other peripheral device (cell phone, mp3 player, etc). Whenever these massive spikes would occur, all events would systematically activate their impulse requirements. A user could activate the Levitate, Floating Sphere, Fire, and

Pull events within the first few seconds of the demo if the headset was not calibrated properly. This was dealt with by adding the dominant features to each of the impulse triggers. Dominant requirements prevented multiple actions from occurring at once, and forced the user to focus on which event should occur within the world instead of just hitting the prodding and hoping for the best.

Incorporating this feedback, we designed a final iteration of the system using the five second timer, a slight alteration to impulse requirements (lowering beta from 50 to 35% for instance), and the dominant feature. From there we tested the experienced users again, all of which had both run through the simulation at a minimum of seven times with using both keyboard and impulse commands (Figure 4). Of the nine documented users, 8 out of 9 were able to perform quicker using the neural headset with a four second event timer over the keyboard input commands by an average range of 10-20 seconds per user. These results show that the headset was able to allow for subjects to increase their efficiency with the system, once they had both experience with the tutorial as well as generating events.



	1	2	3	4	5	6	7	8	9
User Keyboard Time (Average)	60.4	45.8	55.6	47.6	68.7	55.5	70.5	78.8	80.4
User Neural Time (Average)	34.5	48.8	45.6	39.5	55.8	32.2	54.4	55.5	67.6

(Figure 4)

Note: Spiked Timer is equal to five seconds per impulse spike.

4. Conclusion/Future Work

Through our development, we were able to prove that avatar control can be improved through the use of a Neural Impulse Actuator. Design was heavy on the development of events and their ability to communicate between the NIA Client and the Virtual Theatre Environment.

By introducing timer functionality alongside brainwave spikes, it was possible to allow for a quicker completion time of the NIAClient simulation using the neural impulse commands over the use of conventional keyboard interaction. Dominant features were incorporated as well to ensure that only one spike could be activated at a time, and that the headset could simply be tricked into firing every impulse at once.

However, this technology is still in its infant stage. While the ability to design a system to invoke avatar actions exists, further research has to be done in the realm of EEG development and design. As this technology progresses, headsets will need to focus on the following improvements:

1. Better algorithms for dealing with white noise.
2. Greater number of nodes.
3. Higher degree of scanning sensitivity.
4. More impulse parameters (gyroscopes, theta/gamma waves)
5. Decrease in overall price.

Currently, the forerunner in the development of interactive avatar headset is Emotiv with their EPOC headset controller. Their design incorporates fourteen nodes, a gyroscope, and additional wave sensors. However, they are still running into the same difficulties as the NIA in terms of white noise algorithms, and sensitivity[6]. The epoc is completely wireless, which introduces the problem of cell phone and other high frequency interference, especially when used in highly populated areas. While the NIA fixes this with the grounding box, algorithms still need to be designed to completely solve the problem. Furthermore, the epoc controller is three times

as expensive as its NIA cousin, and as there are only a handful of demo games available for use by the device. Development upon the hardware must come before this line of product can become common place within avatar interaction. However, as this technology progresses, greater focus will inevitably be upon how the developer is able to incorporate the headset and its impulses into the development of new events and event triggers. In the end, both the NIA and the EEG devices are only tools for receiving data, and it is how that data is used which can truly define an avatar's experience and a user's immersion.

5. References:

1. Greene, Kate. "Connecting Your Brain to the Game." Rev. of *Emotiv Systems*. *Technology Review*. MIT, 7 Mar. 2007. Web. 17 June 2010. <http://www.technologyreview.com/biztech/18276/>.
2. Leeb, R., Friedman, D., Müller-Putz, G. R., Scherer, R., Slater, M., and Pfurtscheller, G. 2007. Self-paced (asynchronous) BCI control of a wheelchair in virtual environments: a case study with a tetraplegic. *Intell. Neuroscience* 2007 (Apr. 2007), 1-12. DOI= http://dx.doi.org/10.1155/2007/79642
3. Kuikkaniemi, K., Laitinen, T., Turpeinen, M., Saari, T., Kosunen, I., and Ravaja, N. 2010. The influence of implicit and explicit biofeedback in first-person shooter games. In *Proceedings of the 28th international Conference on Human Factors in Computing Systems* (Atlanta, Georgia, USA, April 10 - 15, 2010). CHI '10. ACM, New York, NY, 859-868. DOI= http://doi.acm.org/10.1145/1753326.1753453
4. Emotiv's EPOC: http://www.emotiv.com/apps/sdk/209/
5. OCZ's NIA: http://www.ocztechnology.com/products/ocz_peripherals/nia-neural_impulse_actuator
6. Dakan, Rick. "Emotiv EPOC, Tough Thoughts on the New Mind-reading Controller." Rev. of *Emotiv Systems*. *Joystiq*. Weblongs, INC, 7 Jan. 2010. Web. 11 Aug. 2010. http://www.joystiq.com/2010/01/27/review-emotiv-epoc-tough-thoughts-on-the-new-mind-reading-cont/