

Unleashing Brain Powers

A Study on Development of BCI-enhanced Computer Games

Supervised by:
Prof. Michael R. Lyu
Written by:
Cheung Kwan Yau

Individual Report
Spring Semester 2010-2011



Department of Computer Science and Engineering
The Chinese University of Hong Kong

Abstract

From keyboard and joystick, to Wii-remote and Kinect motion detection, new controllers have always been fuels to bring about new generations of video games. However, when possibilities of motion sensing entertainment are gradually exhausted, one may wonder: what would come next?

We believe Brain-Computer Interface (BCI) might just be a potent candidate.

Our project starts off by studying the consistency and effectiveness of a non-invasive consumer-level BCI, Neurosky Mindset. Afterwards, we attempt to devise a way to improve the BCI control, namely by introducing a way for the users to train themselves to maintain a better control over the brainwave states. In this process, we shall propose our customized algorithm to approximate the attention meter.

Finally, utilizing the modern 3D computer game engine Unreal Engine 3, a complete BCI-enhanced adventure game titled “Psionscape” is developed. The game consists of puzzle elements driven by our newly developed BCI algorithm at the back-end, and this shall further strengthen the belief that BCI-enhanced gaming will be a feasible game genre in the future.

Contents

Abstract.....	1
Contents.....	2
1 Introduction	4
1.1 Motivation.....	4
1.2 Background	6
1.2.1 Research-Level VS Consumer-Level BCIs	6
1.2.2 Brain-Computer Interface Selection	8
1.3 Game Industry Responses to BCI	14
1.4 Project Overview.....	16
2 Studying NeuroSky Mindset.....	20
2.1 Background	20
2.1.1 Brainwaves	20
2.1.2 What Can Mindset Do.....	20
2.1.3 How does Mindset Work	21
2.1.4 Mindset Structure	23
2.1.5 eSense Meters.....	23
2.1.6 Raw Brainwaves	24
2.2 Experiment on eSense	26
2.2.1 Hypothesis.....	26
2.2.2 Methodology.....	26
2.2.3 Result.....	29
2.2.4 Conclusion for result and data analysis	42
3 Brainwave Classifier	43
3.1 Introduction	43
3.2 Working flow of Brainwave Classifier.....	43
3.2.1 Calibration Stage	43
3.2.2 Classifying Stage.....	44
3.3 Brainwave Types.....	45
3.3.1 Type 1	46
3.3.2 Type 2	46
3.3.3 Type 3	47
3.4 Algorithm	48
3.4.1 K-mean clustering	48

3.4.2	Sliding Window	50
3.4.3	Calculating the Brainwave Type	51
3.5	User Interface.....	52
3.5.1	Overall View	52
3.5.2	Individual Channel and Overall Channel.....	53
3.6	Personal Training.....	54
3.6.1	Keeping Type 1	54
3.6.2	Keeping Type 3	55
3.6.3	Oscillating between type 1 and type 3	56
3.7	Experiment on Brainwave Classifier.....	57
3.7.1	Objective	57
3.7.2	Methodology.....	57
3.7.3	Result.....	58
3.8	Control Method.....	62
3.8.1	Keeping Type 1	62
3.8.2	Keeping Type 3	62
3.9	Summary	63
4	Training Method.....	64
4.1	Introduction	64
4.2	Training Flow	64
4.3	Personal Profile	65
4.4	Summary	66
5	Conclusion.....	66
6	Limitations & Difficulties.....	68
6.1	Poor Signal	68
6.2	Curve Matching.....	68
6.3	Passive Control.....	68
7	Reference	70

1 Introduction

1.1 Motivation

No one knows what the first digital game was, since it is debatable upon how one defines a “game”. However, one thing is certain - Since the invention of digital games, human has been actively pursuing new kinds of interfaces which enable communication between the games and the players,



Figure 1.1-1. For PC games, they started with a keyboard

or at least, enable one-way input from the players to the games.

trackball as pointing devices, they served as an additional interface when combined with the keyboard, or sometimes control games even without the keyboard.

Not long after the popularization of the mouse, the game industry evolved to create different controllers to suit different games’ needs.

In the past 10 years, the gaming industry has been a growing multi-billion-dollar business, this shows that the demand of videos games has been growing, and this rocketing demand also attracts a vast investment

In personal computers games, keyboard was one of the first interfaces, where early games usually used the arrow keys (e.g. DOOM, a first person shooter game, used $\leftarrow \uparrow \rightarrow \downarrow$ for movement). With the introduction of mouse and



Figure 1.1-2. A mouse was necessary to play Warcraft II (1995)

on new gaming interfaces, such as Dance Pad in PlayStation and Wii controllers in Wii, which furthered feedback to the snowball (i.e. the demand) positively.

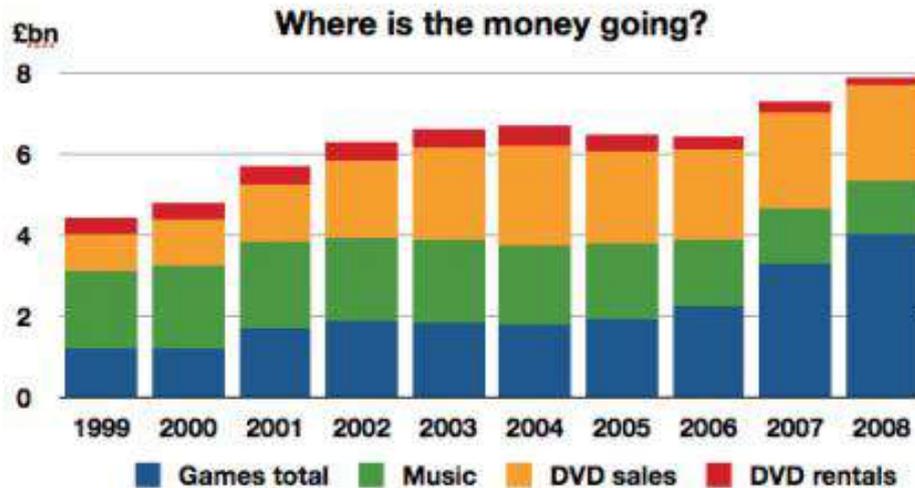


Figure 1.1-3. The video game industry has been growing at lightning speed, even exceeding income generated by the music industry.

(Arthur, 2009)

However, while new interfaces for console games (e.g. touch screens for NDS and remote motion sensors for Wii) has been developed, emergence of new gaming interfaces for PC games seem to slow down after the introduction of game pads, and we think a new gaming interface could perhaps give birth to a new genre of games in the big PC games market, where PCs are very widely owned in almost every family in developed countries.



Figure 1.1-4. With Microsoft's Kinect, players no longer hold any sensors to play

Following the current trend, we could see that the physical world has

almost been captured by interfaces like Microsoft Kinect. Hence one may wonder if anything new would appear in the future gaming world, and what that would be.

Recently, brain computer interfaces for consumer level have been released to the market (See session 2.2.2), making BCI entertainment possible. However, no commercial game has been released onto the market.

Why is that so? Is it because of technical difficulties to utilize the BCI?

We would like to grab that very opportunity by studying the possibility of developing a modern 3D computer game, which can utilize features of BCIs.

1.2 Background

1.2.1 Research-Level VS Consumer-Level BCIs

Some of the game interfaces, such as keyboards and gamepads, which control games directly, including movement, rotation and inventory controls, are usually of high precision, i.e. you wouldn't see a mouse cursor shivering on the screen or floating in random direction.

Research level BCIs could also read the electroencephalography (EEG) in a relatively accurate manner, allowing recognition of many different actions. For example, the Department of Electronic Engineering of the Chinese University of Hong Kong has been working on a BCI project, where the user could input Chinese characters using one's brain waves alone, but the process requires about 1 minute per word and yields 75% accuracy (Oriental-Daily, 2010). For a game related example, The Department of Neurosurgery of Washington University in St. Louis managed to control a classic game "Space Invader" by moving the space ship left or right (note that only 1D linear movement is allowed), where the shooting behavior is governed by time automatically but not by the BCI. The process time for this kind of in-game operation is shorter, and the player could react to the bullets shot by the

aliens on the top of the screen.



Figure 1.2-1 Research-Level BCI-controlled Space Invader

(<http://www.youtube.com/watch?v=T3-mxhDp-u8&feature=related>)

However, consumer-level BCIs are currently not as precise, and are not favorable to be used as direct controls; In addition is the slow recognition speed relative to quick response needed to react to game events (such as shooting a zombie which suddenly popped up in front of the player).

Yet consumer-level BCIs also have their advantages. While a P300 system (the research-level BCI system developed by CUHK mentioned above) may take from 20K to 50K HKD each, consumer-level BCIs are more affordable (cost no more than a few thousand HKDs) by ordinary players. Moreover, they are all very portable and mobile, due to their wireless nature. Another big advantage is the ease to wear the BCI, as they mainly use dry sensors and do not require a very accurate positioning of each electrode (in contrast to research-level BCIs which need to position at least a few dozens of electrodes

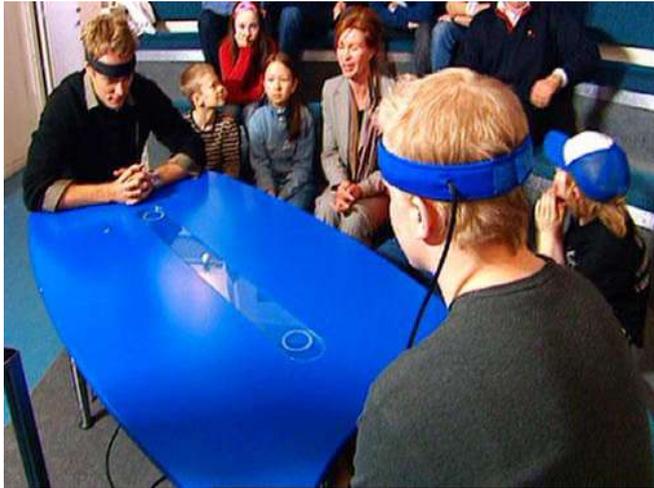


Figure 1.2-2 Mindball was one of the first consumer-level BCIs

before use). Hereafter, the BCIs we talk about would be consumer-level ones.

1.2.2 Brain-Computer Interface Selection

There are vast variety of game-player interfaces in the market, such as Wii-remote and Drum Sticks, but very few of them are intended for personal computer entertainment. However, the consumer level BCIs in the market, up to now, are for that purpose. Therefore connections are easy, usually involving a USB plug-and-play, and those BCIs are usually very light and portable.

Within the ranges of consumer-level BCIs, there are ones with very specific uses.

For instance, 7 years ago, a company called The Interactive Productline which is located in Sweden produced one of the first consumer-level BCIs, named Mindball. A wirelessly controlled ball was place in the middle of a table, the players first attach a BCI onto their forehead, and it will read the EEG of them. As a result, a ball will move from one side to the other player's side when one of the players' focuses more intensively onto the ball, and the ball falls off the edge of loser's side.

However, this kind of BCI did not provide a general use for other

application, and is not viable for development of computer software or games.

In March 2007, NeuroSky, a US-based company, released a headset attached with an EEG sensor, which is to be placed onto the user's forehead. The headset is called Mindset. This should be one of the first BCIs intended for consumer-level computer uses. The single sensor used was a dry one, and is non-invasive (in contrast to medical BCIs which may require insertion of electrodes into the skull for superior accuracy). The device features a decomposition of a whole range of raw brain waves data from the user, including alpha, beta, gamma and theta waves (See Chapter 3 for explanation of brain waves), and also algorithmic values representing "Attention" and "Meditation", consolidated by the raw brain waves data. "Anxiety" and "Drowsiness" are also supported using particular software. The latest firmware even allows the detection of eye blinks, but the underlying principles are not known.



Figure 1.2-3 Wireless with a dry sensor - NeuroSky Mindset

A year later, OCZ Technology released the Neural Impulse Actuator (NIA). The NIA is worn by putting it around the user's forehead, and it is very easy to do so due to the rubber-band-like structure (see Figure 2.2-4). The communication between the NIA and the computer is established by a USB connector (not wireless). There are a total of 3 sensors, which is more than that of Neurosky Mindset. However, the device can only read alpha and beta waves, although there is an overall motor neuron activity (e.g. moving your jaw up can increase its value). Moreover, unlike Neurosky Mindset, there is no processing of raw data, so it is even difficult for developers to tell when the user is in "Attention" state. Yet supposedly, the accurate nature of the 3



Figure 1.2-5 Easy-to-put-on NIA

sensors may yield more precise brain waves which may be mapped to some keystrokes to play certain games.

The Emotiv EPOC is the latest BCI available came in December 2009. This BCI has 14 electrodes and so far is the BCI with the highest number of electrodes. That is very comparable to medical-level BCIs which usually has 19 electrodes. The vast number of electrodes covers different areas around the head, and thus has a lot of features. Therefore it can detect 12 kinds of movement and rotations (e.g. “up”, “left”, “forward”, “zoom”, “turn clockwise”, “turn left” and “sway right”) as supplemented by detecting motor neuron activities. Similar to Neurosky Mindset, it can detect emotions such as “Excitement”, “Engagement”, “Meditation” and “Frustration”. Moreover, it can detect facial expressions like laughing and clenching. Another feature Emotiv EPOC exclusively demonstrated is the ability to make objects disappear in the demo. In

addition to BCI features, it can also measure angular rotation of the head in 2 dimensions (i.e. yaw and pitch, but not roll) as detected by the 2 gyros.



Figure 1.2-4 Highest number of sensors - Emotiv EPOC

To sum things up, we have prepared a comparison table for the BCIs mentioned above:

	Mindball	Mindset	OCZ NIA	Emotiv EPOC
Released	March 2003	March 2007	May 2008	December 2009
SDK Available	No	Yes	Yes	Yes
Connection	Wireless	Wireless	Wired (USB)	Wireless
Number of Electrodes	1	1	3	16
Sensor Type	Dry	Dry	Dry	Saline (Wet)
Raw Data Collection	No	Yes	Yes	Yes
Attention/Engagement	No	Yes	No	Yes
Meditation/Relaxation	No	Yes	No	Yes
Anxiety/Frustration	No	Yes	No	Yes
Drowsiness	No	Yes	No	Yes
Excitement	No	Yes	No	Yes
Push	No	No	No	Yes
Pull	No	No	No	Yes
Lift	No	No	No	Yes
Drop	No	No	No	Yes
Push Left	No	No	No	Yes
Push Right	No	No	No	Yes
Rotate Forward	No	No	No	Yes
Rotate Backward	No	No	No	Yes
Turn Left	No	No	No	Yes
Turn Right	No	No	No	Yes
Tilt Left	No	No	No	Yes
Tilt Right	No	No	No	Yes
Disappearance	No	No	No	Yes
Facial Expressions	No	No	No	Yes
Motor Neurone Activity	No	No	Yes	Yes
Playing Music	No	Yes	No	No
Head Rotation Detection	No	No	No	Yes
Price in USD	\$18700.21	\$159.20	\$138.99	\$299.00
Developer Edition	-	(Free SDK)	(Free SDK)	\$500.00

Table 1.2-1 Comparison between BCIs as of 28th November 2010

To select the most appropriate BCI for our project, we considered a lot

of different features of the above BCIs with weighing.

To begin with, Mindball does not really suit our need because it is of specific use (i.e. for that ball game only), what's more is the terrific price to own one, i.e. \$18700.21, which will make it not probable to become a home entertainment trend. Moreover, there is no possible connection for personal computers, and there is no SDK for development, so we cannot use it to build our game.

The OCZ NIA has the lowest price among the 3 remaining candidates, and it employs the use of dry sensors, which makes the user comfortable with putting this on. It enables raw data feedback to the computer / SDK, and that is useful for customized signal processing. A point to note is that the wired nature of the device may be speculated to reduce the comfort of prolonged use because it hinders the user's movement (e.g. for Mindset, you can go grab a cup of coffee without taking it off). The critical reason for not using OCZ NIA is its failure to detect cognitive states (e.g. Attention). In addition, while it may be possible to detect motor activity, it cannot tell the difference between moving one's jaw and moving one's eyebrows, as complained by the users on forums (Kenner, 2009).

The Emotiv EPOC seems to be very competitive candidate, as it is not only able to detect cognitive states like Neurosky Mindset does, it can also detect motions, which is a very attractive feature for video games. However, while these features are possible thanks to the large number of sensors Emotive EPOC has, saline sensors are employed (not dry). The users may find it troublesome to wet the sensors or their head every time they use it, and it is not just a matter of dropping a few salt solutions onto the sensors but to "make them dripping wet"! Yet even with that, the connection may also be poor. (Emotiv-Administrator, 2010) Moreover, although motion detection is possible, the accuracy is not very high, with some users stating that "it can never follow my thought!" In addition, while there seems to be multiple events to be detected, only a limited number of events can be detected at the

same time, and the user needs to explicitly address them, making EPOC not a very suitable candidate. (Emotiv-Administrator, 2010)

While NeuroSky Mindset is not the most accurate BCI (as it only got 1 pea-sized electrode), it still supports a number of detections such as Attention and Meditation. Moreover, it allows a broader range of raw brain

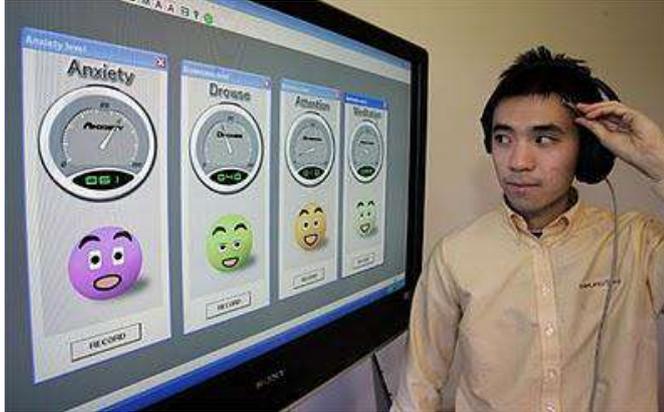


Figure 1.2-6 Some people find it difficult to establish perfect connection

(Notice how the user holds the sensor closely towards the forehead)

better quality than ordinary loudspeakers. Lastly, it has a very low costs compared to other BCIs except NIA, making it more easy to popularize among families as it is more affordable.

However, it also has a drawback (which is also present in other BCIs) – Some people find it difficult to establish perfect connection between the sensor and the users' head. This will be one of the limitations we would like to address later and perhaps during demonstration.

waves data, enabling a more potent signal processing to be carried out ourselves. In addition, as a benefit for gaming experience, it is essentially a wireless headset which can play music, this might enhance the gaming experience by providing surrounded sound of

1.3 Game Industry Responses to BCI

The emergence of different BCIs attracted some attention from the gaming industry, with many of the released or potential products relying on the “Attention” state. (See Chapter 3 for reasons)

For example, Mattel Inc released a non-digital game called MindFlex. In this game, the players need to focus to increase their concentration to raise the ball, and lower the ball by lowering the concentration level, and use a knob to move a ball left or right, with the goal of passing the ball through different obstacles. In fact, this game is using a lite version of NeuroSky Mindset’s chip (the “Mind Force” chip).



Figure 1.3-1 MindFlex (Based on Neurosky Mindset)

Now here comes a big question:

Is there any game-developer-made BCI computer games?

The short answer is: No.

For demonstration purpose, the BCI producers, of course, produced their own game demos.

For example, NeuroSky, the same company which developed the Mindset, built a game called “The



Figure 1.3-2 NeuroSky’s own game - The Adventure of NeuroBoy

Adventure of NeuroBoy” to demonstrate features of Mindset. The game, which comes free-of-charge with the Mindset, has no story at all, but let the player takes control of a character walking around using WASD keys and use the mouse to select an object for one of the 4 purposes: Attract towards player, Push away from player, Levitate and Ignite, which are governed by the 2 states (i.e. “Attention” and “Meditation”).

However, not all hope is lost.

Announced at the 2008 Tokyo Game Show was good news about commercial BCI games. The Japanese game developer Square Enix, which is well-known for its Final Fantasy series, announced the development of the first BCI-enhanced computer game – Judecca. (Fruhlinger, 2008)

Judecca is a first-person shooter game, in which the player is immersed in a world of zombies. The game makes use of NeuroSky Mindset’s “Attention” level. Up to now, the announced BCI features are:

- 1 “After concentrating on a glyph that glows in direct relation to your ability to concentrate, you will open up what’s called your “Devil’s Eye”. Only once you have attained a heightened state of concentration, will you be able to see Judecca’s zombies and kill them.”
- 2 “Those who can tweak their concentration levels even further will be able to walk through walls.”

If that is not descriptive enough, the below is a screenshot which reflects what happens if the “Attention” / “Concentration” level is high enough in Judecca.

An important point to note is that the game is BCI-enhanced but not directly controlled by it, the movement and shooting still relies on keyboard and mouse. With a global game developing company like Square Enix still not working on more BCI features or direct controls, this leads to our speculation that the current game industry is still remain doubt about the accuracy of consumer level BCIs.

1.4 Project

Overview

Chapter 2.2.1 gave us an insight on the current trend of consumer BCIs and also their limitations.

On one hand, we can see that BCIs are rapidly being improved and commercialized. With the lower costs and improving accuracies and features, we predict that in the near future, there will be at least a small to medium sized market for home uses.

On the other hand, we could see there are limitations for current BCIs. The most important part is the difficulties in deciding detected states (e.g. "Meditation") and movements (e.g. "Move forward and Turn Left").

Chapter 2.2.2 concludes by stating that the current computer game industry actually is at the beginning of "trying" to develop BCI games. With a large company like Square Enix (a publicly owned multinational company with thousands of employees), they still limit the game with very few BCI features, possibly because of the limited accuracy for direct movement controls.



Figure 1.3-3 Judecca in action, revealing zombies using Concentration level

We speculate that the lack of developers building BCI games is due to:

- 1) It may be hard to make games which utilize BCI features
- 2) BCI may just unlock too few in-game features. (e.g. revealing zombies)

Therefore, we would like to see if we can tell a different story by dividing our project into 2 phases, with each phase done within each semester.

Firstly, we will study the NeuroSky Mindset, to see how we could operate it, and how we could get data from it. Moreover, we will try to see if there is any trace of correlation between its claimed states (e.g. "Attention" and "Meditation") and the users' feedbacks.

On the other hand, we will study a game engine, Unreal Engine 3, to see if it is possible to make a BCI-enhanced game (at this stage, we do not plan to build a game "directly controlled" by the BCI, as learnt from lessons of the Judecca). If a modern 3D game engine can be modified to produce BCI-enhanced games, then it would be like owning a factory to developers, and they can make BCI-enhanced games pretty much like how they normally do it.

Moreover, we will study how BCI could help facilitates different game events or features.

For the first phase, we will first make a small demo in Unreal Engine 3 environment to demonstrate that it could actually work to combine BCI and an ordinary game engine. Then proceed to demonstrate the possibility of active controls (See Chapter 4 for details about Active/Passive Controls) using NeuroSky Mindset's eSenses (e.g. "Attention").

For the second phase, we will try to analyze raw brain waves data to see if we could devise our own algorithms to calculate values which represent human emotions, and see if we could improve eSenses or develop other senses (e.g. the state of "Fear"). And on the other hand, investigate the

possibility of passive controls by the BCI.

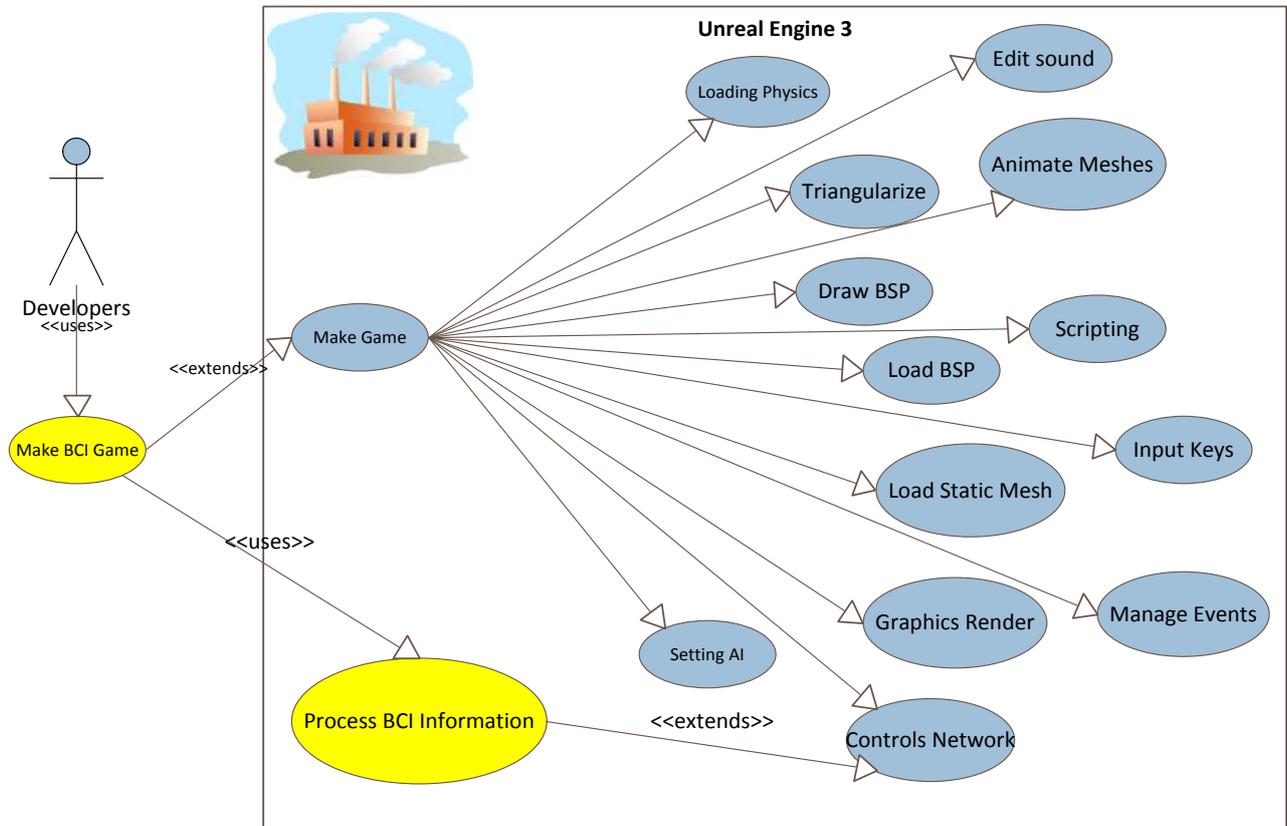


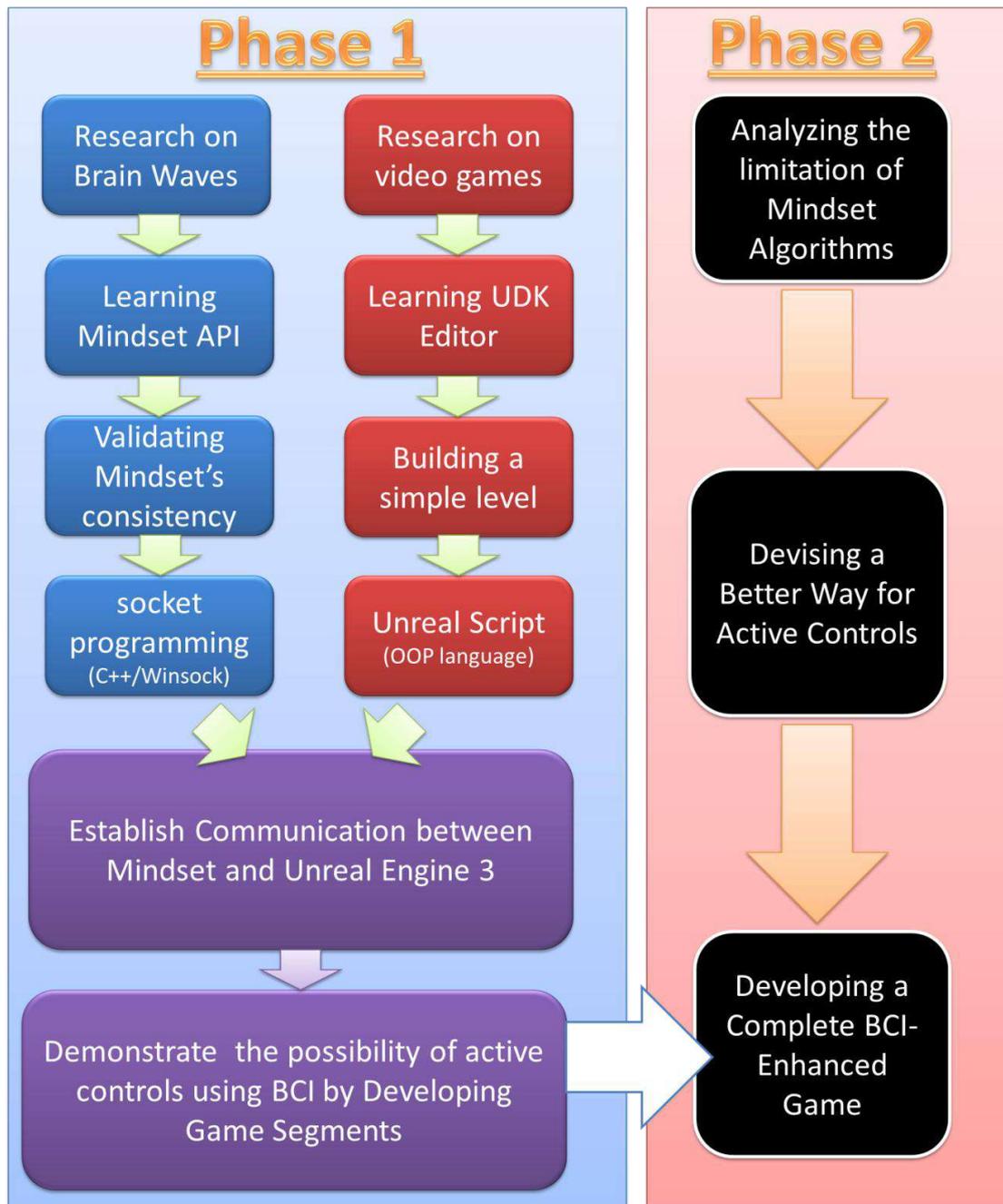
Figure 1.4-1 Use Case Diagram of Modified Unreal Engine 3

Last but not least, we are looking forward to creating a small-sized BCI-enhanced game by integrating different kinds of controls, and evaluate it to see if the players find it more interesting than non-BCI versions.

(Disclaimer)

Throughout our project, we may use the phrase “BCI-enhanced games”. However, by “enhanced” we simply mean the games which are different from those in the current game market in a way that BCI features are absent in present games, we do not attempt to have a bias that “BCI-enhanced” implies an positive impact on the gameplay experience

over “ordinary” games. In fact, the question whether games with BCI features would offer a perceived better gameplay experience is exactly what we would like to find out, after developing a small-scaled game featuring some BCI features in the coming semester.



2 Studying NeuroSky Mindset

2.1 Background

2.1.1 Brainwaves

Brainwaves are the records of electrical activity along the scalp produced by the firing of neurons within the brain. Source of brainwave came from neurons. Neurons are electrically charged. When they receive a signal from the neighbor, it releases ions outside the cell. When many ions of like charge repel each other, it will push out its neighbor and form a wave of ions. When the wave of ions reach the scalp, the electrons inside the electrode or the sensory will be pulled or pushed, which forms a voltage, measure such voltage difference between different electrodes or sensory form the brainwaves.

2.1.2 What Can Mindset Do

Data Collection

Mindset measures the electrical impulses generated by mental activities. This provides a raw unfiltered brainwave measurement. The raw data can be used by researchers and developers to make algorithm to measure mental states. Raw data includes alpha waves, beta waves, gamma waves, delta waves and theta waves.

Mindset also uses algorithm to calculate the mental states, such as attention and meditation which are available as digit input for games and educational applications.

Both raw brainwave data and calculated mental states are updated once per second.

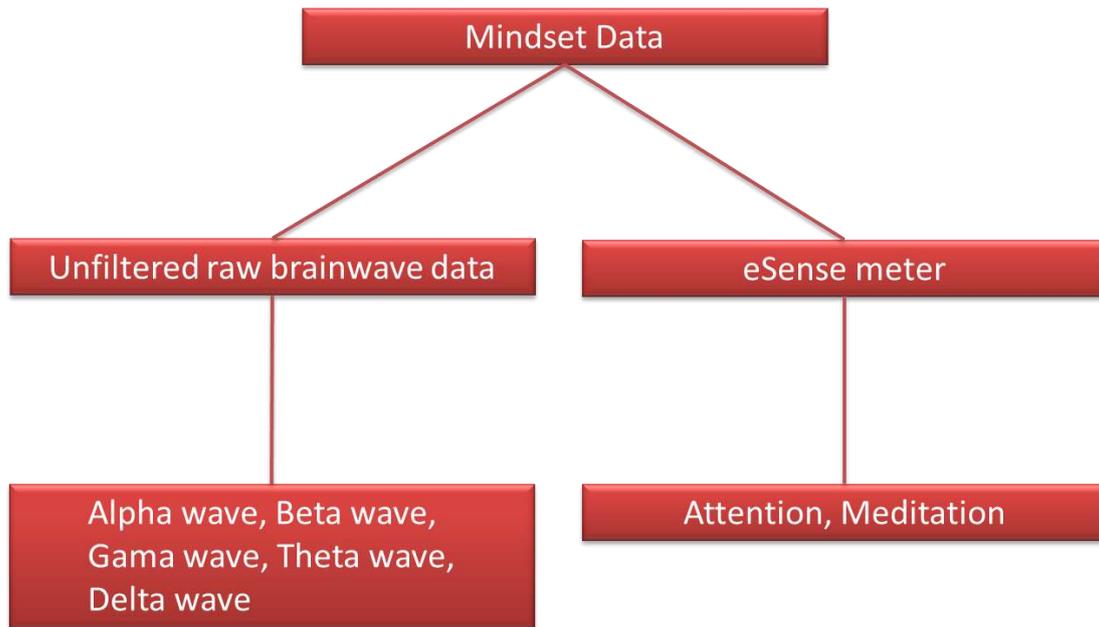


Figure 2.1-1 Classification of mindset data

Mindset SDK

Neurosky provides a Mindset Development Tool (MDT) for developers and researchers to create and publish applications using BCI technology of Mindset. The MDK provides documents, drivers and sample codes for development. Developers can collect Mindset data in different platform (Windows, Mac) and using different language(C, C++, Java).

2.1.3 How does Mindset Work

ThinkGear

Mindset use a technology called ThinkGear to interface wearer's brainwaves. This technology includes the sensor that touches the forehead, the contact and reference points located on the ear pad and the on-board chip that processing all the data. Both eSense Meters and raw data are calculated on the ThinkGear chip. These data are then sent to the computer through Bluetooth.

eSense

eSense™ is a NeuroSky proprietary algorithm for characterizing mental states. The NeuroSky ThinkGear technology amplifies the raw brainwave signal and removes the ambient noise and muscle movement. The eSense algorithm is then applied to the remaining signal, resulting in the interpreted eSense meter values.

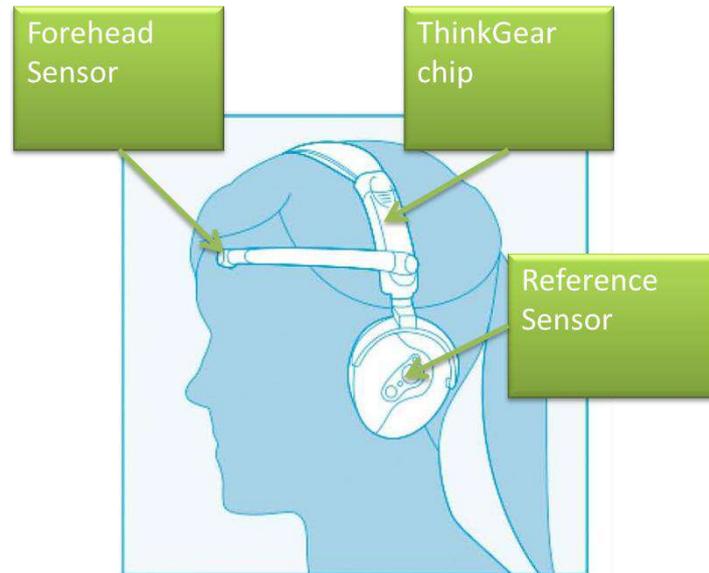


Figure 2.1-2 Key components of Mindset

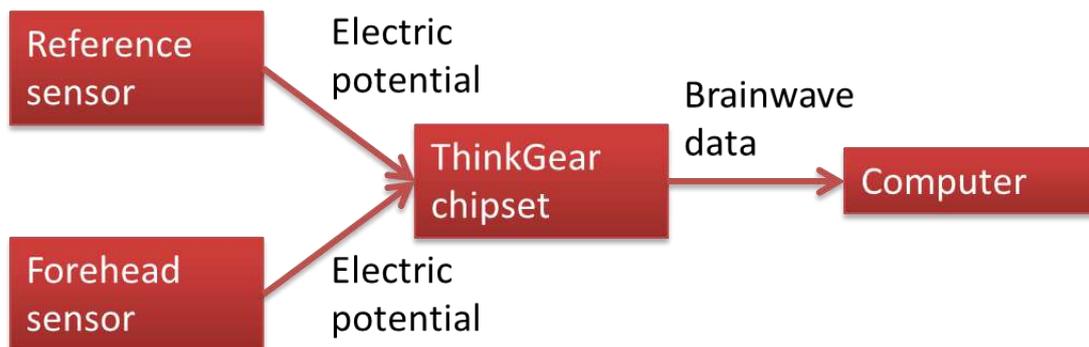


Figure 2.1-3 Data flow diagram from mindset to computer

The data flow of mindset is as follow. First, the forehead sensor and the reference sensor record the electric potential and pass the values to the ThinkGear chipset inside the Mindset. The chipset processes the data and output the raw brainwave data and calculated mental states to the

computer.

2.1.4 Mindset Structure

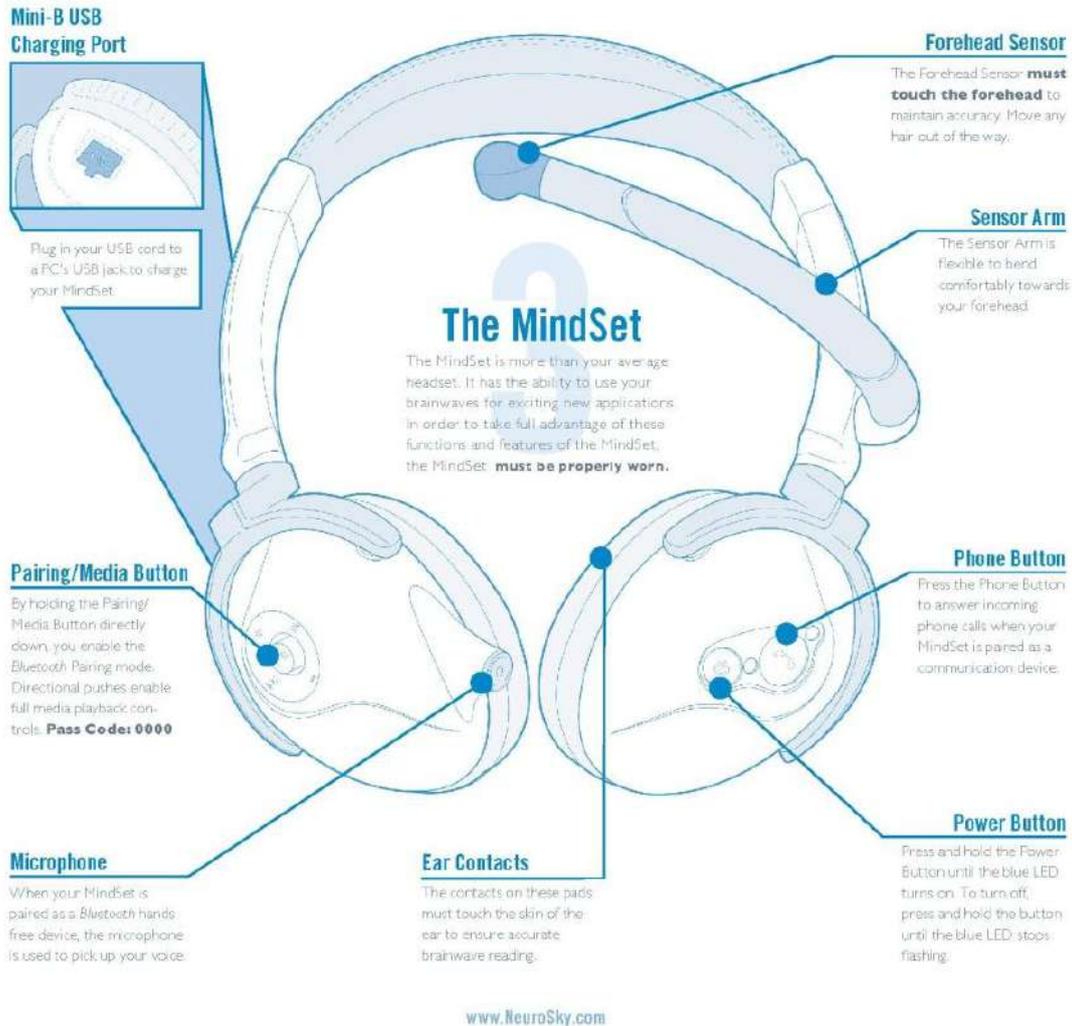


Figure 2.1-4 Complete Mindset components and functions

The Mindset is a head-phone liked BCI. The components of the Mindset include Forehead Sensor, Sensor Arm, Ear Contacts, Microphone, Headphone and different buttons.

2.1.5 eSense Meters

The eSense Meter contains two mental states: Attention and Meditation. The meter ranged from 1 to 100. Values between 40 and 60 at any given

moment in time are considered "neutral". Values from 60 and 80 is considered "slightly elevated", and may be interpreted as levels being possibly higher than normal. Values from 80 to 100 are considered "elevated", meaning they are strongly indicative of heightened levels of that eSense. Similarly, on the other end of the scale, values between 20 and 40 indicates "reduced" levels of the eSense, while values between 1 and 20 indicates "strongly lowered" levels of the eSense.

Both eSense values are updated once per second.

Attention eSense

According to the mindset documents, the eSense Attention meter indicates the intensity of a user's level of mental "focus" or "attention", such as that which occurs during intense concentration and directed (but stable) mental activity. Distractions, wandering thoughts, lack of focus, or anxiety may lower the Attention meter levels.

Meditation eSense

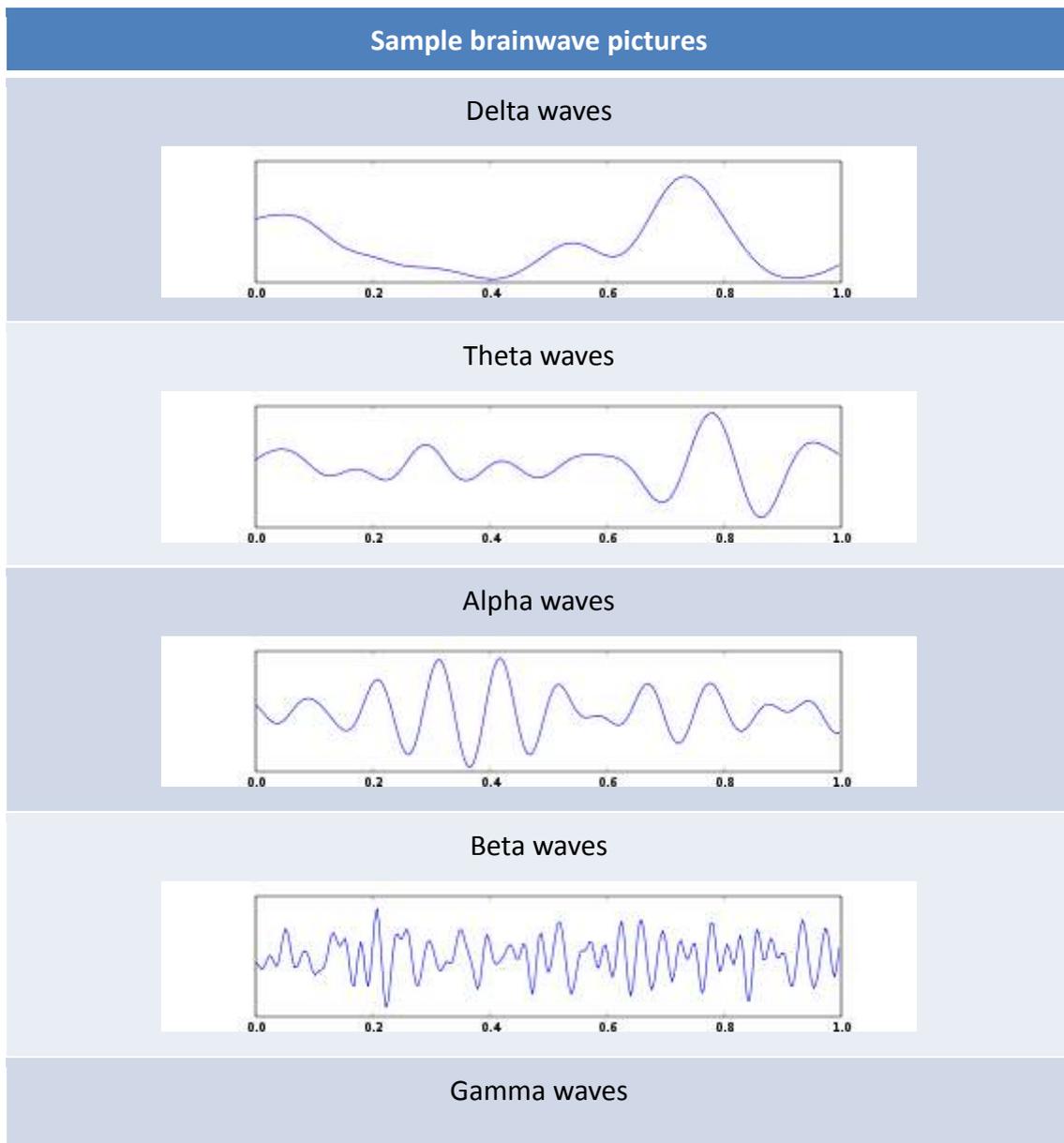
According to the mindset documents, the eSense Meditation meter indicates the level of a user's mental "calmness" or "relaxation". This meter is a measure of mental levels but not physical levels. Simply relaxing all the muscle may not have immediate effect of the eSense meter. Distractions, wandering thoughts, anxiety, agitation, and sensory stimuli may lower the Meditation meter levels.

2.1.6 Raw Brainwaves

Brainwave Type	Frequency range	Mental states and conditions
Delta	0.1Hz to 3Hz	Deep, dreamless sleep, non-REM sleep, unconscious
Theta	4Hz to 7Hz	Intuitive, creative, recall, fantasy, imaginary, dream

Alpha	8Hz to 12Hz	Relaxed, but not drowsy, tranquil, conscious
Low Beta	12Hz to 15Hz	Formerly SMR, relaxed yet focused, integrated
Midrange Beta	16Hz to 20Hz	thinking, aware of self & surroundings
High Beta	21Hz to 30Hz	Alertness, agitation

Figure 2.1-5 Table of different brainwave and its information



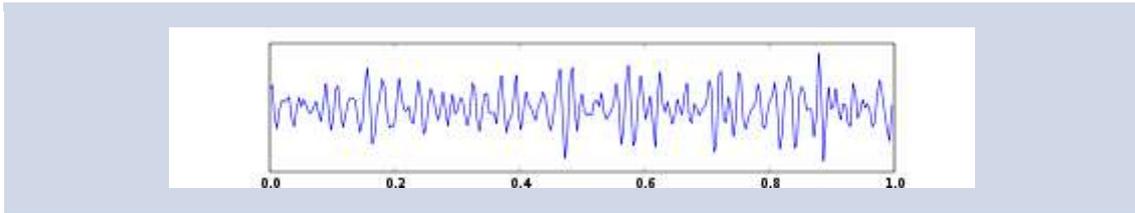


Figure 2.1-6 One second sample of different brainwaves

2.2 Experiment on eSense

An experiment is designed to investigate the relation between eSense meter and mental states. eSense meter includes two mental states: attention and meditation. Mental states include: relaxed, bored, concentrated, fear, excited.

2.2.1 Hypothesis

The eSense meter can reflect the existence or changes of some mental states.

2.2.2 Methodology

We collect the brainwave data of user when watching the movie and use questionnaire to get the feeling of the user watching different part of the movie. Then we will analysis the brainwave data and feeling to see if there is any correlation between mental states and eSense meter.

VMRPlayer with Mindset

The VMRPlayer uses the Video Mixing Renderer 9 (VMR-9) filter to alpha blend one or two running videos and a static image. We integrate the VMRPlayer with the mindset API so when the user is watching the movie, the VMRPlayer will receive packets from the Mindset and records the brainwave data for future analysis.

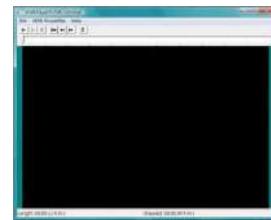


Figure 2.2-1 VMR Player User Interface

Movie

The movie consists of 4 parts. Each parts of the movie have special features which will affect the mental states of the user.

Part 1: String Orchestra, Symphony No. 94 in G Major

Description: The movies contain a string orchestra performance of “Symphony No. 94 in G Major”. The estimated feeling is relaxed.

Special point: There is a strike of loud sound at time 0:36.



Figure 2.2-2 Video clip of Orchestra performance

Part 2: cappella sistina - discorso papa Benedetto XVI (Speech of Pope Benedict XVI in Sistine Chapel)

Description: The movies contain contains a fragment of speech by Pope Benedict XVI. The language spoken by Pope Benedict XVI is Italian and there are no subtitles. The estimated feeling is bored.



Figure 2.2-3 Video clip of Pope speech

Part 3: Don't stare at bikini

Description: The movies are separated into two parts. Each part begins with a picture which changes slightly within 10 seconds. Participant will be given 3 attempts to find out the change. After the 3 attempts, the answers will be told and user will watch the picture again. The estimated feeling is concentrated.



Figure 2.2-4 First Picture showing bikini girl



Figure 2.2-5 Second Picture showing bikini girl

Part 4: Ghost Pop Up



Figure 2.2-6 Video clip before ghost pop up

Description: Participant is told that the ghost will appear anytime and participant needs to be prepared for the ghost. The general feeling of this part is fear. Special point: There is a ghost appear with loud sound at 5:00.

Questionnaire

The questionnaire asks the users their feelings when watching different part of the movie. Participants can select the most suitable word to describe their feelings for each part of the movie.

Figure 2.2-7 Brainwaves sampling feedback form

2.2.3 Result

There are 18 participants in this experiment,

Part 1: String Orchestra, Symphony No. 94 in G Major

	Part 1
Attention	44.79998
Meditation	49.83071

Figure 2.2-8 Part 1 average eSense Value

	Attention	Meditation
Overall	49.29572	53.24579
Part 1	44.79998	49.83071
Difference between overall values and part 1 values	<u>-4.49573</u>	<u>-3.41509</u>

Figure 2.2-9 Compare overall eSense values and part 1 eSense values

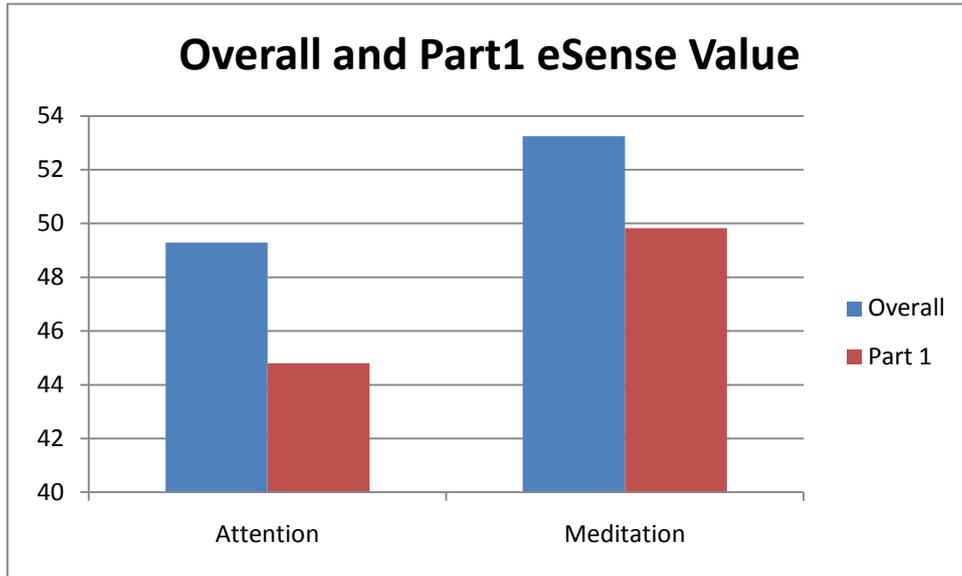


Figure 2.2-10 Overall values and part 1 values

The attention value in part 1 is lower than overall average

Relaxed Participants

The most common feeling of participants in part 1 is relaxed.

	Overall
Attention	47.88266
Meditation	53.2328

Figure 2.2-11 Overall eSense values of relaxed participants

	Part1
Attention	43.67935
Meditation	46.75625

Figure 2.2-12 Part 1 eSense values of relaxed participants

While emotion changed from “relaxed” to “excited” after loud sound

Many people’ feeling changes from relaxed to excite after the strike of

loud sound.

	Attention	Meditation
Before Loud Sound	45.75841	44.45509
After loud sound	40.40862	56.99257
Difference between values before and after loud sound	<u>-5.34979</u>	<u>12.53748</u>

Figure 2.2-13 Compare eSense values before and after the loud sound

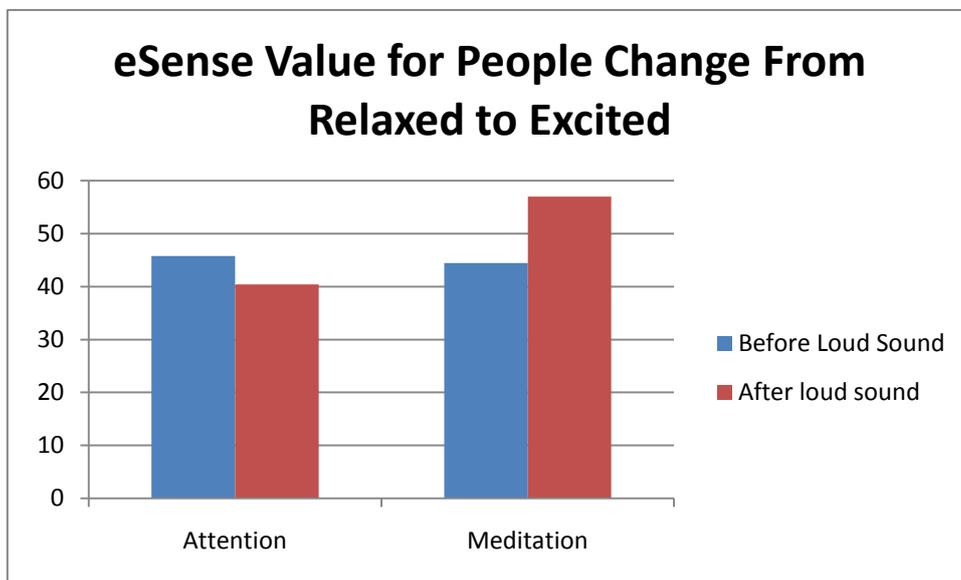


Figure 2.2-14 eSense values before and after the loud sound

The attention value decreases and mediation value increases after the loud sound.

Conclusion

When the participants have a feeling of relaxed, the meditation should be higher than normal level. However, in the observation, the meditation is lower than normal value. The meditation does not reflect the relaxed states in this observation.

When the participants' mental state changes from relaxed to excited

after the loud sound, attention value decreases and meditation value increases. The meditation also does not reflect the change of mental states. One possibility to explain this observation would be: the strike of loud sound does not make the participant excited but help them to relax.

Part 2: Speech of Pope Benedict XVI in Sistine Chapel

	Part 2
Attention	44.19243
Meditation	55.19941

Figure 2.2-15 Part 2 average eSense values

	Attention	Meditation
Overall	49.29572	53.24579
Part 2	44.19243	55.19941
Difference between average values and part 2 values	<u>-5.10329</u>	<u>1.953611</u>

Figure 2.2-16 Compare overall eSense values and part 2 eSense values

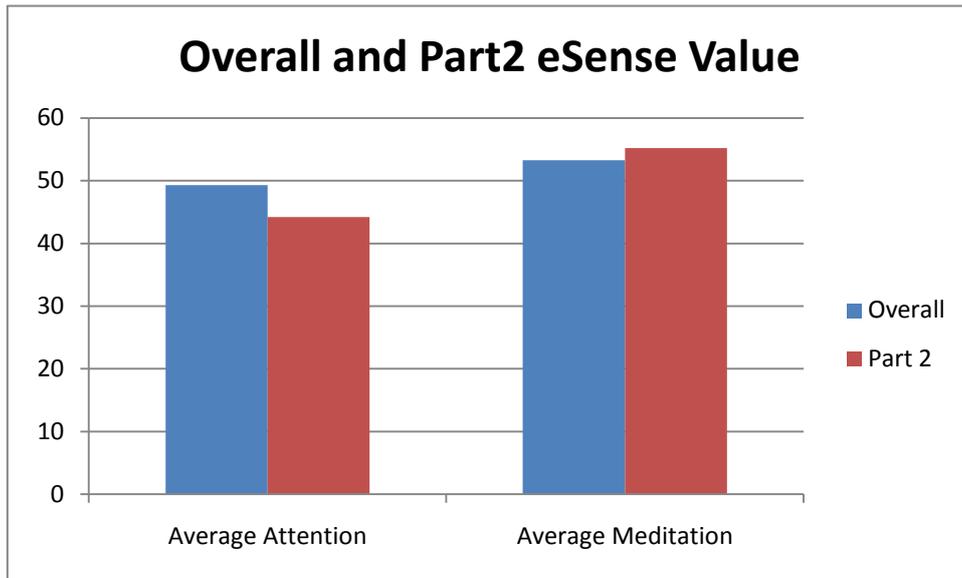


Figure 2.2-17 Overall values and part 2 values

The attention value decreases from first half to second half. The meditation value has no obvious changes.

Bored Participants

Most participants feel bored for both first half and second half of the movie.

Compare eSense values between first half and second half for bored participants

	Attention	Meditation
First Half	43.98032	55.143
Second Half	36.59128	52.59716
Difference between first half and second	<u>-7.38904</u>	<u>-2.54584</u>

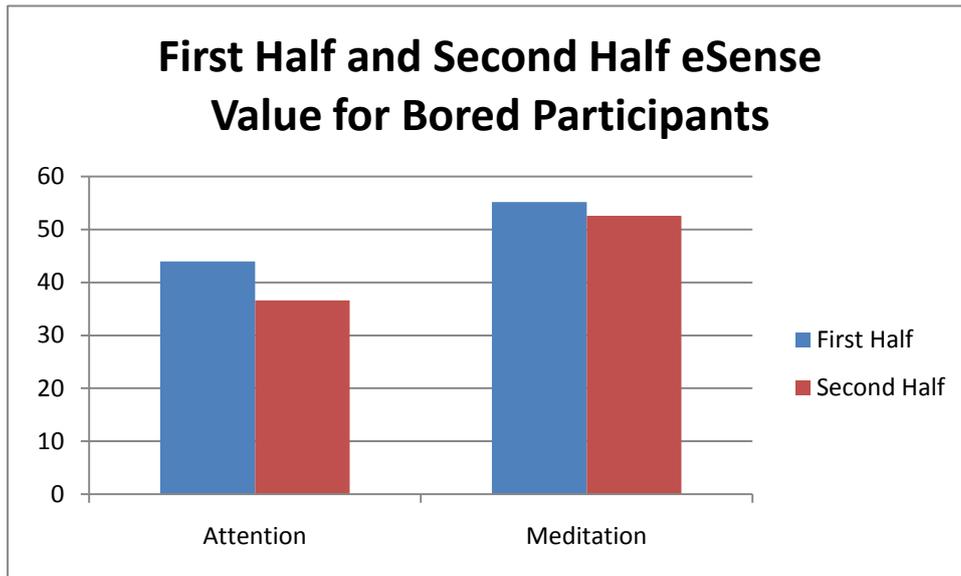


Figure 2.2-18 First half and second half eSense values of bored participants

The attention value decreases from first half to second half. The meditation value has no obvious changes.

Conclusion

When the participants continuously feel bored, there is a decrease in attention. Also, the attention value is lower than normal value for bored participants. There may be some correlations between bored and decrease in attention. One possibility would be the participants lose the focus because they do not understand the Italian that the Pope speaks.

Part 3: Don't stare at bikini

Concentrated Participants

	Attention	Meditation
Overall	44.8396	54.36843
Question 1	40.76725	56.93153
Difference between average values and question 1	<u>-4.07235</u>	<u>2.5631</u>

Figure 2.2-19 Compare part 3 and overall average eSense values of concentrated participants in question 1

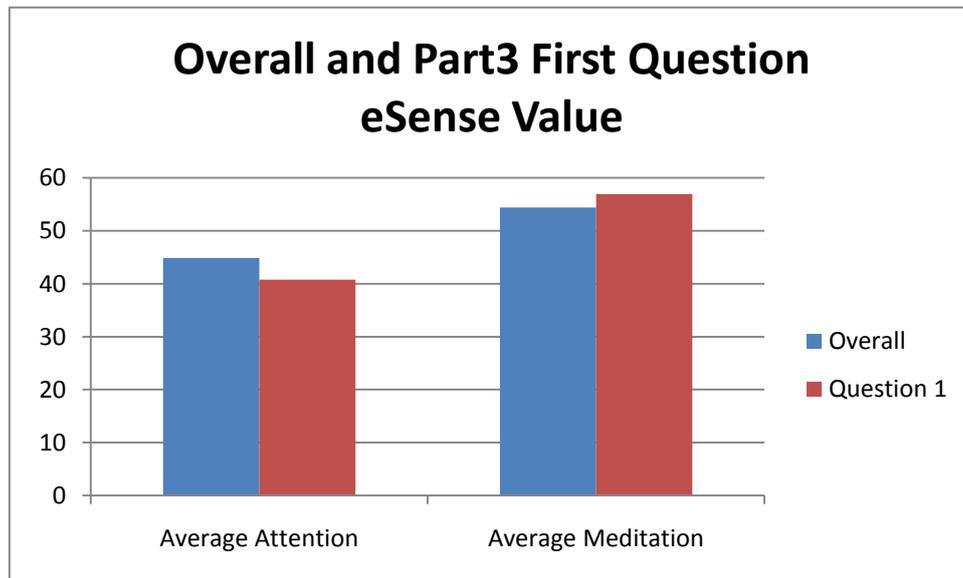


Figure 2.2-20 Overall eSense values and question 1 eSense values

The attention value at question 1 is lower than average value. The mediation value at question 1 is higher than average value.

	Attention	Meditation
Overall	48.59267	57.80879
Question 2	50.97459	56.05645
Difference between average values and question 2	<u>2.38192</u>	<u>-1.75234</u>

Figure 2.2-21 Compare part 3 and overall average eSense values of concentrated participants in question 2

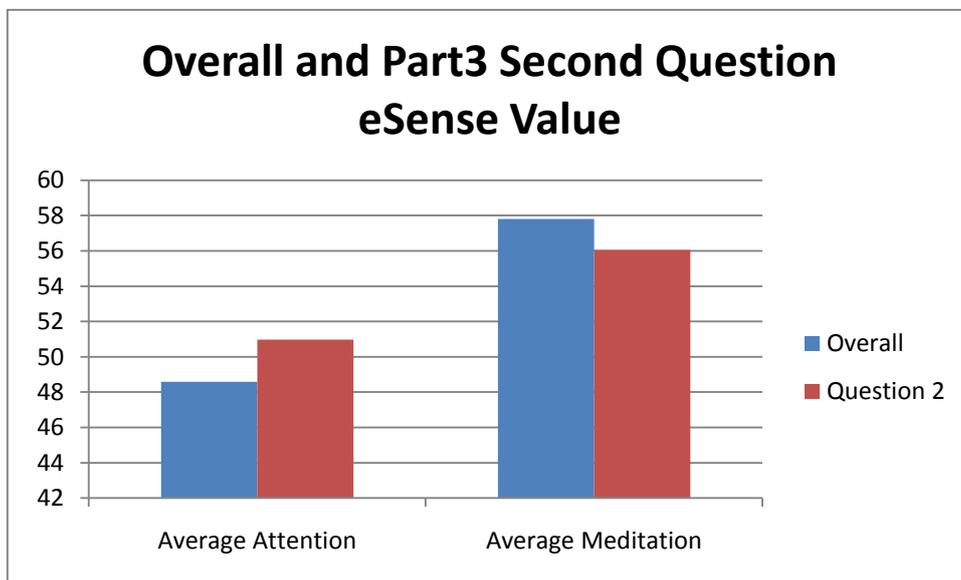


Figure 2.2-22 Overall eSense values and question 2 eSense values

The attention value at question 2 is higher than average value. The mediation value at question 2 is lower than average value.

	Attention	Meditation
Overall	56.67004	55.79628
Answer 1	58.65714	52.41048
Difference between average values and answer 1	<u>1.9871</u>	<u>-3.3858</u>

Figure 2.2-23 Compare part 3 and overall average eSense values of concentrated participants in answer 1

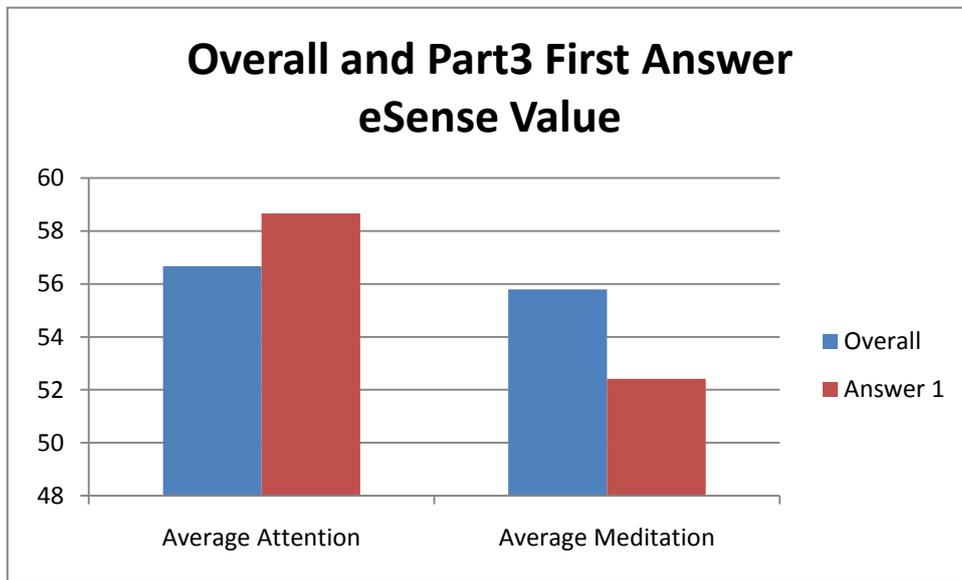


Figure 2.2-24 Overall eSense values and answer 1 eSense values

The attention value at answer 1 is higher than average value. The mediation value at answer 1 is lower than average value.

	Attention	Meditation
Overall	50.41634	53.7065
Answer 2	54.8775	52.63166
Difference between average values and answer 2	<u>4.46116</u>	<u>-1.07484</u>

Figure 2.2-25 Compare part 3 and overall average eSense values of concentrated participants in answer 2

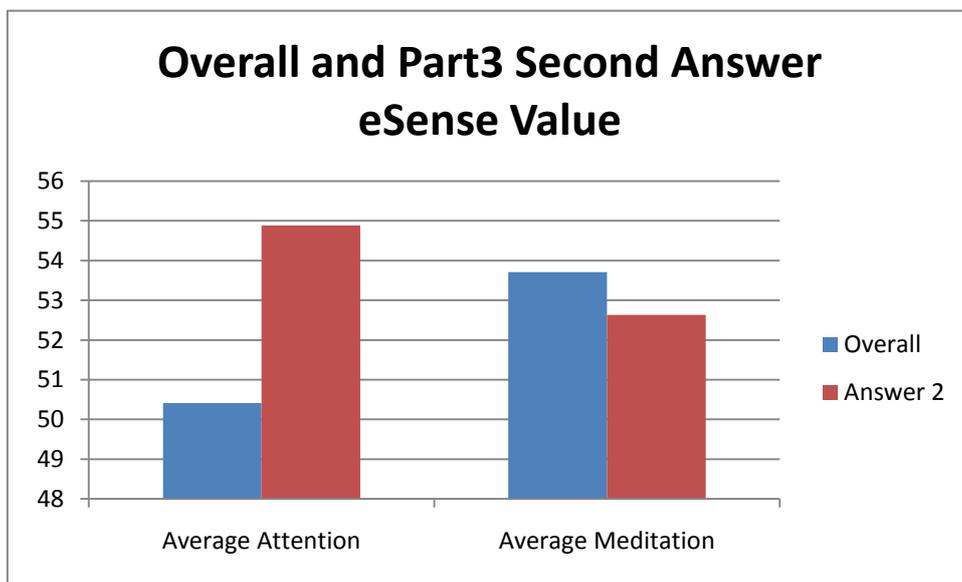


Figure 2.2-26 Overall eSense values and answer 2 eSense values

The attention value at answer 2 is higher than average value. The mediation value at answer 2 is lower than average value.

	Question 1	Question 2	Answer 1	Answer 2
Overall Attention	Q1 < Overall	Q2 > Overall	A1 > Overall	A2 > Overall
Overall Meditation	Q1 > Overall	Q2 < Overall	A1 < Overall	A2 < Overall

Figure 2.2-27 Compare of eSense values for each part with overall value

Conclusion

Some participants are concentrated at question 1, question 2, and answer 1 and answer 2. For attention at question 2, answer 1 and answer 2. The attention is higher than the overall average value which the feeling of concentrated.

The attention value of participant is lower than overall average value but the feeling of participants are concentrated. This error may be caused by participants not fully understand the question when they watch the movie.

For meditation value of each part, it is lower than overall values. The decrease in meditation may cause by bikini girl. Participants become less calm.

Part 4: Ghost Pop Up

	Average
Attention	57.95749
Meditation	51.58219

Figure 2.2-28 Part 4 average attention and mediation

	Attention	Meditation
Overall average	49.29572	53.24579
Part 4 average	57.95749	51.58219
Difference	<u>8.661778</u>	<u>-1.66361</u>

Figure 2.2-29 Compare overall average values and part 4 average values

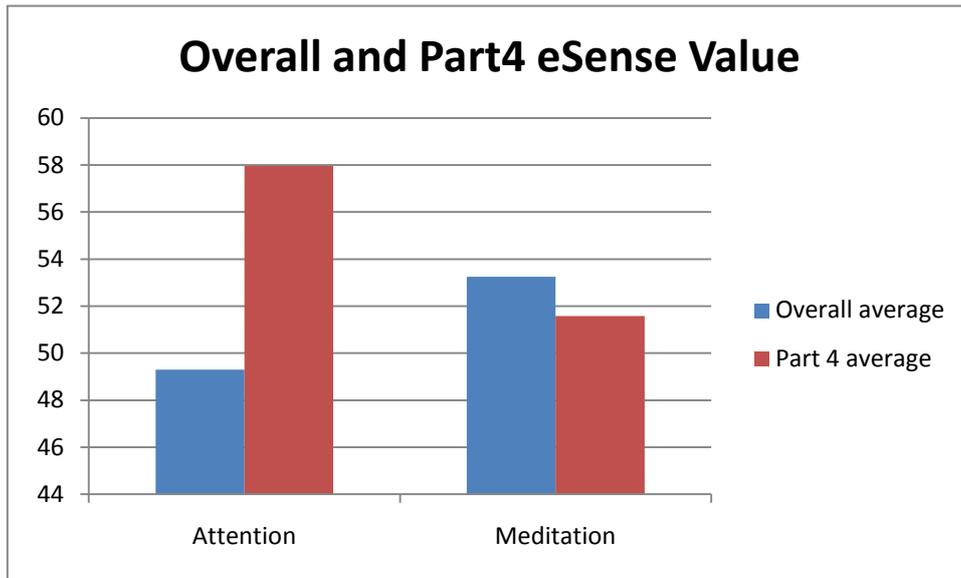


Figure 2.2-30 Compare the overall average values and part 4 average values

The attention value in part 4 is higher than average value. The mediation value at part 4 is slightly lower than average value.

Compare attention and meditation values before and after the ghost appears

The most common feelings of the participants change from concentrated to fear.

	Attention	Meditation
Before Ghost appear	59.18367	52.84439
After Ghost appear	53.51871	50.38266
Difference	<u>-5.66496</u>	<u>-2.46173</u>

Figure 2.2-31 Compare average values before and after the ghost appear for participants' feeling changes from concentrated to fear

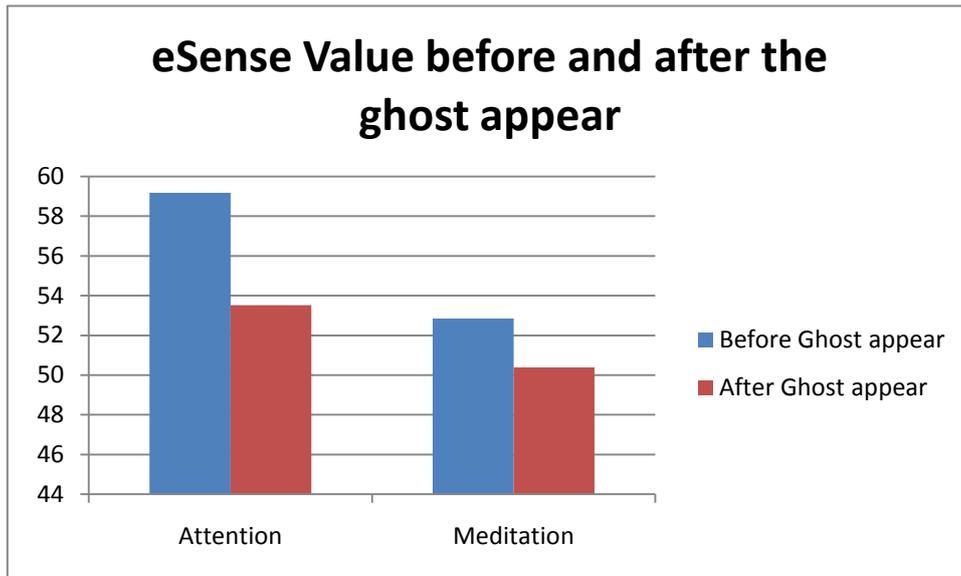


Figure 2.2-32 Compare the average values before and after ghost appear

The attention value and the mediation value decrease after the ghost appears.

Conclusion

In this movie, the participants are told that there will be a ghost appears and to be prepared for the ghost. The attention is higher than normal value. The high attention value supports that the high concentration of the participants since they are aware about the ghost. The decrease in meditation value after the ghost appears support that the participants are less calm because they feel fear.

2.2.4 Conclusion for result and data analysis

The attention and meditation values usually can reflect the changes or existence in mental states.

The attention can correlate to the mental state concentrated. When the participants are concentrated, there is usually an increase in attention value (Part 3 conclusion). When the participants are affected by outside environment (strike of loud sound in part 1 and ghost appear in part 4), there is usually a decrease in attention value (Part 1 and part 4). It reflects the focus of the participants decrease.

Attention can also correlate with mental states boring. There is an obvious decrease in attention when participants are bored (Part 2 conclusion). It can be explained as people become less focus when they are bored.

The relation between meditation value and different mental states are not conclusive. The change in meditation is lower than average value even the participants feel relaxed (Part 1 conclusion). The reason may be the meditation is not measuring the physical relax but the mental relax which is difficult for the participants to identify. However, there are observations which show an decrease in meditation when they are fear (Part 4 conclusion) or change in meditation value after special point (Part 1 loud sound and Part 3 bikini girl).

To conclude, the attention value shows a correlation with concentration and boring while the correlation between meditation value and relaxation are not conclusive and need further investigation.

3 Brainwave Classifier

3.1 Introduction

Brainwave Classifier is an active brainwave control trainer that can classify the brainwave data within a sliding window to different types according to their brainwave pattern. This tool instantly shows the current brainwave states (types) of the user brainwave. The objective to develop this tool is to create an active control brainwave trainer. By using the instant feedback from the Brainwave Classifier, user will train himself/herself maintaining in different brainwaves states (types) or changing from one state to another. This tool shows the possibility of active control using brainwaves.

In this part, the working flow of Brainwave Classifier, brainwave pattern of the three brainwave types, the algorithm used in the Brainwave Classifier and the training result of Brainwave Classifier will be presented.

3.2 Working flow of Brainwave Classifier

The Brainwave Classifier has two stages, the calibration stage and classifying stage. The program first enters the calibration stage and followed by the classifying stage.

3.2.1 Calibration Stage

In the calibration state, user brainwave data will be collected. These data will be used to calculate the value of the peak. In calculate the values of the peak, k-mean clustering algorithm is used. The period of calibration stage is 2 minutes.

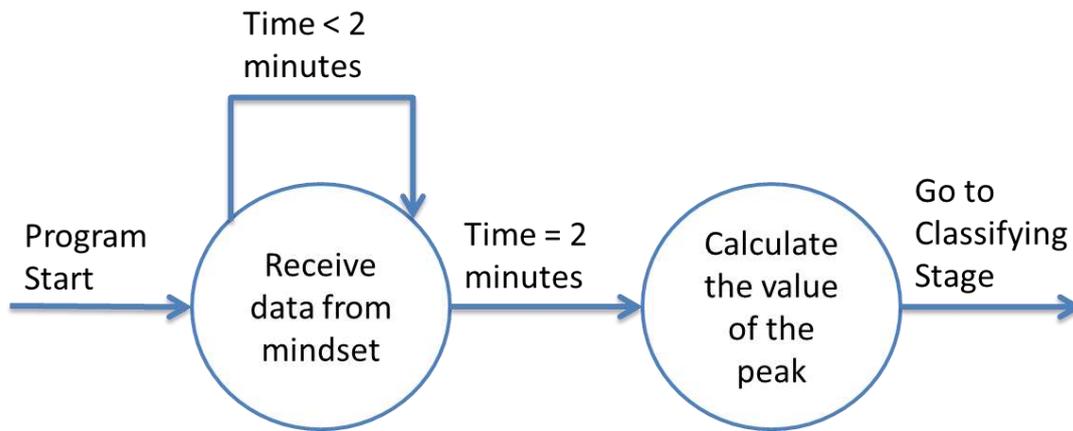


Figure 3.2-1 Program Flow in Calibration Stage

3.2.2 Classifying Stage

In the classifying stage, the program will plot a line chart showing the brainwave data of the latest 15 seconds using a sliding window. The program will calculate the current brainwave type using the latest brainwave data. The resultant brainwave type is showed using color in the chart.

Brainwave Type	Color of line chart
1	Red
2	Yellow
3	Blue

Figure 3.2-2 Color of different brainwave

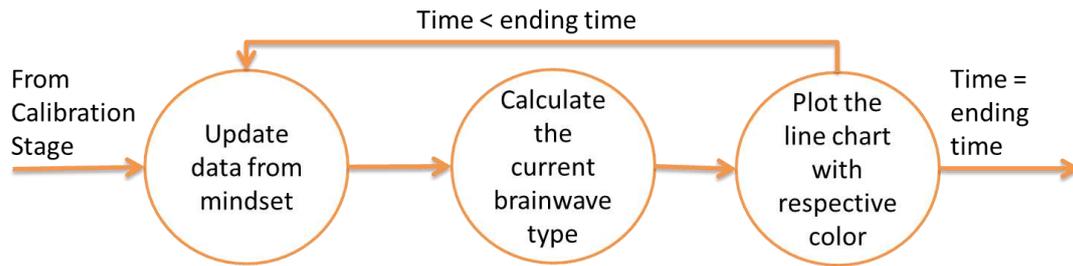


Figure 3.2-3 State Diagram showing the working flow of Brainwave Classifier

In the calibration stage, Brainwave Classifier will first record user brainwave data for 2 minutes.

Then, the data collected will be used to determine the peak of brainwaves using k-mean clustering algorithm.

Then it enters the classifying stage, the Brainwave Classifier will use the latest data (past 15 seconds brainwave data) to calculate the brainwave type. The Brainwave Classifier will update the brainwave type regularly and show the results in the output windows.

3.3 Brainwave Types

Three Brainwave types are defined according to their patterns. Type1 brainwaves mean within the window, there is more than one peak. Type 2 brainwaves mean that within the sliding windows, there is only one peak. Type 3 brainwaves mean that within the sliding windows, there are no peaks.

Brainwaves	Features
Type 1	More than one peaks
Type 2	One peak
Type 3	No peaks

Figure 3.3-1 Features of different brainwave types

3.3.1 Type 1

Brainwaves are classified as type 1 when there is more than one peak within the window. This brainwave pattern means there are great variations in the brainwave data. User showing more type 1 brainwave should have active mental activities.

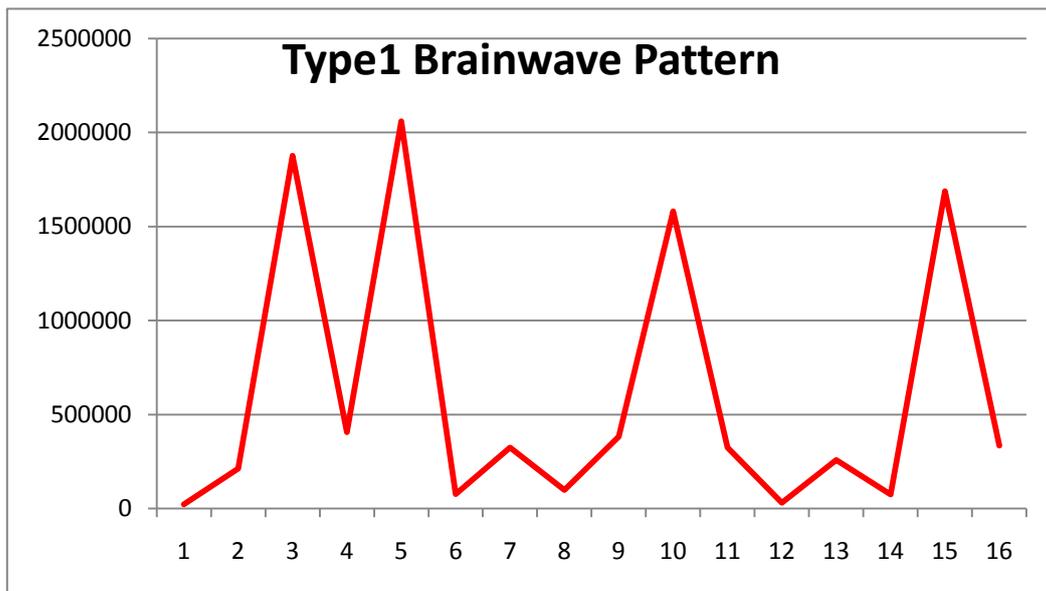


Figure 3.3-2 Sample Type 1 brainwave pattern (window size = 15 data points)

3.3.2 Type 2

Brainwaves are classified as type 2 when there is exactly one peak within the window. This brainwave pattern is the transition state between type 1 and type 3 brainwaves. User showing this brainwave type means the user mental activities is not stable and varies between type 1 and type 3.

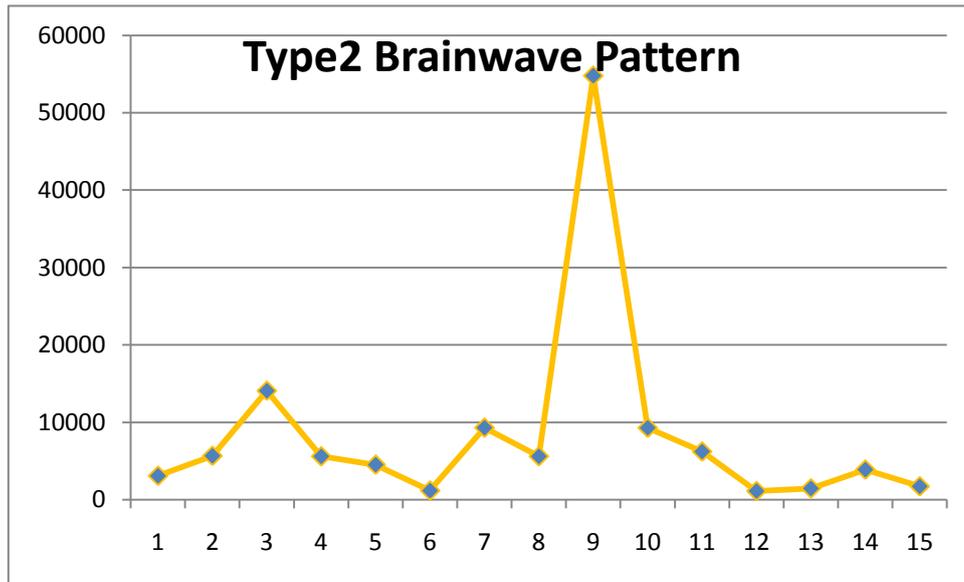


Figure 3.3-3 Sample Type 2 brainwave pattern (window size = 15 data points)

3.3.3 Type 3

Brainwaves are classified as type 3 when there are no peaks within the window. This brainwave pattern means there are small variations in the brainwave data. User showing more type 3 brainwaves should have less active mental activities.

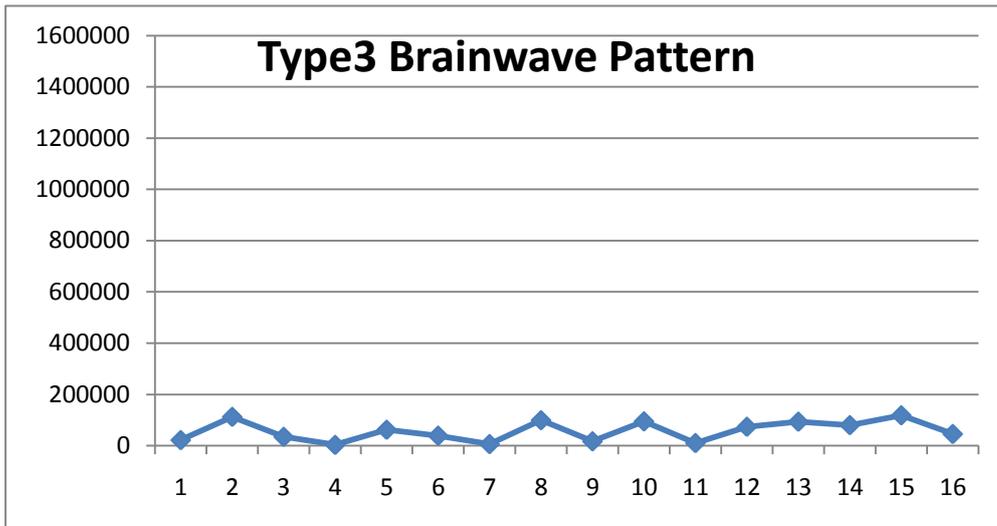


Figure 3.3-4 Sample Type 3 brainwave pattern (window size = 15 data points)

3.4 Algorithm

3.4.1 K-mean clustering

K-mean clustering algorithm is used in the calibration stage. It is used to calculate the value of the peak. When we classify the brainwave, we need to know the value of the peak so we can identify the pattern of the brainwave data.

We use k-mean clustering algorithm to cluster the brainwave data into three clusters. We use the value of the center in the middle cluster as the lowest value of a peak.

The k-mean clustering is defined as:

Given a set of observations $(\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n)$, where each observation is a d -dimensional real vector, k -means clustering aims to partition the n observations into k sets $(k \leq n)$ $\mathbf{S} = \{S_1, S_2, \dots, S_k\}$ so as to minimize the within-cluster sum of squares (WCSS):

$$\arg \min_{\mathbf{S}} \sum_{i=1}^k \sum_{\mathbf{x}_j \in S_i} \|\mathbf{x}_j - \boldsymbol{\mu}_i\|^2$$

where $\boldsymbol{\mu}_i$ is the mean of points in S_i .

Initiation

K-mean algorithm begin with choose the cluster center. In Brainwave Classifier, three data points are randomly chosen as the center of three clusters.

Assignment step

The assignment step will them assign each data point to the closest cluster.

$$S_i^{(t)} = \left\{ \mathbf{x}_j : \|\mathbf{x}_j - \mathbf{m}_i^{(t)}\| \leq \|\mathbf{x}_j - \mathbf{m}_{i^*}^{(t)}\| \text{ for all } i^* = 1, \dots, k \right\}$$

Updating step

The updating step will calculate a new center for each cluster.

$$\mathbf{m}_i^{(t+1)} = \frac{1}{|S_i^{(t)}|} \sum_{\mathbf{x}_j \in S_i^{(t)}} \mathbf{x}_j$$

The assignment step and updating step are repeated until the terminating requirement is reached.

Termination

K-mean clustering terminated when the assignments no longer changed.

Use of k-mean clustering in Brainwave Classifier

After k-mean clustering, the center of the second cluster will be used as a cut off line. The data point above this value will be considered as a peak and the data point below this value will be non-peak.

3.4.2 Sliding Window

Sliding window is used in the classifying stage. Only the latest value of the brainwave is used to calculate the current brainwave type. In the Brainwave Classifier, the window size is 15 seconds.

Time t (seconds)

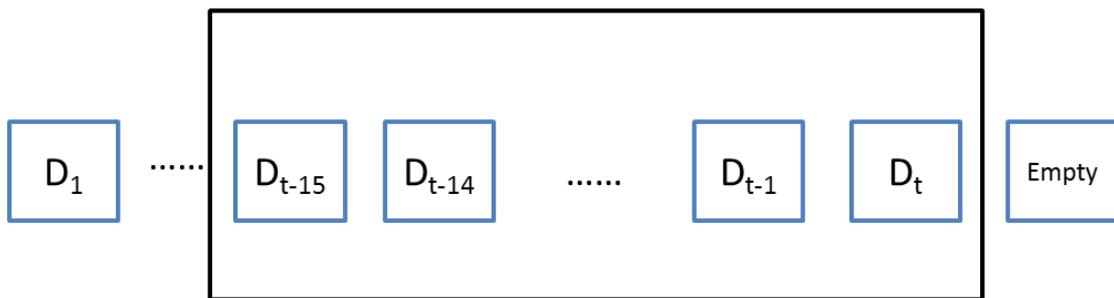


Figure 3.4-1 Sliding Window at Time t

Time $t+1$ (seconds)

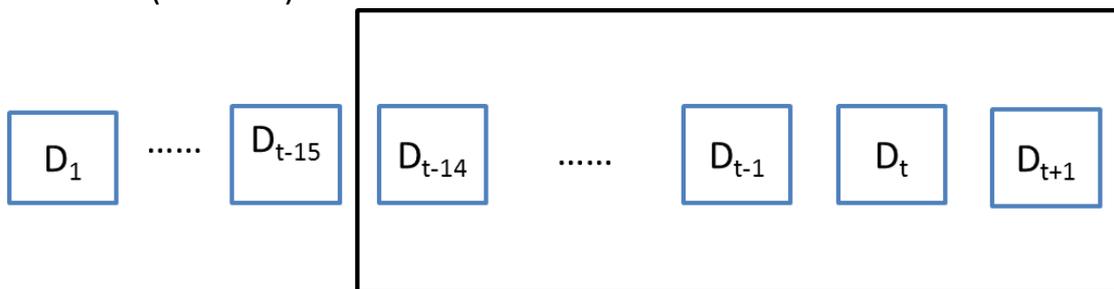


Figure 3.4-2 Sliding Window at Time $t+1$

Every time the Brainwave Classifier collects a new data from Mindset. The sliding window shifts right to use updated data to calculate the new brainwave type.

3.4.3 Calculating the Brainwave Type

Count the Peak

Since the difference between the Type 1, Type 2 and Type 3 brainwave types are the number of peaks in the sliding windows. The method to classify the brainwave type is to count the number of peaks. Using the peak value calculated from the k-mean algorithm, every data point is classified as above peak value or below peak. The number of peaks can be counted by scanning through these data once.

Majority

There are 8 channels of brainwave data. To make the output of the brainwave classifier more user-friendly, an overall brainwave type is returned by majority vote.

In tie-breaking situation, type2 brainwave has a higher priority than type3 brainwaves and a type 3 brainwave has a higher priority than type1 brainwave.

3.5 User Interface

3.5.1 Overall View

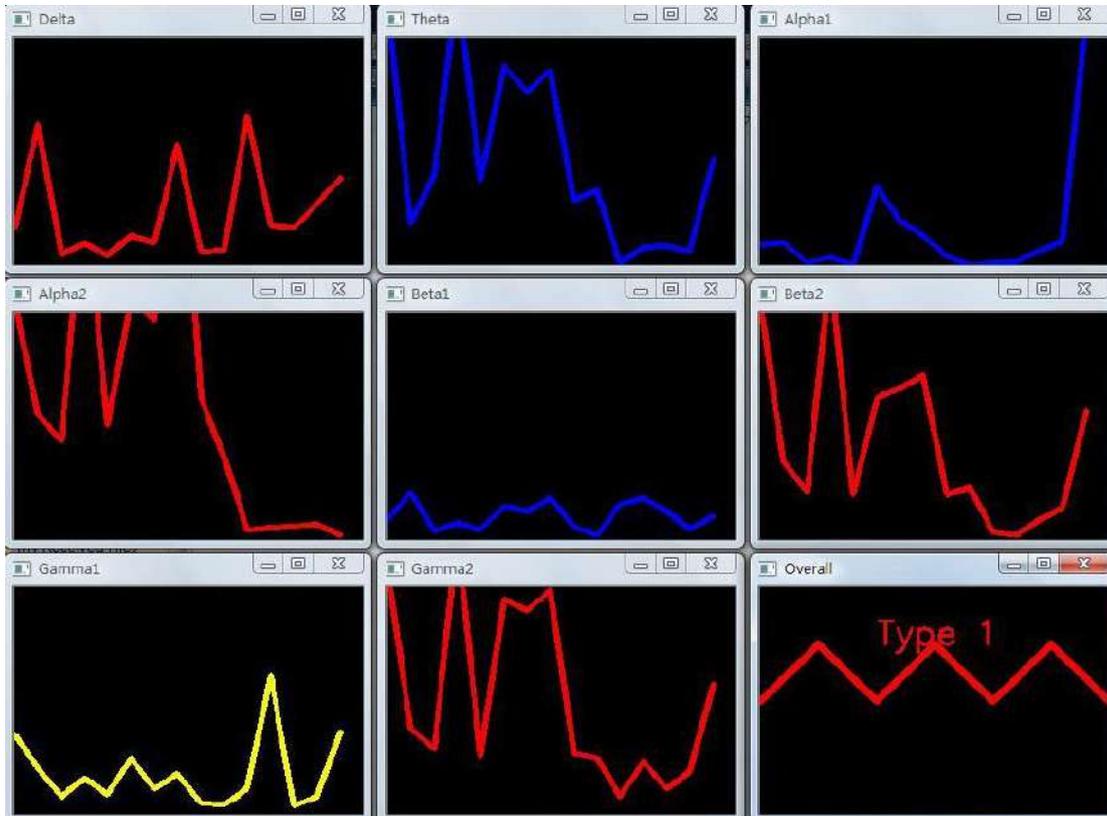


Figure 3.5-1 The whole picture of the Brainwave Classifier

The user interface are created using OpenCV.

The window at the bottom right corner is the overall brainwave window. The overall brainwave window presents the overall brainwave type by the color and simple graph.

The other eight windows are the individual brainwave windows which represent different frequent brainwave. The individual windows present the brainwave type by color and show the exact brainwave pattern by line chart.

3.5.2 Individual Channel and Overall Channel



Figure 3.5-2 Channel with type 1 Brainwave



Figure 3.5-3 Overall Channel with type 1 brainwave

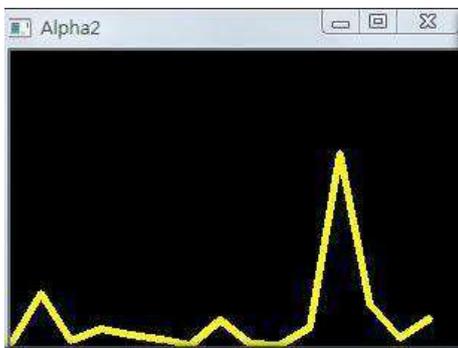


Figure 3.5-4 Channel with Type 2 Brainwave

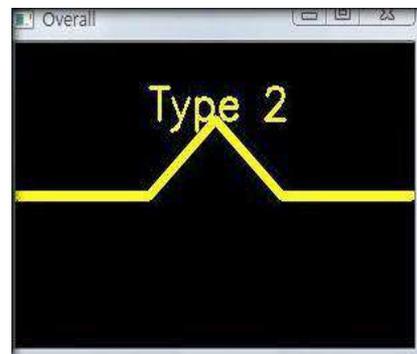


Figure 3.5-5 Overall Channel with type 2 brainwave

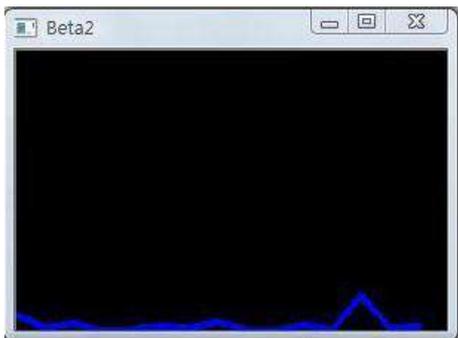


Figure 3.5-6 Channel with type 3 Brainwave



Figure 3.5-7 Overall Channel with type 3 brainwave

3.6 Personal Training

As long term users of Brainwave Classifier, we fully train ourselves and are able to control the brainwave states. The controls include keeping a specific brainwave type or changing between different brainwave types. We found that we can use brainwave pressure or image different music to control different type brainwave.

The following parts are going to demonstrate how well we can control our brainwave using Brainwave Classifier.

3.6.1 Keeping Type 1

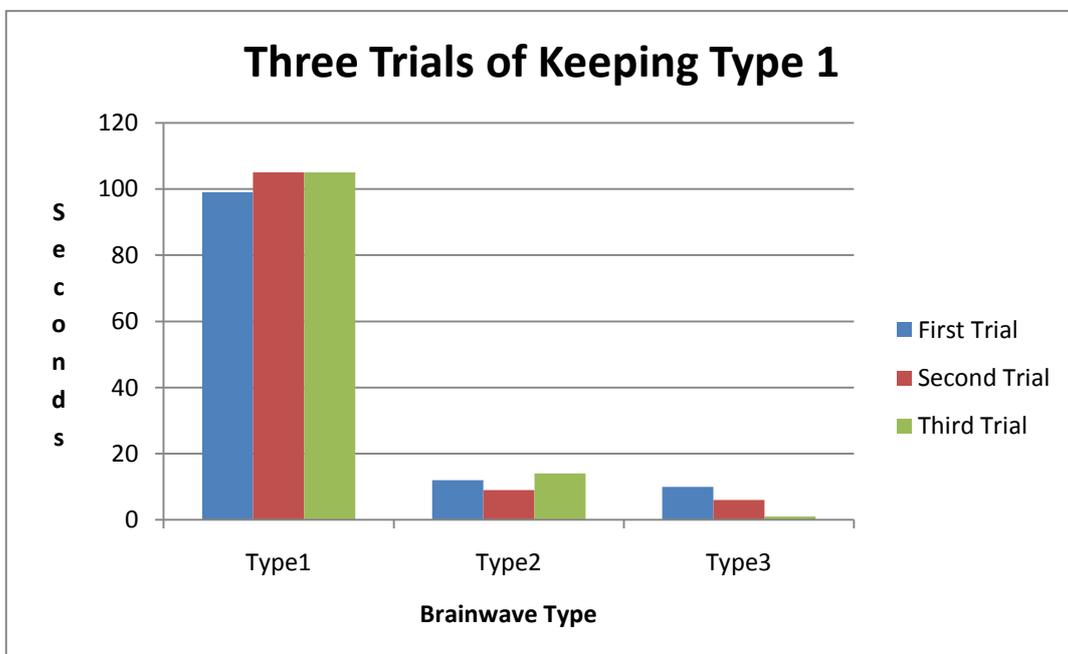


Figure 3.6-1 Bar chart showing keeping type 1 time

We use the brainwave for three times and each trial longs for 2 minutes. In each trial, we are going to keep our brainwave at type 1. The results show that we are able to maintain our brainwave states at type 1 most of the time. We can hold the type 3 brainwave pattern for about 100 seconds out of 120 seconds.

3.6.2 Keeping Type 3

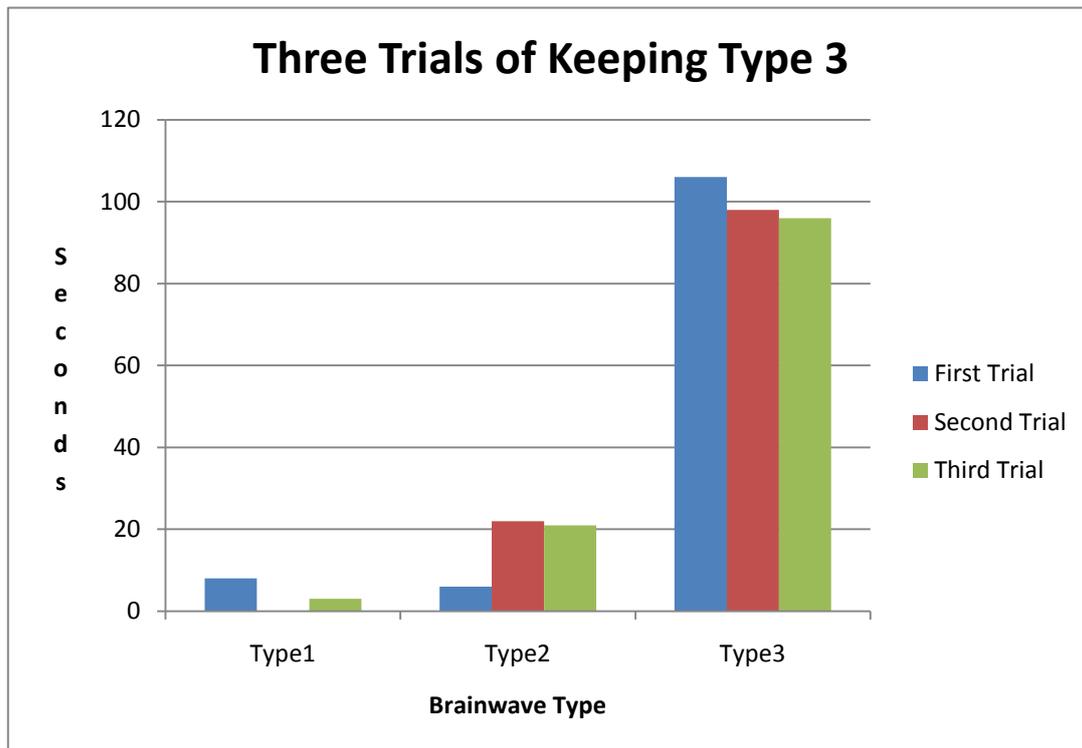


Figure 3.6-2 Bar chart showing keeping type 3 time

We use the brainwave for three times and each trial lasts for 2 minutes. In each trial, we are going to keep our brainwave at type 3. The results show that we are able to maintain our brainwave states at type 3 most of the time. We can hold the type 3 brainwave pattern for about 100 seconds out of 120 seconds.

3.6.3 Oscillating between type 1 and type 3

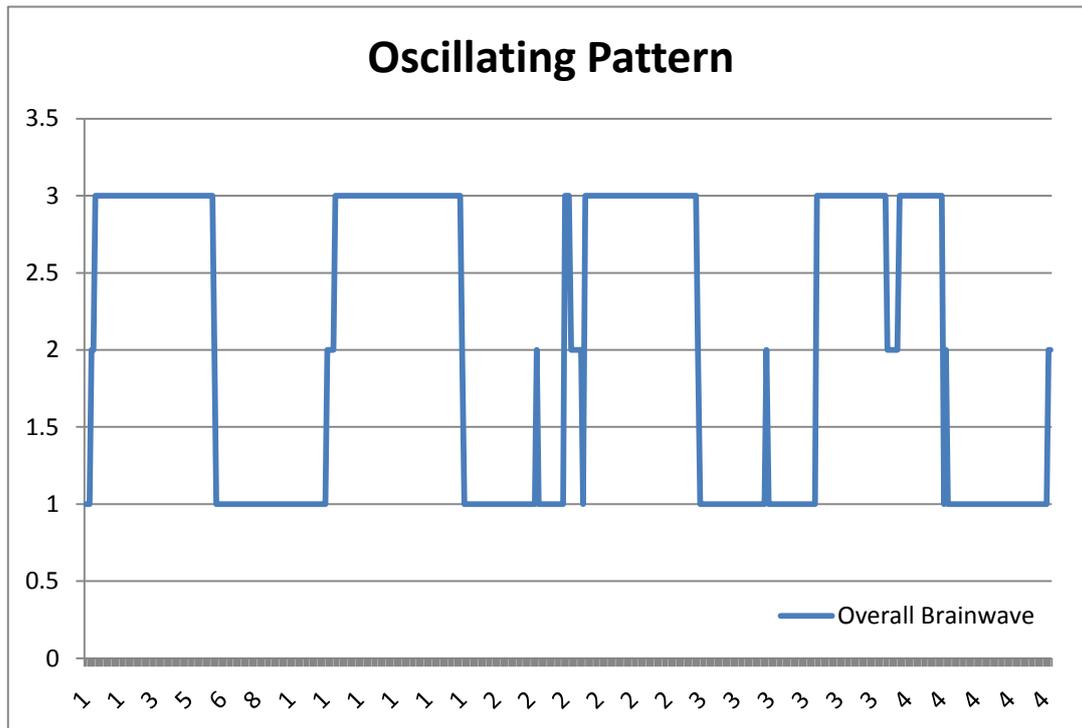


Figure 3.6-3 Line chart showing the oscillating pattern produced

This personal testing lasts for 10 minutes. We try to maintain the brainwave type3 and then change to type1 after 1 minute. We change our brainwave type to one another once per minute.

The pattern on the line chart shows that for most of the time, the brainwave types 1 and type 3 are maintainable. We can maintain our brainwave type for at least 1 minute and we can change the brainwave types from one another within 5 to 10 seconds.

This shows that we can completely control type1 and type3 brainwaves.

3.7 Experiment on Brainwave Classifier

We would like to investigate whether non-experienced user are able to use Brainwave Classifier to train themselves.

3.7.1 Objective

The objective of this experiment is to show that Brainwave Classifier help some people training their brainwave.

3.7.2 Methodology

Users will have 15 minutes to use the Brainwave Classifier. They will use the Brainwave Classifier to train their brainwaves, trying to control the brainwave types.

After the training, user will be asked to maintain type 1 and type 3 brainwaves for 2 minutes as much as possible. By compare the time that they can maintain at type 1 and type 3, it may show that users can control their brainwave in some degree.

3.7.3 Result

Nine participants have do the experiment and all of them are male.

Overall Result

Keeping Type 1

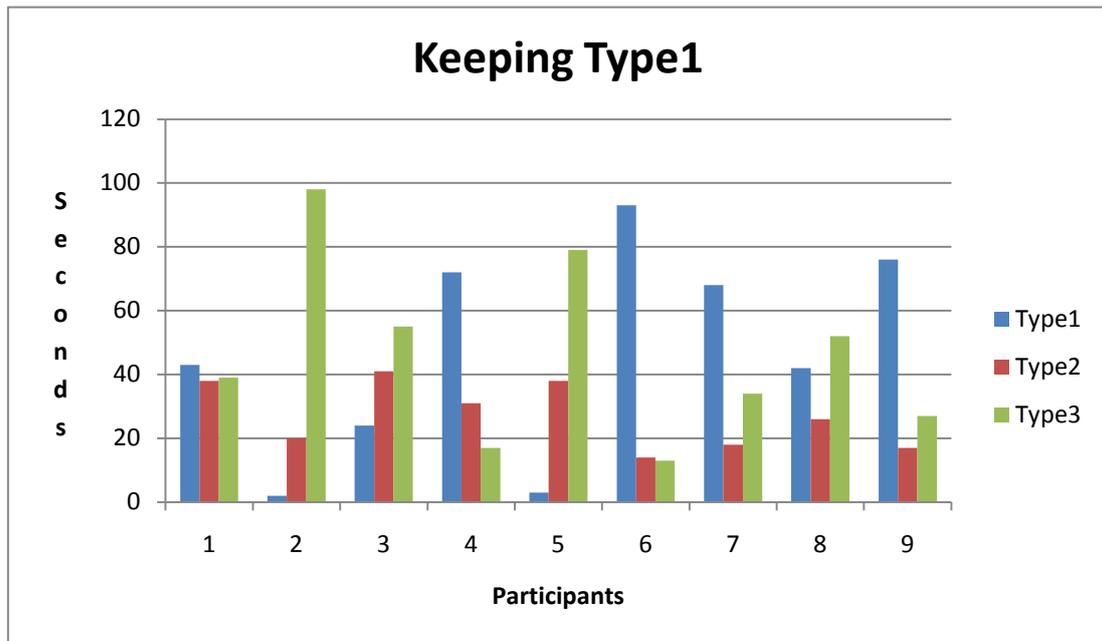


Figure 3.7-1 Bar chart showing result in Keeping Type 1 Brainwave

For keeping type1 experiment, the participants are asked to keep the type 1 brainwave (pattern with more than one peak) as long as they can within two minutes.

The result shows that 5 participants have maintain longer type 1 brainwave than type2 and type 3 while the remaining 4 participants have more type3 brainwave than type1 brainwave.

Keeping Type 3

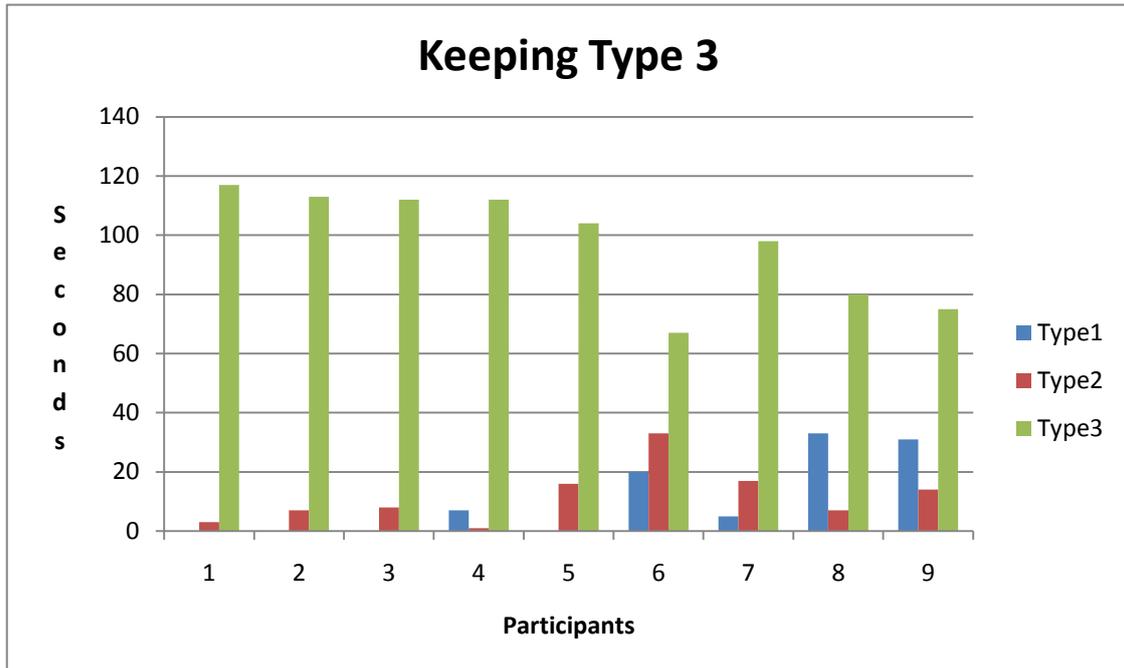


Figure 3.7-2 Bar chart showing result in Keeping Type 3 Brainwave

For keeping type3 experiment, the participants are asked to keep the type 3 brainwave (pattern with more than one peak) as long as they can within two minutes.

The result shows that all participants can maintain longer type3 brainwave than type1 and type2. Some participants can even keeping type3 brainwave for more than 100 seconds out of 2 minutes.

This results show that type3 brainwave is more easier to control than type1, since all participants are able to control type3 brainwave while only half of the participants are able to control type1 brainwave.

Successful Users of Keeping Type 1

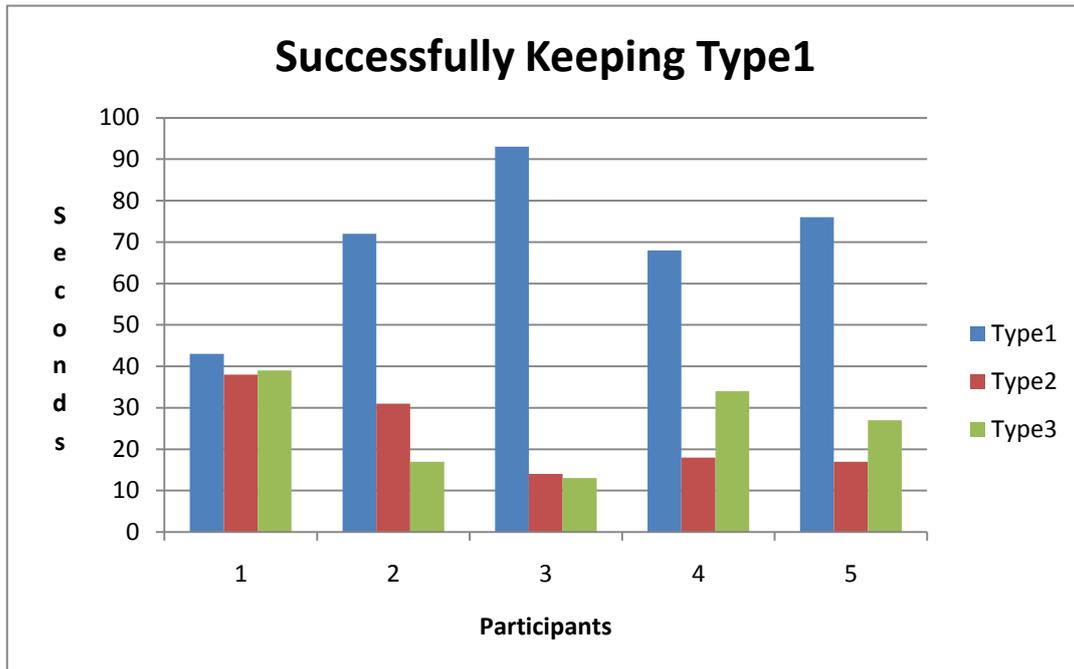


Figure 3.7-3 Bar chart showing data of those who can control Type 1

If we only consider the successful user in keeping type1, we find that the performance in keeping type 1 is not as good as keeping type 3.

This results show that keeping type1 is more difficult than keeping type3. The time that keeping type3 is about 100 seconds but the time that keeping type1 is only about 70 seconds. Moreover, only half of the participants can successfully control type1 brainwave while all participants can control type3 brainwave.

To conclude, keeping type1 brainwave seems more difficult and needs more training when comparing to type3 brainwave.

Compare Keeping 1 and Keeping 3

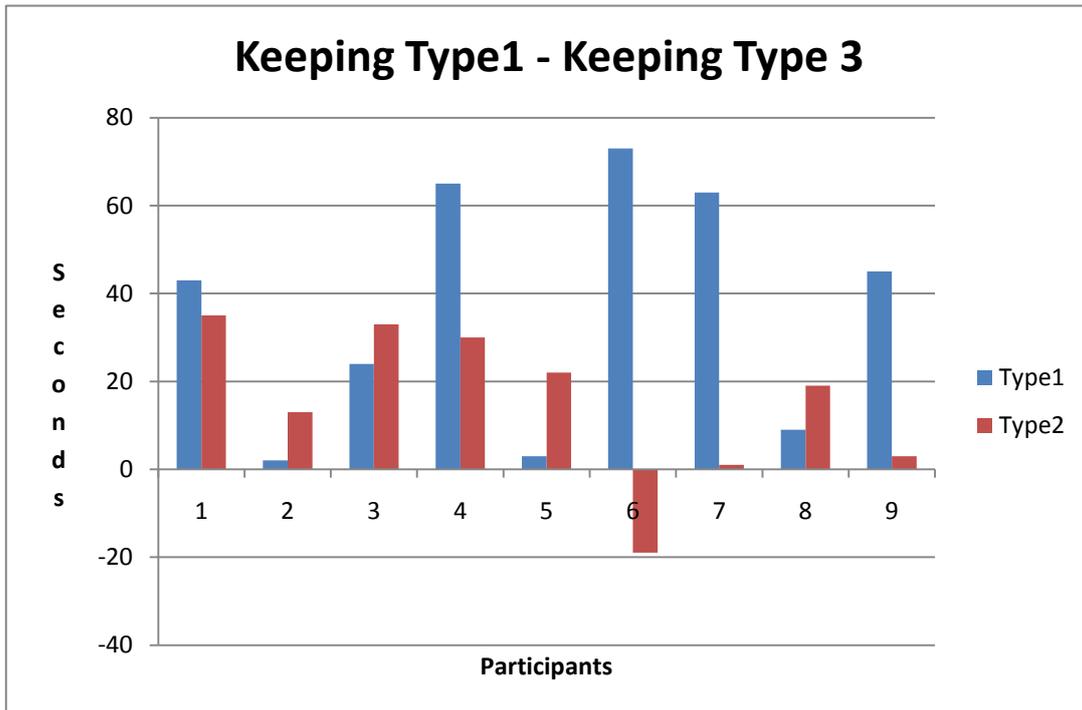


Figure 3.7-4 Bar chart showing difference between Keeping Type 1 and Keeping Type 3

We compare the brainwave data in experiment of keeping type1 and experiment of keeping type3. The chart above is produced by time in keeping type1 minus time in keeping type3. We find that their generally more type1 and type2 in keeping type1 experiment than keeping type3 experiment.

Although not all participants can maintain longer time in type1 than type3, they are able to produce more type1 and type2 brainwave when they try to produce type1 brainwave. It means the participants can control the brainwave in some degree using Brainwave Classifier.

To conclude, Brainwave Classifier is quite successful. Most participants can control their brainwave in some degree.

3.8 Control Method

3.8.1 Keeping Type 1

Keeping type1 brainwave is relatively difficult when comparing to keeping type3 brainwave. Different participants have different ways to maintain their type1 brainwave. The method includes, image some music, small muscle movement, concentrate on surroundings people or scene, and increase the brain pressure.

Keeping type1 brainwave is more difficult than keeping type3. Only half of the participants can show significant control on type1. For those successful participants, 5 to 10 minutes are needed to achieve some degree of control.

3.8.2 Keeping Type 3

Keeping type3 brainwave is relatively easy when comparing to keeping type1 brainwave. Most people are able to maintain type3 brainwave when they keep themselves at resting state or keeping very calm, not making large muscle movement.

Keeping type3 brainwave is quite easy to control. Most participants are able to achieve maintain type3 brainwave within several minutes.



3.9 Summary

In this semester, we develop Brainwave Classifier, which act as a brainwave trainer. With some training, we are able to control our brainwave to use the Brainwave Classifier with high accurate. We are able to maintain about 100 seconds out of 120 seconds for both type1 and type3. We are also able to change quickly from type1 and type3 and vice versa.

We also examine this tool with normal users; we find that they can also control their brainwave in some degrees. In the experiment, we found that type1 brainwave is more difficult to control than type3 brainwave. On the other hand, keeping type1 brainwave does not have any general method while keeping type3 can generally be achieved by keeping calm or at rest.

To conclude, Brainwave Classifier can help the user to train their brainwave. In the experiment, participants only use the Mindset for about 15 minutes but they already show some degree of control on brainwave. As experienced users of Brainwave Classifier, we even show an excellent control of brainwave with high accuracy.

4 Training Method

4.1 Introduction

From the development of Brainwave Classifier, we conclude a training method for users to learn and identify their ways to control the brainwaves. This training method provides a working flow for user to follow and discover their brainwave pattern.

4.2 Training Flow

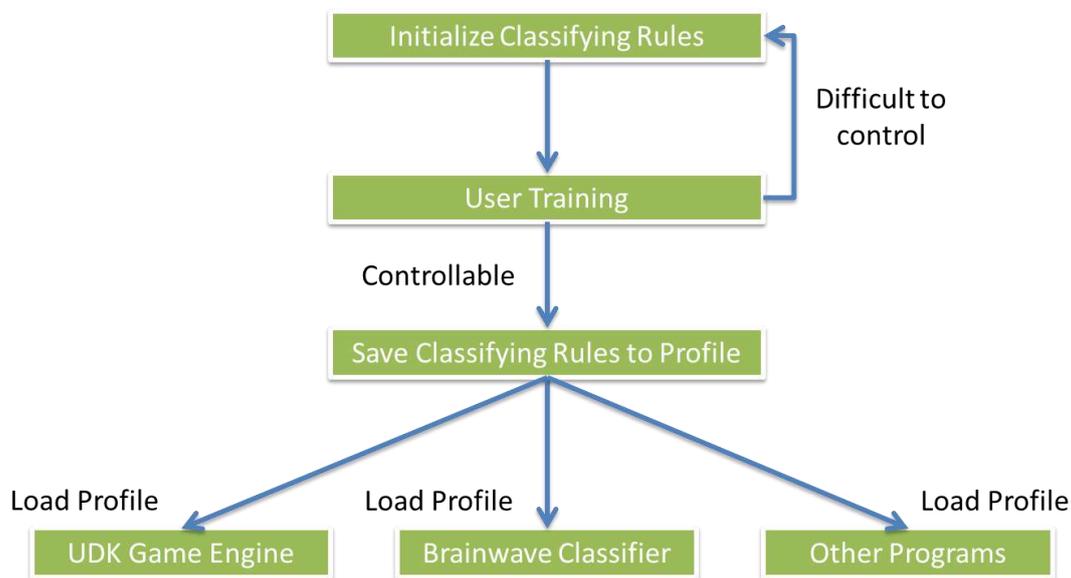


Figure 4.2-1 Training flow using Brainwave Classifier

The users can discover a controllable brainwave pattern by following the working flow below:

First, the user first initializes the rules to classifying brainwaves, for example, number of brainwave types and brainwave pattern of each brainwave type. Then the users use the brainwave classifier and train themselves to control these brainwave types.

If users feel difficult to control the defined brainwave types, they can modify the brainwave classifying rules of each type. If users feel the current brainwave patterns are controllable, a personal profile that contains the

brainwave classifying rules and relevant data is created.

Using this profile, different programs can read the personal profile and classify the user brainwave using the brainwave classifying rules that are most suitable to the user.

4.3 Personal Profile

Personal profile is universal to all Mindset program. Mindset program will read the personal profile and use the brainwave classifying rules in the profile. The main advantage of using a personal profile is user does not need to learn new method to control their brainwave for different programs.

Possible Profile Format:

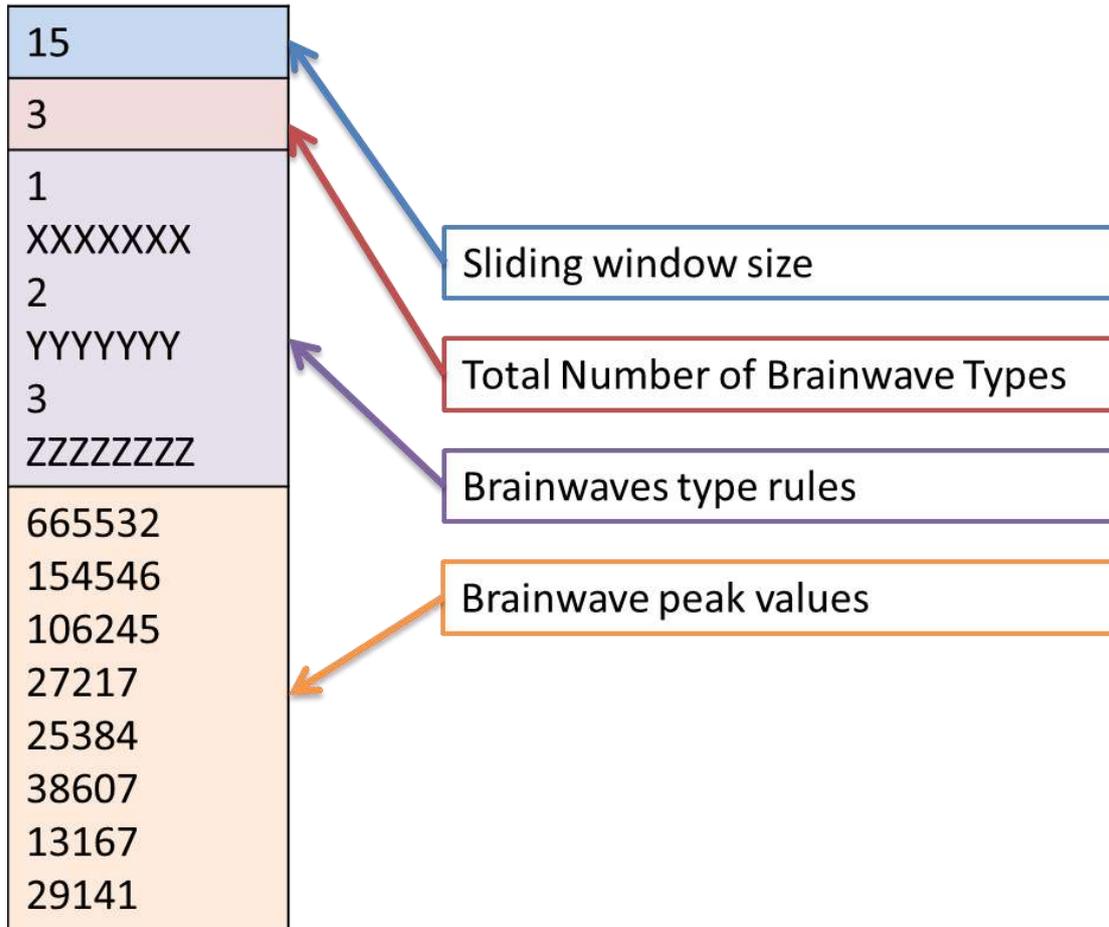


Figure 4.3-1 Possible Profile Format

4.4 Summary

This training method is a new idea in controlling brainwave. It suggests an idea of personal profile to help user easier to control their brainwave.

5 Conclusion

This project has introduced a brainwave training tool, Brainwave Classifier, an experiment based on this tool and its results and a new training method discovered from the experience of using Brainwave Classifier.

Brainwave Classifier is a active control trainer that helps the user to train his/her brainwave. The Brainwave Classifier classifies the brainwave pattern within the sliding windows into three types. This tool provides an instant

feedback on brainwave which helps the user to learn how to change or maintain different brainwave states.

The k-mean clustering algorithm and sliding window are some techniques used in the Brainwave Classifier. OpenCV is used to create a user-friendly interface.

As we are experienced user of Brainwave Classifier, we are able control different brainwave type with high accuracy. The personal testing shows we are able to control type1 and type3 and we are able to smoothly change from one state to another.

An experiment on Brainwave Classifier is carried out. Most participants show good control on type3 brainwave and some participants show some degree of control on type1 brainwave. The result of the experiment shows that the Brainwave Classifier is quite reliable as the user only has limited time to use it.

The new training method is a working flow that user can follow and train their brainwave. It also brings an idea of personal profile.

The development of Brainwave Classifier is quite successful. The experiment shows people can control their brainwave in some degree. It also helps to conclude a new brainwave training method.

6 Limitations & Difficulties

6.1 Poor Signal

The Poor Signal from Mindset is how poor the signal measured by Mindset. The value ranges from 0 to 200. A non-zero value means the existence of noise contamination. Poor signals may be caused by a number of different things, such as, poor contact of sensor, excessive motion of user, excessive environmental electrostatic noise.

In this semester, we still cannot figure out a complete way to get rid of poor signal. However, we find that the poor signal will come out less frequently if the Mindset are tightly attached on the user' head and the user does not have very large body movement.

Since our experiment in this semester do not rely on any special points, we analysis a longer period of time to see if the user can control the brainwave. The effect of poor signal on the experiment is not significant.

6.2 Curve Matching

In this semester, we have tried to use some curve matching method, such as compare area under curve, using Frechet distance, to find repeated pattern or find similar curve. However, the result in using those methods to analysis the brainwave pattern is not successful. There are not significant findings when we use these methods. Therefore, results of using curve matching technique in analysis brainwave are not presented in this report.

6.3 Passive Control

In the last semester, a passive control is one of our targets to be achieved in this semester. However, we face several problems in developing a passive control. First, a large amount of data is needed and analysis on these data is difficult. We lack a proper algorithm in BCI to produce new algorithms for passive control.

Therefore, before develop the passive control; we decide to test different algorithms by develop a simple active control, which is Brainwave Classifier. After developing the Brainwave Classifier, we find that it is very successful and may help develop an active control trainer. Therefore, we shift our BCI development from passive control to an active control trainer.

7 Reference

- Arthur, C. (2009, June 9). *Are downloads really killing the music industry? Or is it something else?* Retrieved 11 23, 2010, from Guardian:
<http://www.guardian.co.uk/news/datablog/2009/jun/09/games-dvd-music-downloads-piracy>
- Bishop M.C. (1995). *Neural Networks for Pattern Recognition*. Oxford, England: Oxford University Press.
- DevMaster.net. (2010, 11 29). *List All Engines*. Retrieved 11 29, 2010, from DevMaster.net: <http://www.devmaster.net/engines/list.php>
- Emotiv-Administrator. (2010, 07 30). *EPOC (not good) Experiences*. Retrieved 11 28, 2010, from Emotiv Main Forum:
<http://emotiv.com/forum/messages/forum4/topic732/message4310/#message4310>
- Fear, E. (2009, 6 26). *The Top 10 Game Engines*. Retrieved 11 29, 2010, from Develop-Online.net:
<http://www.develop-online.net/features/519/The-Top-10-Game-Engines-No-1-Unreal-Engine-3>
- Fruhlinger, J. (2008, 10 9). *Brains-on with NeuroSky and Square Enix's Judecca mind-control game*. Retrieved 11 28, 2010, from engadget:
<http://www.engadget.com/2008/10/09/brains-on-with-neurosky-and-square-enix-judecca-mind-control-ga/>
- Gregory, J., & Lander, J. (2009). Third-party SDKs and Middleware. In J. Gregory, & J. Lander, *Game Engine Architecture* (p. 31). A K Peters, Ltd.
- Kenner, C. (2009, 4 1). *Emotivated vs Neurosky*. Retrieved 11 28, 2010, from Facebook:
<http://www.facebook.com/topic.php?uid=9489703974&topic=8230>
- Ko, M., Bae, K., Oh, G., & Ryu, a. T. (2009). A Study on New Gameplay Based on Brain-Computer Interface. *Digital Games Research Association (DiGRA)*.
- Lalor, E., Kelly, S., Finucane, C., Burke, R., Reilly, R., & McDarby, G. (2004). Brain Computer Interface Based on the Steady-State VEP for Immersive Gaming Control. *Biomed Tech*.
- Moon, T. K., & Stirling, W. C. (1999). *Mathematical Methods and Algorithms for Signal Processing*. Prentice Hall.
- NeuroSky. (2009, December). *Brainwave EEG Signal*. Retrieved from NeuroSky

- Website: <http://dev.www.neurosky.com/Academics/AcademicPapers.aspx>
- NeuroSky. (2009, September). *NeuroSky's eSense Meters and Detection of Mental State*. Retrieved from NeuroSky Website:
<http://dev.www.neurosky.com/Academics/AcademicPapers.aspx>
- NVIDIA. (2010, 1 17). *NVIDIA PhysX SDK Features*. Retrieved 11 30, 2010, from NVIDIA Developer Zone:
http://developer.nvidia.com/object/physx_features.html
- OGRE. (2010). *Testimonials*. Retrieved 11 29, 2010, from OGRE 3D:
<http://www.ogre3d.org/about/testimonials>
- Oriental-Daily. (2010, 11 05). *中大研腦電波輸入漢字*. Retrieved 11 27, 2010, from Oriental Daily:
http://orientaldaily.on.cc/cnt/news/20101105/00176_045.html
- Pelletier, S. (2002, Fall). *Computing the Fréchet distance between two polygonal curves*. Retrieved from
<http://www.cim.mcgill.ca/~stephane/cs507/Project.html>
- PixelMineGames. (2010, 11 11). *nFringe:Features*. Retrieved 11 30, 2010, from PixelMineGames:
<http://wiki.pixelminegames.com/index.php?title=Tools:nFringe:Features>
- Porter, J. (2010). *UDN*. Retrieved 11 30, 2010, from Calling DLLs from UnrealScript (DLLBind): <http://udn.epicgames.com/Three/DLLBind.html>
- Reynders, D., & Wright, E. (2003). Practical TCP/IP and Ethernet networking. In D. Reynders, & E. Wright, *UDP (User Datagram Protocol)* (p. 131). The Great Britain: IDC Technologies.
- Scaleform.com. (2010). *Scaleform Gfx Core Technology*. Retrieved 11 30, 2010, from Scaleform Corporation: <http://www.scaleform.com/products/gfxtech>
- Stevens, W. R., & Wright, G. R. (1994). *TCP/IP illustrated: The protocols*. Canada: Addison Wesley.
- Tatum, W. O. (2008). *Handbook of EEG Interpretation*. Demos Medical Publishing.
- Walsh, P. (2008). Advanced 3D game programming with DirectX 10.0. In P. Walsh, *MIP Maps* (p. 432). United States of America: Wordware Publishing, Inc.
- Ward, J. (2008, 04 29). *What is a Game Engine?* Retrieved 11 29, 2010, from Game Career Guide:
http://www.gamecareerguide.com/features/529/what_is_a_game_.php?page=2
- Yoh, M.-S., Kwon, J., & Kim, S. (2010). NeuroWander: a BCI game in the form of interactive fairy tale. In *Proceedings of the 12th ACM international conference*

adjunct papers on Ubiquitous computing (pp. 389--390). ACM.

陳國志, 楊文鎮, 張蕪英, 林灶生, 張韶芹, & 劉崇志. (2009). 植基於無線腦波儀之心境模式判讀與應用.

Unleashing Brain Powers

A Study on Development of BCI-enhanced Computer Games

Supervised by:
Prof. Michael R. Lyu
Written by:
Liu Kwan Chak

Individual Report
Spring Semester 2010-2011



Department of Computer Science and Engineering
The Chinese University of Hong Kong

Table of Contents

1. Abstract.....	3
2. Introduction.....	4
2.1. Motivation	4
2.2. Background	6
2.2.1. Research-Level VS Consumer-Level BCIs	6
2.2.2. Brain-Computer Interface Selection	8
2.2.3. Game Industry Responses to BCI.....	14
2.3. Project Overview.....	16
3. Summary on Our BCI Algorithm	20
4. Building Our BCI Computer Game	21
4.1. Introduction to Game Engines.....	21
4.1.1. What is the structure of a game engine?	21
4.1.2. Why even bothering using a game engine?	24
4.2. Why Unreal Engine 3?	26
4.3. Speedy Guide to the UDK.....	29
4.3.1. Primitive Modeling	30
4.3.2. Asset Creation/Management and Lighting	31
4.3.3. Visual Programming and Animation	33
4.4. Integration of UDK and Mindset	38
4.4.1. Introduction to UnrealScript	38
4.4.2. Communication between UDK and Mindset.....	41
4.4.3. Ways of BCI controls.....	44
4.4.4. Demonstrating Active Control.....	46
5. Advanced BCI Game Development.....	49
5.1. Introduction	49
5.2. Game Plan	49

5.2.1.	Main Idea	49
5.2.2.	Game Plot.....	52
5.3.	WalkThrough.....	53
5.3.1.	SHB 9/F	53
5.3.2.	SHB 5/F and 6/F	55
5.4.	BCI Integration.....	56
5.4.1.	Launcher Program.....	56
5.4.2.	Use of Mental State Number.....	60
5.5.	Features	64
5.5.1.	Interactive Menu	64
5.5.2.	Text-To-Speech Support.....	69
5.5.3.	Realistic Scenes	71
5.6.	BCI-Driven Game Puzzles	75
5.6.1.	Teleporter Activation	75
5.6.2.	Mind Shield.....	78
5.6.3.	Telekinesis Elevator.....	81
6.	Conclusion	84
7.	Limitation & Difficulties	85
7.1.	Getting stuck in UDK.....	85
7.2.	Poor signal from Mindset.....	86
7.2.1.	Explanation	86
7.2.2.	Relation with the BCI Game.....	86
8.	Division of Labor.....	87
9.	References.....	88

1. Abstract

From keyboard and joystick, to Wii-remote and Kinect motion detection, new controllers have always been fuels to bring about new generations of video games. However, when possibilities of motion sensing entertainment are gradually exhausted, one may wonder: what would come next?

We believe Brain-Computer Interface (BCI) might just be a potent candidate.

Our project starts off by studying the consistency and effectiveness of a non-invasive consumer-level BCI, Neurosky Mindset. Afterwards, we attempt to devise a way to improve the BCI control, namely by introducing a way for the users to train themselves to maintain a better control over the brainwave states. In this process, we shall propose our customized algorithm to approximate the attention meter.

Finally, utilizing the modern 3D computer game engine Unreal Engine 3, a complete BCI-enhanced adventure game titled “Psionscape” is developed. The game consists of puzzle elements driven by our newly developed BCI algorithm at the back-end, and this shall further strengthen the belief that BCI-enhanced gaming will be a feasible game genre in the future.

2. Introduction

2.1. Motivation

No one knows what the first digital game was, since it is debatable upon how one defines a “game”. However, one thing is certain - Since the invention of digital games, human has been actively pursuing new kinds of interfaces which enable communication between the games and the players, or at least, enable



Figure 2-1. For PC games, they started with a keyboard

one-way input from the players to the games.

In personal computers games, keyboard was one of the first interfaces, where early games usually used the arrow keys (e.g. DOOM, a first person shooter game, used $\leftarrow\uparrow\rightarrow\downarrow$ for movement). With the introduction of mouse and trackball as pointing devices,

they served as an additional interface when combined with the keyboard, or sometimes control games even without the keyboard.

Not long after the popularization of the mouse, the game industry evolved to create different controllers to suit different games’ needs.

In the past 10 years, the gaming industry has been a growing multi-billion-dollar business, this shows that the demand of videos games has been growing, and this rocketing demand also attracts a vast investment on new gaming interfaces, such as Dance Pad in PlayStation and Wii controllers in Wii, which furthered feedback to the snowball (i.e. the demand) positively.



Figure 2-2. A mouse was necessary to play Warcraft II (1995)

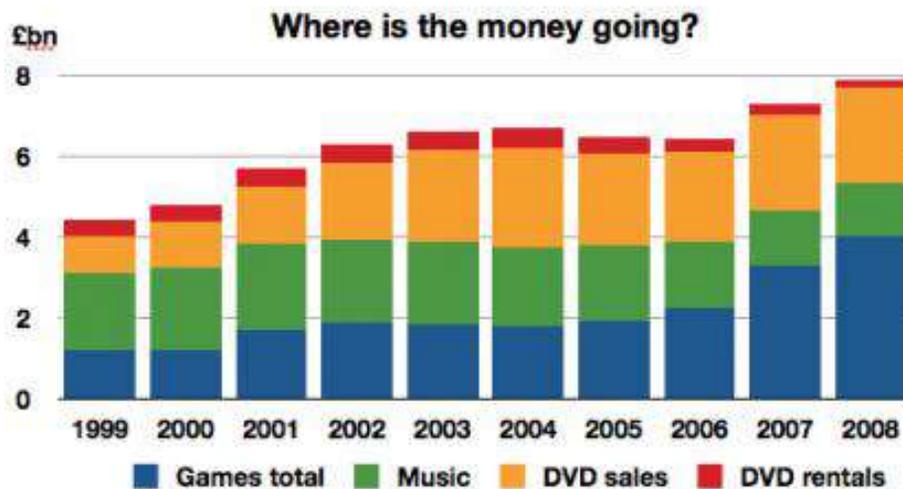


Figure 2-3. The video game industry has been growing at lightning speed, even exceeding income generated by the music industry.

(Arthur, 2009)

However, while new interfaces for console games (e.g. touch screens for NDS and remote motion sensors for Wii) has been developed, emergence of new gaming interfaces for PC games seem to slow down after the introduction of game pads, and we think a new gaming interface could perhaps give birth to a new genre of games in the big PC games market, where PCs are very widely owned in almost every family in developed countries.

Following the current trend, we could see that the physical world has almost been captured by interfaces like Microsoft Kinect.

Hence one may wonder if anything new would appear in the future gaming world, and what that would be.



Figure 2-4. With Microsoft's Kinect, players no longer hold any sensors to play

Recently, brain computer interfaces for consumer level have been released to the market (See session 2.2.2), making BCI entertainment possible. However, no commercial game has been released onto the market.

Why is that so? Is it because of technical difficulties to utilize the BCI?

We would like to grab that very opportunity by studying the possibility of developing a modern 3D computer game, which can utilize features of BCIs.

2.2. Background

2.2.1. Research-Level VS Consumer-Level BCIs

Some of the game interfaces, such as keyboards and gamepads, which controls games directly, including movement, rotation and inventory controls, are usually of high precision, i.e. you wouldn't see a mouse cursor shivering on the screen or floating in random direction.

Research level BCIs could also read the electroencephalography (EEG) in a relatively accurate manner, allowing recognition of many different actions. For example, the Department of Electronic Engineering of the Chinese University of Hong Kong has been working on a BCI project, where the user could input Chinese characters using one's brain waves alone, but the process requires about 1 minute per word and yields 75% accuracy (Oriental-Daily, 2010). For a game related example, The Department of Neurosurgery of Washington University in St. Louis managed to control a classic game "Space Invader" by moving the space ship left or right (note that only 1D linear movement is allowed), where the shooting behavior is governed by time automatically but not by the BCI. The process time for this kind of in-game operation is shorter, and the player could react to the bullets shot by the aliens on the top of the screen.



Figure 2-5 Research-Level BCI-controlled Space Invader

(<http://www.youtube.com/watch?v=T3-mxhDp-u8&feature=related>)

However, consumer-level BCIs are currently not as precise, and are not favorable to be used as direct controls; In addition is the slow recognition speed relative to quick response needed to react to game events (such as shooting a zombie which suddenly popped up in front of the player).

Yet consumer-level BCIs also have their advantages. While a P300 system (the research-level BCI system developed by CUHK mentioned above) may take from 20K to 50K HKD each, consumer-level BCIs are more affordable (cost no more than a few thousand HKDs) by ordinary players. Moreover, they are all very portable and mobile, due to their wireless nature. Another big advantage is the ease to wear the BCI, as they mainly use dry sensors and do not require a very accurate positioning of each electrode (in contrast to research-level BCIs which need to position at least a few dozens of electrodes before use). Hereafter, the BCIs we talk about would be consumer-level ones.

2.2.2. Brain-Computer Interface Selection

There are vast variety of game-player interfaces in the market, such as

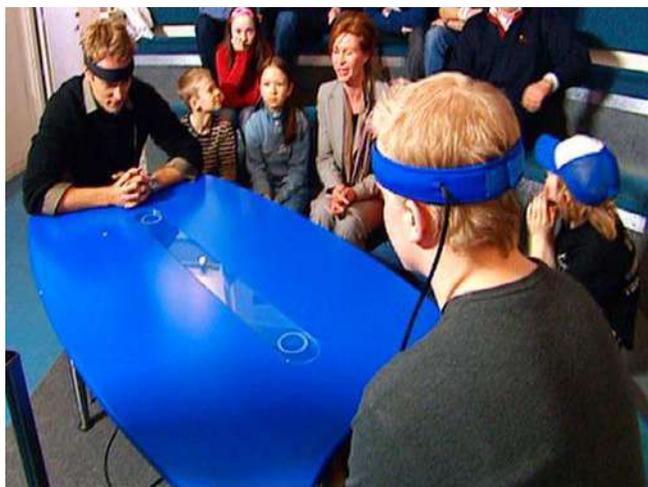


Figure 2-6 Mindball was one of the first consumer-level BCIs

those BCIs are usually very light and portable.

Wii-remote and Drum Sticks, but very few of them are intended for personal computer entertainment. However, the consumer level BCIs in the market, up to now, are for that purpose.

Therefore connections are easy, usually involving a USB plug-and-play, and

Within the ranges of consumer-level BCIs, there are ones with very specific uses.

For instance, 7 years ago, a company called The Interactive Productline which is located in Sweden produced one of the first consumer-level BCIs, named Mindball. A wirelessly controlled ball was place in the middle of a table, the players first attach a BCI onto their forehead, and it will read the EEG of them. As a result, a ball will move from one side to the other player's side when one of the players' focuses more intensively onto the ball, and the ball falls off the edge of loser's side.

However, this kind of BCI did not provide a general use for other application, and is not viable for development of computer software or games.

In March 2007, NeuroSky, a US-based company, released a headset attached with an EEG sensor, which is to be placed onto the user's forehead. The headset is called Mindset. This should be one of the first BCIs intended for consumer-level computer uses. The single sensor used was a dry one, and is non-invasive (in contrast to medical BCIs which may require insertion of electrodes into the skull for superior accuracy). The device features a decomposition of a whole range of raw brain waves data from the user, including alpha, beta, gamma and theta waves (See Chapter 3 for explanation of brain waves), and also algorithmic values representing "Attention" and "Meditation", consolidated by the raw brain waves data. "Anxiety" and "Drowsiness" are also supported using particular software. The latest firmware even allows the detection of eye blinks, but the underlying principles are not known.



Figure 2-7 Wireless with a dry sensor - NeuroSky Mindset

A year later, OCZ Technology released the Neural Impulse Actuator (NIA). The NIA is worn by putting it around the user's forehead, and it is very easy to do so due to the rubber-band-like structure (see Figure 2.2-4). The communication between the NIA and the computer is established by a USB connector (not wireless). There are a total of 3 sensors, which is more than that of Neurosky Mindset. However, the device can only read alpha and beta waves, although there is an overall motor neuron activity (e.g. moving your jaw up can increase its value). Moreover, unlike Neurosky Mindset, there is no processing of raw data, so it is even difficult for developers to tell when the user is in "Attention" state. Yet supposedly, the accurate nature of the 3 sensors may yield more precise brain waves which may be mapped to some keystrokes to play certain games.



Figure 2-8 Easy-to-put-on NIA

detect 12 kinds of movement and rotations (e.g. “up”, “left”, “forward”, “zoom”, “turn clockwise”, “turn left” and “sway right”) as supplemented by detecting motor neuron activities. Similar to Neurosky Mindset, it can detect emotions such as “Excitement”, “Engagement”, “Meditation” and “Frustration”. Moreover, it can detect facial expressions like laughing and clenching. Another feature Emotiv EPOC exclusively demonstrated is the ability to make objects disappear in the demo. In addition to BCI features, it can also measure angular rotation of the head in 2 dimensions (i.e. yaw and pitch, but not roll) as detected by the 2 gyros.

The Emotiv EPOC is the latest BCI available came in December 2009. This BCI has 14 electrodes and so far is the BCI with the highest number of electrodes. That is very comparable to medical-level BCIs which usually has 19 electrodes. The vast number of electrodes covers different areas around the head, and thus has a lot of features. Therefore it can



Figure 2-9 Highest number of sensors - Emotiv EPOC

To sum things up, we have prepared a comparison table for the BCIs mentioned above:

	Mindball	Mindset	OCZ NIA	Emotiv EPOC
Released	March 2003	March 2007	May 2008	December 2009
SDK Available	No	Yes	Yes	Yes
Connection	Wireless	Wireless	Wired (USB)	Wireless
Number of Electrodes	1	1	3	16
Sensor Type	Dry	Dry	Dry	Saline (Wet)
Raw Data Collection	No	Yes	Yes	Yes
Attention/Engagement	No	Yes	No	Yes
Meditation/Relaxation	No	Yes	No	Yes
Anxiety/Frustration	No	Yes	No	Yes
Drowsiness	No	Yes	No	Yes
Excitement	No	Yes	No	Yes
Push	No	No	No	Yes
Pull	No	No	No	Yes
Lift	No	No	No	Yes
Drop	No	No	No	Yes
Push Left	No	No	No	Yes
Push Right	No	No	No	Yes
Rotate Forward	No	No	No	Yes
Rotate Backward	No	No	No	Yes
Turn Left	No	No	No	Yes
Turn Right	No	No	No	Yes
Tilt Left	No	No	No	Yes
Tilt Right	No	No	No	Yes
Disappearance	No	No	No	Yes
Facial Expressions	No	No	No	Yes
Motor Neurone Activity	No	No	Yes	Yes
Playing Music	No	Yes	No	No
Head Rotation Detection	No	No	No	Yes
Price in USD	\$18700.21	\$159.20	\$138.99	\$299.00
Developer Edition	-	(Free SDK)	(Free SDK)	\$500.00

Table 2.2-1 Comparison between BCIs as of 28th November 2010

To select the most appropriate BCI for our project, we considered a lot of different features of the above BCIs with weighing.

To begin with, Mindball does not really suit our need because it is of specific use (i.e. for that ball game only), what's more is the terrific price to own one, i.e. \$18700.21, which will make it not probable to become a home entertainment trend. Moreover, there is no possible connection for personal computers, and there is no SDK for development, so we cannot use it to build our game.

The OCZ NIA has the lowest price among the 3 remaining candidates, and it employs the use of dry sensors, which makes the user comfortable with putting this on. It enables raw data feedback to the computer / SDK, and that is useful for customized signal processing. A point to note is that the wired nature of the device may be speculated to reduce the comfort of prolonged use because it hinders the user's movement (e.g. for Mindset, you can go grab a cup of coffee without taking it off). The critical reason for not using OCZ NIA is its failure to detect cognitive states (e.g. Attention). In addition, while it may be possible to detect motor activity, it cannot tell the difference between moving one's jaw and moving one's eyebrows, as complained by the users on forums (Kenner, 2009).

The Emotiv EPOC seems to be very competitive candidate, as it is not only able to detect cognitive states like Neurosky Mindset does, it can also detect motions, which is a very attractive feature for video games. However, while these features are possible thanks to the large number of sensors Emotive EPOC has, saline sensors are employed (not dry). The users may find it troublesome to wet the sensors or their head every time they use it, and it is not just a matter of dropping a few salt solutions onto the sensors but to "make them dripping wet"! Yet even with that, the connection may also be poor. (Emotiv-Administrator, 2010) Moreover, although motion detection is possible, the accuracy is not very high, with some users stating that "it can never follow my thought!" In addition, while there seems to be multiple events to be detected, only a limited number of events can be detected at the same time, and the user needs to explicitly address them, making EPOC not a very suitable candidate. (Emotiv-Administrator, 2010)

While NeuroSky Mindset is not the most accurate BCI (as it only got 1 pea-sized electrode), it still supports a number of detections such as Attention and Meditation. Moreover, it allows a broader range of raw brain waves data, enabling



Figure 2-10 Some people find it difficult to establish perfect connection
(Notice how the user holds the sensor closely towards the forehead)

a more potent signal processing to be carried out ourselves. In addition, as a benefit for gaming experience, it is essentially a wireless headset which can play music, this might enhance the gaming experience by providing surrounded sound of better quality than ordinary loudspeakers. Lastly, it has a

very low costs compared to other BCIs except NIA, making it more easy to popularize among families as it is more affordable.

However, it also has a drawback (which is also present in other BCIs) – Some people find it difficult to establish perfect connection between the sensor and the users' head. This will be one of the limitations we would like to address later and perhaps during demonstration.

2.2.3. Game Industry Responses to BCI

The emergence of different BCIs attracted some attention from the gaming industry, with many of the released or potential products relying on the “Attention” state. (See Chapter 3 for reasons)

For example, Mattel Inc released a non-digital game called MindFlex. In this game, the players need to focus to increase their concentration to raise the ball, and lower the ball by lowering the concentration level, and use a knob to move a ball left or right, with the goal of passing the ball through different obstacles. In fact, this game is using a lite version of NeuroSky Mindset’s chip (the “Mind Force” chip).

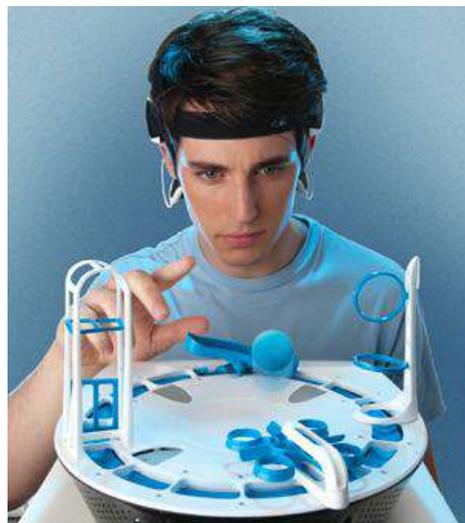


Figure 2-11 MindFlex (Based on Neurosky Mindset)

Now here comes a big question:

Is there any game-developer-made BCI computer games?

The short answer is: No.



Figure 2-12 NeuroSky’s own game - The Adventure of NeuroBoy

For demonstration purpose, the BCI producers, of course, produced their own game demos.

For example, NeuroSky, the same company which developed the Mindset, built a game called “The Adventure of NeuroBoy” to demonstrate features of Mindset. The game, which comes free-of-charge with the Mindset, has no story at

all, but let the player takes control of a character walking around using WASD keys and use the mouse to select an object for one of the 4 purposes: Attract towards player, Push away from player, Levitate and Ignite, which are governed by the 2 states (i.e. “Attention” and “Meditation”).

However, not all hope is lost.

Announced at the 2008 Tokyo Game Show was good news about commercial BCI games. The Japanese game developer Square Enix, which is well-known for its Final Fantasy series, announced the development of the first BCI-enhanced computer game – Judecca. (Fruhlinger, 2008)

Judecca is a first-person shooter game, in which the player is immersed in a world of zombies. The game makes use of NeuroSky Mindset’s “Attention” level. Up to now, the announced BCI features are:



Figure 2-13 Judecca in action, revealing zombies using Concentration level

1. “After concentrating on a glyph that glows in direct relation to your ability to concentrate, you will open up what’s called your “Devil’s Eye”. Only once you have attained a heightened state of concentration, will you be able to see Judecca’s zombies and kill them.”
2. “Those who can tweak their concentration levels even further will be able to walk through walls.”

If that is not descriptive enough, the below is a screenshot which reflects what happens if the “Attention” / “Concentration” level is high enough in Judecca.

An important point to note is that the game is BCI-enhanced but not directly controlled by it, the movement and shooting still relies on keyboard and mouse. With a global game developing company like Square Enix still not working on more BCI features or direct controls, this leads to our speculation that the current game industry is still remain doubt about the accuracy of consumer level BCIs.

2.3. Project Overview

Chapter 2.2.1 gave us an insight on the current trend of consumer BCIs and also their limitations.

On one hand, we can see that BCIs are rapidly being improved and commercialized. With the lower costs and improving accuracies and features, we predict that in the near future, there will be at least a small to medium sized market for home uses.

On the other hand, we could see there are limitations for current BCIs. The most important part is the difficulties in deciding detected states (e.g. “Meditation”) and movements (e.g. “Move forward and Turn Left”).

Chapter 2.2.2 concludes by stating that the current computer game industry actually is at the beginning of “trying” to develop BCI games. With a large company like Square Enix (a publicly owned multinational company with thousands of employees), they still limit the game with very few BCI features, possibly because of the limited accuracy for direct movement controls.

We speculate that the lack of developers building BCI games is due to:

- 1) It may be hard to make games which utilize BCI features
- 2) BCI may just unlock too few in-game features. (e.g. revealing zombies)

Therefore, we would like to see if we can tell a different story by dividing our project into 2 phases, with each phase done within each semester.

Firstly, we will study the NeuroSky Mindset, to see how we could operate it, and how we could get data from it. Moreover, we will try to see if there is any trace of correlation between its claimed states (e.g. “Attention” and “Meditation”) and the users’ feedbacks.

On the other hand, we will study a game engine, Unreal Engine 3, to see if it is possible to make a BCI-enhanced game (at this stage, we do not plan to build a game “directly controlled” by the BCI, as learnt from lessons of the Judecca). If a modern 3D game engine can be modified to produce BCI-enhanced games, then it would be like owning a factory to developers, and they can make BCI-enhanced games pretty much like how they normally do it.

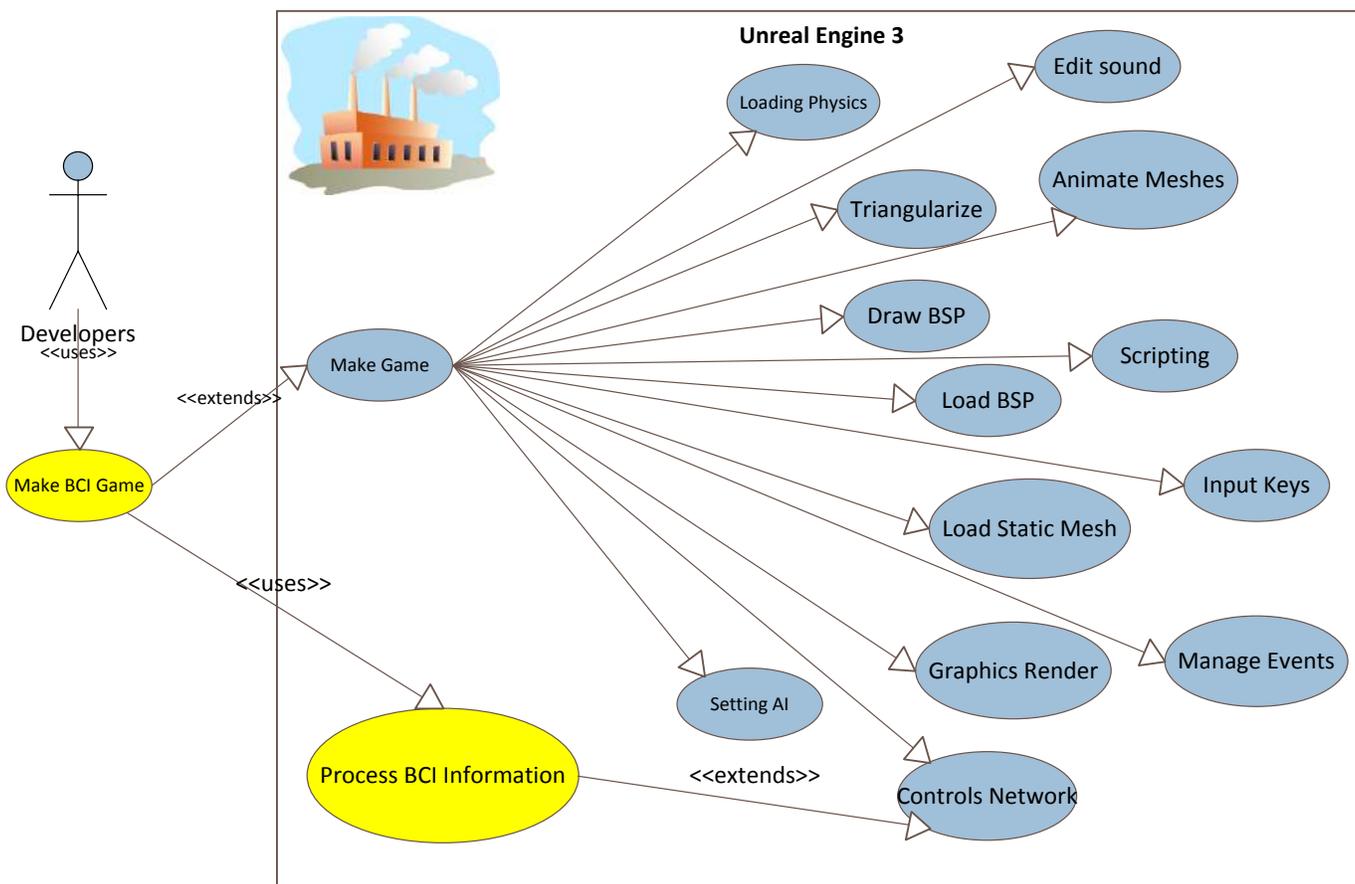


Figure 2-14 Use Case Diagram of Modified Unreal Engine 3

Moreover, we will study how BCI could help facilitates different game events or features.

For the first phase, we will first make a small demo in Unreal Engine 3 environment to demonstrate that it could actually work to combine BCI and an ordinary game engine. Then proceed to demonstrate the possibility of active controls (See Chapter 4 for details about Active/Passive Controls) using NeuroSky Mindset's eSenses (e.g. "Attention").

For the second phase, we will try to analyze raw brain waves data to see if we could devise our own algorithms to calculate values which represent human emotions, and see if we could improve eSenses or develop other senses (e.g. the state of "Fear"). And on the other hand, investigate the possibility of passive controls by the BCI.

Last but not least, we are looking forward to creating a small-sized BCI-enhanced game by integrating different kinds of controls, and evaluate it to see if the players find it more interesting than non-BCI versions.

(Disclaimer)

Throughout our project, we may use the phrase "BCI-enhanced games". However, by "enhanced" we simply mean the games which are different from those in the current game market in a way that BCI features are absent in present games, we do not attempt to have a bias that "BCI-enhanced" implies an positive impact on the gameplay experience over "ordinary" games. In fact, the question whether games with BCI features would offer a perceived better gameplay experience is exactly what we would like to find out, after developing a small-scaled game featuring some BCI features in the coming semester.

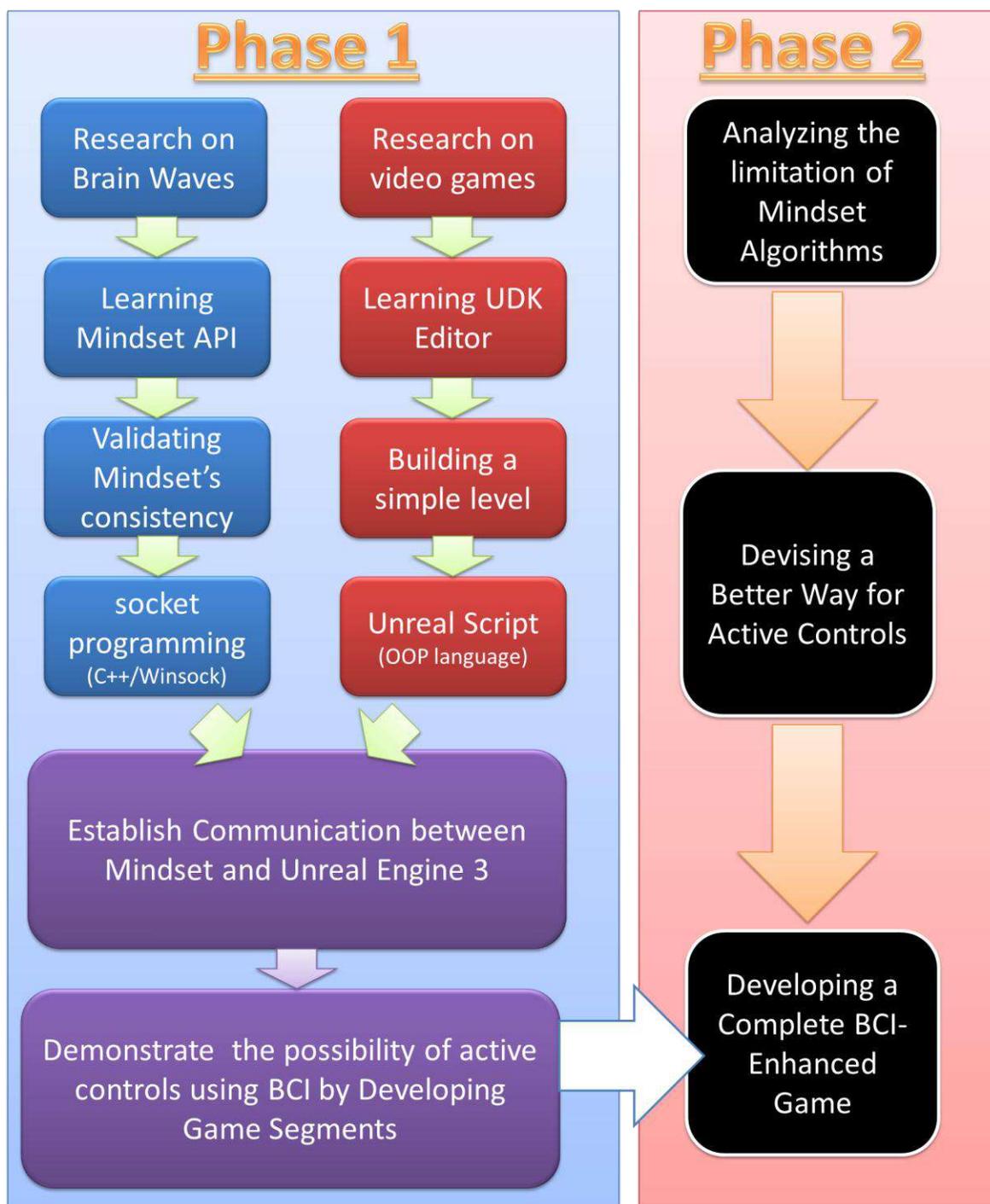


Figure 2-15 Complete Project Overview

3. Summary on Our BCI Algorithm

Thanks to my partner, Donald Cheung, we have a more controllable brain wave state classification algorithm.

After a period of calibration, the user's current brain waves can be classified into one of the three types.

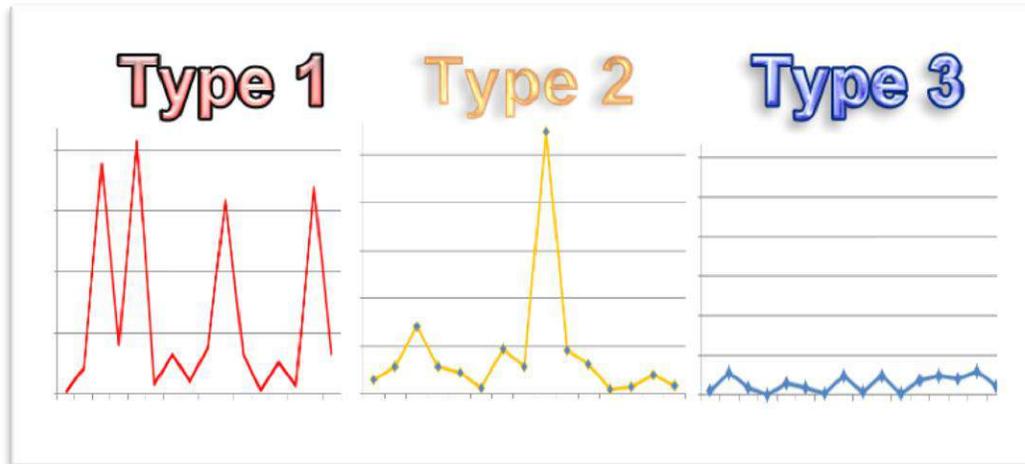


Figure 3-1 Different Types of Classified Brain Waves

As seen in Figure 3-1, the Type 1 wave usually has a series of high magnitude peaks, while Type 3 wave typically consists of a series of very low magnitude cycles.

This observation leads us to manipulate these three types of brain waves, by assuming that the respective types could reflect the level of “Mind Power” of the BCI-enhanced game players. This finding is essential to the development of our BCI game.

4. Building Our BCI Computer Game

4.1. Introduction to Game Engines

This sub-chapter will briefly introduce different aspects of game engines.

4.1.1. What is the structure of a game engine?

In a nutshell, game engines are basically black boxes which utilize third-party software (which are called “middleware”), by using respective APIs, to enhance the functionality of the engines. (Gregory & Lander, 2009).

The middleware can control a wide variety of things. For instance, “Scaleform GFX” is a vector graphics rendering engine used to create in-game Adobe Flash based graphical user interfaces (GUI) such as main menu and skill selection GUI (Scaleform.com, 2010), while “Nvidia PhysX” is integrated into some game engines so as to boost the efficiency to render 3D scenes with preset physical

properties such as blowing a piece of cloth up with air.

(NVIDIA, 2010)

There are even middleware specifically designed to draw dynamic

and random

trees (e.g. SpeedTree) so developers do not need to spend time on writing codes to draw trees in different games.



Figure 4-1 In Batman: Arkham Asylum, the wing of Batman is rendered with “cloth physics” using Nvidia PhysX

While the communication between the game and the operating system is established by codes of the engine or its associated middleware, the developers need to build “Level” files to be fed into the game engine. The file should contain utterly everything inside the game world on a particular level, including but not limited to, object properties (e.g. coordinates, dimensions and textures), events (e.g. spawning an enemy bot upon entering a room), animations and lighting information. After that, the engine just waits for players’ input to trigger different in-game events designated in the level data.

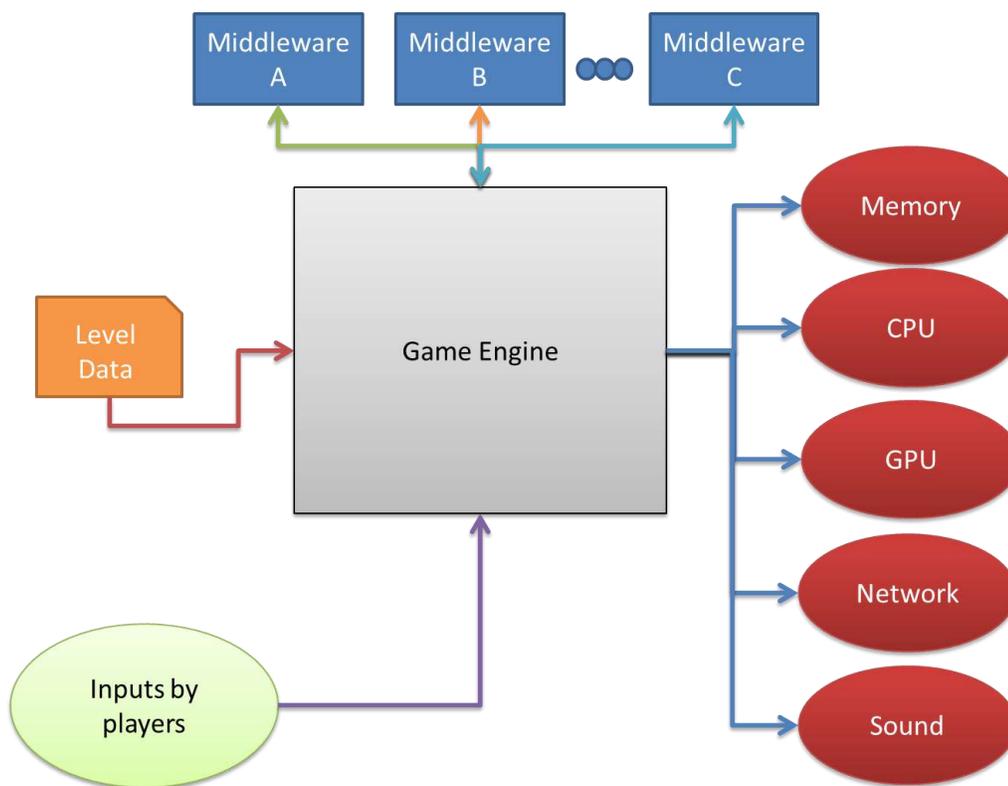


Figure 4-2 Simplified Context Diagram of Game Engines

The developers using a game engine do not really need to consider the hardware problems because they are in the scope inside the game engine.

In each pre-designed period of time, the game engine will update the game information by consulting the game logics (e.g. damage done to a particular NPC, Deaths, and destruction of objects) and for each (usually shorter) time event called “tick”, the game engine will visualize the changes to the player by rendering accordingly.

Without a game engine, programmers will have to explicitly type in codes into specific location of the source files to update the game logic, and command the rendering engine in a line-by-line manner

```
double last = timeGetTime();
while (!endGame)
{
    if ( (timeGetTime()-last)>1/frequency)
    {
        game_logic();
        last = timeGetTime();
    }
    renderWorld();
    renderCharacters();
}
```

Figure 4-3 Pseudo codes for a simple game engine

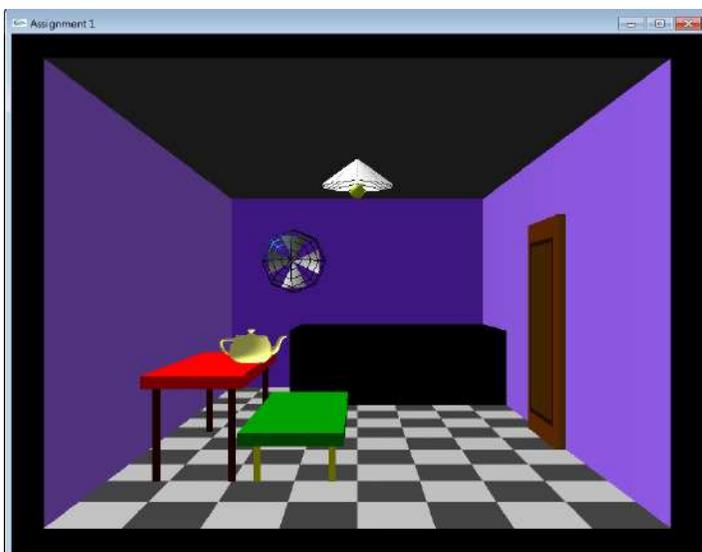


Figure 4-4 Using OpenGL alone to create an interactive scene offered a taste of the principle of game engines, but codes are explicitly written to control objects in a relatively low-level manner and hard to reuse. (CSC3260 Assignment 1 2009-2010, done by K.C. LIU) in-game events by harness the power of game engines (See Figure 4.1.2-2).

(instead of object-by-object). Moreover, the control over peripherals (even I/O) is also needed to be addressed one-by-one, slowing down the production process. Moreover, it is difficult to reuse codes of a previous game and port into another to-be game. In contrast, developers sometimes just create multiple games using only different artworks and

4.1.2. Why even bothering using a game engine?

In the early days of video games (arcade game era), the hardware development was very rapid, and each game is programmed and built with their own codes and models to maximize its attraction to gamers by catching up to the latest technologies.

While, for each game, programmers take tremendous efforts just to attach the game to the kernel with the right drivers and to handle the memory, the codes are all threw into rubbish bins, because the latest hardware would not be compatible with the recycled codes. We call this kind of game development the “Monolithic Approach”.

```
FYP_main()
{
    // codes for main menu
    ...
    // codes for level1
    ...
    // codes for level2
    ...
    // codes for level3
    ...
    // codes for end game credits
    ...
    GOTO main menu
}
```

Figure 4-5 Monolithic Game Development

As each game and their levels were being programmed one-by-one, this encouraged creativity because the developers no need to reuse any of the codes from a previously written game, and it was still manageable for small teams at that time (as the games themselves were also relatively small).

However, one may find it difficult to maintain the codes for different levels, especially when those lines of codes were copy-and-pasted onto different levels (e.g. initialization of stats for the playable character). Moreover, it was troublesome to divide work to different team members.

Since the late 1980s (Ward, 2008), in-house game engines have been developed to build multiple games. From software engineering points of view,

software reliability should be enhanced by the reuse of codes written in the game engine modules.

Moreover, the division of work could be speed up by employing an game engine. For example, artists could focus on character creation using artwork modules from the game engine, while the programmers could focus on writing in-game objects properties, and story writers could work on writing dialogues.

The number of game engines developed and employed has been growing. And modern game engines could sometimes create different games by changing only artworks and scripts (See Figure 4.1.2-2). Moreover, development kits of the game engines could be distributed to the mod communities, so that new contents could be added by the fans continuously, while the contents created may attract more players to play the games, or keep the fans loyal to the game for a longer period of time, thus boosting sales of the games and their sequels.



Figure 4-6 Assassin Creed 2 (left) and Prince of Persia: The Forgotten Sands (Right) both used Anvil engine developed by Ubisoft. Seemingly only the artworks and postprocessing filters (one is blue and one is brick-red) are different.

Besides, game engines nowadays are usually cross-platform, meaning that the games created are not just playable on PCs, but also on consoles like PS3 and Xbox360. Therefore games could be launched on different platforms on almost the same release dates.

The portability nature of games created by game engines, together with the hardware and graphics abstractions, when incorporated with our BCI modules, should be more than enough to prove that it is not difficult to create BCI-enhanced video games in general.

4.2. Why Unreal Engine 3?

When you type “game engine” in Google or Wikipedia, you will probably be amazed to see an almost endless list - there are over hundreds of game engines out there (DevMaster.net, 2010), so here we would just select a few of them and compare them to see why Unreal Engine 3 was selected. After consultation with Dr. Or Siu Hang in our department, we had targeted 3 major game engines available in the market. Figure 4.1.2-1 shows a comparison between them.

In fact, we also investigated a variety of engines. Interestingly, in the open-source community, a 3D rendering engine called OGRE, is used to make games. (OGRE, 2010) However, while it does support a lot of features (such as 3D API support, material/shader support, meshes and animation), it is not an exactly a game engine, features such as sound have to be supplemented by other engines. This makes it unfavorable to make games easily when compared to suite-like game engines like Unreal Engine 3.



Figure 4-7 OGRE is not even a game engine, but still popular.

While open source projects usually are praised for their higher performance and less bugs due to collaborative work by wider range of developers, Reality Factory does not support as many features as other game engines (See comparison table we created). This may be due to the fact that there are just too many open-source game engines out there, and fans do not focus their power on this one. Moreover, the resources game developer companies usually are far greater, so the quality of the commercial game engines is greater.

		Reality Factory	CryEngine	Unreal Engine 3
Platform	Windows	Yes	Yes	Yes
	Linux	No	No	Yes
	Mac	No	No	Yes
	PS2	No	Yes	Yes
	PS3	No	Yes	Yes
	PSP	No	No	Yes
	Xbox	No	Yes	Yes
	Xbox360	No	Yes	Yes
	Wii	No	No	Yes
Cost	License	Open-source	Commercial	Commercial
	Price	Free	Comes with Crysis	Free for Non-commercial
Documentation	Level Editor	Yes	Yes	Yes
	Asset Creation	Yes	Yes	Yes
	Programming	Yes	Yes	Yes
	Engine Architecture	No	No	Yes
	Knowledge Database	No	No	Yes
	Video Tutorials	No	No	Yes
	Demo w/ Source Codes	No	No	Yes
Networking	Client-Server	No	Yes	Yes
	Peer-to-Peer	Yes	No	Yes
Graphics	Hardware Acceleration	No	Yes	Yes
Shadows	Shadow Mapping	Yes	No	Yes
	Shadow Volume	No	Yes	Yes
	Projected Planar	No	No	Yes
Texturing	Multi-Texturing	Yes	Yes	Yes
	Bump mapping	Yes	Yes	Yes
	Mip mapping	Yes	Yes	Yes
Animation	Keyframe Animation	Yes	Yes	Yes
	Skeletal Animation	Yes	Yes	Yes
	Facial Animation	No	No	Yes
Physics	Collision Detection	Yes	Yes	Yes
	Rigid Body	Yes	Yes	Yes
	Vehicle Physics	No	Yes	Yes
AI	Pathfinding	Yes	Yes	Yes
	Scripted	Yes	Yes	Yes
	FSM	No	No	Yes
Scene Management	BSP	Yes	Yes	Yes
	Portals	Yes	Yes	Yes
	LOD	No	Yes	Yes

Figure 4-8 Comparison of Some Popular Game Engines

On the other hand, CryEngine, a commercial game engine developed and employed to make the game Crysis, has several advantages over Reality Factory.



Figure 4-10 Realistic 3D rendering in Crysis - A game built by CryEngine

Firstly, the engine supports more platforms than Reality Factory, enabling the production of games spanning across a larger gamer market.

Secondly, the sandbox CryEngine employed is a more What-you-see-is-what-you-get

(WYSIWYG) that includes different game objects in the editor real-time.

Moreover, the hardware acceleration and the shadow volume rendering create a more realistic 3D graphics rendering without high loss in rendering speed. In addition, there is a smart LOD management built into the game engine, making adjustable rendering possible.

However, no matter it is the number of platforms supported, network types supported, the unique lightmass feature, or the higher rendering performance (e.g. shadow mapping / LOD), Unreal is the winner. Criticism leveled against was the engine gearing towards first-person shooter (FPS) games but there has been more than a hundred games being



Figure 4-9 Dungeon Defenders - an online multiplayer strategy game powered by Unreal Engine 3, demonstrating flexibility in graphics and types of games developed.

developed by it and many are not FPS but ranged from fighter games (e.g. Mortal Kombat) to Driving game (e.g. Intellidrive, a game developed by US government).

In addition to the user-friendliness in using the development toolkits (we tried all of the above engines), the flexibility in controlling the rendering properties, models and even I/O (which will enable us to communicate with the Mindset!) really made it stand out. No wonder the game engine is the number 1 in “Top 10 Game Engines” (Fear, 2009). That’s why we finally choose to use the Unreal Engine 3.

4.3. Speedy Guide to the UDK

The Unreal Engine consists of more than 18 middleware to provide functionalities ranging from 3D rendering, flash-like menus, and animation to lightings and scripting. Moreover, there is an integrated environment called Unreal Development Kit (UDK) which provides an interface to harness the power of the engine. On a side note, the UDK also compiles objects written in an object-oriented programming language called “UnrealScript”, so that technical developers could create customized content or events using the engine.

We spent **months** to study how to **master** the UDK and here’s a summary demonstrating our understanding, and at the same time, trying not to get our readers bored. ☺

4.3.1. Primitive Modeling

As you can see in the figure, there are 4 viewports. The bottom-left corner is the game perspective viewport, and in a clockwise direction are the top view, front view and the side view respectively.

On the left is a toolbar for drawing solid primitives by Constructive Solid Geometry (CSG) stored using Binary Space Partition (BSP). CSG is actually a method to draw solids by using Boolean operators, specifically “intersection”, “union” and “difference”, while BSP is a data structure of storing such solid geometry due to its efficiency to be

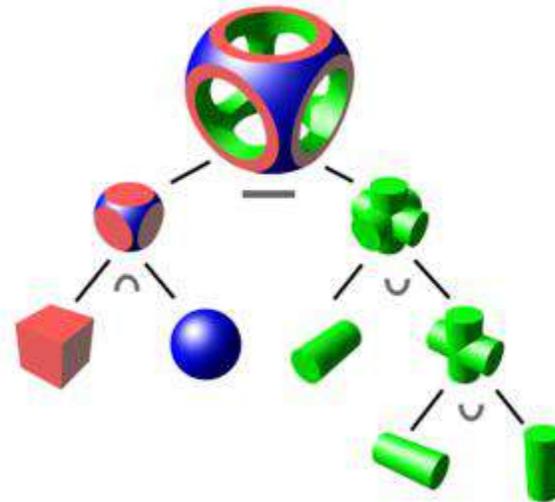


Figure 4-11 Effects of CSG operators (from Wikipedia)

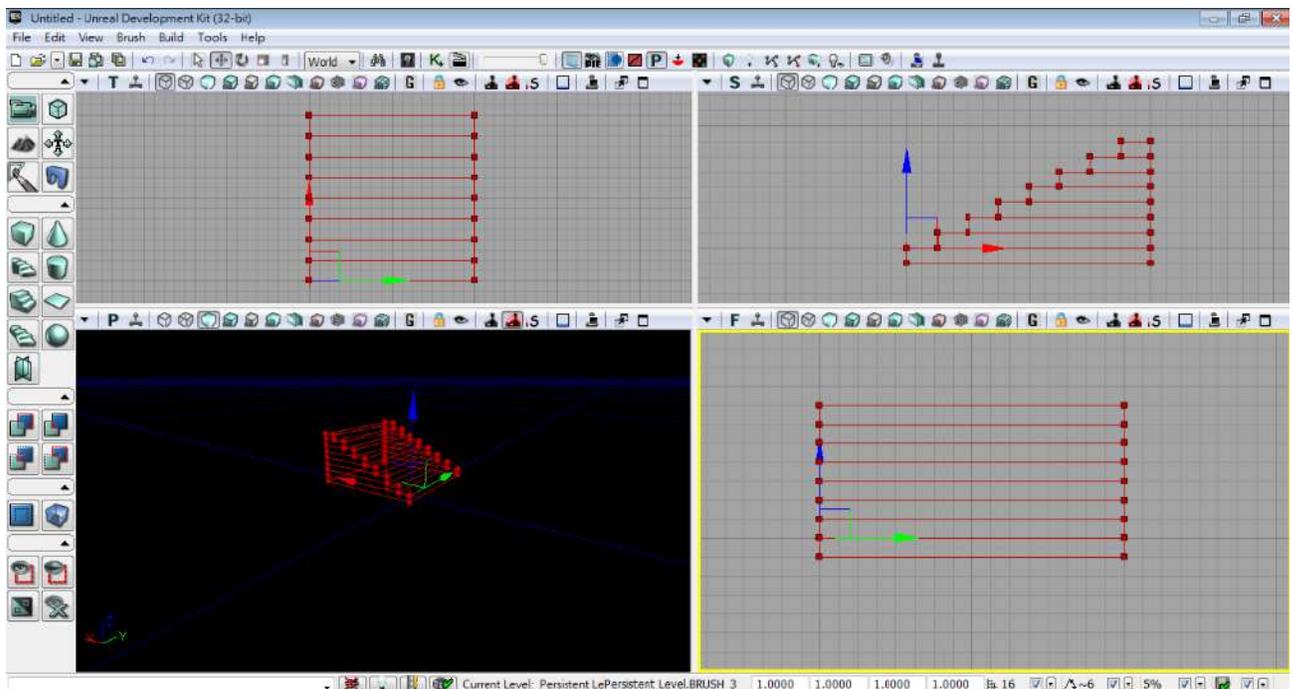


Figure 4-12 UDK (Editor)

drawn (It is just like accessing a tree resulting from a 3D quick-sort).

4.3.2. Asset Creation/Management and Lighting

Using the primitives, we can draw a series of objects. But that would not be realistic or meaningful. Therefore we should create a texture for the objects.

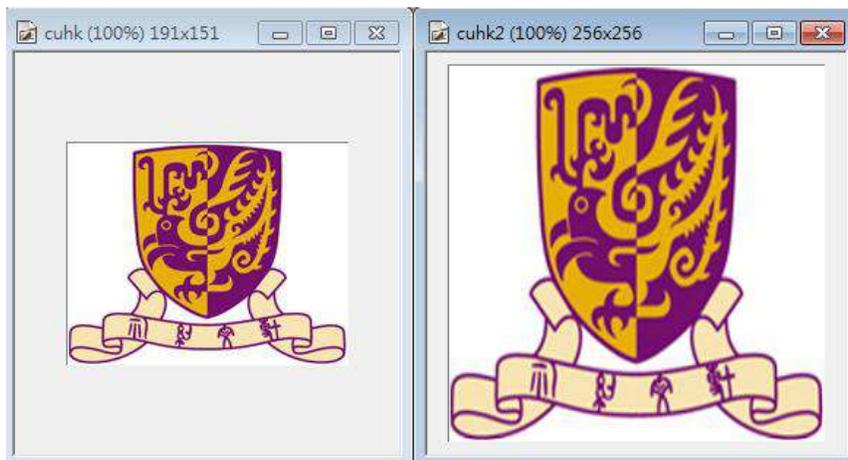


Figure 4-13 Bigger pictures with lengths of powers of 2 (Right) are most suitable for MIP-mapping in UDK because more MIP maps could be generated without precision loss

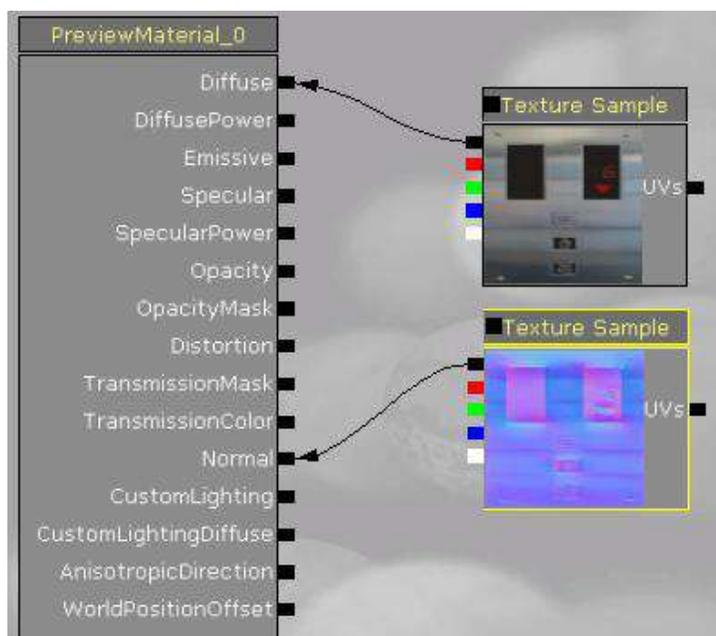


Figure 4-14 Material Editor in UDK

However, Unreal Engine 3 employs “MIP mapping” for textures, in which pre-calculated texture maps are prepared in sizes with sides always divisible by 2 (e.g. 256x256 pixels), and the Unreal Engine will choose the map with a size closest to the object’s plane (e.g. Texturing a 200x200

rectangle would use the 256x256 map instead of 128x128 map). Due to the employment of MIP mapping, the image used should be of dimensions which can be consecutively divided by 2 so that precision is not lost. For example, halving 5x2 would yield 3x2 where supposedly 2.5x2 should be chosen instead but there is no such thing as “0.5” pixels. (Walsh, 2008)

By applying suitable textures onto the BSP models we created, we could create meaningful scenes, such as the Lift Lobby of an engineering building in CUHK.

However, note that the object could not be seen without light. Therefore light sources such as point light and directional light should be placed in suitable locations. In this example, we added static point light sources with lightmass (static ray tracing technique employed by UDK). With the texture mapping involving not just the “diffuse” texture (i.e. the “real” material of the object), but also the “normal” texture (i.e. data of roughness of the surface), the lightmass can approximate the overall reflection on the material and project the light information to the camera to visualize the texture-mapping result.

When objects are modeled and textured, we can convert them into static meshes using built-in conversion algorithms in UDK, and the meshes, like other assets (e.g. textures /

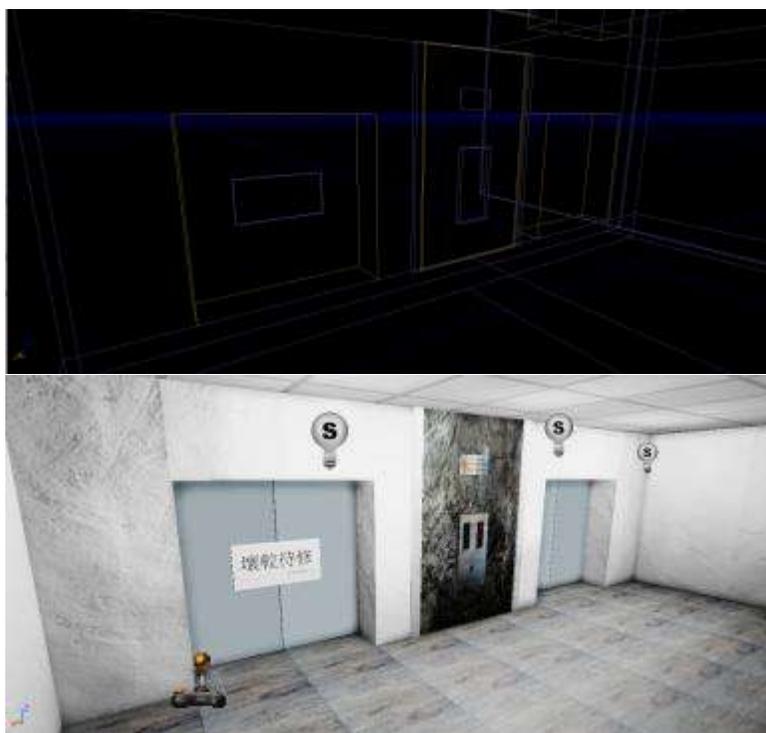


Figure 4-15 Wireframe of BSP models (Top) and the models after texture-mapping (Bottom)

sound cues) will be categorized into the content browser, so later we could just drag-and-drop the different assets to use them.

Finally, for collision detection (e.g. a bullet will be blocked by a wall, and we have to define the boundary on the wall to tell the engine where to stop that bullet), we have to set up collision information by using the static mesh editor provided by UDK.

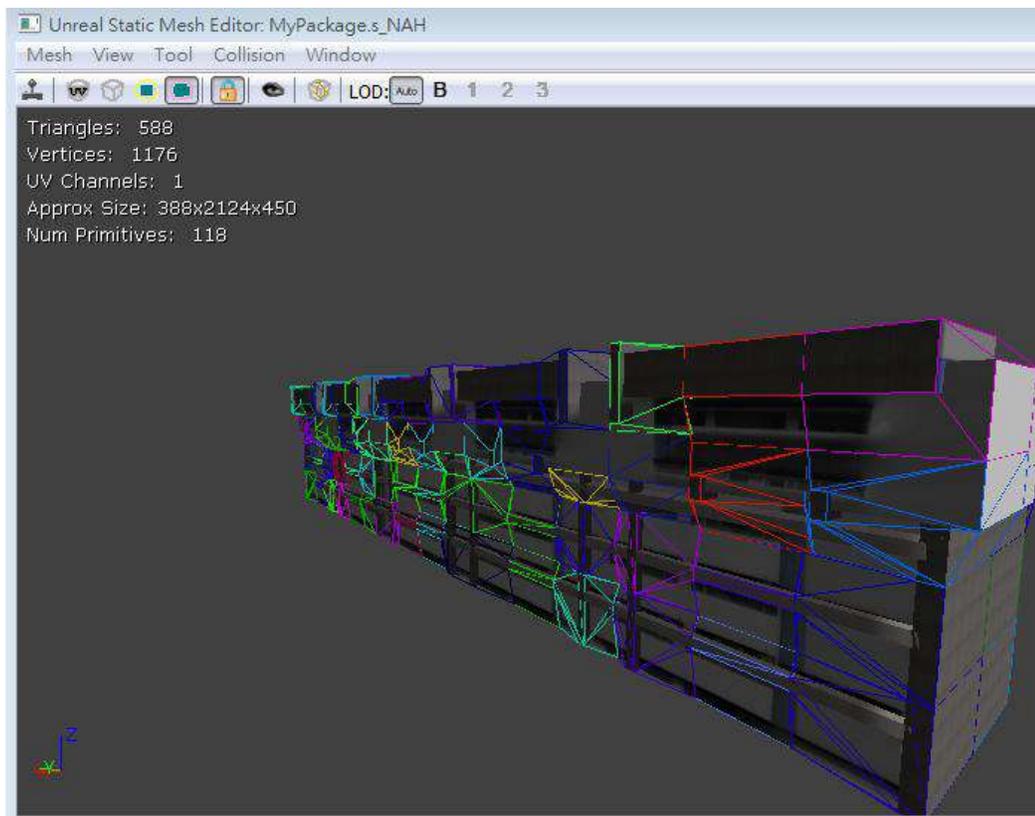


Figure 4-16 Replicating the Humanity Building of [New Asia College](#) of CUHK, with collision information

4.3.3. Visual Programming and Animation

UDK included a sub-system called Kismet which allows a part of programming being done visually.

Basically, Kismet surrounds about “Events”, “Variables”, “Action”, “Comparison” and “Matinee”, where “Matinee” is a sub-system to control animation which we will discuss a little bit later.

Programming done in Kismet usually is triggered by an event, and comparison may be done on variables, and decide which actions to be done, or whether an animation sequence should be played.

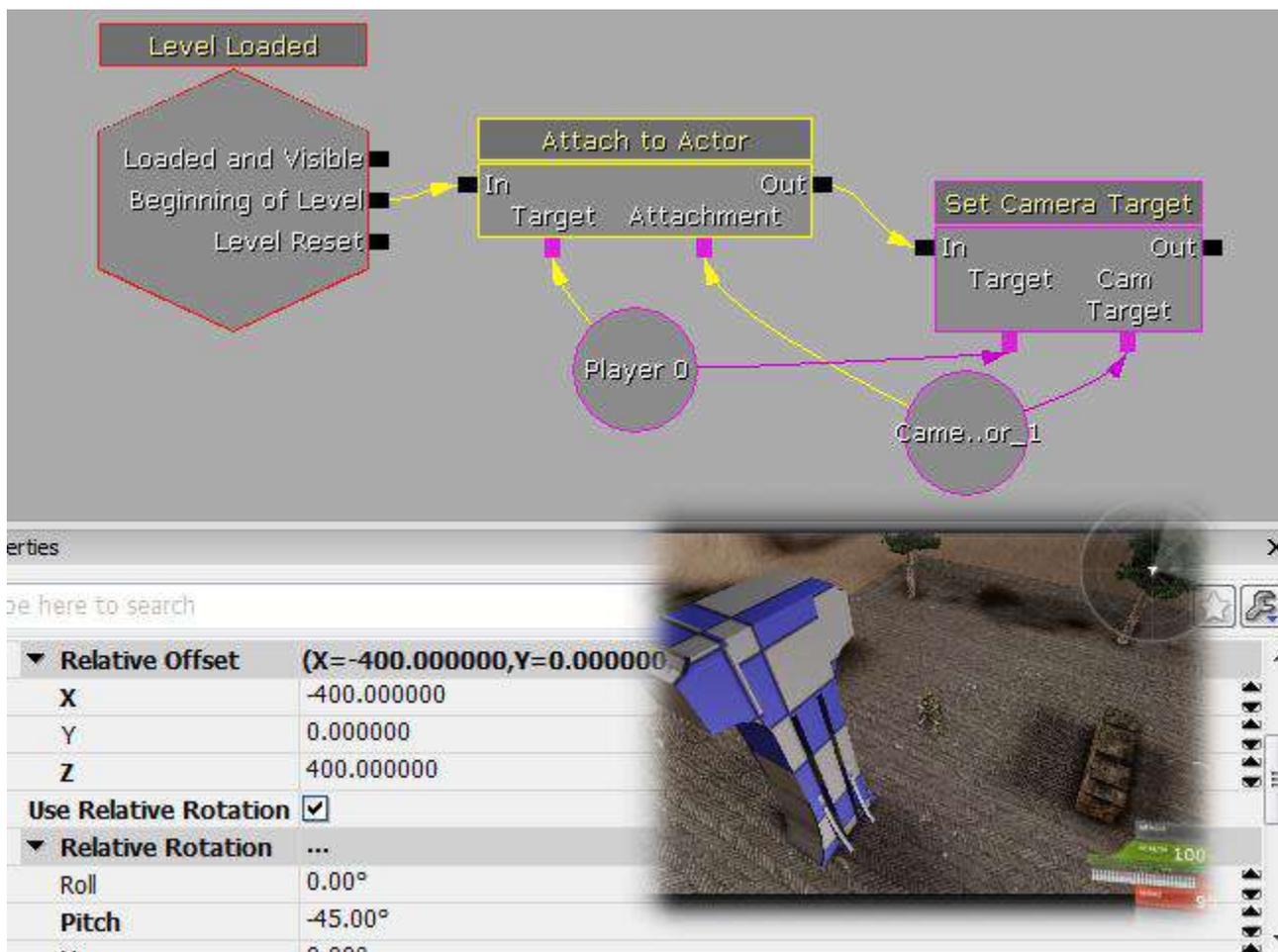


Figure 4-17 Third-person view done in Kismet (Outer) and its effect (Bottom-Right)

As you can see in Figure 4.3.3-1, the event “Level Loaded” attach the camera onto the player in a way that the camera is positioned behind and higher than the player, to create a third-person view. The background is our creation of a

United College section, showing the UC water tower without textures and some trees created by SpeedTree, another middleware integrated in UDK.

For animation, the matinee employs a key-frame animation system, which interpolates “key-frames” (i.e. snapshots of object in different timeslots) to create animation.

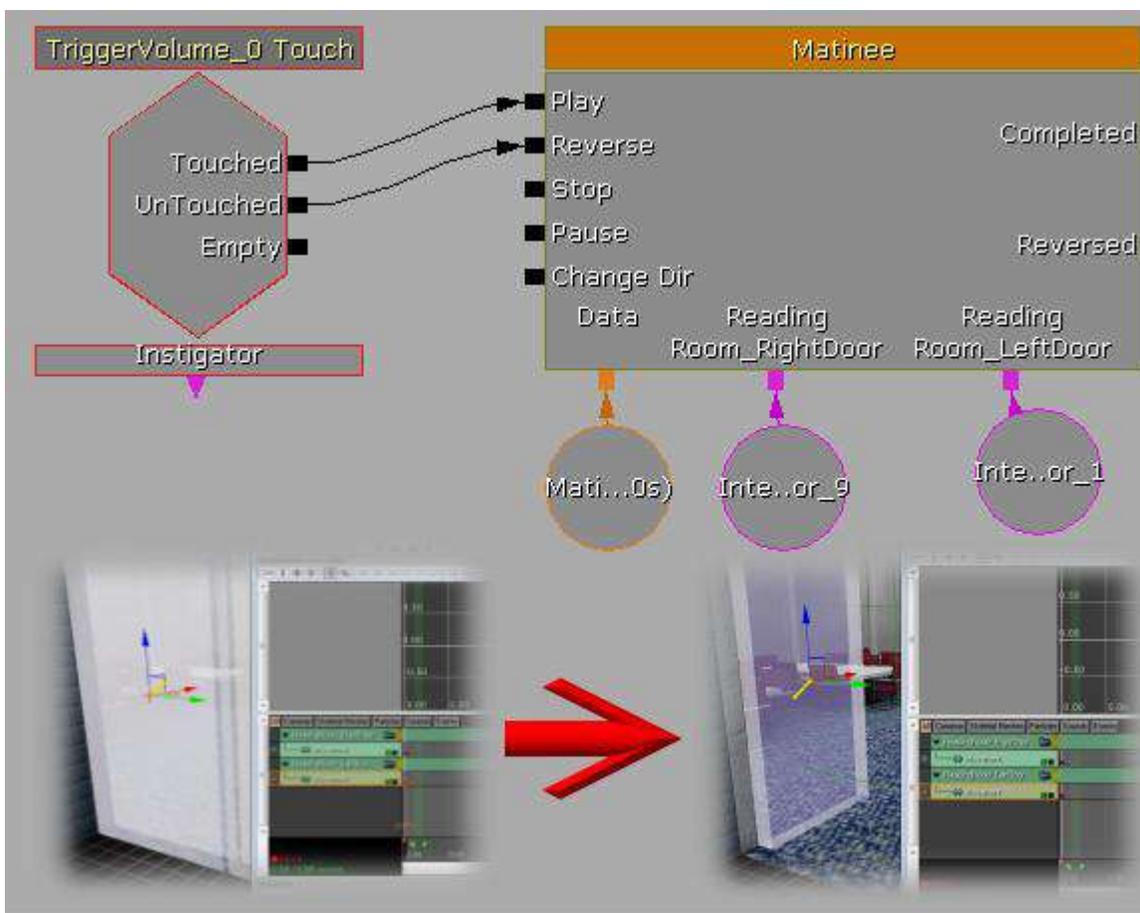


Figure 4-18 Kismet setup (Top) and keyframes setup in Matinee (Bottom)

In Figure 4.3.3-2, we aim to open the doors when the player approaches them. To do it, we first place a trigger volume (i.e. to define a space which can trigger an event), then set in the Kismet that when the player touches that volume, it will play a Matinee animation to open the doors; and the doors close when the player leaves the volume. An important point to note is that the collision detection

follows the objects, so when the doors are opened, the collision is still correct, so the players are not blocked by the “old” doors’ rigid body when the doors are opened.

Kismet and Matinee seem to be very powerful alone, so...

Can we simply rely on Kismet alone?

Imagine a line of codes you write in C or Java represents a single rectangular block inside the Kismet, how many blocks will there be? With only dozens of components, you can already see how confusing it is to connect one component to another one.

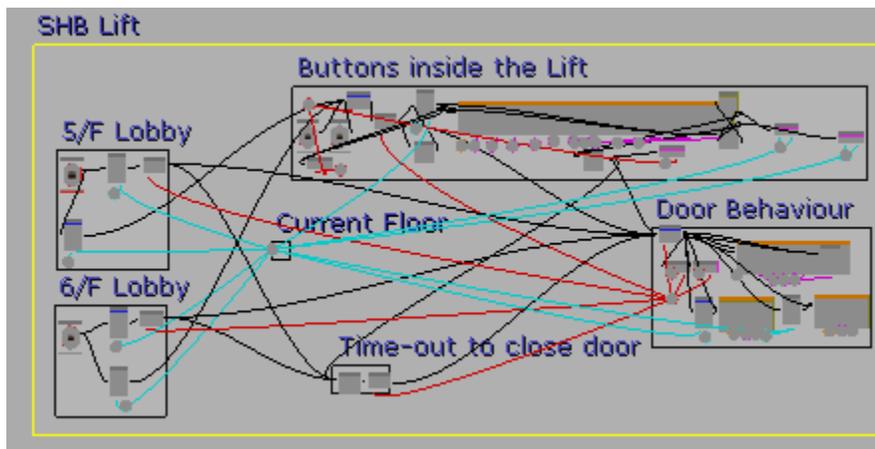


Figure 4-19 Confusing interface in Kismet just to control a lift travelling between 2 floors.

Although the two systems control a lot of things in the game world, they cannot control “everything”. For example, we cannot bind keys, create customized events, control networks, and define AI behavior in the game. And most importantly, there is no default component for us to connect to the Mindset!

Moreover, the Kismet data is not transferrable from one map (a .udk file) to another map, so if there is a feature which is to be shared among all the maps in a game, Kismet is really not a good way to do the programming.

Lastly, there is also a scalability problem if we only rely on Kismet. Imagine that we want to place 100 different spawning points on a map, then design the scripting / programming such that each spawning points could be used to spawn out an enemy robots (i.e. “UTBot” controlled by the AI), we can do this entirely in Kismet. We can call up 100 different “Actor Factory” which is a default module of Kismet, with each of them being set a different “Spawn Point” but the same spawning object (i.e. the robot). However, placing the 100 “Actor Factory” in the Kismet window will make them very difficult to manage, which is an easily imaginable problem. Even worse, if we want to change the number of spawning points, we have to create another batch of “Actor Factory”, so we understand that using Kismet can cause a scalability problem. However, there is another module called “Console Command”. The console command module is a component which calls commands with behaviors defined using UnrealScript labelled the “exec function” calls. In this case, we can simply call up only 1 “Console Command” module in Kismet and set the command to “Addbots 100” to achieve the same purpose as placing 100 default modules! This demonstrated the scalability power of combining UnrealScript with Kismet and the “highly customizable” feature of UnrealScript.

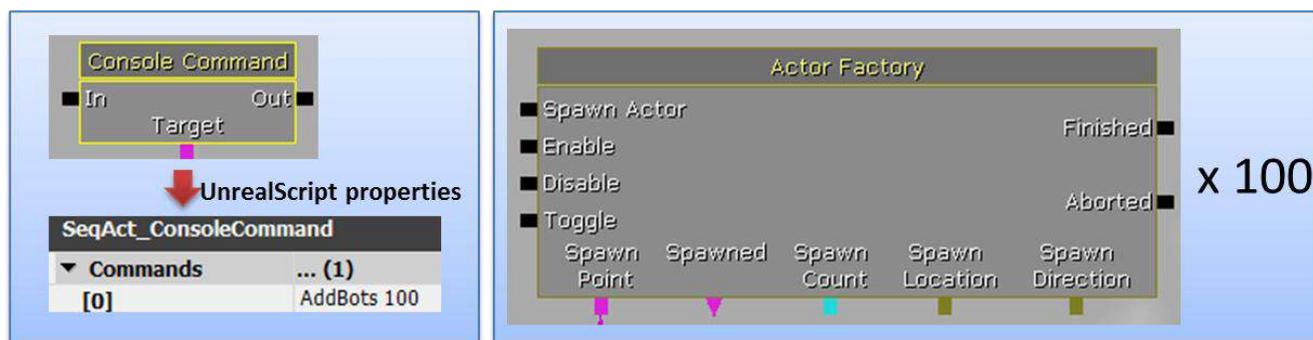


Figure 4-20 UnrealScript is highly scalable - 1 customized UnrealScript module (Left) can be as powerful as 100 or even more repeating "default" Kismet modules (Right)

As a conclusion, we cannot entirely rely on Kismet, while combining UnrealScript with Kismet seems to be a good solution.

Therefore, we shall proceed to learn UnrealScript.

4.4. Integration of UDK and Mindset

4.4.1. Introduction to UnrealScript

UnrealScript is an object-oriented programming language very similar to Java, in a sense that programs are written as classes and compiled into byte-codes (*.u files), and also multiple Inheritance is not allowed. But while operator overloading is strongly supported (e.g. string “+” integer), method overriding are not, with exception of methods with optional parameters.



Figure 4-21 UnrealScript byte-codes

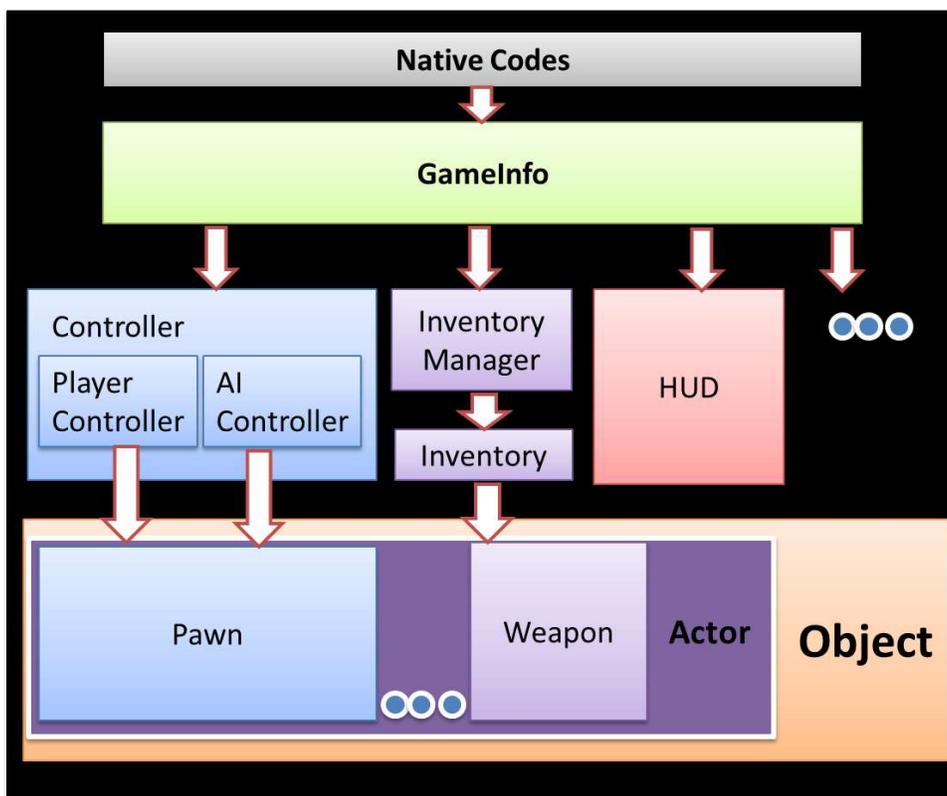


Figure 4-22 Simplified Abstract view of classes of the scripting engine (e.g. many classes are actually actors, including GameInfo itself)

The Unreal Engine first loads and execute native codes which mainly loads the maps, then almost all of the remaining work, such as deciding which script files to load, are left to “GameInfo” (which is also a script file).

Controllers, including PlayerController and AIController, are invisible entities used to define how to control pawns, which are moving actors representing the players or enemies such as UTBot (e.g. Robotic enemies in Unreal Tournament 3).

Inventory of each pawn is governed by InventoryManager, and usually composed of at least 1 weapon (e.g. a link gun).

HUD in the figure stands for “Head Up Display”. This class manages how the player reads important information from the screen during gameplay. For example, in UTHUD (default HUD for the game Unreal Tournament 3), the Hitpoints of the player and Ammunition left for the active weapon are shown.

Besides, of course there are many more classes which spin about GameInfo. However, there are **over 2350** different classes packaged in UDK (September 2010 Edition), we do not intend to explain them one-by-one.

Although there are many classes out there, development is still easy thanks to a Microsoft Visual Studio Plugin called nFringe, which modifies Visual Studio’s IntelliSense to pre-parse symbols in the classes, so that “Auto-complete” is enabled to find a particular class, method or variable name, and at the same time, enables syntax highlighting for UnrealScript. (PixelMineGames, 2010) Despite the fact that nFringe normally costs up to \$10000 for each project developed using it, we are able to use it for free due to non-commercial nature of our project.

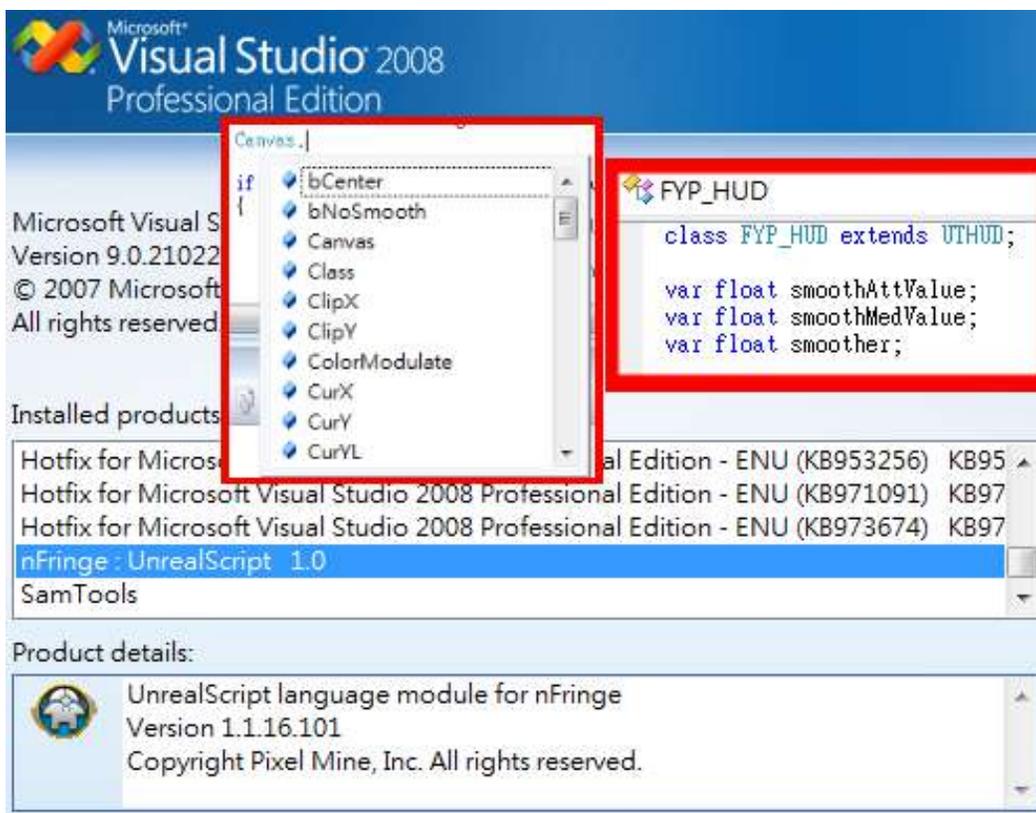


Figure 4-23 nFringe enables syntax highlighting and Auto-complete for UnrealScript

Of course, there is still one more important feature needed – Compilation of UnrealScript source codes (*.uc) into byte-codes (*.u)! With nFringe, we can set the action done by the compile button, and run the UDK compiler on the source codes instead.

On a side note, we also found out a software called UnCodeX, which lists out all the classes available as a tree, but we think the auto-complete feature in nFringe is already enough.

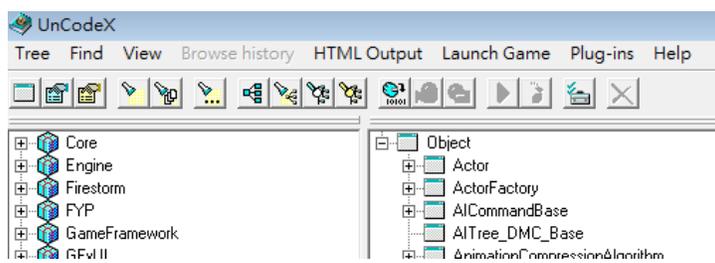


Figure 4-24 UnCodeX lists all classes out as trees.

4.4.2. Communication between UDK and Mindset

Game engines do not usually support a hacking of I/O (except key binding), but Unreal Engine 3 supports a dynamic-link library (DLL) binding (Porter, 2010), which supposedly allows us to connect it to the Mindset in our own way.

However, the use of DLL libraries may be too dependent on the platforms. By **exhausting** the UDK documentations, we discovered the possibility to control of TCP connections using TCPLink (codes hidden as native codes), and here is how we are going to solve the Mindset connection problem with advantages over DLLs.

We shall make use of TCP connections. Firstly, since not all personal computers nowadays could offer a high clock rate of CPU or GPU for the complicated rendering done in modern 3D games, additional resources consumption by the Mindset may not be desired and some computers simply cannot afford it. By making use of the TCP connections, one low computational power computer could be used to connect to the Mindset, and act as a TCP server, sending the BCI related data through a TCP socket to the computer running the Unreal Engine 3 upon request. This model also enables the ability to decouple the decision process (e.g. whether this level of Attention should trigger any event) and let the TCP server shares the burden of the client, leaving more CPU power for game logics processing.

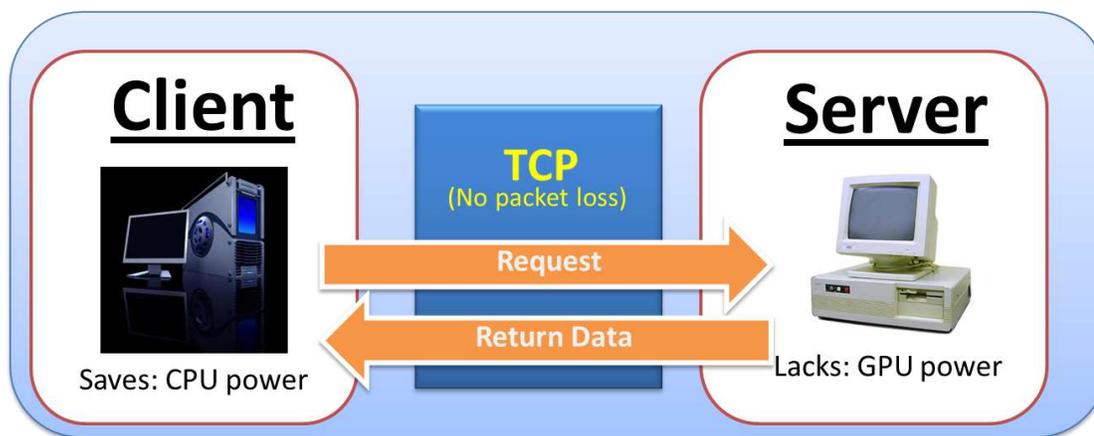


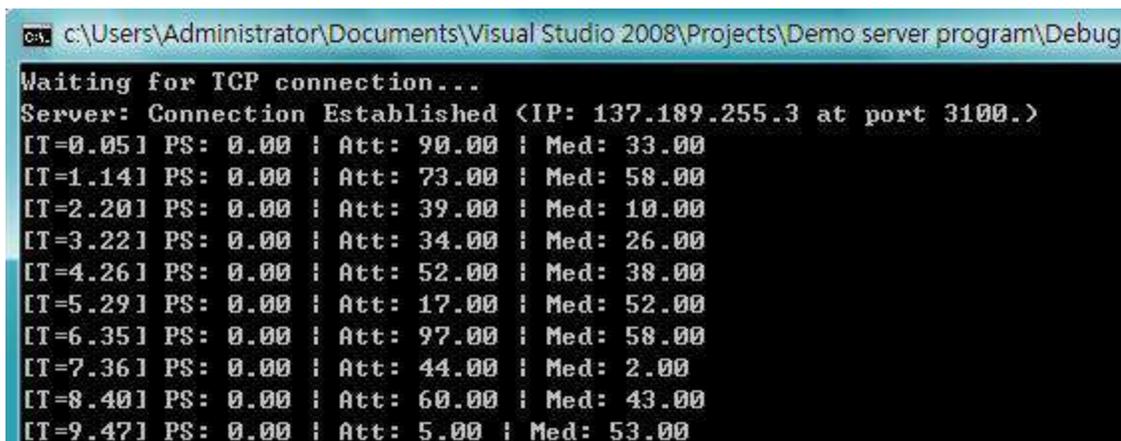
Figure 4-25 TCP Model – with the server connecting to Mindset

Moreover, the server-client model may also allow a hub of Mindset data, and may be used to trigger events in multiple game clients or simply allow more gameplay features (e.g. one player may have a skill to detect a drop of another player's Attention level and back-stab him in the game).

So is it bad if the player only got 1 computer at home? Don't worry, we thought about that too! If the player has only 1 computer, the set up can still run smoothly, the only modification is set "localhost" as the server address to be resolved in the client. In addition, the employment of TCP model reduces the machine dependency of the BCI game. While DLLs are operating system specific (e.g. Windows), TCPLink is primitive to UnrealScript and supports multiple OS.

Some people may also worry about data loss during network transmission, especially when the data flows between different computers. This is usually the issues with network packets sent using the User Datagram Protocol (UDP) in which the sender of the packets will never request to see if the receiver actually got it and the receiver will never respond upon reception of the packet. If packet loss occurs, there will not be any retransmission of the lost packet from the sender. (Reynders & Wright, 2003) As a result, there will be at least 2 second window-period in which the attention and meditation meters remain static and not updated, thus affecting the gameplay (e.g. when the player needs to open a locked door to escape from a horde of zombies using his/her attention level, the multi-seconds delay could lead to death of the playable character and annoy the player). However, in the case of TCP (Transmission Control Protocol), the kernel of the operating system will trace to see if the packet is sent to receiver successfully. When a TCP packet is lost, the send will perform a retransmission until the receiver confirms its reception. In this way, the BCI data will never be really lost. (Stevens & Wright, 1994) Even though retransmission takes time, but if the connection is established through a local area network (LAN), the time for retransmission will be negligible relative to the 1 second delay in running Mindset API, so it will not affect the gameplay.

For demonstration purposes, we wrote a C++ program which uses Mindset API (ThinkGear API) to mine data from the Mindset, and establish a listening socket using Winsock as the program runs in Windows (we can switch it to Linux by including a different header file).



```
cs. c:\Users\Administrator\Documents\Visual Studio 2008\Projects\Demo server program\Debug
Waiting for TCP connection...
Server: Connection Established <IP: 137.189.255.3 at port 3100.>
[T=0.051 PS: 0.00 | Att: 90.00 | Med: 33.00
[T=1.141 PS: 0.00 | Att: 73.00 | Med: 58.00
[T=2.201 PS: 0.00 | Att: 39.00 | Med: 10.00
[T=3.221 PS: 0.00 | Att: 34.00 | Med: 26.00
[T=4.261 PS: 0.00 | Att: 52.00 | Med: 38.00
[T=5.291 PS: 0.00 | Att: 17.00 | Med: 52.00
[T=6.351 PS: 0.00 | Att: 97.00 | Med: 58.00
[T=7.361 PS: 0.00 | Att: 44.00 | Med: 2.00
[T=8.401 PS: 0.00 | Att: 60.00 | Med: 43.00
[T=9.471 PS: 0.00 | Att: 5.00 | Med: 53.00
```

Figure 4-26 Our C++ based Mindset-TCP server program which serves the TCP client written in UnrealScript

On the other hand, we programmed a Mindset data streamer in UnrealScript. We first force the “GameInfo” to create a TCP client object after loading the map, and we changed the properties of the TCP client such that for every 1 second (similar to the delay between data are fetch from the Mindset using Mindset API) in the game, it will send a TCP packet to the server program. When the program receives a request, it will reply the same socket with Mindset data (e.g. Attention, Meditation and Poor Signal) in our designed format. The TCP client object will then modify the variables we defined in “GameInfo”, which registers the respective Mindset data in UDK’s data structure.

In this way, the Mindset data can stream into the UDK.

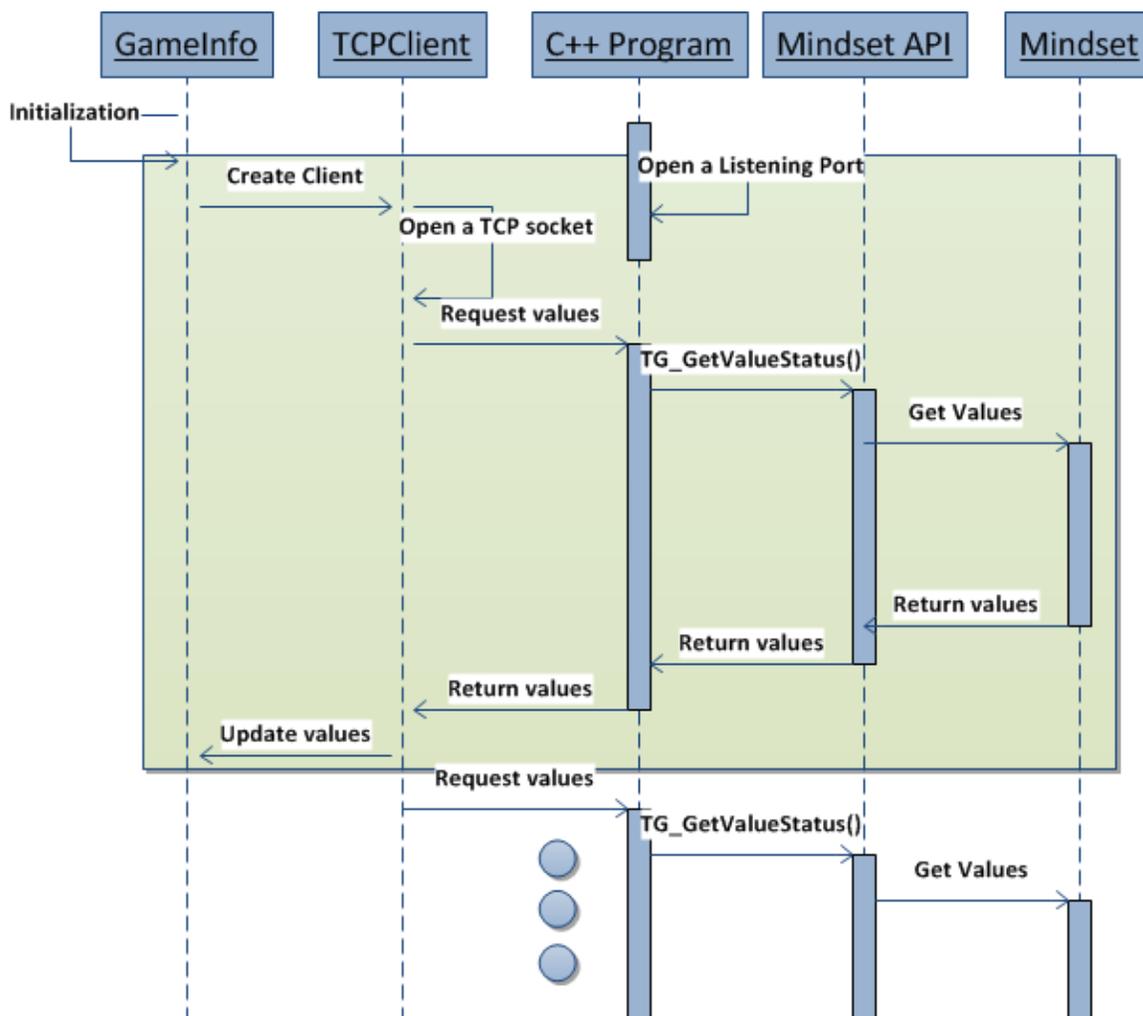


Figure 4-27 Sequence Diagram of BCI values update

4.4.3. Ways of BCI controls

As lessons learnt from previous chapters, we understand that it is difficult to use Mindset as a means of direct control of the player (e.g. Bi-directional movement).

However, utilizing specific brain waves patterns, or simply using the level of Attention or Meditation states provided by eSenses, can trigger certain events within a certain pre-defined in-game areas; we define it as “Active Control”.

	BCI Features	Detailed Example
Active Control	Healing	The player's Hitpoint regenerates over time if the meditation level is over 50%
	Moving object linearly	Within a locus on a line, the attention level directly changes the object's position on that line
	Opening Doors	Opening locked door slowly when attention level is across 50%, and the door will close if the attention level declines
	Passing through walls	Walk through walls with a high meditation level, with collision being turned off for the wall or the player.
	Removing Boss Shield	Weaken the armor of monsters with strength proportional to the attention level
	Shooting enemies	Rockets are created to shoot an originally undefeatable boss if the attention level is over 50%
	Simulating rotation	When attention level is over 50%, the object rotates in clockwise direction. The rotation stops if it falls under 50%
	Spawning AI bots	Helper robots are built using scrap metals and "mind power" (i.e. attention level over 60%)
	Toggling Lights	Turn on lights in a dark area when the meditation is lower than 50%; or brightness varies directly with the meter
	Visualizing hidden objects	Can turn off "hidden" property of bonus weapons or ghosts when meditation level is over 70%

Table 4.4-1 Some of our ideas of Active Controls

On the other hand, we thought of consolidating BCI data for a regular period of time, say, an timeframe of 20 seconds, to recognize whether the player is under pre-defined mental states (e.g. fear) which take time to form and be recognized, or to give us extra information, so as to alter some of the background information or events. We shall call this “Passive Controls”. (If it is too hard to be understood, please read the chapter “Future Work”) This kind of control are much

more difficult as relatively more samples need to be processed, and sometimes a pattern should be approximately matched first, therefore we do not plan to demonstrate that out in this phase.

4.4.4. Demonstrating Active Control

In session 4.4.2, we demonstrated how the Mindset data can enter the UDK using TCP connections. And the values are stored in the “GameInfo”. Here we will have a quick look at how we managed to demonstrate the possibility of active control in UDK using the BCI.

There are a few components we need to make.

Firstly we have to build components which tell the game whether the player has a high attention or meditation values, where we can use that component in Kismet.

Moreover, we have to let the player senses that he/she is actually playing a game connected to the Mindset, therefore we have to hack the HUD to show the current states of the player. However, since the Mindset API only updates data every second (i.e. a limitation), the player may not want to see big jumps in the BCI values every now and then. Therefore we explicitly installed codes to smooth the values into a curve by drawing the values (we call these values the “smoothValues” in the script) closer to the latest BCI values in every “tick” of the game.

$$S_t = A_{t-1} + \frac{A_t - A_{t-1}}{T}$$

(where $S_t, S_{t-1}, A_t, A_{t-1}, T \in R$)

Figure 4-28 Smoothing method programmed inside our HUD class. (S: smoothValue, A: Attention, t: time, T: Number of ticks per second.)

```
function DrawGameHud()
{
    super.DrawGameHud(); //Draw HP and Ammo bars

    if ( (Player Is Alive) && (He Is Not In "Spectating" State) )
    {
        Set smoothValues nearer to latest BCI values;
        Draw Attention Bar;
        Draw Meditation Bar;
    }
}
```

Figure 4-29 Pseudo-codes of the drawing function inside our customized HUD class

Combining the component we wrote for attention detection (i.e. the “has Attention” module which returns true if the attention level exceeds 50%), we can use the following Kismet set up to raise the New Asia College Water Tower (君子塔) from the ground using our mind alone!

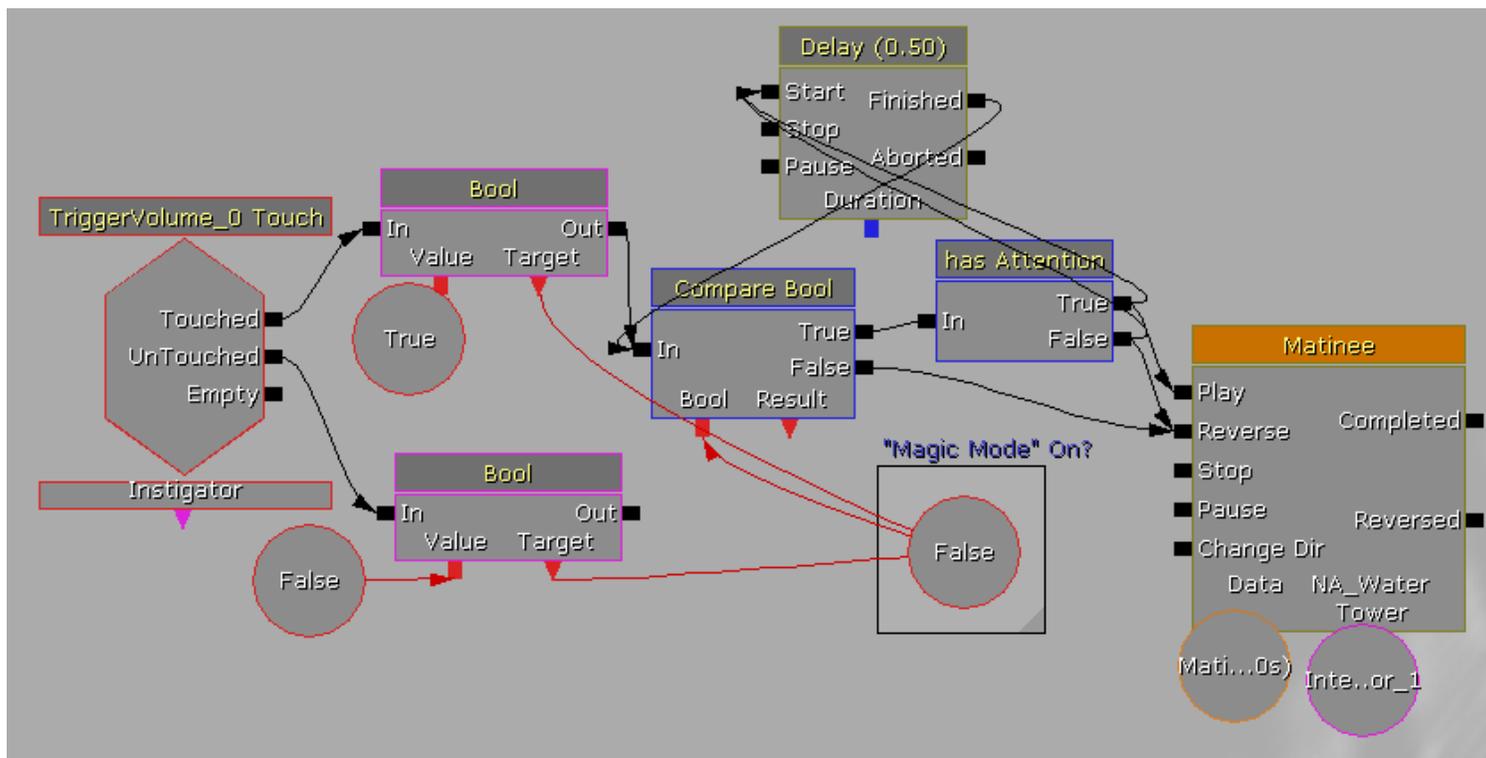


Figure 4-30 Kismet sequence for an active BCI control

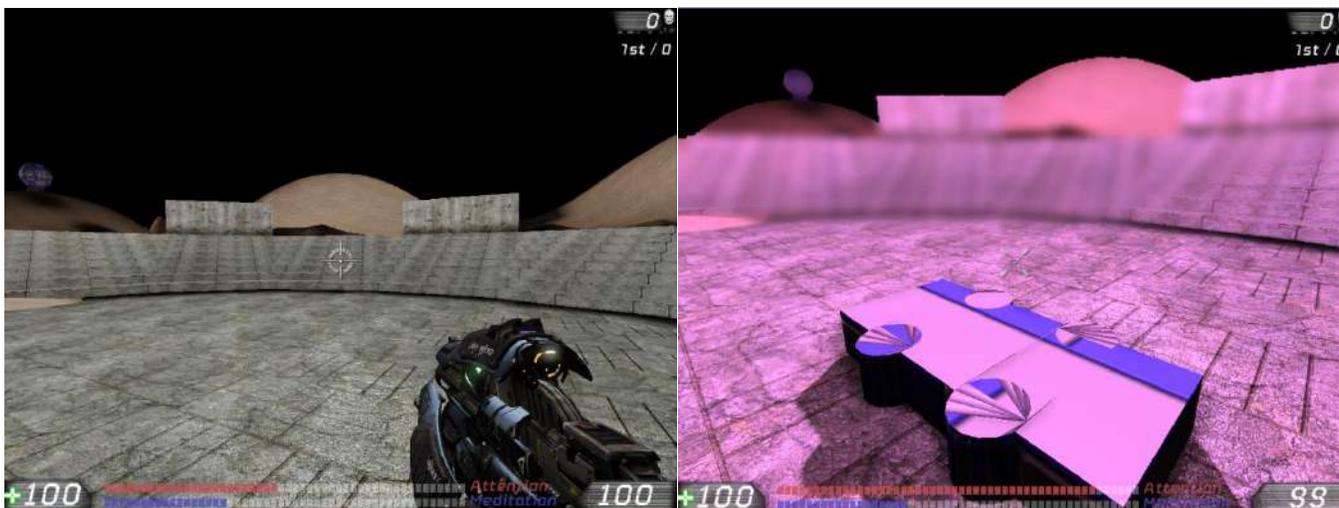


Figure 4-31 Effects of magic mode being turned off (Left) and on (Right) due to current attention level (Realtime display of attention and meditation are shown by red and blue bars at the bottom respectively)

When the player steps into a pre-defined space volume (which is a post-processing trigger volume AND an event trigger volume) placed at the center of the New Asia College Round Centre, the “Magic Mode” will be turned on, and if the attention level of the player is high enough, our module “has Attention” will trigger the raising effect of the water tower. If the player steps outside the trigger volume (i.e. not close to the tower), or the attention level falls below 50%, our module will let the tower fall down, so the player will feel the urge to focus so as to bring the tower up again!

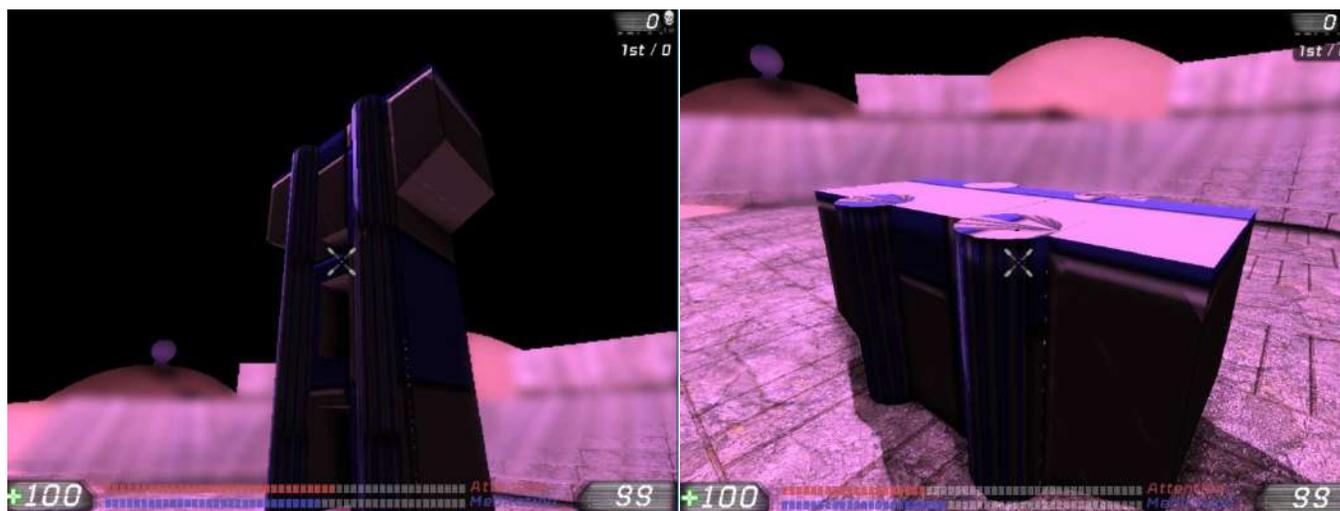


Figure 4-32 Tower being raised high (Left) and the effect of declined attention level (Right)

5. Advanced BCI Game Development

5.1. Introduction

The previous chapter shed some light on how the Unreal Engine 3 can communicate with the Neurosky mindset API and talked about the essential basis of the ways to utilize the Unreal Development Kit, especially the editor. Although a few demos are created, they are only segments which are not linked up together, thus the argument that “BCI-enhanced game development is feasible” is not too strong so far.

In this chapter, we shall move on to develop a first-person adventure game called “Psionscape”. The game we develop will be a very short one but it is solely because we have very limited human resources. Of course, in today’s world, modern game development teams for even small companies usually consists of around 100 members, (Novak & Moore, 2010) , so the BCI-enhanced game can be easily scaled up.

5.2. Game Plan

5.2.1. Main Idea

The game title is called “Psionscape”. It actually comes from the words “Psionics” and “Escape”.

Psionics usually refers to the ability to use the human mind to induce paranormal phenomena; this normally includes telekinesis, in which objects are manipulated remotely using only the mind. In broader term, this is quite similar to “magic” and “witchcraft”, but they require usually not just the mind, but fictional things like “Mana”, “Magic Wand” or “Potions”, to induce the different phenomena. Nevertheless, we use the term Psionics here, because players in Psionscape can take control of a variety of “magical” events by controlling solely their brain activity as

reflected by the mental states detected using the customized algorithm we detected.

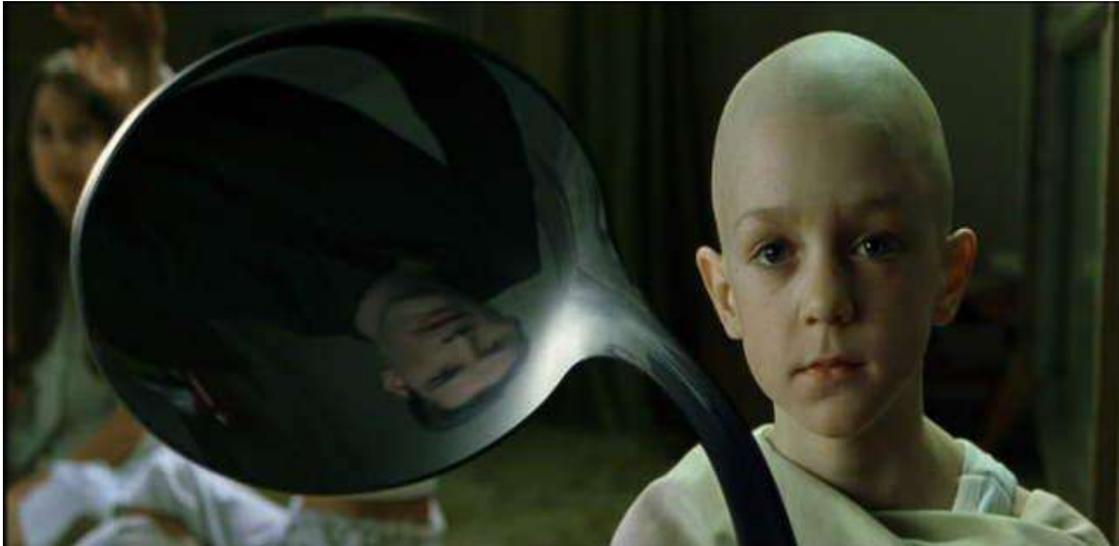


Figure 5-1 Spoon-bending, a famous example of telekinesis demonstrated in The Matrix (psionics usage example)

As an overview, we shall introduce a variety of in-game psionic abilities. Firstly the player can use their “Mind Power” (to be discussed later) to repair a broken teleporter (a machine which can transport the player across remote distances) by maintaining a high Mind Power. Secondly, the player can block hostile projectiles such as rockets and bullets by raising a shield which entirely depends on how well the player can maintain the Mind Power level. Finally, to be an interesting adventure game, “platformer” (i.e. involving jumping between platforms) elements shall be introduced with the psionic ability, such as converting debris into an elevator so as to transport the player to a destination.

“Escape” will be the main plot of the game. The player is required to escape the main building inside the game, yet like most of the games in the consumer market, the path to the goal is never easy.

Examples of the game elements include the followings. Firstly, the game map is not a regular cube or flat land, it will instead consist of

different paths, where some of the paths are broken and impassible. The story shall help facilitates the rationality of broken paths and this forces the players to solve the puzzles we introduced to the game so as to proceed. Moreover, the puzzles are mostly related to the use of Mind Power to make this game special, although we still have to leave some of the puzzles solvable without using the Mind Power (which shall be easier) to leave time for the beginner players to rest between Psionic tasks. Last but not least, some typical exciting game elements such as first-person shooter, which requires the player to destroy the hostile robot in order to clear a path to reach a key.

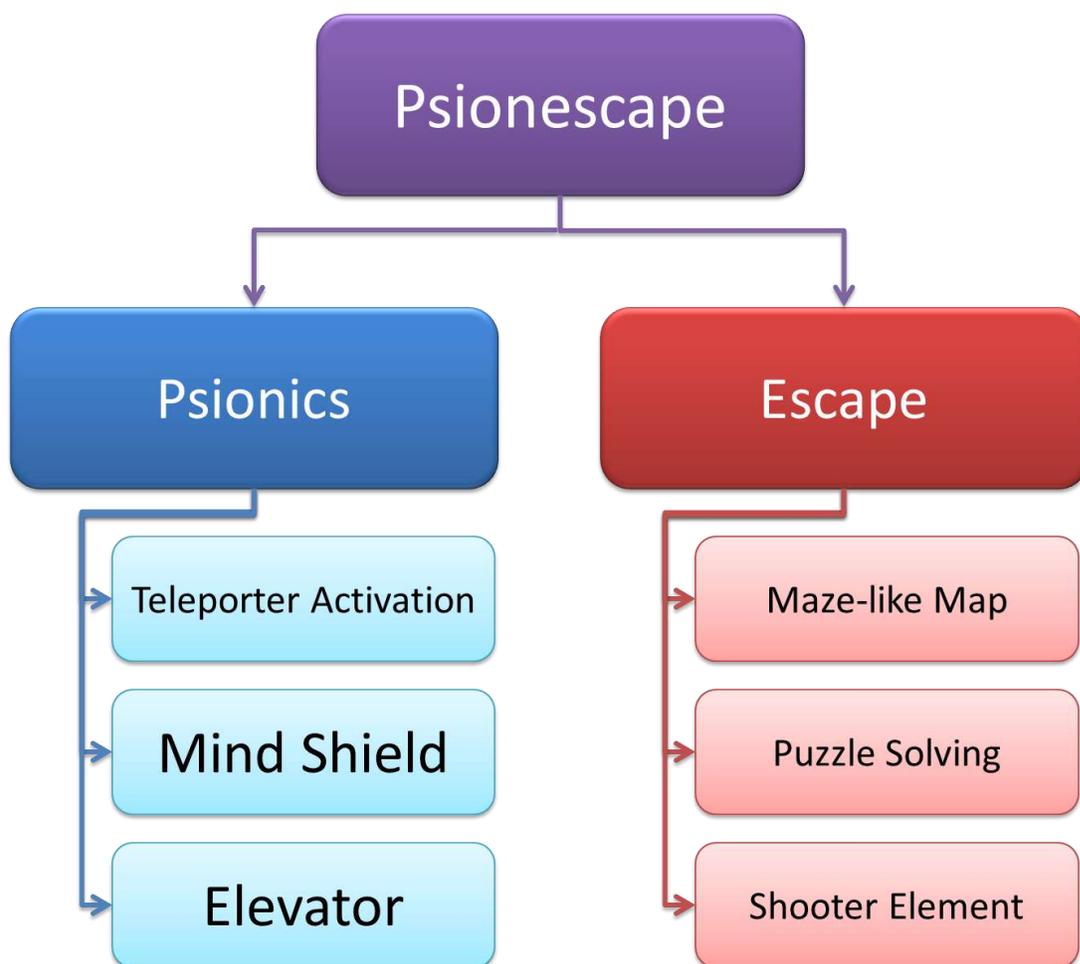


Figure 5-2 Overview of the game elements

When you first run the game, you will notice there are different logos shown in the splash screen.



Figure 5-3 Splash Screen of Psionescape

In the bottom-left hand corner, there are logos for Unreal, Scaleform, and Fonix, where Scaleform enables in-game flash support (mainly for interactive interface designs) and Fonix enables text-to-speech support in real-time, and both of them are the main middleware we use in the game (we shall discuss them later).

The main logo for the game is basically that we use for fire exit in daily life, so players will be very familiar with the game's main idea when they see the logo.

5.2.2. Game Plot

Every game must have a game plot and Psionescape is not an exception.

It is now year 2036. The protagonist of the game is the world's first intelligent robot with psionic powers, who has just been successfully

developed by the Department of Computer Science and Engineering of The Chinese University of Hong Kong.

Originally this robot development project is carried out secretly, but no matter how secret this can be, a technologically hostile country has sent special agents to either retrieve or destroy the robot.

The game starts with the robot just awoken from its development chamber, not knowing anything. However, the on-site Artificial Intelligence system (named Edward) installed in the Ho Sin-Hang Engineering Building (SHB) briefs the protagonist that the special agents have destroyed most part of the SHB just to search for him and that he has to escape the building to not get destroyed and sent for hostile “dissection”.

With the help of Edward and the protagonist’s psionic power, he managed to escape the building at the end.

5.3. WalkThrough

The game features a partial replication of the 5/F, 6/F, and 9/F of the Ho Sin-Hang Engineering Building of CUHK.

5.3.1.SHB 9/F

(The following figure is a labeled floor plan of this game map which can be followed during the reading of this sub-chapter.)

The game begins in a normally inaccessible room (1) located inside Room904, which is the common laboratory for Faculty of Engineering students and staff members.

After a briefing from the AI system through telecommunication, the player shall know about the basic controls of the game, such as how to open a door and interact with objects.

The player then enters room 904, only to find that the door at (3) is locked. Therefore the player must explore the whole Room904 to find and interact with the switch located at (2).

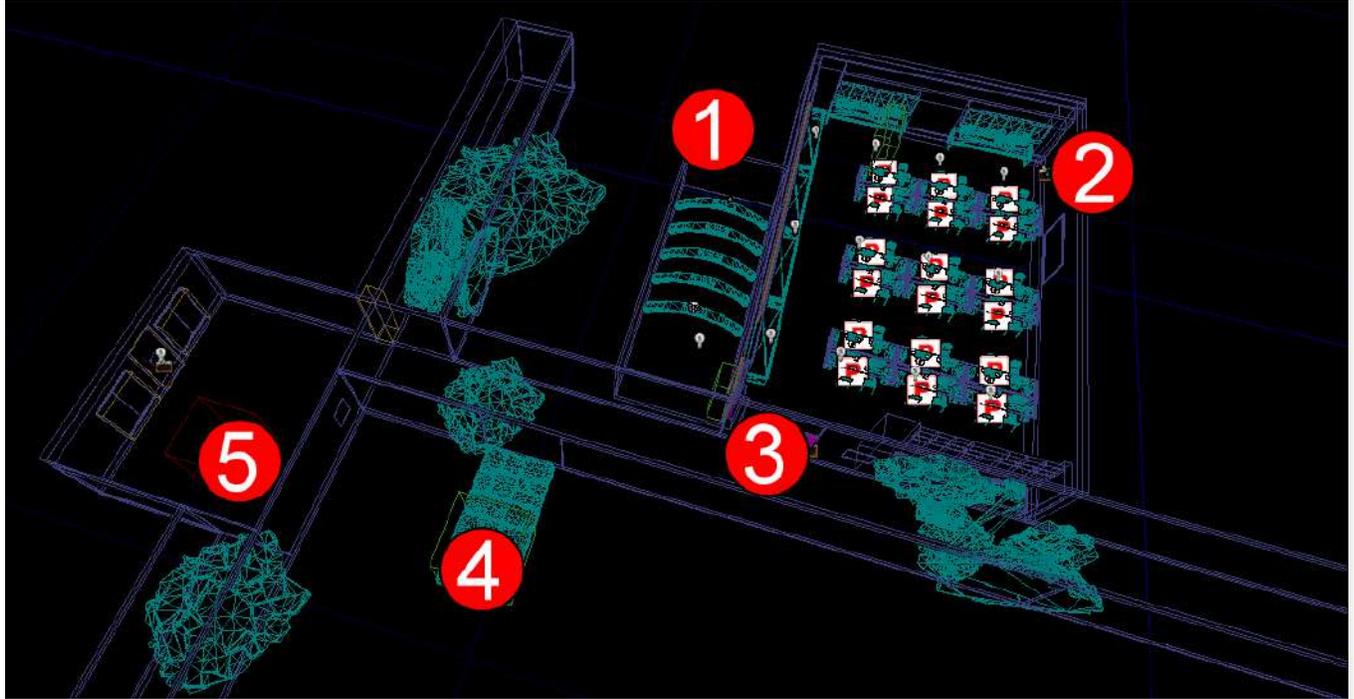


Figure 5-4 Labeled 3D Floor Plan for the in-game SHB 9/F

However, exiting through (3) is just the first step. Although the player may see that there're lifts at the far side of the corridor at (5), there are many rubble resulted from a huge explosion (caused by the special agents) scattering all over the place and blocked the way to the lift.

The player then walks around to find a broken teleporter at (4), at this point, there would be an option to attempt to repair the teleporter, but a high level of Mind Power must be maintained.

Upon successful repair of the teleporter, the player can travel to (5) and reach the lift lobby. Taking the lift down can bring the player to the next game map.

5.3.2.SHB 5/F and 6/F

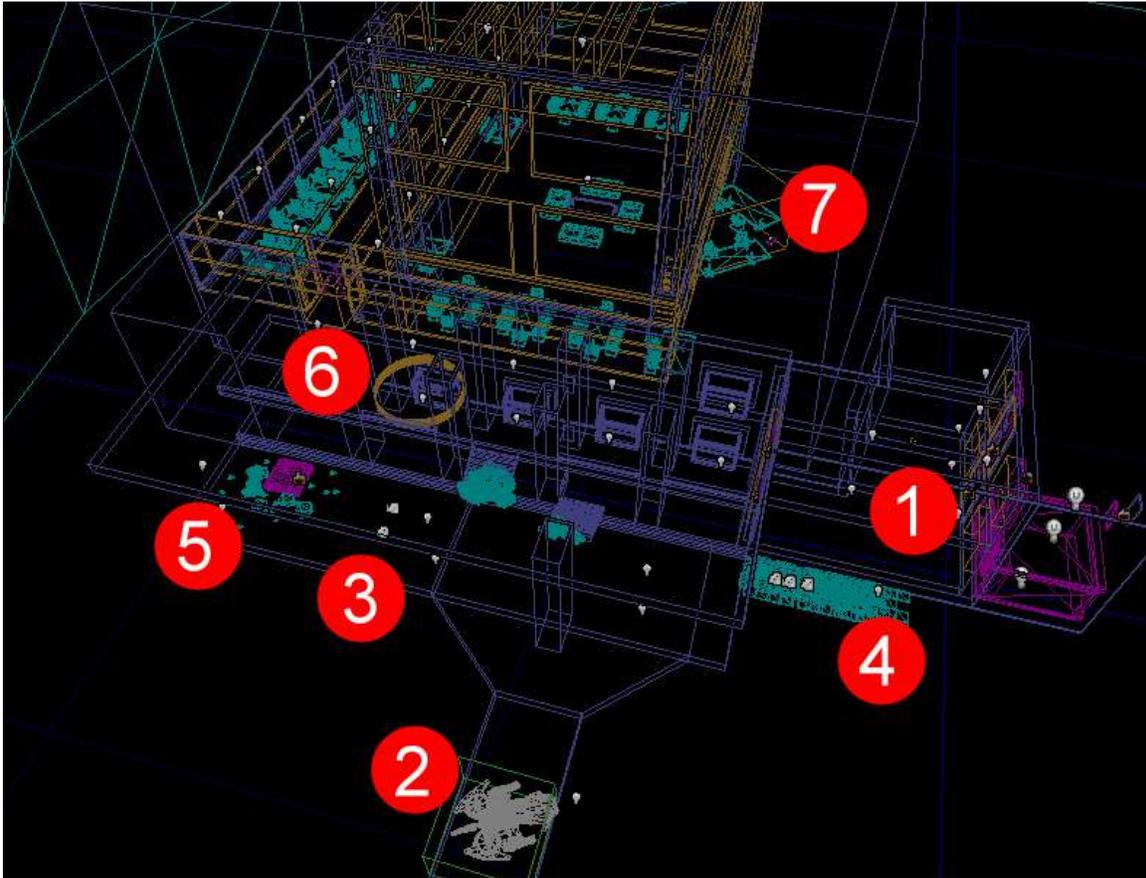


Figure 5-5 Labeled 3D Floor Plan for SHB 5/F and 6/F

Following the above 3D floor plan, the player on this game map will continue at point (1) on 5/F. Immediately, he can choose to take the lift to the 6/F. However, the path is broken at point 6 and he cannot travel to point (7) normally.

The player can also walk to point (2) to investigate an aircraft which continuously shoots out bullets towards point (4). After the investigation, the player will know that the aircraft's weaponing system is malfunctioning so that the bullets are pouring out. Moreover, the aircraft is locked so that the player cannot escape the building with it yet.

In search for the key, the player will approach point (3), however, there is a special agent (robot guard) there which patrols around. If the player gets too close, the guard will shoot the player with rockets.

Therefore, the player must find a way to eliminate the guard, and there are hints that a shock rifle is located at point (4).

In order to avoid getting killed by the bullets fired from the aircraft, the player can create a shield which can block any projectiles. Yet the “Mind Shield” requires the player to maintain a high level of Mind Power.

After killing the robot with the newly acquired rifle, the player can proceed to point (5), which is an area consisting of more regular rubble. Here, the player can investigate the pile of rubble to find a flat platform, which can be animated as an elevator using the mind power. This part of the game is tricky, because the player must first lower the platform by lowering the mind power, such that he can jump on top of the platform. Afterwards, the mind power must be raised to almost maximum to allow the player to jump onto 6/F.

With a thorough search inside the reading room on 6/F, the player will find the key at point (7), and bringing the key to the aircraft will enable the player to unlock the aircraft and successfully escape the building, essentially winning the game.

5.4. BCI Integration

5.4.1. Launcher Program

Let us have a quick revision on how we make the connection between the BCI and the game engine.

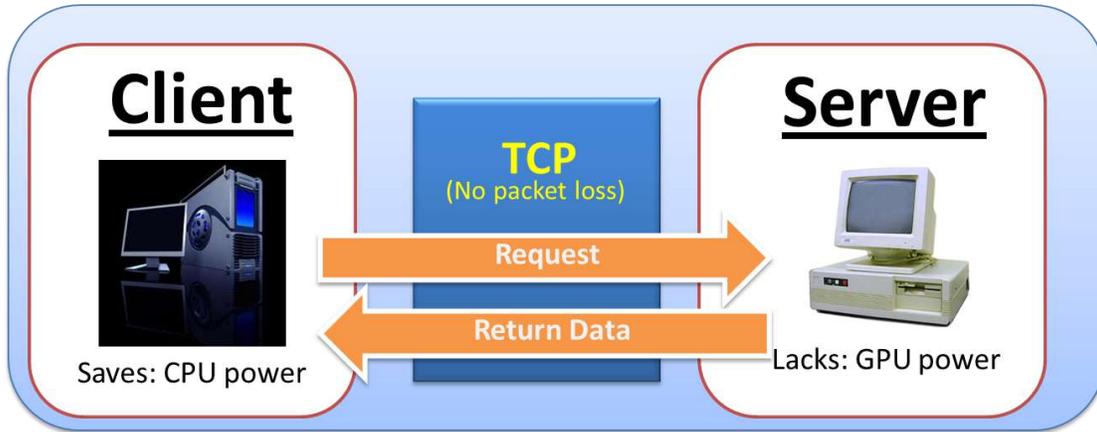


Figure 5-6 Generic Model for BCI-Game-Engine Integration

Although this model is very friendly to allow connections between 2 computers and save respective resources, and also enables the possibility of a networked game, since we are trying to make a single-player game, it is very likely that the player would like to run the whole package in a single computer.

A problem preventing us from making a workable game using solely the method mentioned in the previous chapter is that the TCP server program may not be started when the game starts, because the player may not have the idea to run the server program separately.

In response to this, we have developed a special game launcher program.

The launcher first fires up a profile manager which detects whether this is the first time the player plays the game. If it is the first time, it will calibrate the BCI with our algorithm for two minutes, so as to define what the average brain activity is like, then save it into a text profile. Otherwise, when the program detects that there is an existing profile text file saved previously, the program will simply load from that profile.

After that, the launcher would fork out 2 processes.

One of the processes would be the BCI server program. As soon as it starts, it will start querying the values of different types of brainwaves.

On top of that it also includes the brainwave classifier discussed in the previous chapter, and continuously calculates and stores the current mental state numbers (from 1 to 3).

With the mental state number ready, the TCP server thread is also set to listen to a specific port and ready for sending the mental state number out every second, to any specific TCP clients.

In our case, the TCP server is set to only response to TCP request containing the string “FYP” so that it is not vulnerable to uninvited parties such as hackers. In this sense, the string can also serve as a password for the BCI data TCP server.

On the other hand, the game engine is fork()-ed up to call out the splash screen and begin loading the main menu map.

Since the default game type of Unreal Engine 3 is, in fact, the actual Unreal Tournament game with the game type “UTDeathMatch”, the main menu serves as a purpose to set the game type into the one we created, which is “PsionscapeGame”.

With a console command “open SHB9F?game=PsionscapeGame” being run in the main menu’s “Play” button, the game should begin as it should be (i.e. have the correct game type), and also at the beginning of the game, the mental state can be passed correctly from the TCP server to the client.

Below is a diagram for better understanding of the structure of the launcher program.

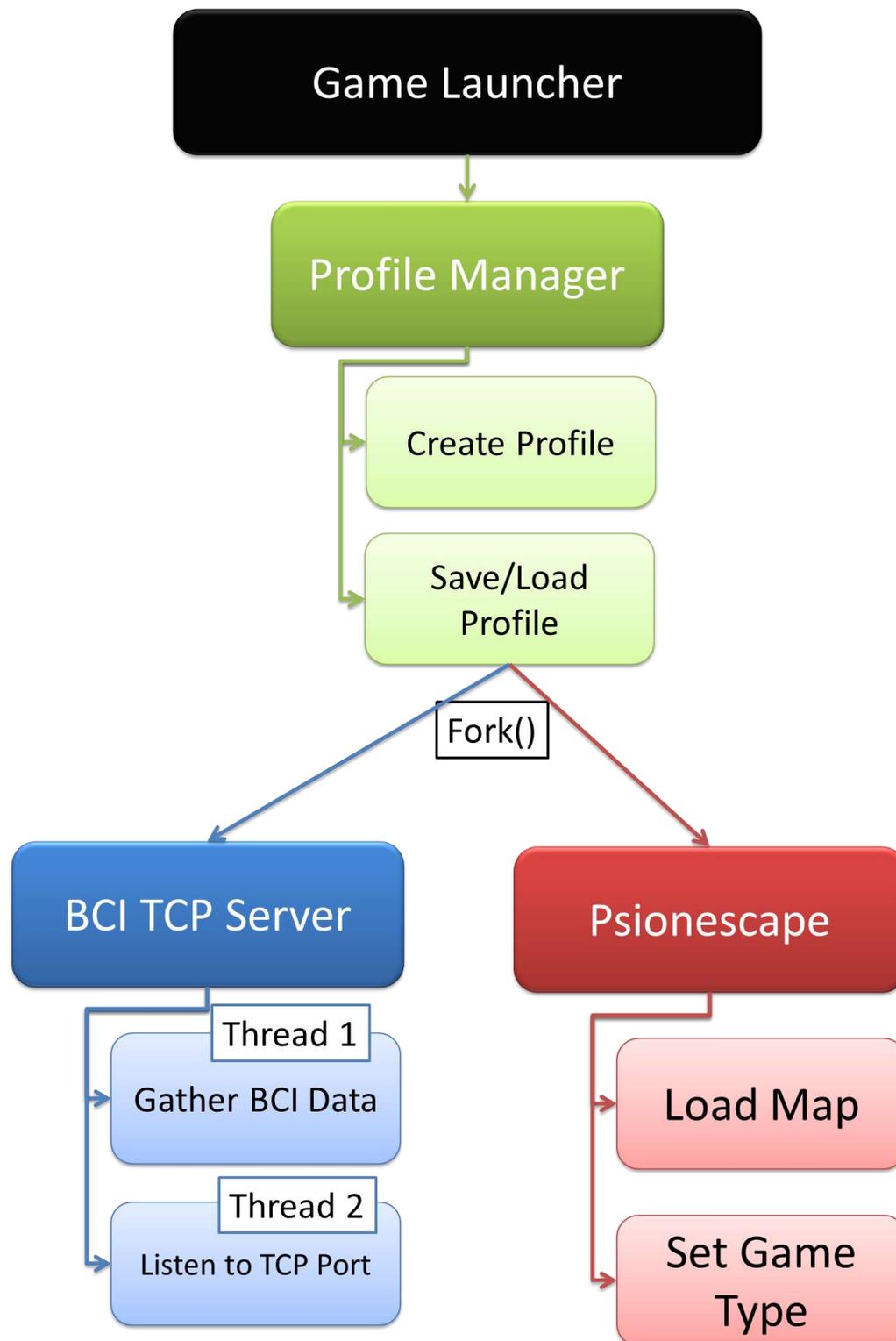


Figure 5-7 Overview of the Game Launcher

5.4.2. Use of Mental State Number

Although the customized BCI classification algorithm enables calibration and training modes, and it also enables the classification of 3 mental states, the output is, however, just a number.

Therefore, in order to use this as an interesting and visually stimulating in-game tool, a scoring system is being introduced into the game.

```
For each second passed
  oldMP ← MP
  Request Mental State Number N from BCI
  Parse the Number from the TCP Packet
  If N equals 1
    MP ← MP + 10.0
  End If
  If N equals 2
    MP ← MP + 5.0
  End If
  If N equals 3
    MP ← MP - 15.0
  End If
End For

t ← time past since last update of MP
smoothMP ← oldMP + (MP - oldMP) * t
```

Figure 5-8 Game Loop to Update the Mind Power Value with smoothing

Assuming that the three states measure the level of attention of the player, where state 1 presumably implies a high attention level, state 2 implies a normal attention level and state 3 implies a low attention level. We can make up a linear meter which has values ranging from 0 to 100 in real numbers to reflect the change and persistence of each mental state.

The game logics loop runs through itself frequently within every second, and we can modify the loop following the above pseudo-code algorithm.

The gauge we introduce is called Mind Power.

For each second passed in the game, the TCP client in the game will request a state number from the BCI server program, and when it detects the state number as 1, the Mind Power meter will increase by 10. Similarly, the Mind Power is increased by 5 for state number 2 but decreased by 15 for state number 3.

Similar to the problem we encountered in the last semester, the query can only be done every second, but not in a shorter time frame, due to the limitation of the Neurosky Mindset itself.

Hence, in order to produce a smooth transition effect between each second, the Mind Power used in-game is actually a smoothed value using the last two Mind Power values. This can be done using linear interpolation between the two values against the milliseconds passed after the last update.

On the other hand, we have to display the Mind Power gauge inside the game, such that the player can view the change, rate of change and actual value of the Mind Power, for better determination of in-game actions. For example, in order to maintain a shield to survive through a blast of bullets, the player has to maintain a very high level of Mind Power because a fall of Mind Power beyond certain level will suddenly disable

the shield, and so the best approach would be to start the shield when the Mind Power is already very high.

The implementation of the visual gauge is similar to the one we discussed in the previous chapter. We simply need to reprogram the Heads-Up-Display or HUD in short, in the game engine.



Figure 5-9 The Mind Power Gauge in the game

Finally, even though the smoothed Mind Power value is passed into the Unreal Engine 3 using UnrealScript, it is not immediately usable inside the game for events triggered using Kismet.

In order to solve this problem, we have programmed a Kismet component using UnrealScript, which is called the Mind Power Synchronizer.

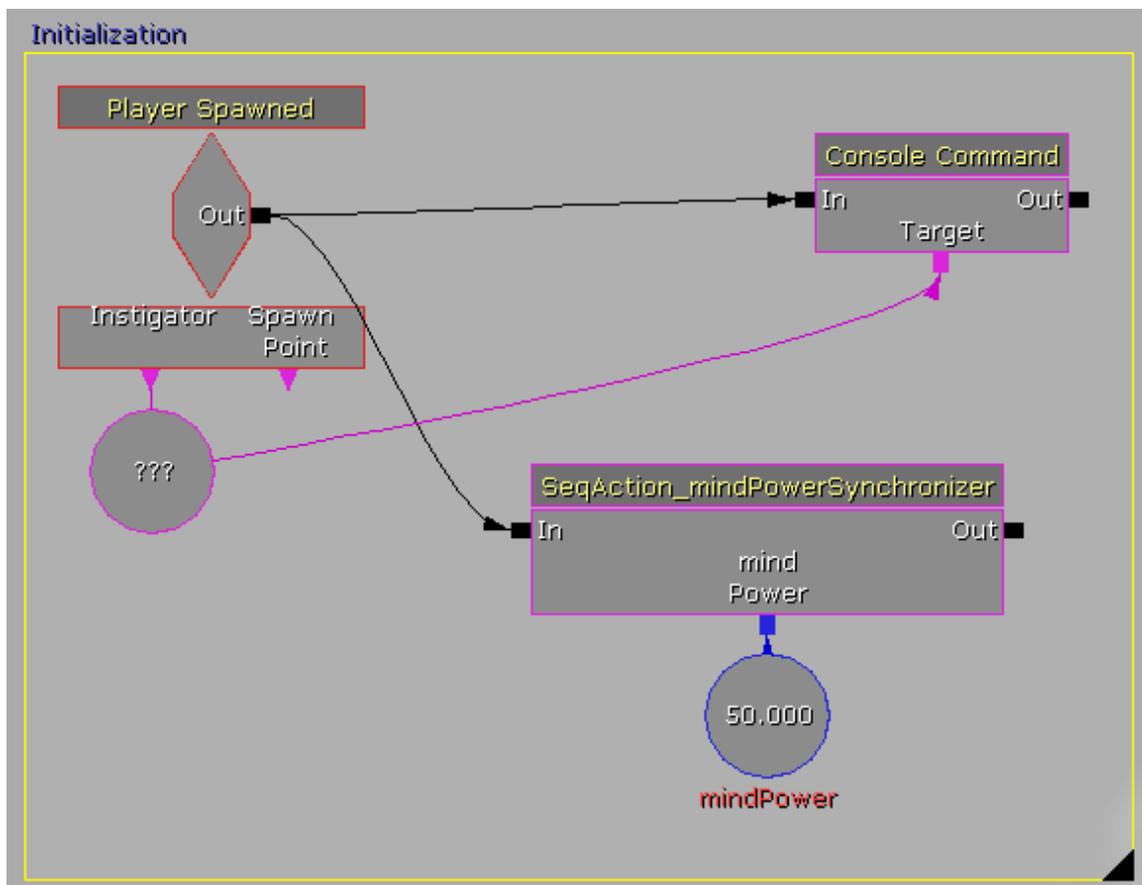


Figure 5-10 Customized Kismet Component to Synchronize the Mind Power Values

The Mind Power Synchronizer actually does not synchronize the value by itself. Instead, it provides a channel for the special TCP client running in the background to update the Mind Power value in the Kismet. Since it looks more exciting and realistic to use the smoothed Mind Power values, the Mind Power Synchronizer only allows the update using the smoothed values, but not the actual ones.

5.5. Features

5.5.1. Interactive Menu

Interactive menus are now necessary components for most of the games; they allow the interaction between the player and specific objects in the game world, by providing an interface for people to select adventure options.

In the adventure-action game series called Mass Effect, the adjustment knob like interactive menu is being introduced. With the huge success of the game, the interactive menu is surely one of the components which are widely accepted.



Figure 5-11 Interactive Dialogue in the game, Mass Effect

The interactive dialogue provides up to 6 choices regarding each scene, and a description text can be inserted on the top which does not fade away until the player selects an option, this ensures that the player has read about the text and this is much better than showing different buttons onto the screens alone.

Of course, it is not a must for us to follow the Mass Effect's interactive menu, but based on our gaming experience, it would be interesting to try to replicate the same user interface.

In order to do that in the Unreal Engine 3, we have to use the Scaleform Gfx middleware, which essentially provides a way for us to integrate Adobe Flash files into the game.

Fortunately, we do not need to program it from scratch, because one of the experts on the UnrealScript forum has done a similar work. (UDKC, 2010)

To begin with, a flash template together with a circle consists of 6 parts, a dot at the center and an arrow has to be created. A point of note though, is that the arrow is somehow grouped with the dot, because the dot provides a center of rotation for the arrow to follow the direction of mouse movement.

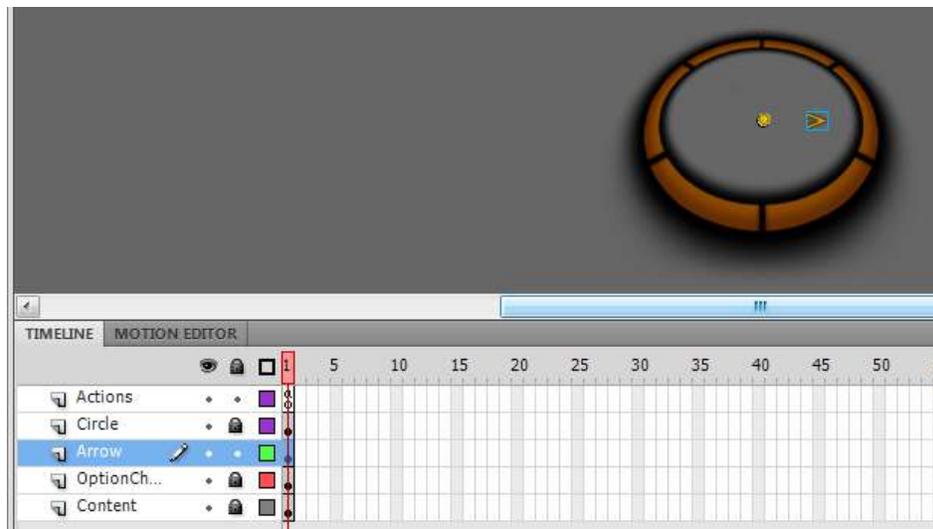


Figure 5-12 Flash Template

While the template may look simple, the functionality of the menu is not solely depends on the basic components shown in the template.

Next, the components inside the template must be coded to mimic the behaviors like those in the Mass Effect interactive dialogue. For this purpose, Action Script 2.0 has been used. Although newer versions of Action Script have been released, the Unreal Engine 3 does not support it.

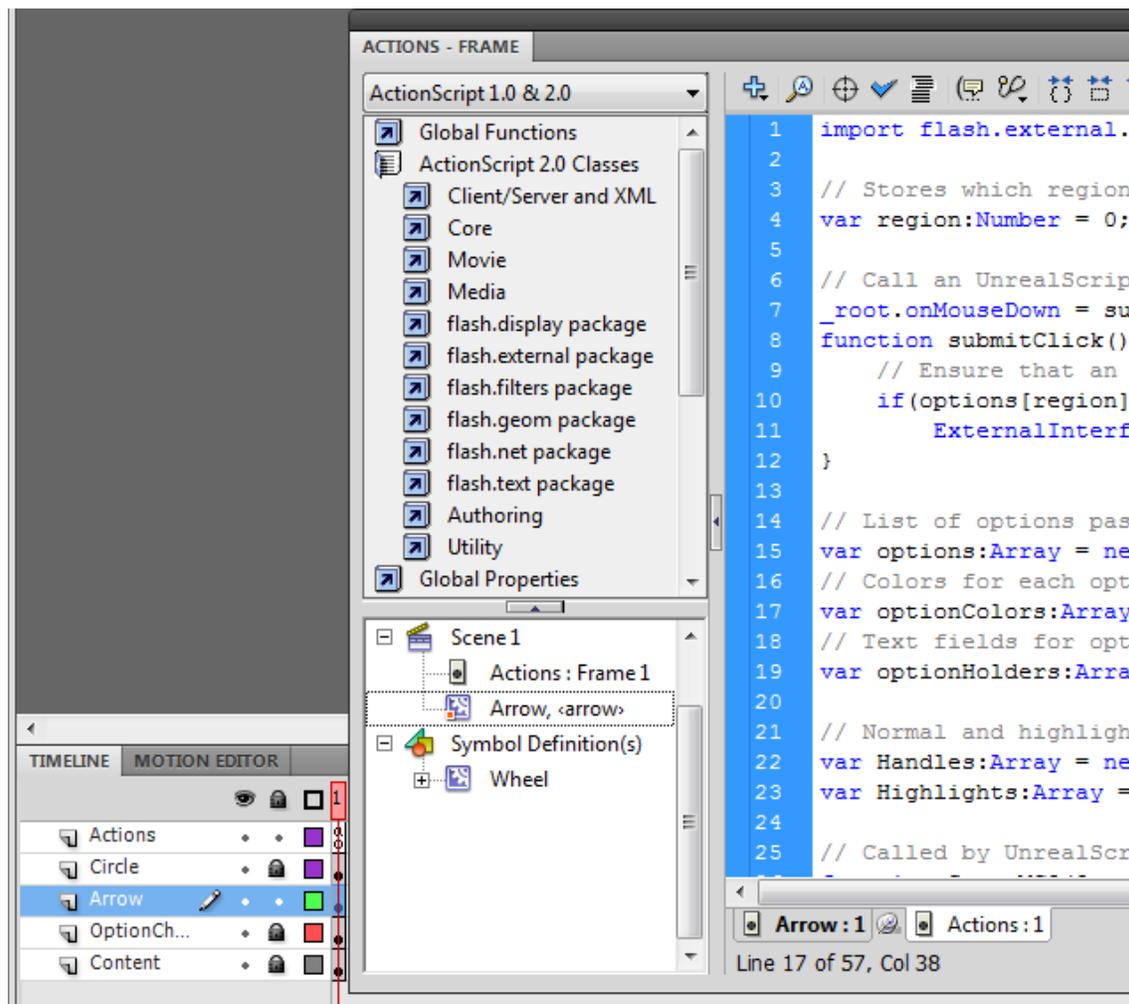


Figure 5-13 Action Script 2.0 in the Flash

When the coding in the Flash file is completed, the file has to be imported into the Unreal Engine 3 to act as a “GFX UI” scene, and to be called out in Kismet.

Normally, this will require the work of a series of “FSCCommand” for the Kismet to communicate with the Flash file, but the setting of such

links of FSCCommand calls is visually simplified as special Kismet Components called “SeqAction_BeginInteraction”, “GfxUI_Interaction” and “SeqEvent_OptionSelected” are created using UnrealScript.

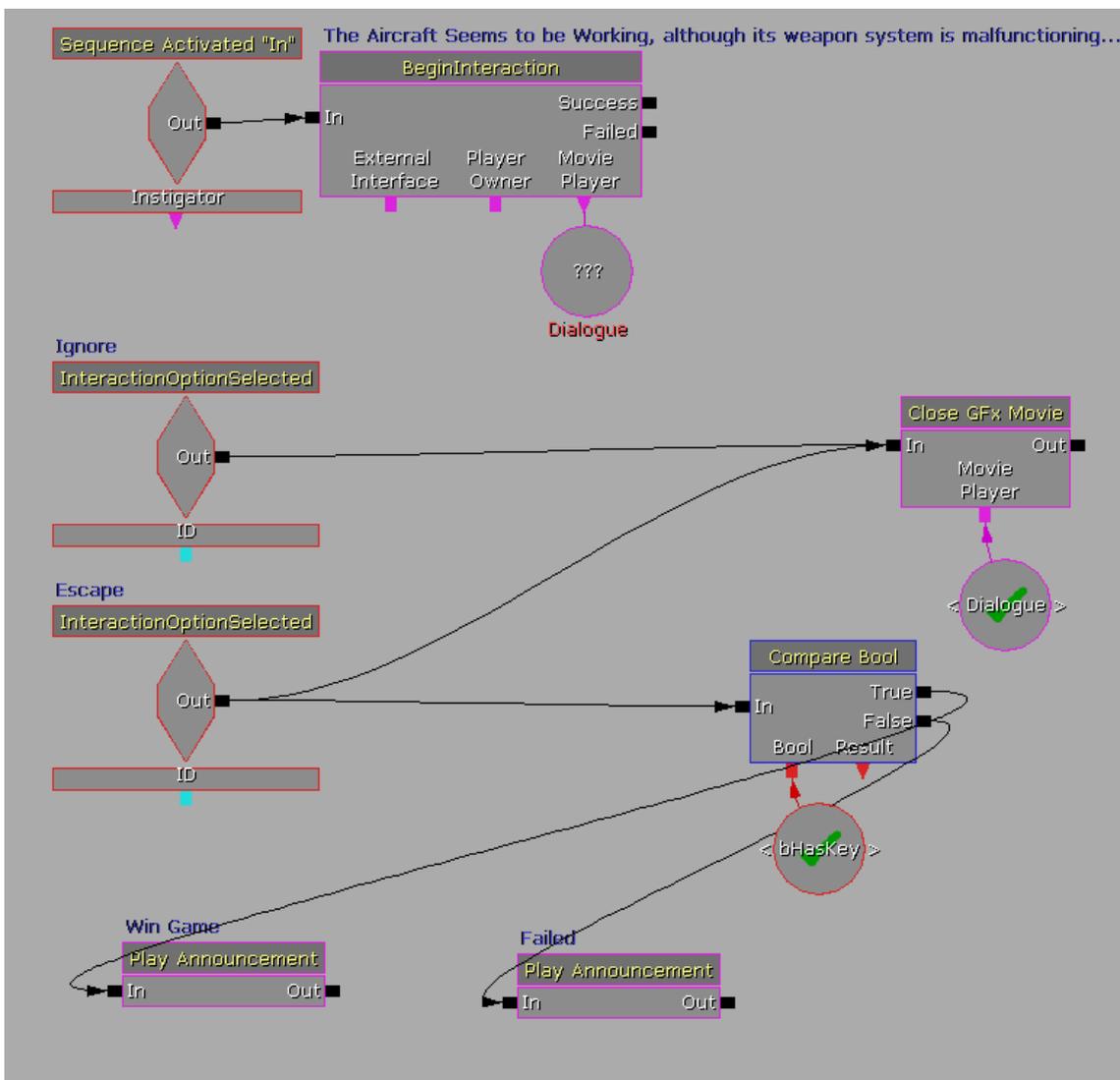


Figure 5-14 Example of Interactive Menu Setup

Therefore, we just need to set up conditions which triggers an interaction (such as pressing a button), and link it to the component “BeginInteraction”, then the scene will be loaded together with locking the player’s input and disabling the HUD (for better cinematic views). And we

also have to set up the strings which are to be displayed on the screen together with options to be selected.

Then, the events “InteractionOptionSelected” has to be set one-by-one for each option and be linked to different “effector” components, such as showing up a message or activating a machine in the game.

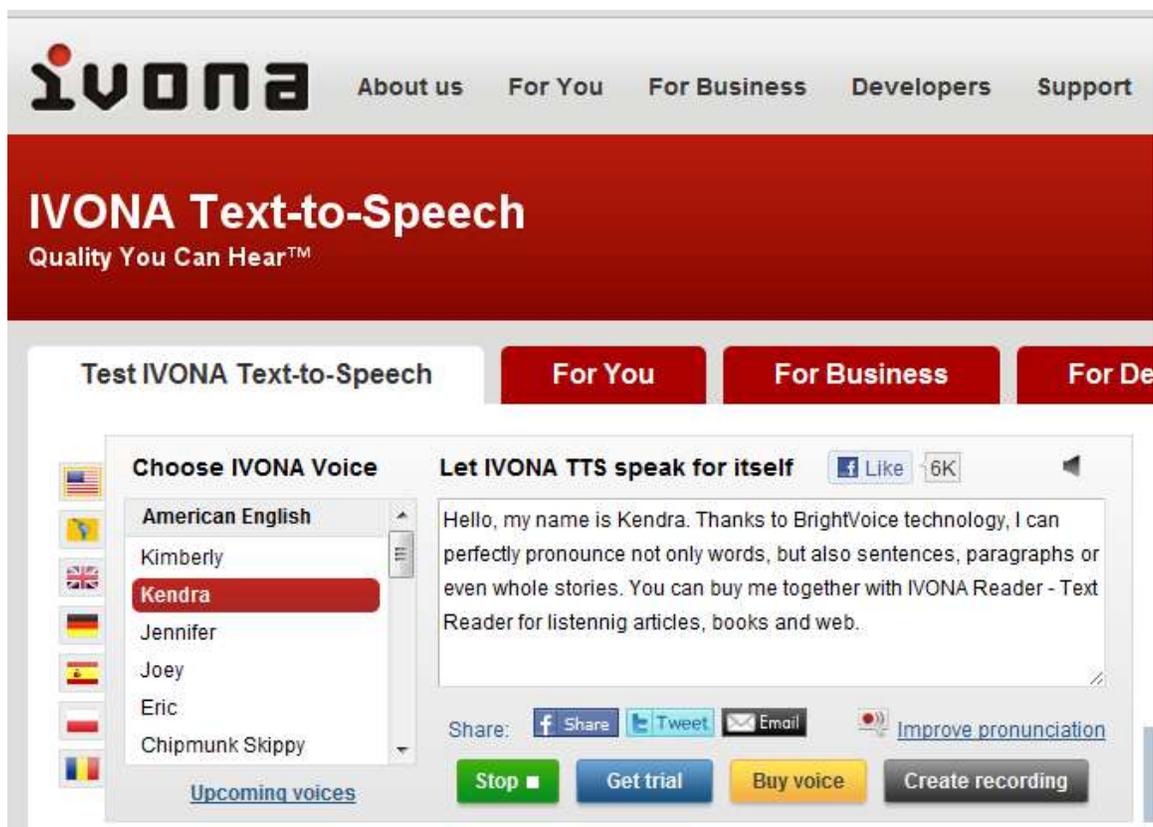


Figure 5-15 Example of a finished Interactive Menu

5.5.2. Text-To-Speech Support

According to the game plot, there will be an Artificial Intelligence voice guiding the player through the game, but in order to make it sound non-human while proper English is being pronounced, we decided to make that sound a product of Text-To-Speech synthesis.

There are several ways for us to achieve that goal, for example, we can first use some external Text-To-Speech converter to build up sound wave files from text. One of the available websites for such task is the IVONA Text-To-Speech system developed by the Polish IT company with the same brand name. (IVONA, 2011) The IVONA system comes with a variety list of voice synthesizers, simulating voices of different accents and genders.



The screenshot shows the IVONA website's interface for testing text-to-speech. At the top, the IVONA logo is displayed alongside navigation links: "About us", "For You", "For Business", "Developers", and "Support". Below the logo is a red banner with the text "IVONA Text-to-Speech" and the tagline "Quality You Can Hear™".

The main content area features a "Test IVONA Text-to-Speech" section with three tabs: "For You", "For Business", and "For De". Under the "For You" tab, there is a "Choose IVONA Voice" section. It includes a list of voices for "American English": Kimberly, Kendra (highlighted in red), Jennifer, Joey, Eric, and Chipmunk Skippy. To the left of this list are small icons representing different languages: US, UK, Germany, Spain, France, and Italy. Below the voice list is a link for "Upcoming voices".

To the right of the voice selection is a section titled "Let IVONA TTS speak for itself" with a "Like" button showing 6K likes. Below this title is a text box containing the sample text: "Hello, my name is Kendra. Thanks to BrightVoice technology, I can perfectly pronounce not only words, but also sentences, paragraphs or even whole stories. You can buy me together with IVONA Reader - Text Reader for listennig articles, books and web." Below the text box are social sharing buttons for Facebook, Twitter, and Email, along with a link to "Improve pronunciation". At the bottom of this section are four buttons: "Stop", "Get trial", "Buy voice", and "Create recording".

Figure 5-16 IVONA, a Text-to-Speech service

Afterwards, we can import the sound wave files into the Unreal Development Kit's Content Browser, and we then need to create sound playback nodes inside Kismet and link them up with the Interactive Dialogues.

However, this process has a few disadvantages.

To begin with, the process of integrating each sound wave file into the game is tedious and meaningless. Moreover, whenever we want to update a script inside the interactive dialogue box, we have to launch the Text-To-Speech synthesizer again only to generate a wave file, making the update process very unfriendly to developers.

Fortunately, the middleware Fonix VoiceIn has already been integrated as a part of the Unreal Engine 3, so that we do not have to re-program it from the beginning.



Figure 5-17 The Middleware Which Enables Text-To-Speech Support

In Psionscape, we have re-configured the interactive menu mentioned above in such a way that there will be a Boolean value called “bSpeakItOut”, which, by default, is set to false.

```
function bool Start(optional bool StartPaused=false)
{
    super.Start(StartPaused);
    // Start at the first frame
    Advance(0.f);
    PlayerOwner = GetPC();
    // Flush PlayerInput and stop the Pawn
    PlayerOwner.PlayerInput.ResetInput();
    PlayerOwner.Pawn.ZeroMovementVariables();
    // Cache all Option events
    PlayerOwner.WorldInfo.GetGameSequence().FindSeqObjectsByClass(class'SeqEvent
    if(bSpeakItOut)
        PlayerOwner.SpeakTTS(SpeakString, PlayerOwner.PlayerReplicationInfo);
    return true;
}
```

Figure 5-18 The Most Important Piece of Codes Which Enables Text-To-Speech

If the Boolean value is set to true inside Kismet, a Text-to-Speech message would be spoken out using Fonix VoiceIn in real-time. Changes to the script no longer needs to re-create a new sounds wave file because the whole Text-to-Speech process in run inside the game but not the editor.

5.5.3. Realistic Scenes

Since we have introduced how we edit the level in the previous chapter, we will only briefly discuss the process.

Basically we build everything using the CSG brushes inside the Unreal Development Kit's Editor. Since we are trying to replicate some parts of the real world's scenes in the Ho Sin-Hang Engineering Building (SHB), we need to first take a list of photos.



Figure 5-19 Reading Room at SHB 6/F

However, since the photos shot using ordinary cameras are prone to distortion like a slight fish-eye view due to light projection paths, we need to reconfigure the photos manually using photo editing software, and in our case, the Ulead PhotoImpact X3.



Figure 5-20 Outdoor view of the SHB

Moreover, as most of the texture in real world are not smooth surface, so we need to use the tool CrazyBump to generate a normal map for the picture, and together they can form a textured material inside the Unreal Editor.

With the right shapes and materials we can make up different objects such as chairs, desks and even computers and curtains.

To further improve the end result, we need to convert it into static meshes and edit different properties such as collision blocks.

With the static meshes ready, some pattern can be replicated more conveniently. For example, we just need to spend a few hours to make a

realistic chair, and then use it multiple times by dragging them from the Content Browser.



Figure 5-21 SHB Room 904

Last but not least, we have also included many additional static meshes to make the game look more interesting, such as rubbles, teleporters and weapon pickup locations, to create the feeling of a half-destroyed SHB in the game.



Figure 5-22 AIX Computers and Realistic Ceiling and Floor

5.6. BCI-Driven Game Puzzles

Making a BCI-enhanced game impressive is, of course, to develop puzzles actually utilizing the newly developed Mind Power meter.

5.6.1. Teleporter Activation

Due to the explosion caused by the special agents, there are rubbles blocking the path to the lift lobby. Although the player sees the lift lobby, he cannot reach it by simply jumping.



Figure 5-23 Block of Path to the Lift Lobby

However, exploring the corridor reveals a strange looking object on a balcony, and that is a broken teleporter (a machine which can transport the player between two locations instantly).

Investigating the teleporter gives the option for the player to repair it. However, this requires a high level of Mind Power over 5 seconds.

The player can try fixing it for unlimited number of times until it works. On successful attempt, the teleporter will be functional, and can warp the player to the lift lobby, where the lift can be taken to transport him to the fifth floor.

An interesting point to note is that when the teleporter is functional, there will be a preview of the destination, so that the player will know where this teleporter leads to.



Figure 5-24 Hint prompting the player to fix the teleporter

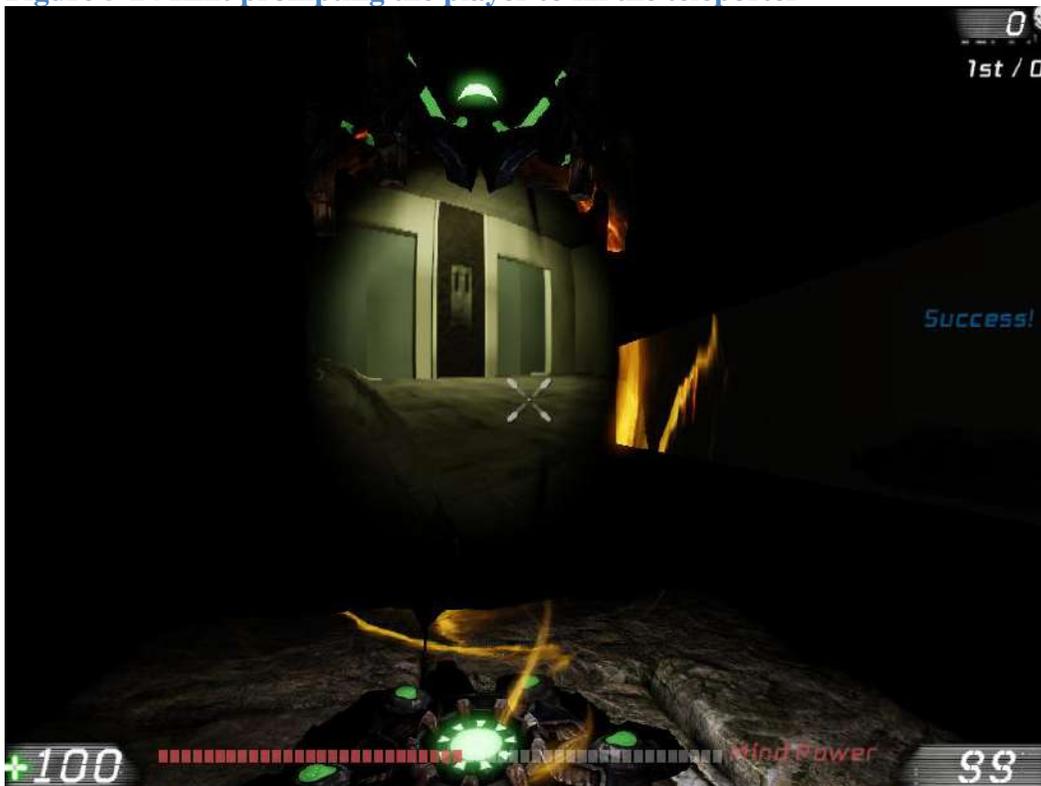


Figure 5-25 Preview of Destination when the teleporter is fixed

5.6.2. Mind Shield

One of the most common kinds of magical abilities available in many “Magic” games nowadays is the “magic shield”. A magic shield is usually summoned up in games related to wizardry, and the shield reduces or nullifies any damage caused to the spell casters.

This helps spawn up the idea that “Psionic Power” can also share a similar ability.



Figure 5-26 Patrolling Guard Equipped with a Rocket Launcher

In the game, there will be a robot patrolling around an area, where the robot is equipped with a rocket launcher. When the player walks close to the robot, the player may be killed within a few rounds of rocket hits.

Exploring the area will reveal a secret location where a shock rifle is laid on the ground, ready for pick up. However, it is guarded by a steady stream of bullets fired from a nearby malfunctioning aircraft.



Figure 5-27 Secret area with a weapon pickup

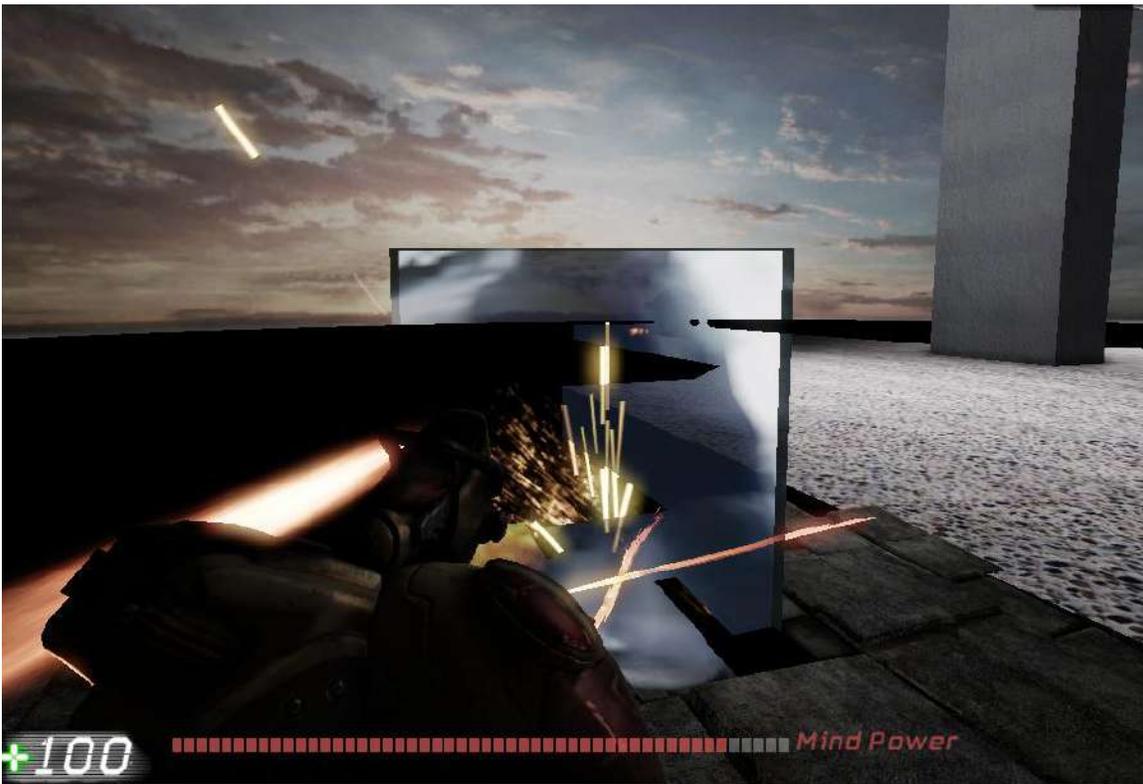


Figure 5-28 Mind Shield in Action

This is where the Mind Shield comes in handy.

The Mind Shield can be activated and “summoned” up by pressing the “R” key, which is programmed to work only when the Mind Power is high enough. If the player fails to maintain a Mind Power over half of the meter, the Mind Shield will automatically fail and disappear.

Nevertheless, if the player can maintain such a Mind Shield, it can be used to block all projectiles in front of the player, effectively protecting him to travel past the stream of bullets so as to get the shock rifle.

With the rifle acquired, the player can simply go back to the robot guard, and fire up the rifle towards it. However, at this stage, the player must be able to “Cool Down” his Mind Power to deactivate the Mind Shield so that the shield will not block the pulse attack.

In this way, the control over Mind Power can be trained.



Figure 5-29 The Mind Shield must be deactivated to use the Shock Rifle

5.6.3. Telekinesis Elevator

Telekinesis is a very famous trick for psionics and magic, so we decided to introduce it into the game. Originally, telekinesis only means to move something remotely, but in order to make more sense in the game, we decided to integrate the puzzle solving element into it.

In one occasion, the player will find that it is impossible to travel from the fifth floor of the SHB to the sixth floor, because the staircase is broken. Even if the player takes the working lift to the sixth floor, it will be found that there is a huge hole blown away from the ground and it is impossible to jump through it.



Figure 5-30 Broken ways making it impossible to reach 6/F

However, if the player searches around, he will notice that there is a group of rubbles lying on the ground, and among them is a flat platform.

If the player is curious enough and investigate it, there will be an interactive menu popping up and asking whether the player would like to animate that into an elevator.

When the player decides to do so, the Mind Power level will be used to exactly control the height of the elevating platform, meaning that, for a higher level of Mind Power, the platform will be moved upper, and for a lower level of Mind Power, it will fall down.

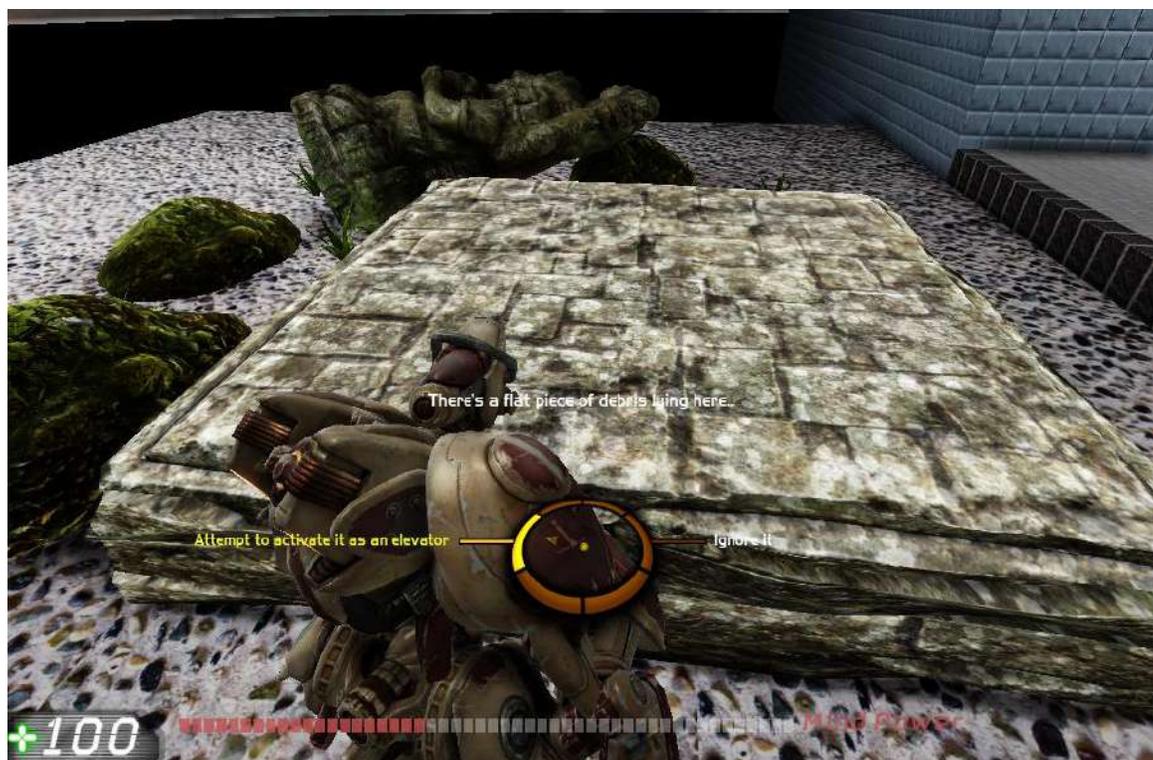


Figure 5-31 Interactive Menu for Activating the "Elevator"

Since the player must first jump onto the platform before moving upwards, he must be able to take control over his Mind

Power in order to lower the Mind Power value and thus the elevator's position. Similarly, after jumping onto the platform, the player must also have the ability to increase the Mind Power so as to make the platform high enough for the player to reach the sixth floor.

In this manner, this demo game also serves as a purpose to train up the ability of the player to use the BCI, and such advantage is not seen in other programs so far.

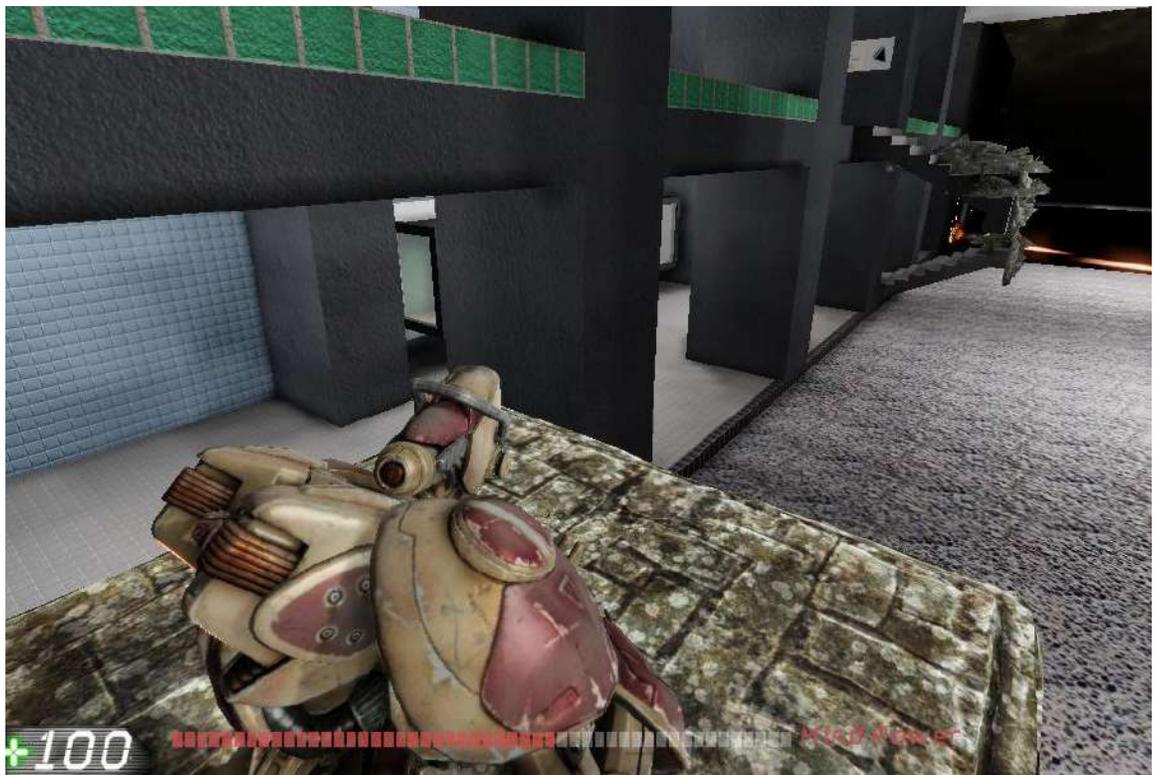


Figure 5-32 Controlling the height of the elevator via Mind Power

6. Conclusion

Given only the manpower of one to two persons, we managed to create a small but complete BCI-enhanced game.

The game is comprehensive in the way that basic gaming elements such as level editing, interactive menus, puzzle solving, shooter element and game plot are all included.

Adding on top is the BCI element we call Mind Power, which does not only make it an alternative input for the game, but also adds in new challenge to the game itself (e.g. Maintaining a Mind Shield to block hostile bullets) and making the story of the game sound more reasonable (e.g. Using telekinesis to “make up” an elevator using a pile of rubble).

With a simple BCI like the Neurosky Mindset, we managed to make one reliable input which works well in the game, as demonstrated above.

We demonstrated that BCI-enhanced game development is not a very difficult task and believe that the gaming industry should take into consideration the possibility of creating such kind of games in the near future.

7. Limitation & Difficulties

7.1. Getting stuck in UDK

While this report may be, hopefully, quite descriptive (in a high-level view) such that the readers could understand how the UDK and UnrealScript work. However, no matter it is the decision of game engines (especially when we do not know which game engines allow us to hack them to suit our needs), how to edit levels, how to write programs using UnrealScript (e.g. the syntax), and even more importantly how the structure of classes is organized in UDK (as said earlier, there are over 2350 classes which use or inherit from each other), the classes are simply “very convoluted and troublesome” (said by Dr. Or Siu Hang during CSCI4120 lessons).

As the beginning, we found that there is a vast number of documentation available and treated that as an advantage, but we did not figure out that there were actually

reasons behind which made this thick pile of documentation

necessary – there are really a lot of things to be learnt.

Moreover, some details are not provided. For example, on our first attempt to use the TCP client in UnrealScript, there

were even sample codes on the net which demonstrates how to

set several properties of the client object, but we did not even know how and where (i.e. in which class) should we create that object.

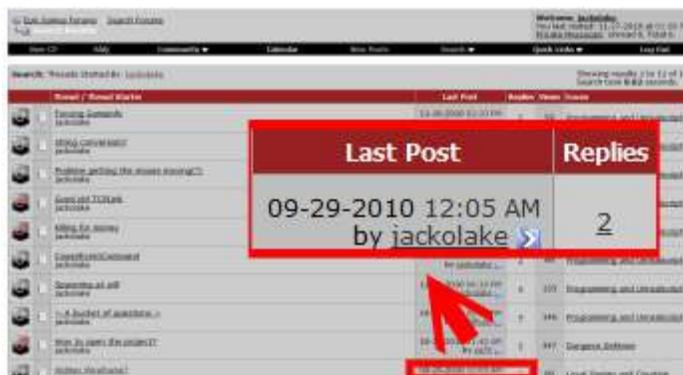


Figure 7.1 [Long list of threads](#) was posted onto UDK forum, but there had been only a few replies, and the replies were always detailed enough.

The time spent on the tedious work was tremendous, yet it is very hard to demonstrate all of the efforts involved.

7.2. Poor signal from Mindset

7.2.1. Explanation

The Poor Signal from Mindset is how poor the signal measured by Mindset. The value ranges from 0 to 200. A non-zero value means the existence of noise contamination. Poor signals may be caused by a number of different things, such as, poor contact of sensor, excessive motion of user, excessive environmental electrostatic noise.

The poor signal affects the data collection from mindset because attention and meditation value will not be updated when poor signals is non-zero. It means there will be some missing data while we collecting the data if there is noise.

When experiment was carried out, some users found difficulties to make the poor signal values to become zero. They may need several minutes to adjust the position of the mindset.

7.2.2. Relation with the BCI Game

During the game, if poor signals are received, the classifier would cease to detect the latest brain wave types, so the type will be unchanged for the time being. This could lead to a very high or very low Mind Power reflected in the game.

8. Division of Labor

Beginning from last summer, I have been the project manager of this team. Originally, the FYP topic is not like this, and the steering of the direction of work is mostly done by me.

In the first semester, I work closely with my partner, Donald Cheung, to gather various information about the Neurosky Mindset, but programs utilizing the ThinkGear API and the experiments are carried out by him. At the same time, beginning from nothing at all, I started by research on computer game development and picked one of the best modern 3D game engines. However, the Unreal Engine 3 is so powerful that “newbies” like me really need a lot of time to study and practice using it, so most of the time I have to learn the basis of how the game engine work and, more importantly, how to master the UnrealScript. Therefore, at the end of the semester, I managed to develop demo scenes to demonstrate preliminary BCI-integrated active control features.

In the second semester, I single-handedly created the game Psionscape, starting from the game plot, the concept arts and the actual level design and coding. The level design alone is already tedious because photos have to be taken and edited to suit the development needs. Even more difficult is to introduce various game elements into the game to make it more impressive. Despite the small game size when compared with the industry standard (due to the ridiculously small manpower I have), the game is a complete one, meaning that it has a story and can be driven through the beginning to the end. Combining with our close work with Donald, I consider this project a very successful learning experience.

9. References

- Arthur, C. (2009, June 9). *Are downloads really killing the music industry? Or is it something else?* Retrieved 11 23, 2010, from Guardian:
<http://www.guardian.co.uk/news/datablog/2009/jun/09/games-dvd-music-downloads-piracy>
- DevMaster.net. (2010, 11 29). *List All Engines*. Retrieved 11 29, 2010, from DevMaster.net: <http://www.devmaster.net/engines/list.php>
- Emotiv-Administrator. (2010, 07 30). *EPOC (not good) Experiences*. Retrieved 11 28, 2010, from Emotiv Main Forum:
<http://emotiv.com/forum/messages/forum4/topic732/message4310/#message4310>
- Fear, E. (2009, 6 26). *The Top 10 Game Engines*. Retrieved 11 29, 2010, from Develop-Online.net: <http://www.develop-online.net/features/519/The-Top-10-Game-Engines-No-1-Unreal-Engine-3>
- Fruhlinger, J. (2008, 10 9). *Brains-on with NeuroSky and Square Enix's Judecca mind-control game*. Retrieved 11 28, 2010, from engadget:
<http://www.engadget.com/2008/10/09/brains-on-with-neurosky-and-squareenixs-judecca-mind-control-ga/>
- Gregory, J., & Lander, J. (2009). Third-party SDKs and Middleware. In J. Gregory, & J. Lander, *Game Engine Architecture* (p. 31). A K Peters, Ltd.
- IVONA. (2011). *IVONA Text-to-Speech*. Retrieved April 19, 2011, from IVONA:
<http://www.ivona.com/?tk=EMC4SR2X&gclid=CNTv-MXIQKgCFU9SHAod1icyIA>
- Kenner, C. (2009, 4 1). *Emotivated vs Neurosky*. Retrieved 11 28, 2010, from Facebook:
<http://www.facebook.com/topic.php?uid=9489703974&topic=8230>
- Novak, J., & Moore, M. E. (2010, April 29). *Game Industry Career Guide: Growth of an Industry*. Retrieved April 18, 2011, from Game Career Guide:
http://www.gamecareerguide.com/features/847/game_industry_career_guide_growth_.php?page=3

- NVIDIA. (2010, 1 17). *NVIDIA PhysX SDK Features*. Retrieved 11 30, 2010, from NVIDIA Developer Zone: http://developer.nvidia.com/object/physx_features.html
- OGRE. (2010). *Testimonials*. Retrieved 11 29, 2010, from OGRE 3D: <http://www.ogre3d.org/about/testimonials>
- Oriental-Daily. (2010, 11 05). *中大研腦電波輸入漢字*. Retrieved 11 27, 2010, from Oriental Daily: http://orientaldaily.on.cc/cnt/news/20101105/00176_045.html
- PixelMineGames. (2010, 11 11). *nFringe:Features*. Retrieved 11 30, 2010, from PixelMineGames: <http://wiki.pixelminegames.com/index.php?title=Tools:nFringe:Features>
- Porter, J. (2010). *UDN*. Retrieved 11 30, 2010, from Calling DLLs from UnrealScript (DLLBind): <http://udn.epicgames.com/Three/DLLBind.html>
- Reynders, D., & Wright, E. (2003). Practical TCP/IP and Ethernet networking. In D. Reynders, & E. Wright, *UDP (User Datagram Protocol)* (p. 131). The Great Britain: IDC Technologies.
- Scaleform.com. (2010). *Scaleform GfX Core Technology*. Retrieved 11 30, 2010, from Scaleform Corporation: <http://www.scaleform.com/products/gfxtech>
- Stevens, W. R., & Wright, G. R. (1994). *TCP/IP illustrated: The protocols*. Canada: Addison Wesley.
- UDKC. (2010, July 22). *ScaleForm - Mass Effect style interaction system*. Retrieved April 19, 2011, from UDKC: http://udkc.info/index.php?title=Tutorials:ScaleForm_-_Mass_Effect_style_interaction_system
- Walsh, P. (2008). Advanced 3D game programming with DirectX 10.0. In P. Walsh, *MIP Maps* (p. 432). United States of America: Wordware Publishing, Inc.
- Ward, J. (2008, 04 29). *What is a Game Engine?* Retrieved 11 29, 2010, from Game Career Guide: http://www.gamecareerguide.com/features/529/what_is_a_game_.php?page=2