

Unleashing Brain Powers

A Study on Development of BCI-enhanced Computer Games

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

FYP Report
Fall Semester 2010-2011



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

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 will start off by studying the consistency and effectiveness of a non-invasive consumer-level BCI, Neurosky Mindset, then devise and carry out an interesting experiment to collect brain wave samples for analysis later on.

We will demonstrate that BCI could actually be integrated into modern or future computer games in various ways, by utilizing the Mindset as active/passive controls in game demos developed by a modern 3D computer game engine, Unreal Engine 3, then conclude by evaluating the reception of such kind of entertainment.

Table of Contents

1. Abstract.....	1
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
1.2.3. Game Industry Responses to BCI.....	14
1.3. 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.....	22
2.1.4. Mindset Structure	24
2.1.5. Relevant Brain waves	25
2.2. eSense Meters	26
2.3. Experiment on eSense.....	27
2.3.1. Hypothesis	27
2.3.2. Methodology.....	27
2.3.3. Result.....	31
2.3.4. Conclusion for result and analysis.....	45
3. Building Our BCI Computer Game.....	46
3.1. <i>Introduction to Game Engines</i>	46
3.1.1. What is the structure of a game engine?.....	46
3.1.2. <i>Why even bothering using a game engine?</i>	49

3.2.	Why Unreal Engine 3?.....	51
3.3.	Speedy Guide to the UDK	54
3.3.1.	Primitive Modeling.....	55
3.3.2.	Asset Creation/Management and Lighting	56
3.3.3.	Visual Programming and Animation.....	58
3.4.	Integration of UDK and Mindset	63
3.4.1.	Introduction to UnrealScript.....	63
3.4.2.	Communication between UDK and Mindset.....	66
3.4.3.	Ways of BCI controls	69
3.4.4.	Demonstrating Active Control.....	71
4.	Conclusion	74
5.	Limitation & Difficulties	75
5.1.	Getting stuck in UDK.....	75
5.2.	Poor signal from Mindset.....	76
6.	Future Work.....	77
6.1.	Passive Controls	77
6.2.	Small-scaled BCI-enhanced Game.....	78
7.	References	79
8.	Appendix	82
8.1.	Our Efforts in This Semester.....	82
8.2.	Questionnaire	83
8.3.	Brainwaves sampling Feedback	86

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, or at least, enable

one-way input from the players to the games.



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

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

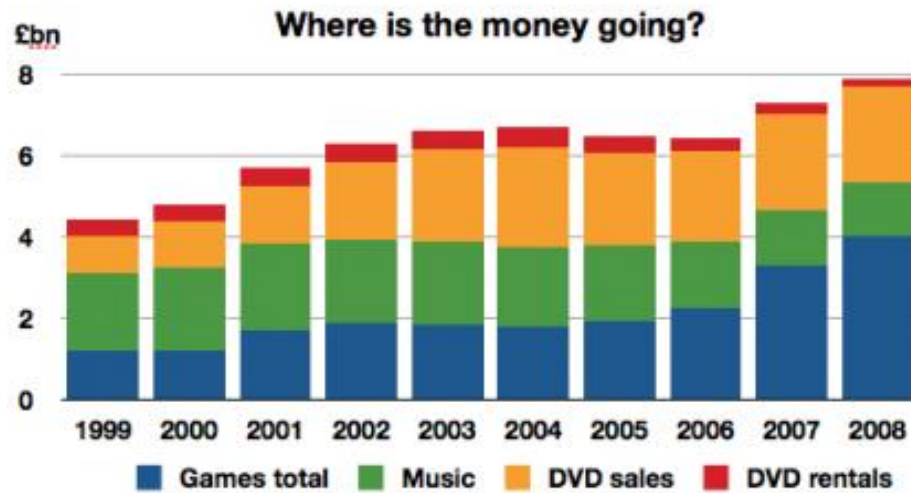


Figure 1.1-3. The video game industry has been growing at lightning speed, even exceeding income generated by the music industry.

(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.

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 1.1-4. 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.

1.2. Background

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

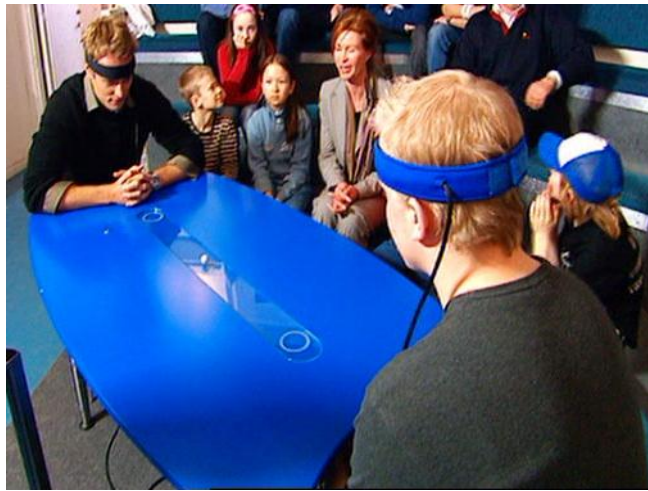


Figure 1.2-2 Mindball was one of the first consumer-level BCIs

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



Figure 1.2-4 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 1.2-5 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 waves data, enabling

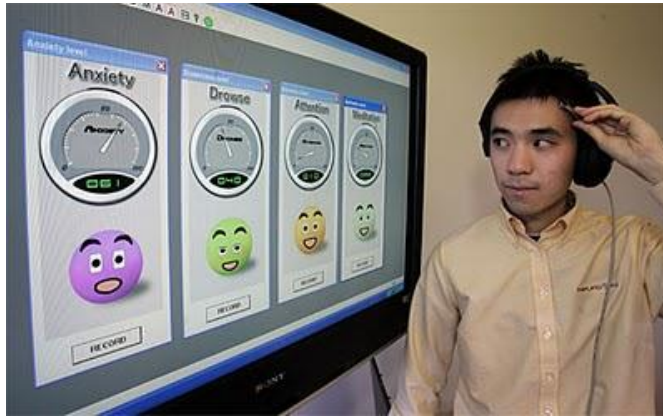


Figure 1.2-6 Some people find it difficult to establish perfect connection

(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.

1.2.3. 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 1.2.3-1 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.



Figure 1.2.3-2 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:

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.



Figure 1.2.3-3 Judecca in action, revealing zombies using Concentration level

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.

1.3. 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.

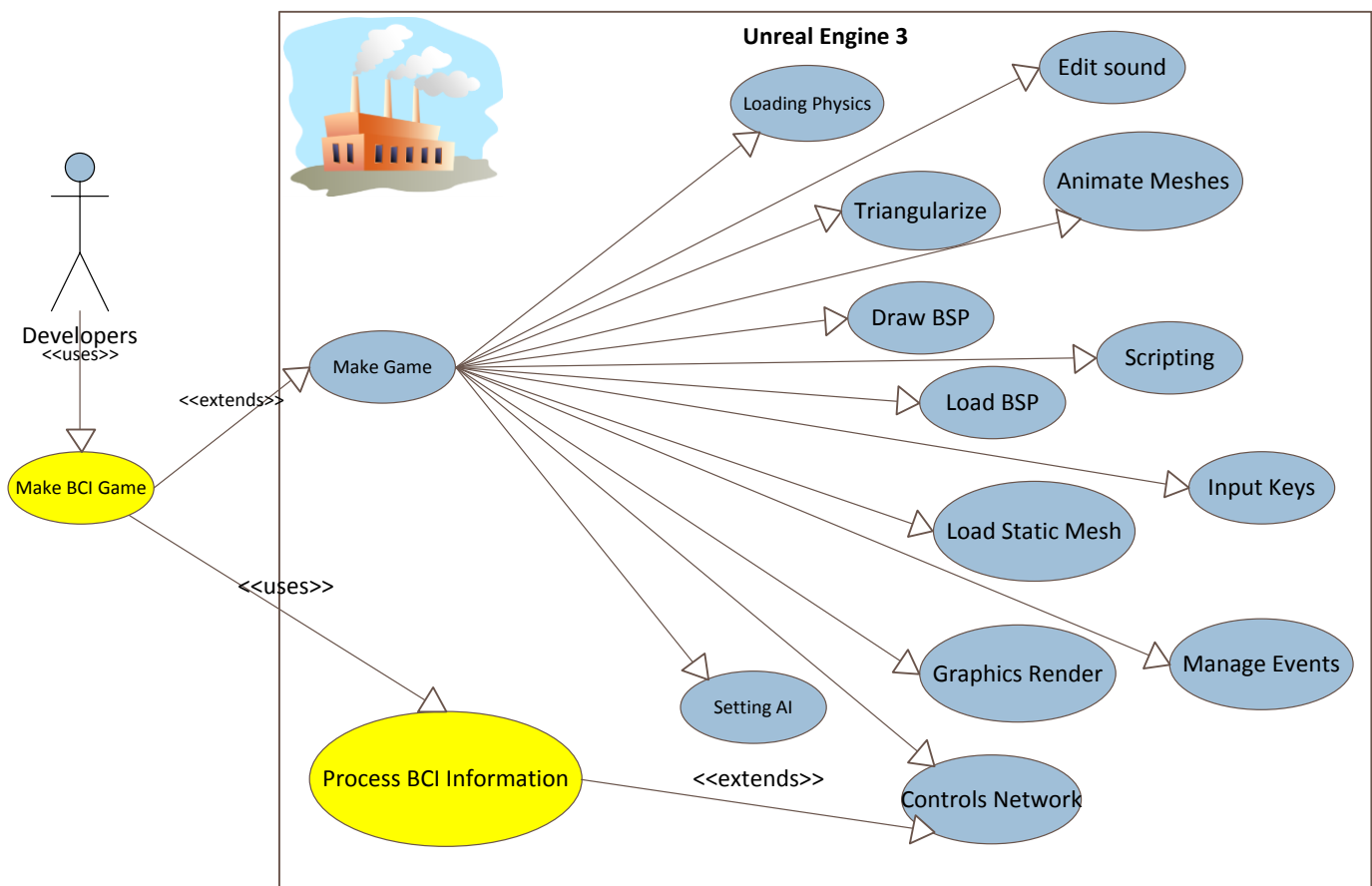


Figure 1.2.3-4 Use Case Diagram of Modified Unreal Engine 3

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.

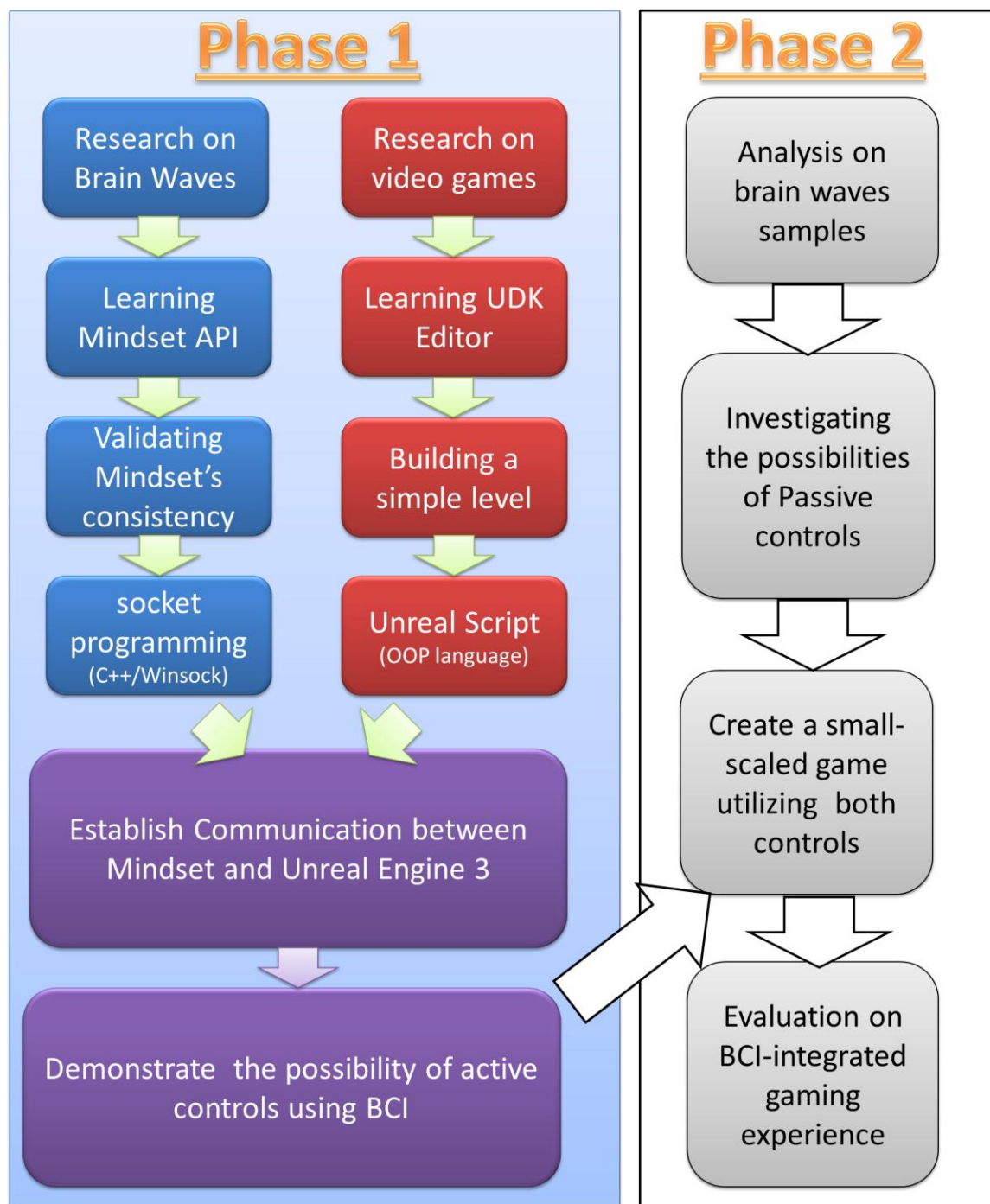


Figure 1.2.3-5 Complete Project Overview

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.

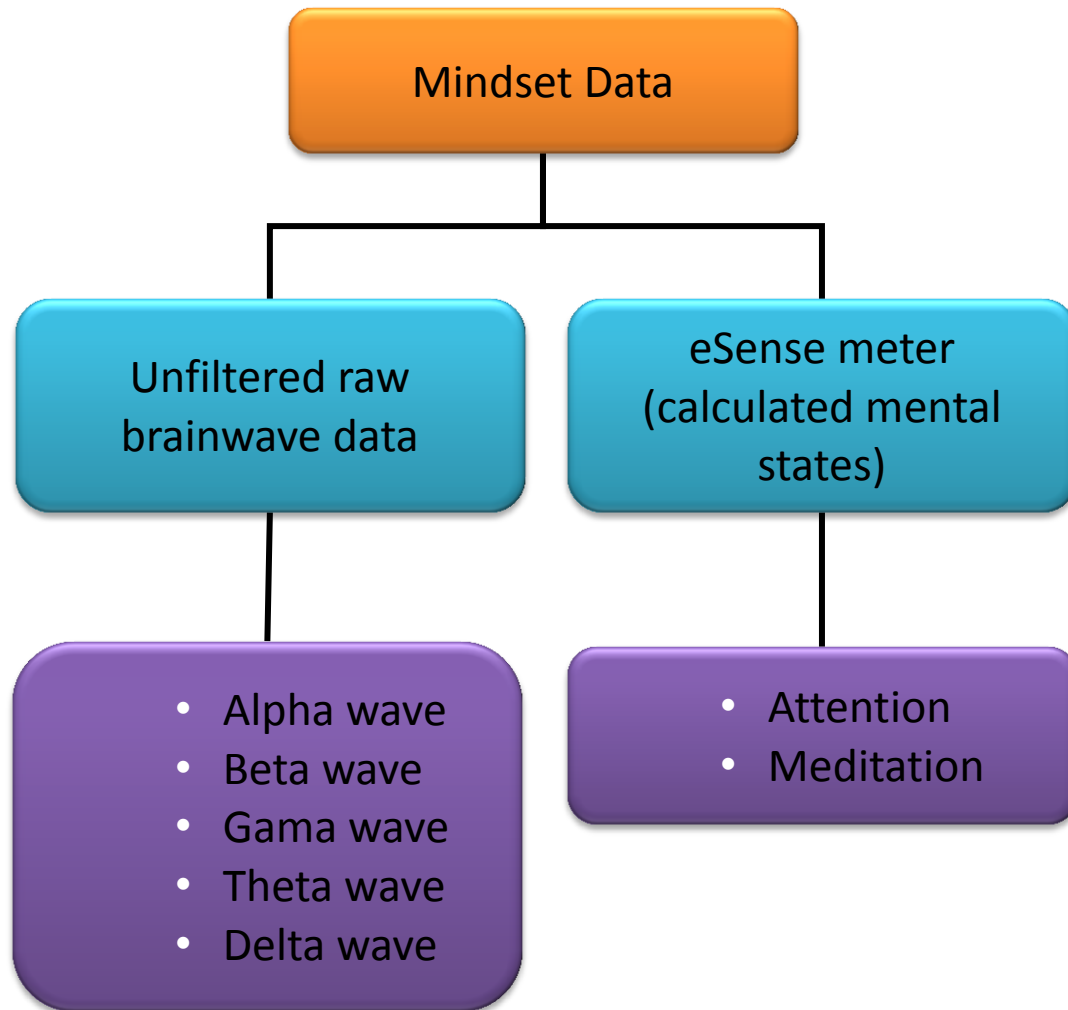


Figure 3.1.2-1 Data that can be collected from Mindset

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 brainwave 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 reference points located on the ear pad and the on-board chip that processing 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's 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.

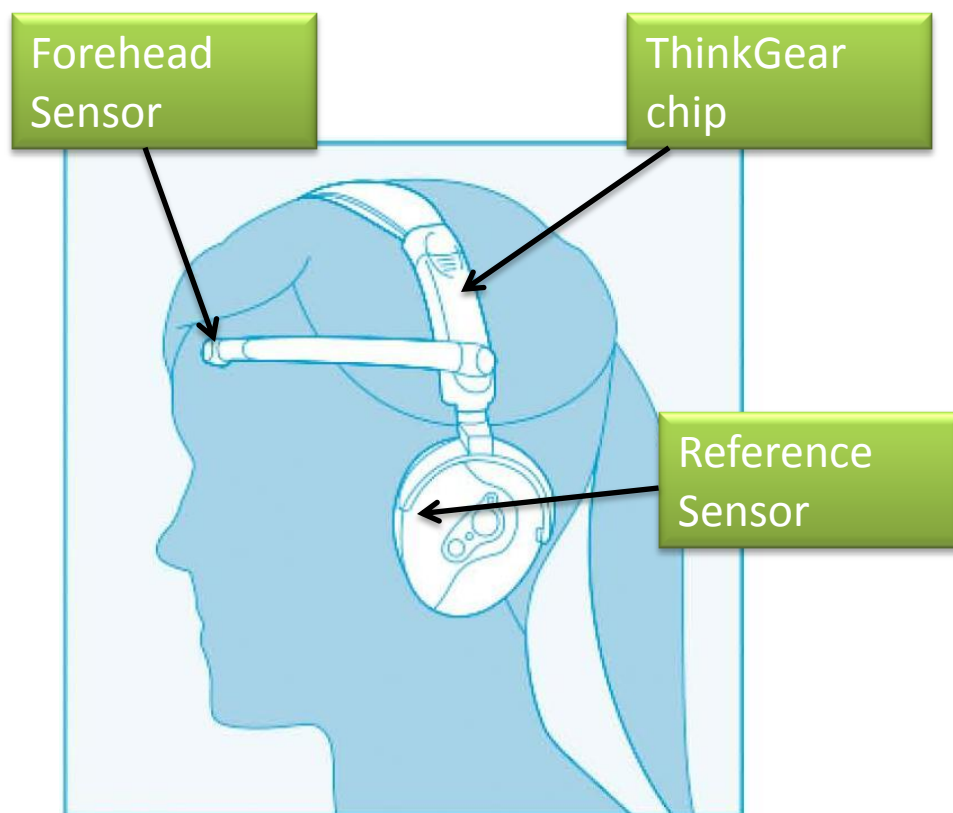


Figure 3.1.3-1 Key component for ThinkGear technology

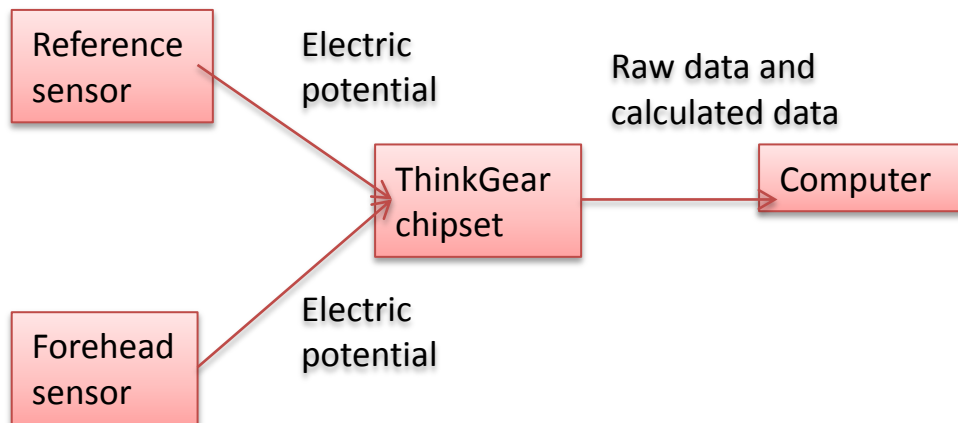


Figure 3.1.3-2 Data flow of Mindset

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 as digit input to the computer.

2.1.4. Mindset Structure

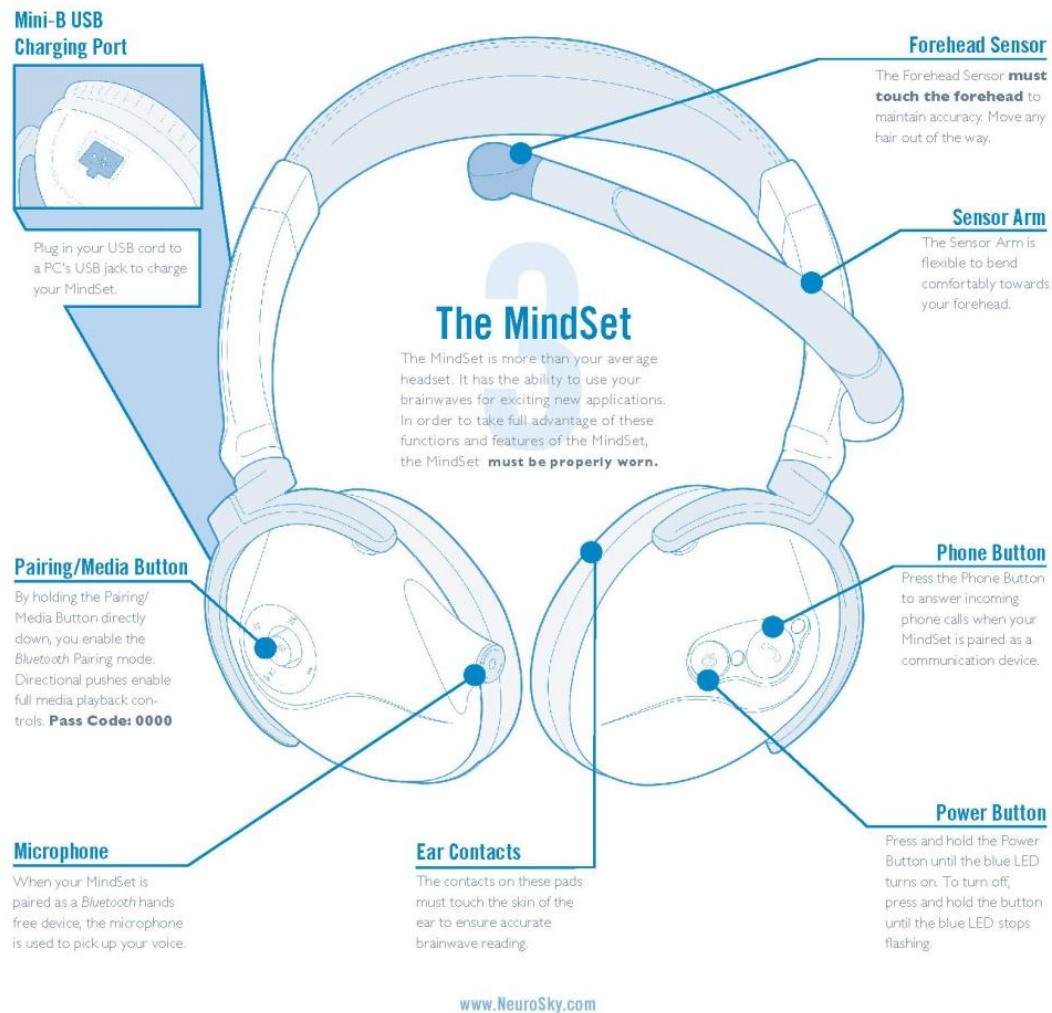


Figure 3.1.4-1 Physical structure of the mindset

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.

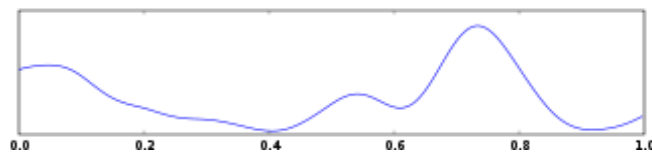
2.1.5. Relevant Brain waves

Table of different brainwave and its information

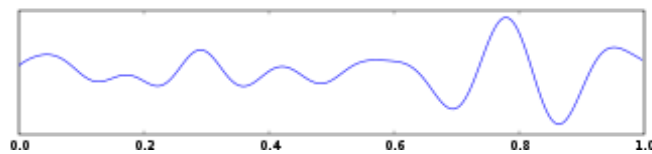
Brainwave Type	Frequency range	Mental states and conditions
Delta	0.1Hz to 3Hz	Deep, dreamless sleep, non-REM sleep, unconscious
Theta	4Hz to 7Hz	Intuitive, creative, recall, fantasy, imaginary, dream
Alpha	8Hz to 12Hz	Relaxed, but not drowsy, tranquil, conscious
Low Beta	12Hz to 15Hz	Formerly SMR, relaxed yet focused, integrated
Midrange Beta	16Hz to 20Hz	thinking, aware of self & surroundings
High Beta	21Hz to 30Hz	Alertness, agitation

One second sample of different brainwaves

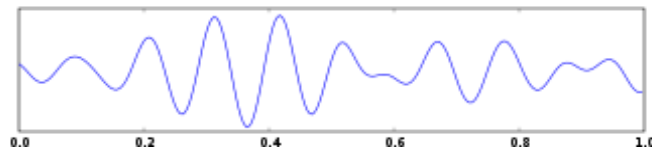
Delta waves



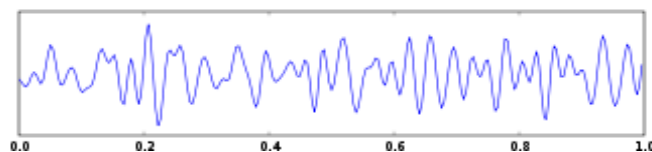
Theta waves



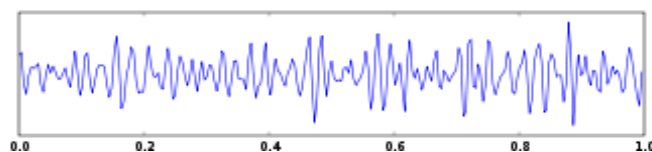
Alpha waves



Beta waves



Gamma waves



2.2. eSense Meters

The eSense Meter contains two mental states: Attention and Meditation.

The meter ranged from 1 to 100. A value between 40 and 60 at any given moment in time is considered "neutral". A value between 60 and 80 is considered "slightly elevated", and may be interpreted as levels being possibly higher than normal.

Values between 80 and 100 are considered "elevated", meaning they are strongly indicative of heightened levels of that eSense. Similarly, on the other end of the scale, a value between 20 and 40 indicates "reduced" levels of the eSense, while a value 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.

2.3. Experiment on eSense

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

2.3.1. Hypothesis

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

2.3.2. Methodology

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

VMRPlayer with Mindset

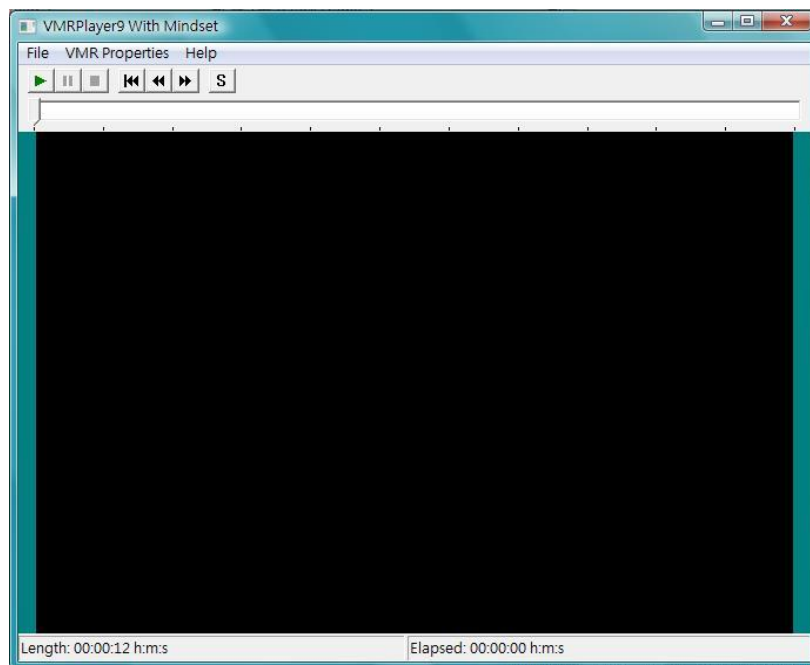


Figure 3.4.2-1 VMRPlayer with mindset

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.

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

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 mental state is relaxed.

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

Figure 3.4.2-2 The string 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 mental state is bored.

Figure 3.4.2-3 Speech of Pope Benedict XVI

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 mental state is concentrated.



Figure 3.4.2-4 Bikini and growing brown box

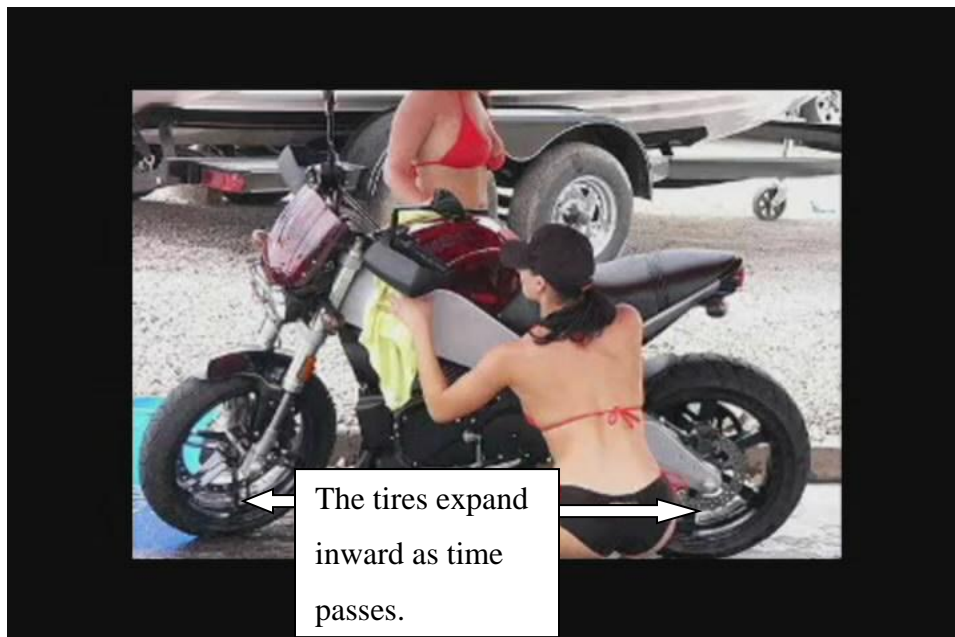


Figure 3.4.2-5 Bikini and the expanding tires

Part 4: Ghost Pop Up



Figure 3.4.2-6 Movie before ghost appears

Description: Participant is told that the ghost will pop up anytime and participant needs to be prepared for the ghost. The estimated mental state of this part is fear.

Special point: There is a ghost pop up with loud sound at 5:00.

Questionnaire

The questionnaire asks the participants about 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.

Brain Waves Sampling Feedback Form

* Required

For Clip#1 (the string orchestra), which of the followings best describes your general emotion? *

- ☐ Bored
- ☐ Concentrated
- ☐ Excited
- ☐ Fear
- ☐ Relaxed
- ☐ Other:

Figure 3.4.2-7 Brainwaves sampling feedback form

See Appendix for full questionnaire.

2.3.3.Result

There are 18 participants in this experiment, includes 16 males and 2 females and all of them are around 20 years old.

Average attention and meditation value are used for analysis. There are four parts in the movies and data analysis will be done to correlate mental state and change in eSense at each part.

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

Part 1 average attention and mediation for all participants

	Part 1 average
Attention	44.79998
Meditation	49.83071

Compare overall average eSense values and part 1 average values for all participants

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

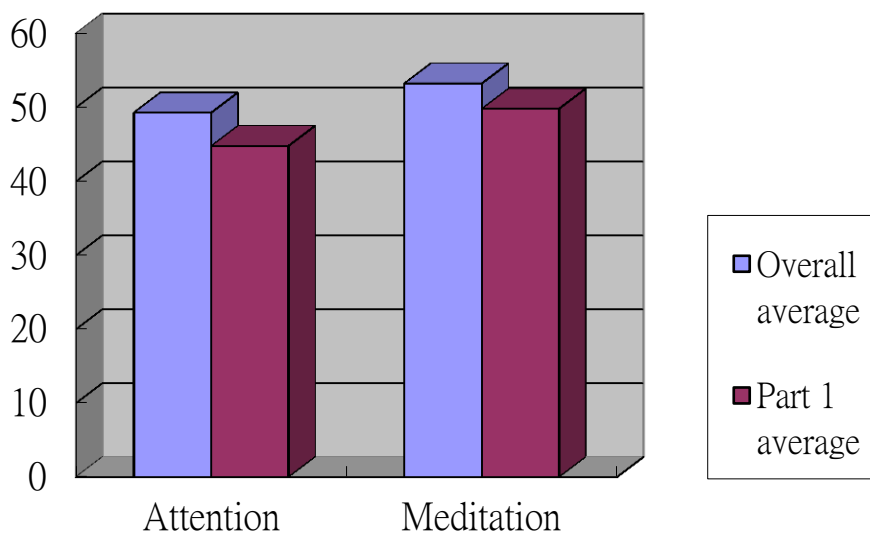


Figure 3.4.3-1 Comparison of overall average eSense values and part 1 average values for all participants

The average attention and meditation values in part 1 is lower than overall average values.

Relaxed Participants

Since about half of participants in part 1 is relaxed, the data of relaxed participants are selected and uses for comparison with other participants.

Overall eSense values of relaxed participants

	Overall average
Attention	47.88266
Meditation	53.2328

Part 1 eSense values of relaxed participants

	Part 1 average
Attention	43.67935
Meditation	46.75625

Compare overall average eSense values and part 1 average values for relaxed participants

	Attention	Meditation
Overall average	47.88266	53.2328
Part 1 average	43.67935	46.75625
Difference between average values and part 1 values	<u>-4.20332</u>	<u>-6.47655</u>

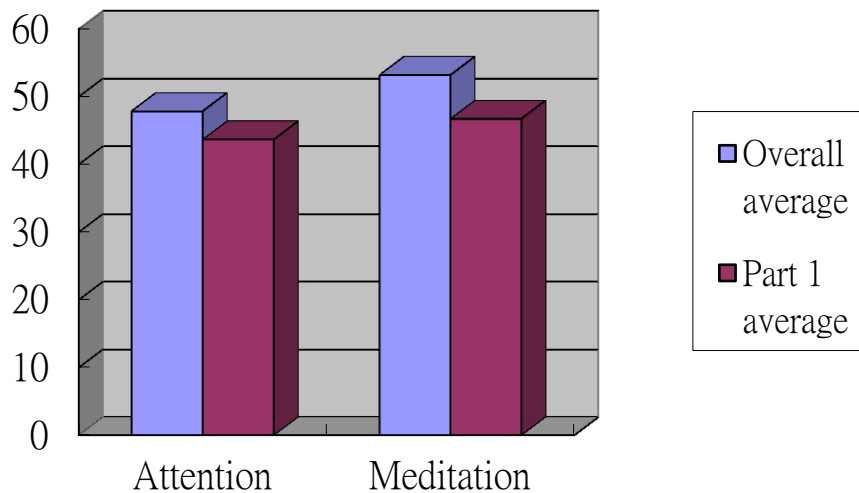


Figure 3.4.3-2 Comparison of overall average eSense values and part 1 average values for relaxed participants

The average attention and meditation values for relaxed participants in part 1 is lower than overall average values.

Comparing relaxed participants and all participants, the eSense values in part 1 are both lower the overall average values.

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

About half of participants’ feeling changes from relaxed to excited after the strike of loud sound. Analysis is done by comparing eSense values before and after the strike of

loud sound for these participants

	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>

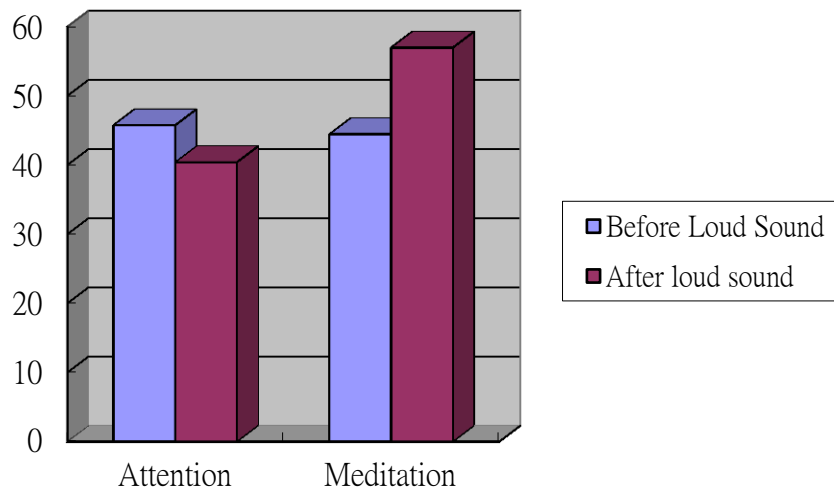


Figure 3.4.3-3 Compare eSense values before and after the strike of loud sound for participants who emotion changes from relaxed to excited

The attention value decrease and the meditation value increases after the loud sound.

Conclusion

According to the definition of meditation given by Mindset, 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 value 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 average attention and meditation for all participants

	Part 2 average
Attention	44.19243
Meditation	55.19941

Compare overall average eSense values and part 2 average values for all participants

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

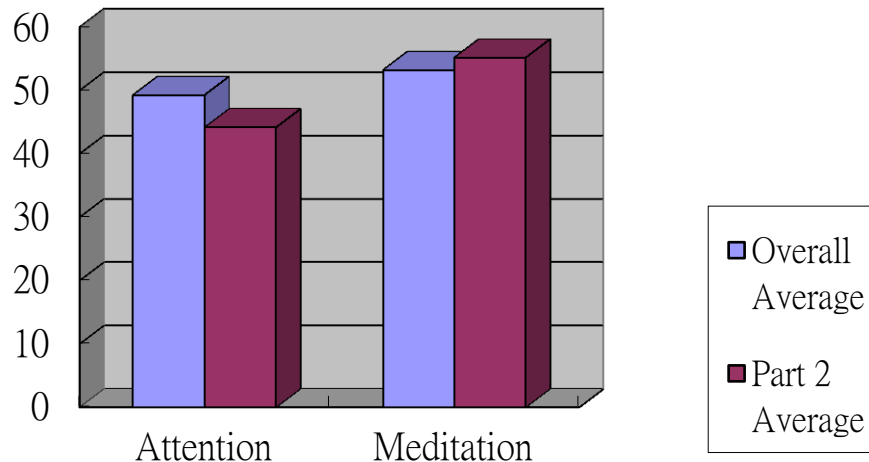


Figure 3.4.3-4 Compare overall average eSense values and part 2 average values for all participants

The average attention value of Part 2 is lower than overall average value. There are no obvious changes in meditation value.

Bored Participants

Most participants feel bored during the whole Pope' Speech. The data analysis is done by investigating the changes in eSense value during the speech.

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

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

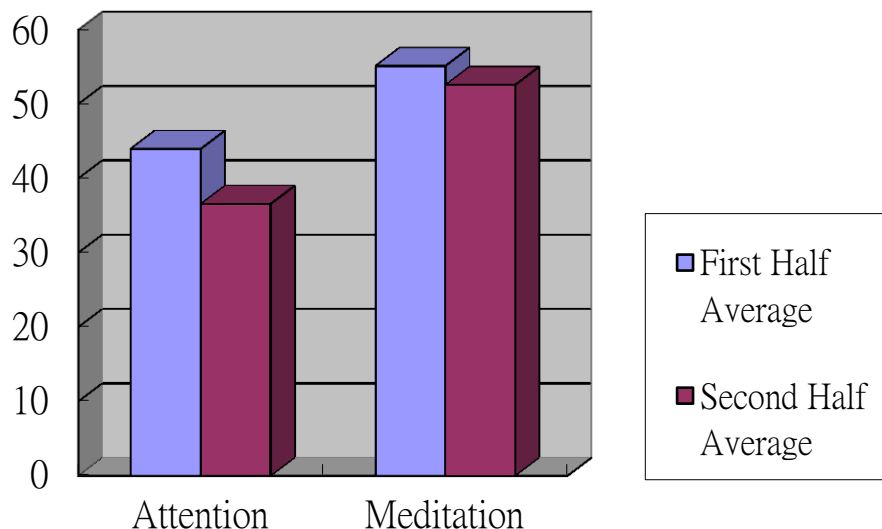


Figure 3.4.3-5 Compare eSense values between first half and second half for bored participants

Comparing values in first half and second half of Pope's speech, the average attention value of bored participants decreases and the average 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 overall average 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

The video for part 3 ask the participants to find out the changes in the movie. We assume the participants are concentrated and the data will be useful for finding correlation between eSense and concentration.

Compare overall average eSense values and part 3 average values for concentrated participants watching question 1

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

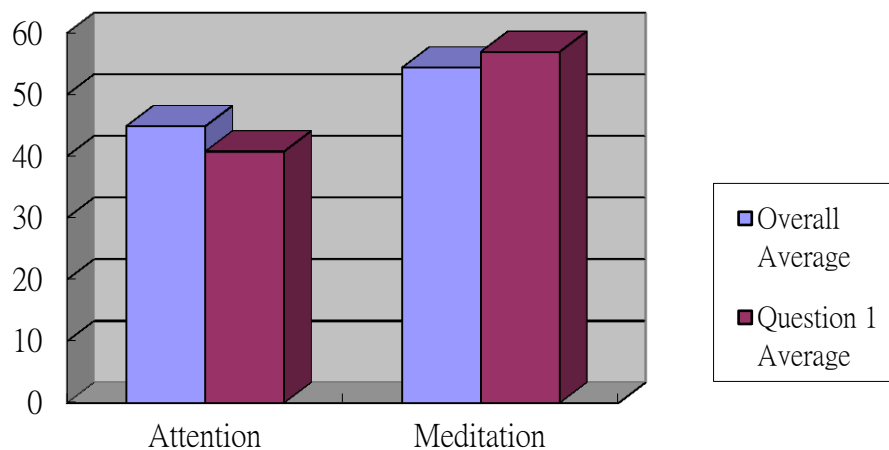


Figure 3.4.3-6 Compare overall average eSense values and part 3 average values for concentrated participants watching question 1

The average attention value during question 1 is lower than overall average value. The average meditation value during question 1 is higher than overall average value.

Compare overall average eSense values and part 3 average values for concentrated participants watching question 2

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

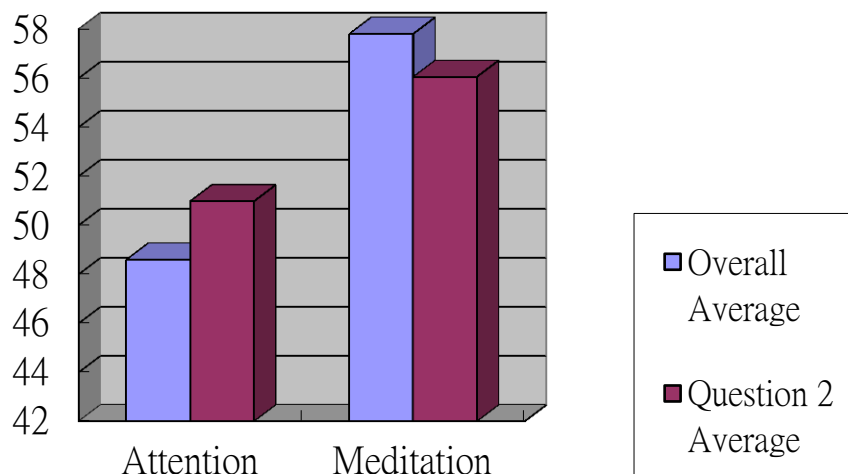


Figure 3.4.3-7 Compare overall average eSense values and part 3 average values for concentrated participants watching question 2

The average attention value during question 2 is higher than overall average value. The average meditation value when during question 2 is lower than overall average value.

Compare overall average eSense values and part 3 average values for concentrated participants watching answer 1

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

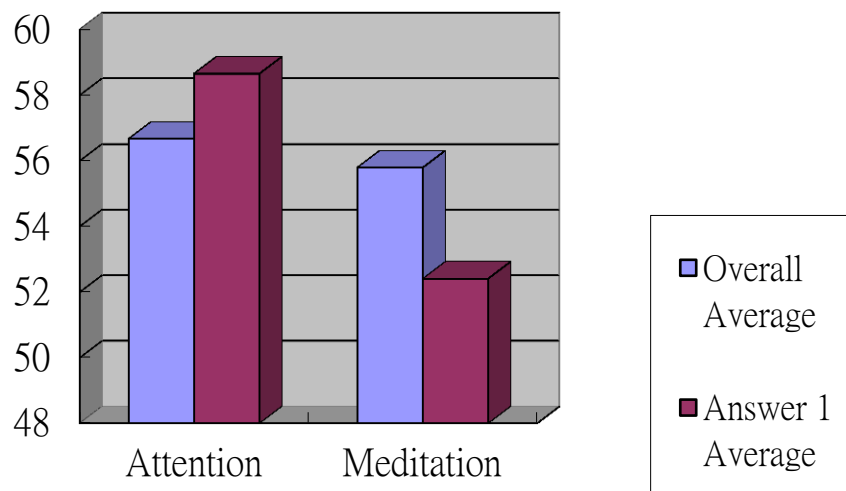


Figure 3.4.3-8 Compare overall average eSense values and part 3 average values for concentrated participants watching answer 1

The average attention during answer 1 is higher than overall average value. The average meditation value during answer 1 is lower than overall average value.

Compare overall average eSense values and part 3 average values for concentrated participants watching answer 2

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

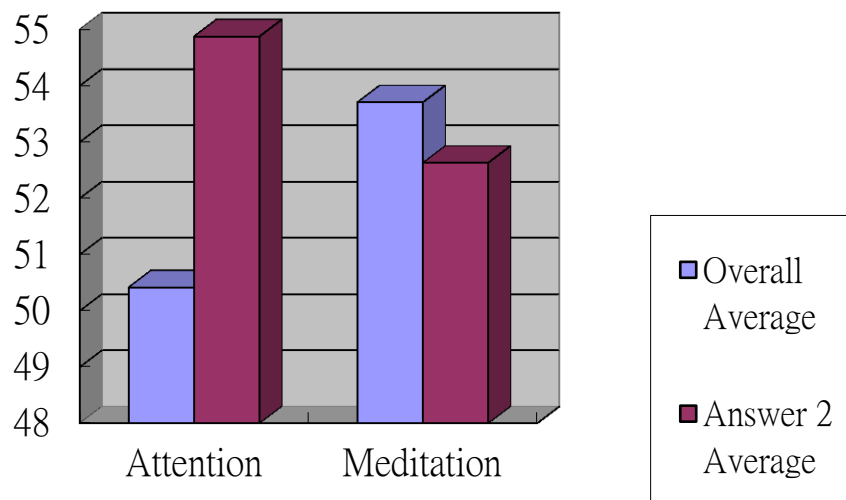


Figure 3.4.3-9 Compare overall average eSense values and part 3 average values for concentrated participants watching answer 2

The average attention during answer 2 is higher than overall average value. The average meditation value during answer 2 is lower than overall average value.

Table of Comparing average eSense values for each part with overall average values

	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

Conclusion

Some participants are concentrated at question 1, question 2, 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.

For attention at question 1, 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, the average value at part 3 is lower than overall average values. This observation may be caused by the bikini girl in the movie. Participants become less calm when they saw the bikini girl.

Part 4: Ghost Pop Up

Part 4 average attention and mediation for all participants

	Average
Attention	57.95749
Meditation	51.58219

Compare overall average eSense values and part 4 average values for all participants

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

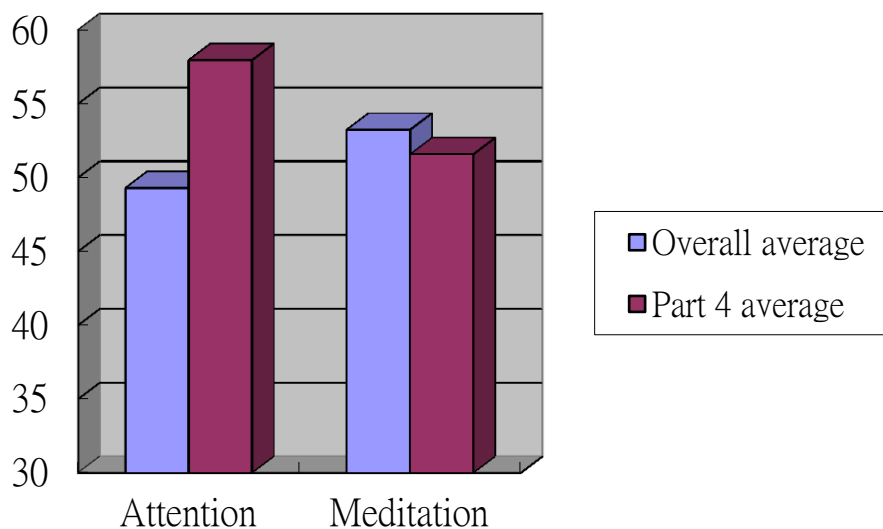


Figure 3.4.3-10 Compare overall average eSense values and part 4 average values for all participants

The average attention value for participants in part 4 is higher than overall average value.
 The average meditation value for participants in part 4 is lower than overall average values.

Compare attention and meditation values before and after the ghost pop up

About half of the participants' emotion change from concentrated to fear. Only data of participants' emotions changes from concentrated to fear is used.

Compare average values before and after the ghost appear for participants' feeling changes 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>

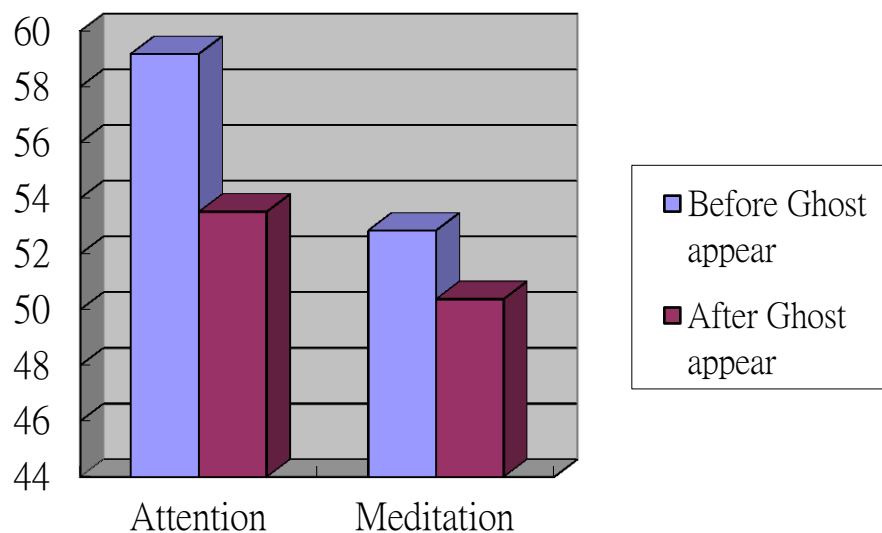


Figure 3.4.3-11 Compare attention and meditation values before and after the ghost pop up

The average attention and meditation values both decreased after the ghost pop up.

Conclusion

In this movie, the participants are told that there will be a ghost popped up 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.

2.3.4. Conclusion for result and 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. Building Our BCI Computer Game

3.1. Introduction to Game Engines

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

3.1.1. 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 3.1.1-1 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.

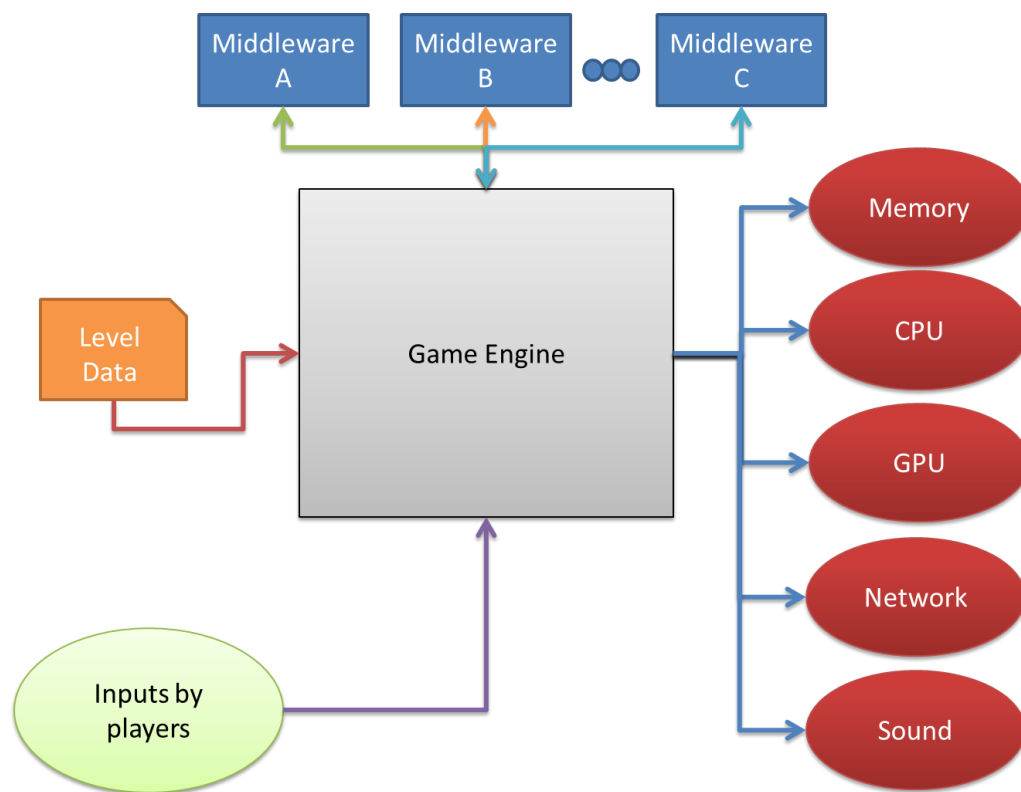


Figure 3.1.1-2 Simplified Context Diagram of Game Engines

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 3.1.1-3 Pseudo codes for a simple game engine

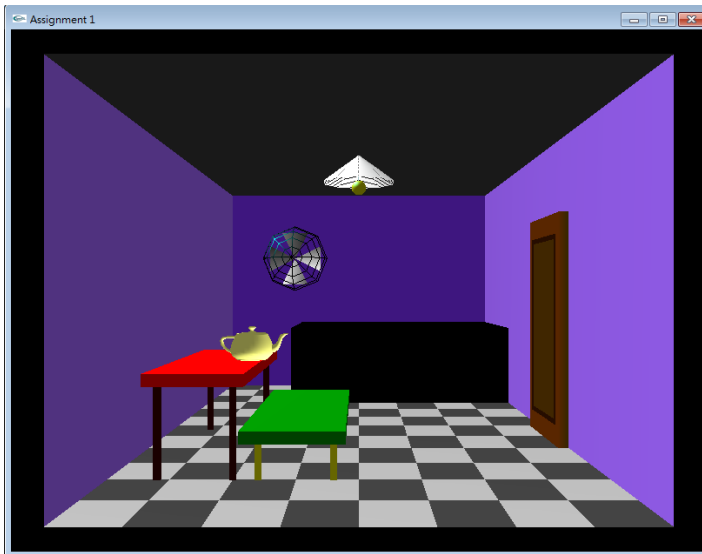


Figure 3.1.1-4 Using OpenGL alone to create an interactive scene offered a taste of the principle of game engines, but codes are explicitly written to control objects in a relatively low-level manner and hard to reuse. (CSC3260 Assignment 1 2009-2010, done by K.C. LIU)

(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

in-game events by harness the power of game engines (See Figure 4.1.2-2).

3.1.2. 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 3.1.2-1 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 3.1.2-2 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.

3.2. 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 3.1.2-1 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 3.1.2-2 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 3.1.2-4 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)



Figure 3.1.2-3 Dungeon Defenders - an online multiplayer strategy game powered by Unreal Engine 3, demonstrating flexibility in graphics and types of games developed.

games but there has been more than a hundred games being 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.

3.3. 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. ☺

3.3.1.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

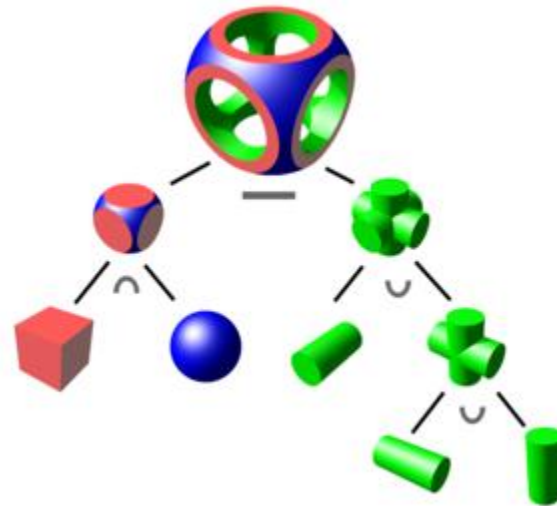


Figure 3.3.1-1 Effects of CSG operators (from Wikipedia)

BSP is a data structure of storing such solid geometry due to its efficiency to be

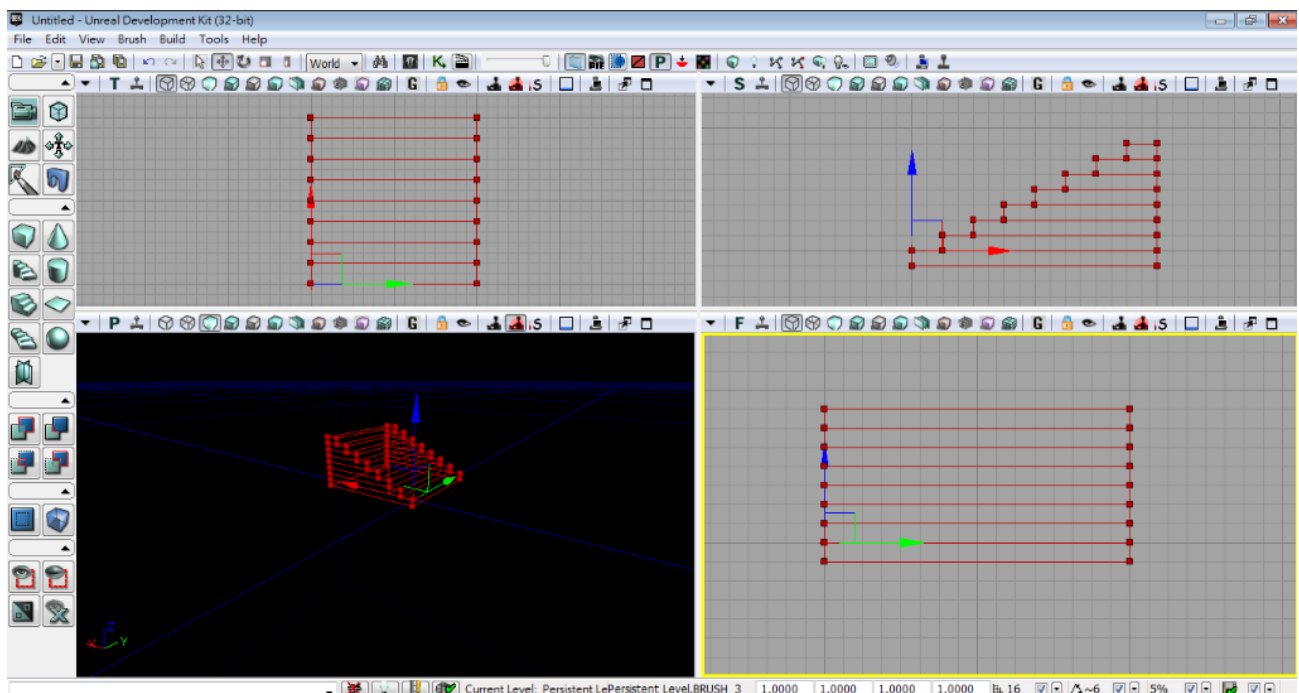


Figure 3.3.1-2 UDK (Editor)

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

3.3.2. 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.

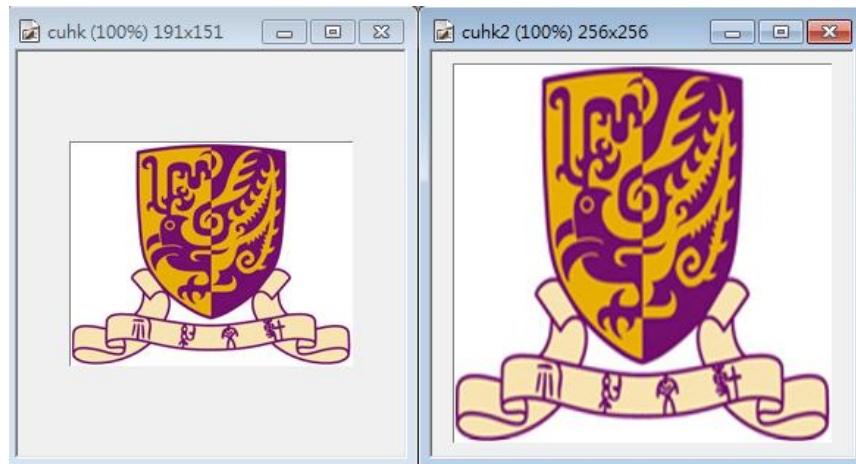


Figure 3.3.2-1 Bigger pictures with lengths of powers of 2 (Right) are most suitable for MIP-mapping in UDK because more MIP maps could be generated without precision loss

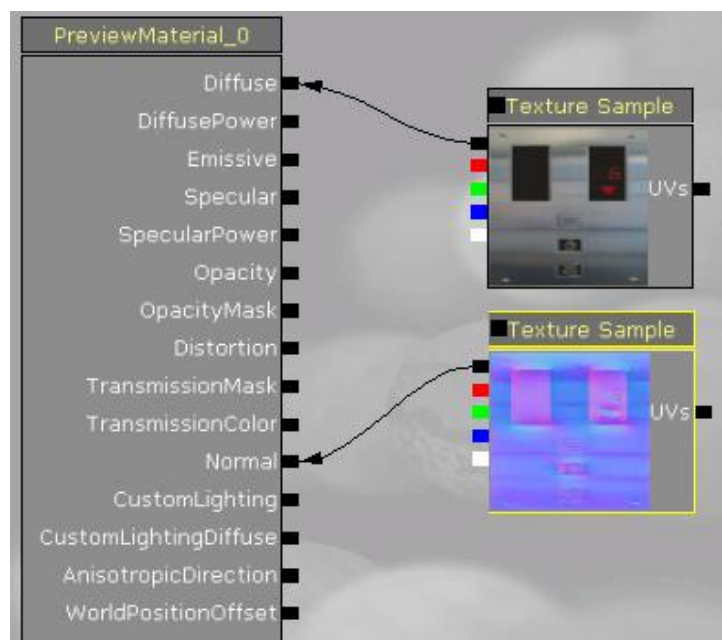


Figure 3.3.2-2 Material Editor in UDK

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 /

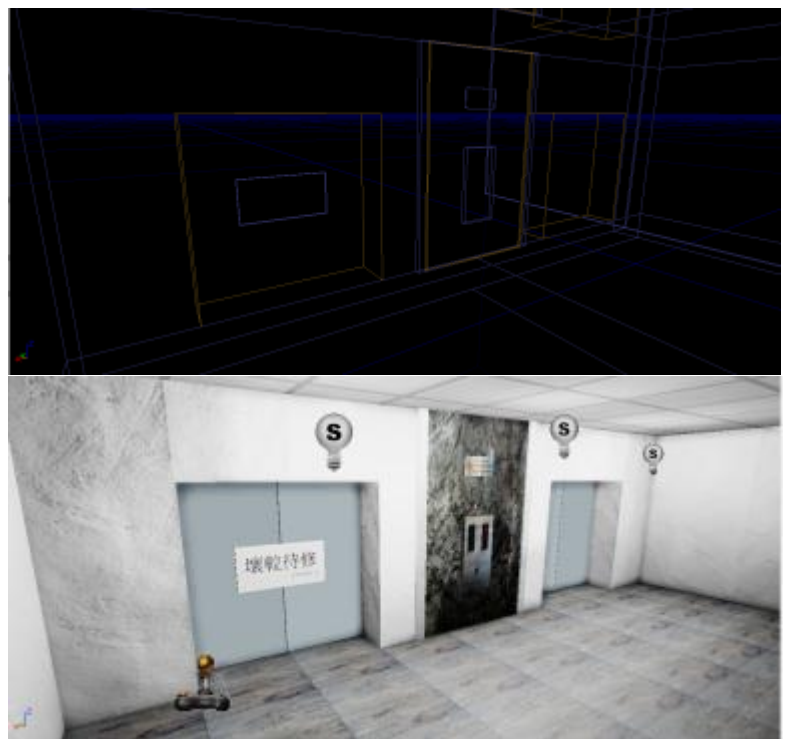


Figure 3.3.2-3 Wireframe of BSP models (Top) and the models after texture-mapping (Bottom)

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.

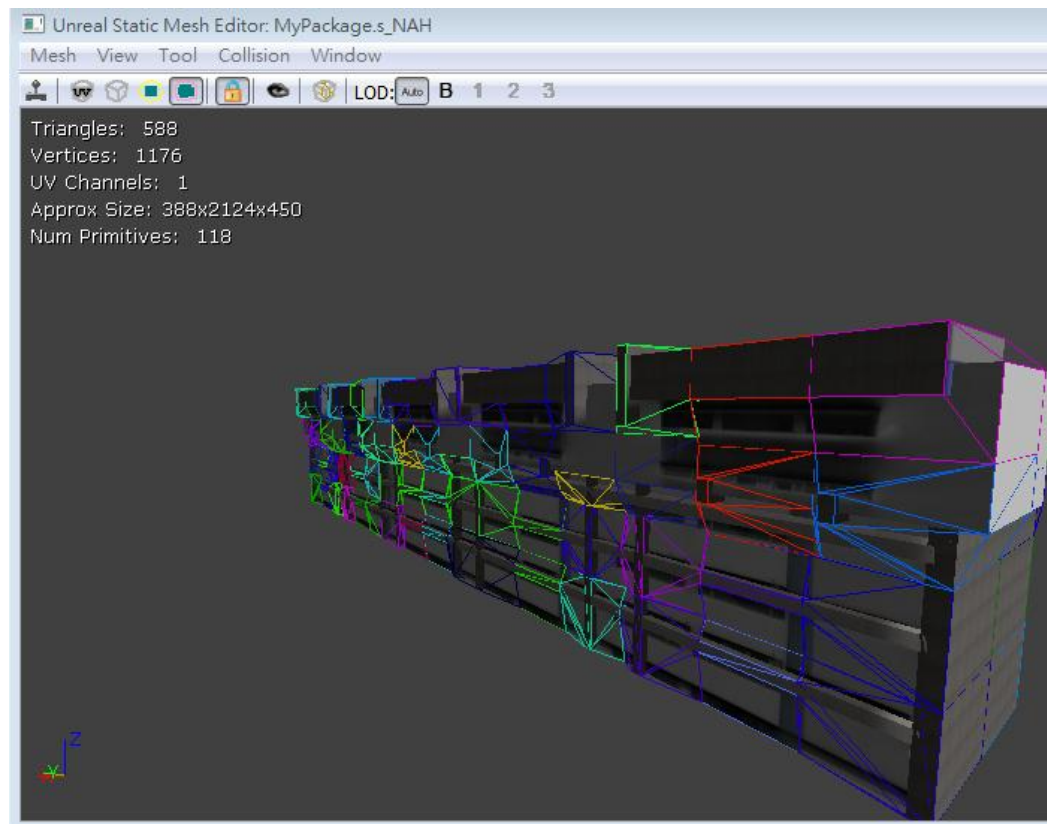


Figure 3.3.2-4 Replicating the Humanity Building of [New Asia College of CUHK](#), with collision information

3.3.3. 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.

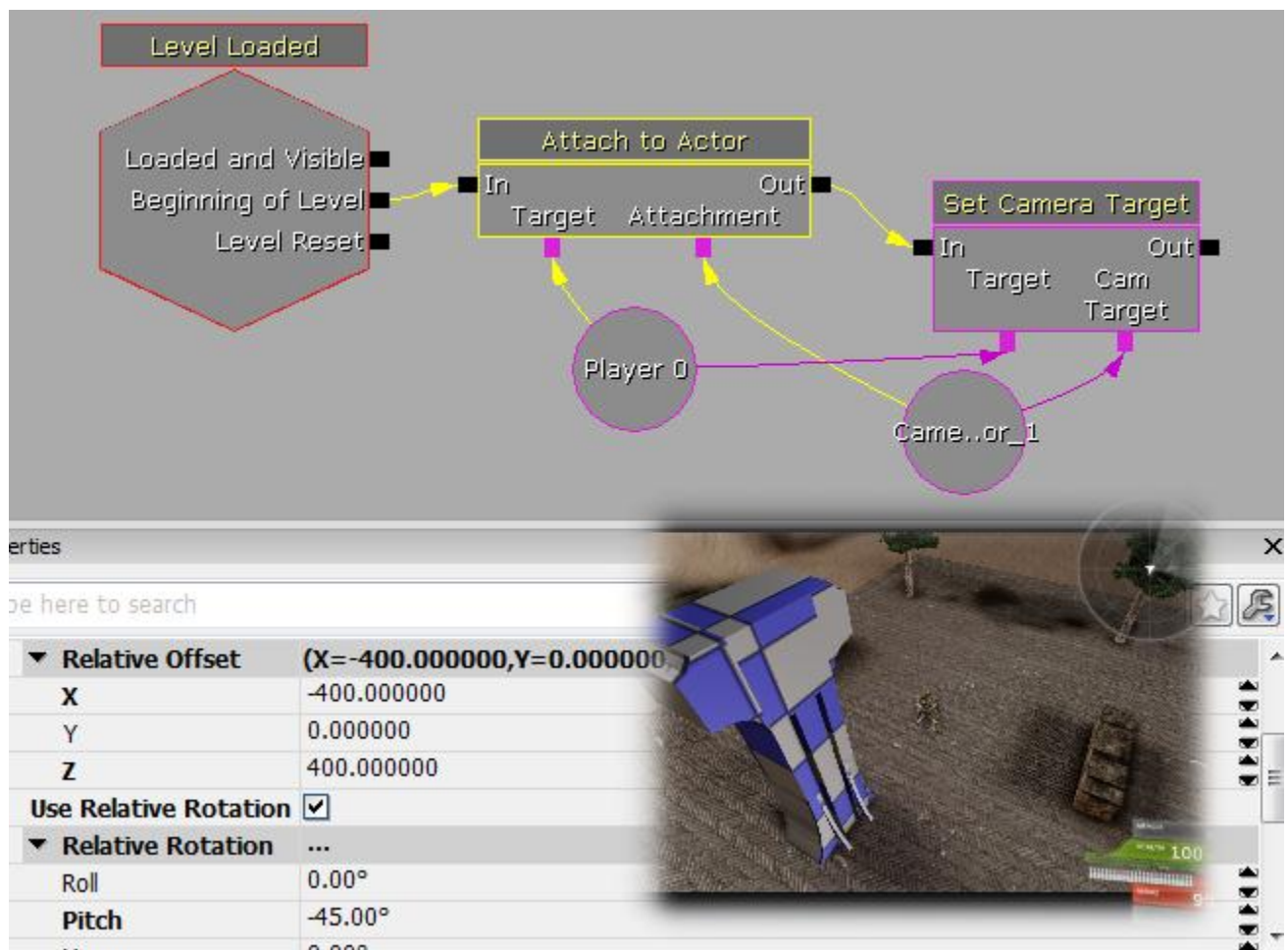


Figure 3.3.3-1 Third-person view done in Kismet (Outer) and its effect (Bottom-Right)

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.

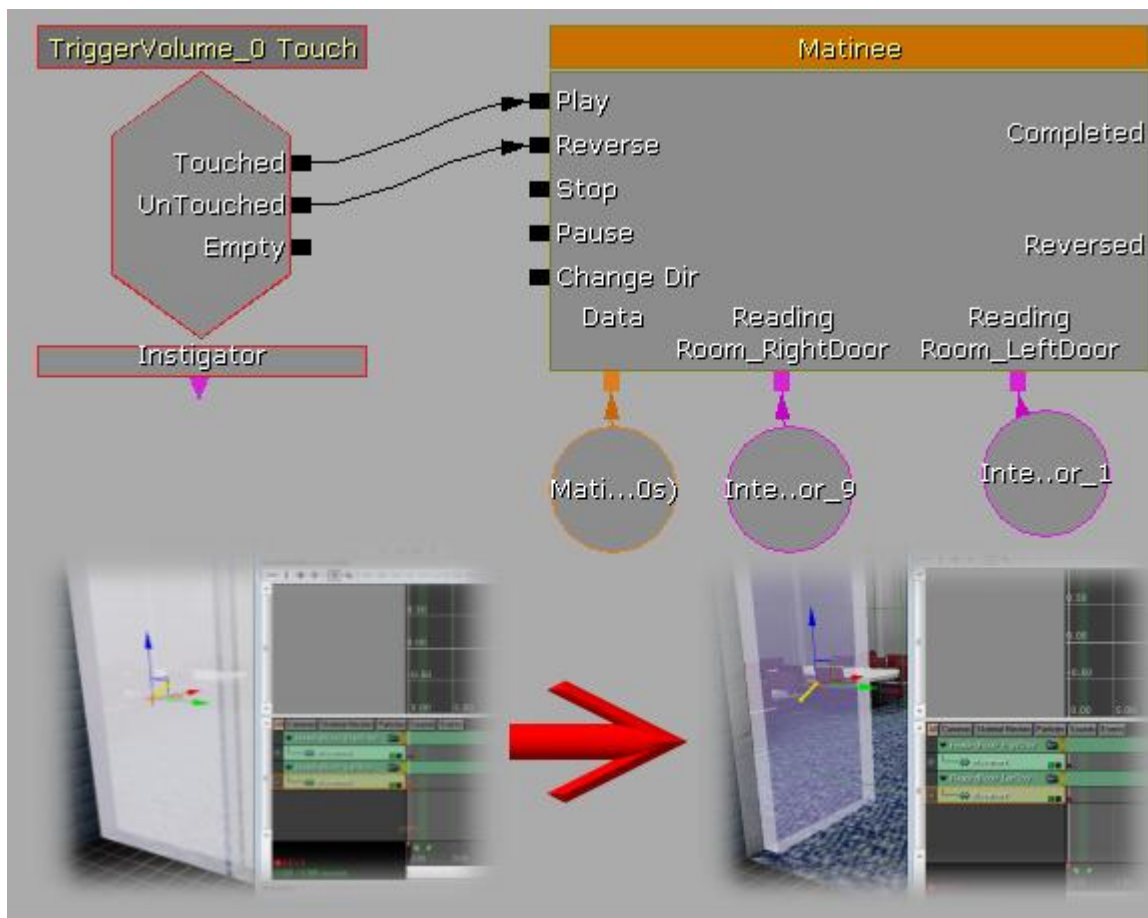


Figure 3.3.3-2 Kismet setup (Top) and keyframes setup in Matinee (Bottom)

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.

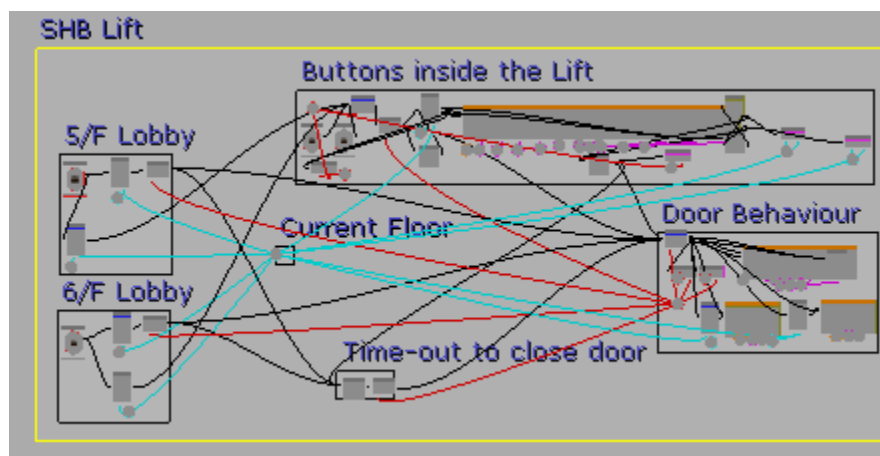


Figure 3.3.3-3 Confusing interface in Kismet just to control a lift travelling between 2 floors.

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.

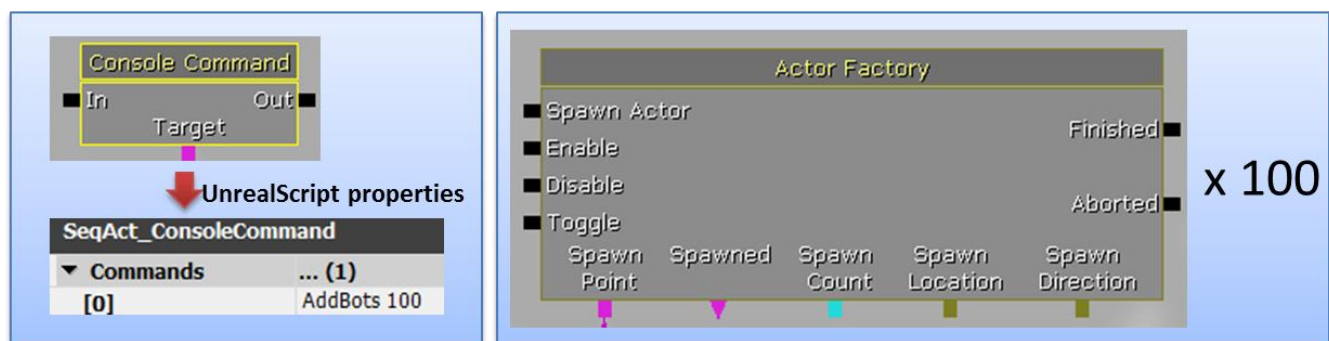


Figure 3.3.3-4 UnrealScript is highly scalable - 1 customized UnrealScript module (Left) can be as powerful as 100 or even more repeating "default" Kismet modules (Right)

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!

3.4. Integration of UDK and Mindset

3.4.1. 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.

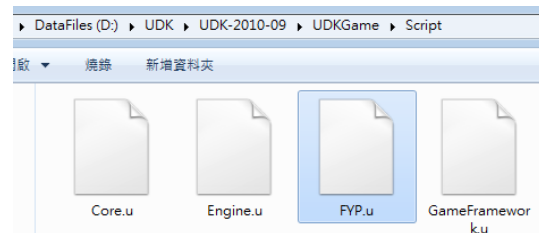


Figure 3.4.1-1 UnrealScript byte-codes

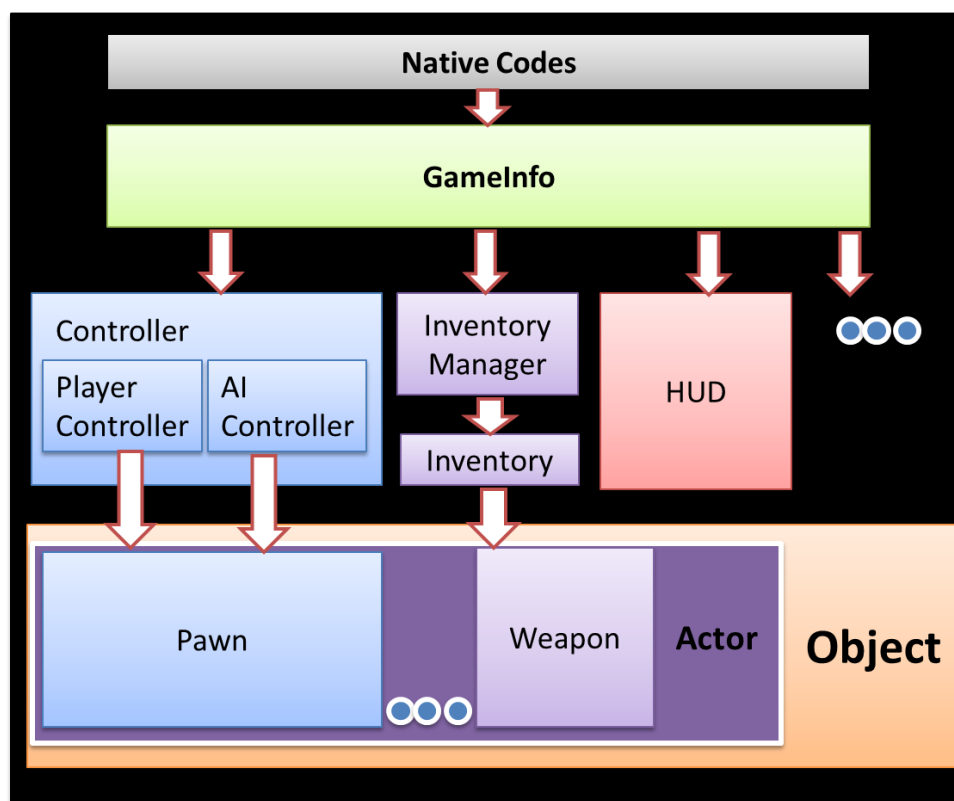


Figure 3.4.1-2 Simplified Abstract view of classes of the scripting engine (e.g. many classes are actually actors, including GameInfo itself)

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.

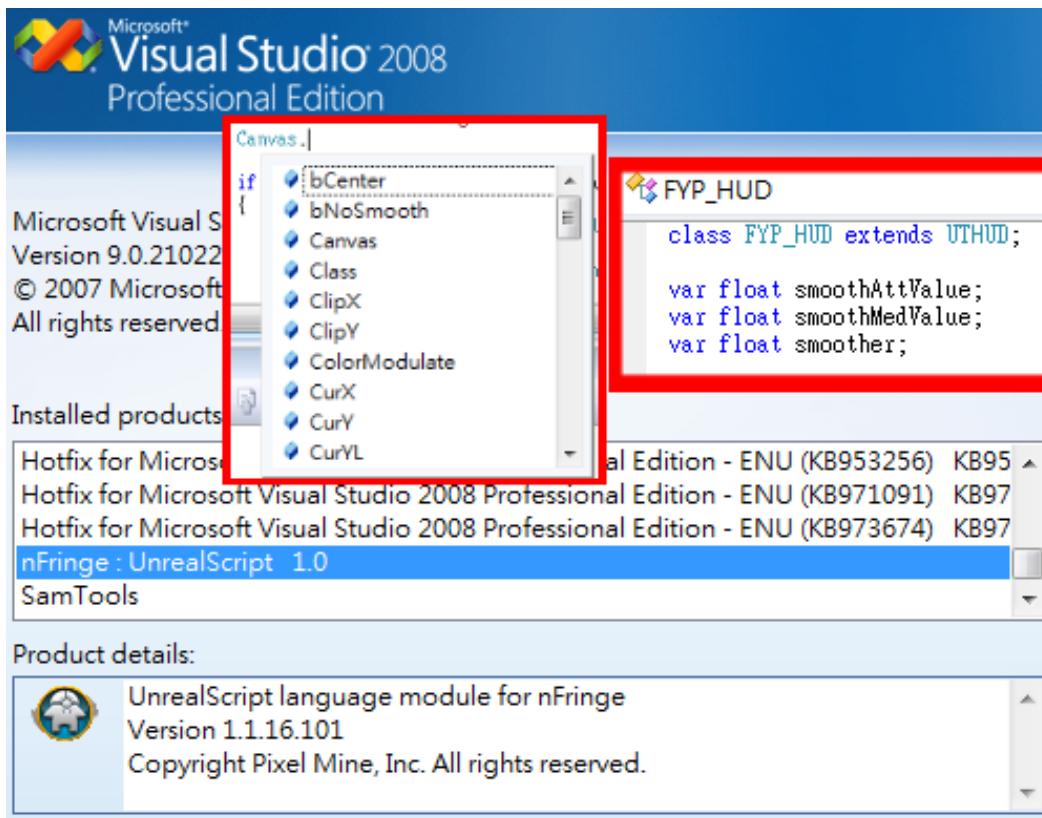


Figure 3.4.1-3 nFringe enables syntax highlighting and Auto-complete for UnrealScript

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.

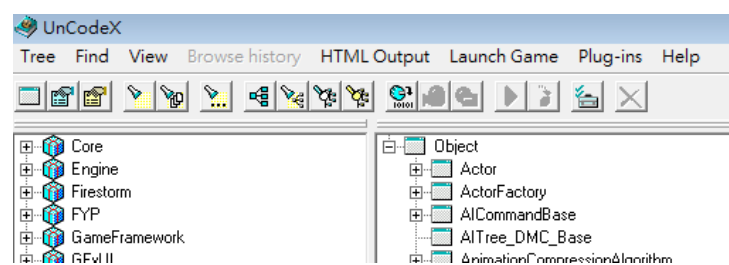


Figure 3.4.1-4 UnCodeX lists all classes out as trees.

3.4.2. 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.

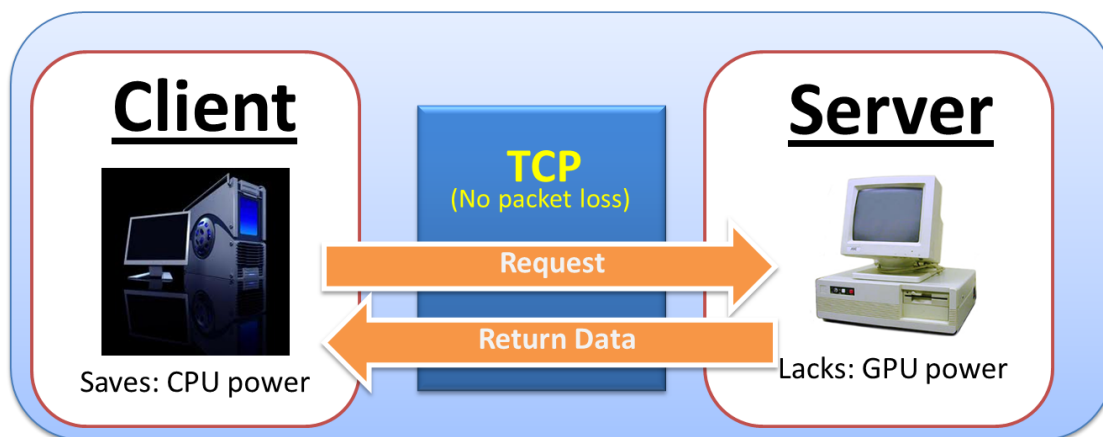


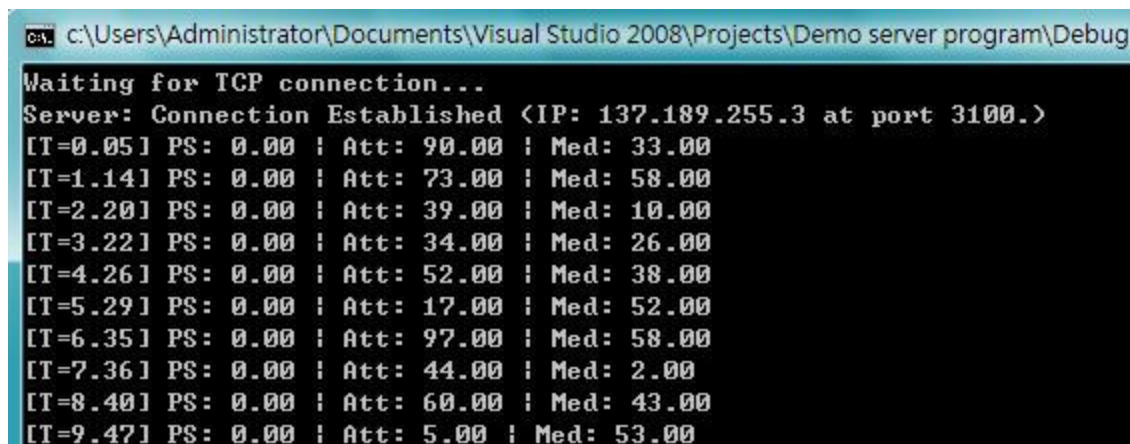
Figure 3.4.2-1 TCP Model – with the server connecting to Mindset

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).



```

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=0.05] PS: 0.00 | Att: 90.00 | Med: 33.00
[T=1.14] PS: 0.00 | Att: 73.00 | Med: 58.00
[T=2.20] PS: 0.00 | Att: 39.00 | Med: 10.00
[T=3.22] PS: 0.00 | Att: 34.00 | Med: 26.00
[T=4.26] PS: 0.00 | Att: 52.00 | Med: 38.00
[T=5.29] PS: 0.00 | Att: 17.00 | Med: 52.00
[T=6.35] PS: 0.00 | Att: 97.00 | Med: 58.00
[T=7.36] PS: 0.00 | Att: 44.00 | Med: 2.00
[T=8.40] PS: 0.00 | Att: 60.00 | Med: 43.00
[T=9.47] PS: 0.00 | Att: 5.00 | Med: 53.00

```

Figure 3.4.2-2 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.

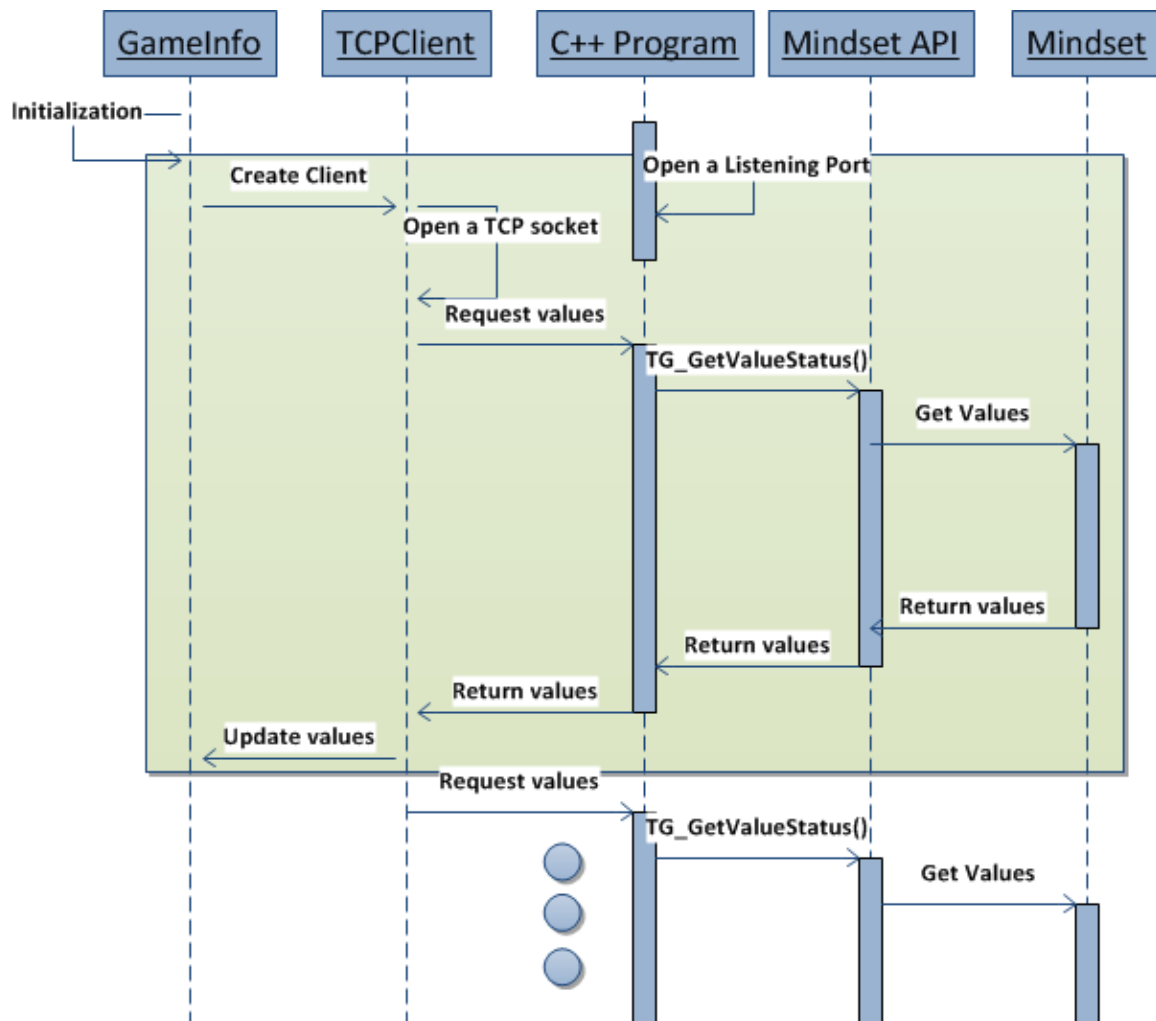


Figure 3.4.2-3 Sequence Diagram of BCI values update

3.4.3. 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 3.4.3-1 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.

3.4.4. 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 3.4.4-1 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 3.4.4-2 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!

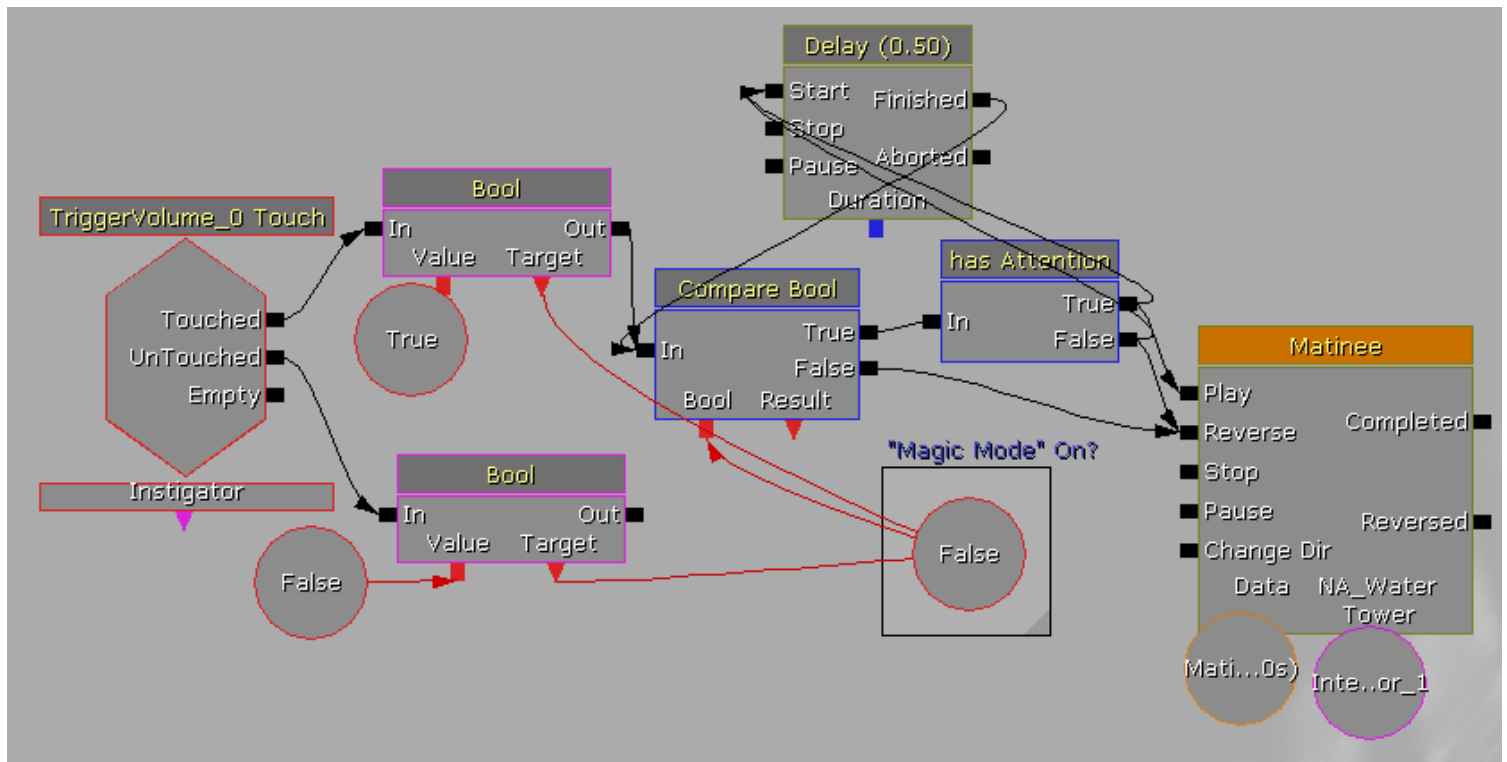


Figure 3.4.4-3 Kismet sequence for an active BCI control

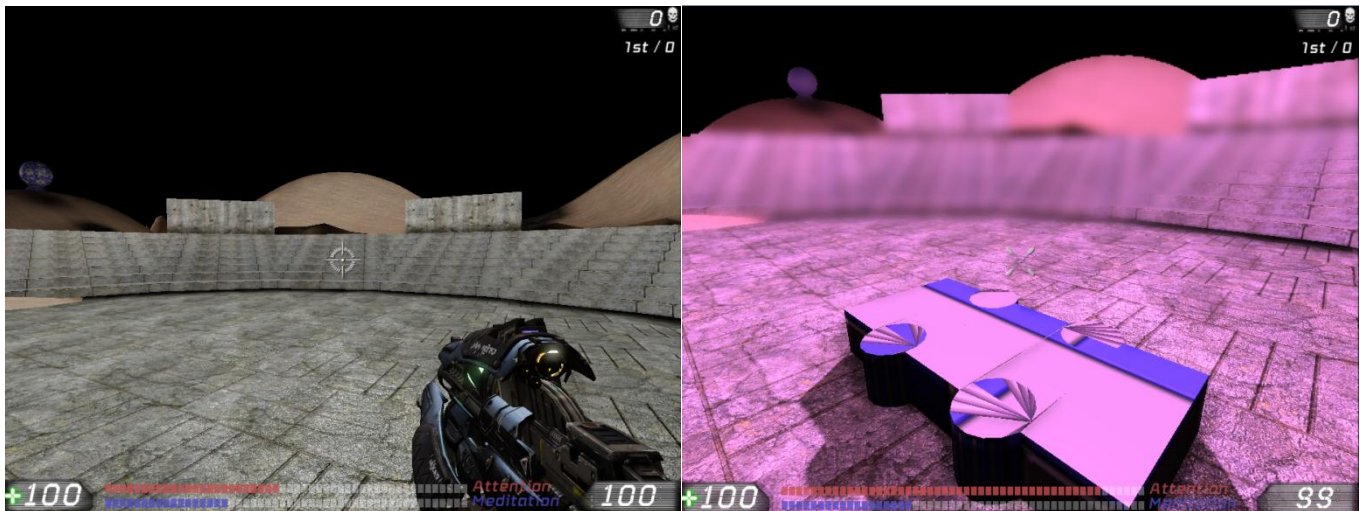


Figure 3.4.4-4 Effects of magic mode being turned off (Left) and on (Right) due to current attention level (Realtime display of attention and meditation are shown by red and blue bars at the bottom respectively)

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!

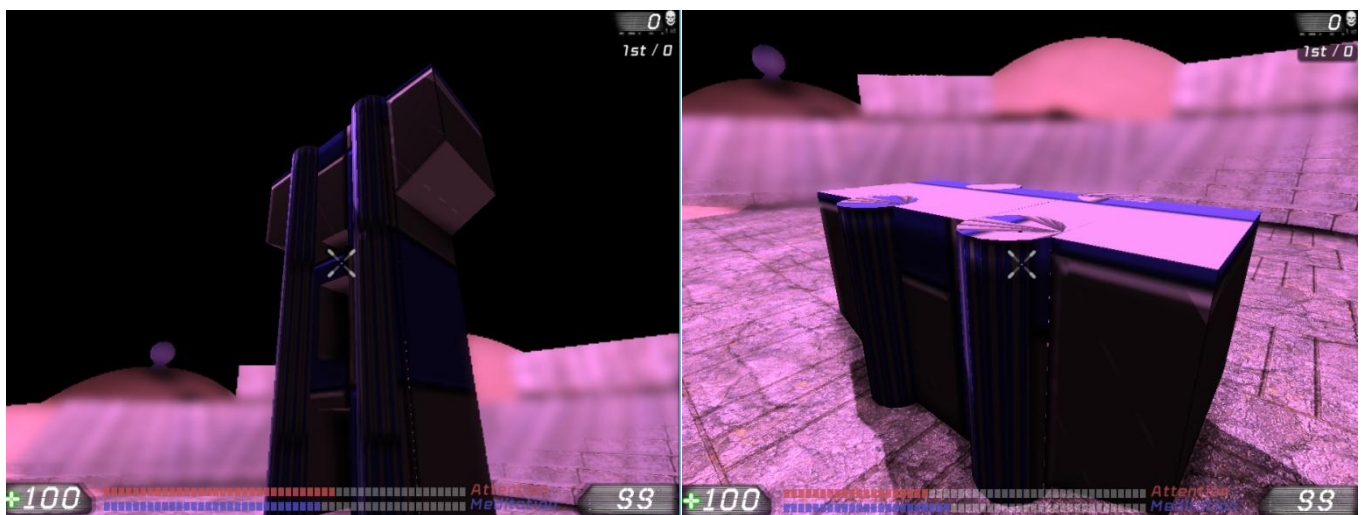


Figure 3.4.4-5 Tower being raised high (Left) and the effect of declined attention level (Right)

4. Conclusion

We have performed market research and while the game market is growing, there is no commercially available BCI-enhanced game.

We first investigated the apparent correlation between Mindset eSense values and users emotion, and the validity of Mindset eSenses. From the experiment, we found that the attention value is correlated with the two mental states of participants (concentration and boring). The attention value is higher when participant is concentrated and the attention value is lower when the participant is bored. On the other hand, the experiment on meditation is not conclusive. This result suggests that attention is more suitable to develop a passive control in the next semester. The possibility of meditation value needs further investigation.

Since Mindset seems to generate values of certain level of validity, we can proceed to create a BCI-enhanced game using the values read from it.

With UnrealScript programming, we built components in the Unreal Development Kit, which communicates via TCP with the Neurosky Mindset to get brain waves related information such as eSenses. This proves the feasibility of connecting at least one game engine with a BCI, showing the possibility of producing BCI-enhanced games.

Furthermore, we defined the two types of controls which utilize the Mindset data – “Active” / “Passive” controls, and then proceeded to conclude that active controls are feasible, by making use of Kismet and UnrealScript to trigger in-game events using eSenses.

5. Limitation & Difficulties

5.1. 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.

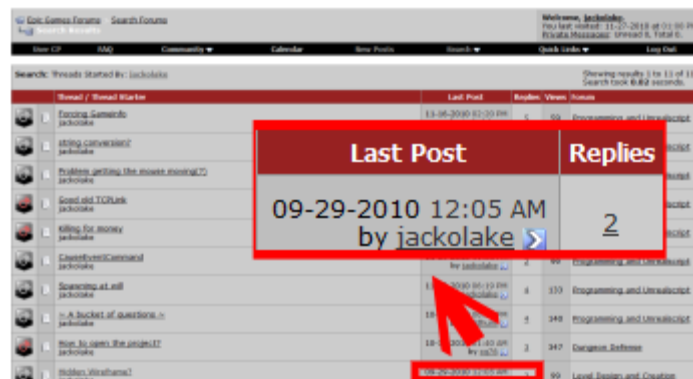


Figure 5.1-1 Long list of threads Jackie was posted onto UDK forum, but there had been only a few replies, and the replies were always detailed enough.

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

5.2. Poor signal from Mindset

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.

6. Future Work

6.1. Passive Controls

If we monitor the states of the player for a longer period of time, the data will be of higher decisive power, which enables us to change some of the game settings or features automatically.

For instance, while current games available on the market usually offer an option for the players to change the “Game Difficulty” in game menus, none of them could do it automatically. If we can monitor the level of frustration or anxiety of the player for a longer period of time, we may conclude that the game difficulty is currently too high for the skills of the player for that timeframe, thus automatically adjust the game difficulty downward.



Figure 6.1-1 Static Game Difficulty Settings in the game DragonballZGT



Figure 6.1-2 Fallout 3 has speech checks based on almost random chances (e.g. “34%”).
We want to change that.

Another ideas we came up with, is the ability to detect lies. In some RPG games such as the Fallout series, there are usually skills or abilities which govern the character’s power to convince an NPC (Non-player character) over something, such as persuading a shopkeeper to sell items at a lower price because you use the items to

fight a righteous war. However, the chances of success are usually random. If we can find a brain wave pattern corresponding to “lying”, we could further increase the Role-Playing content by letting the player interacts with the NPC - if the BCI detects a “lying” signal throughout the conversation, the speech check is failed.

There are many more passive control ideas, and should be **revolutionary** to the game industry.

6.2. Small-scaled BCI-enhanced Game

With the development of both active and passive controls on hand, we would like to leap onto creating a small-scaled BCI-enhanced game to further prove that making BCI-enhanced games are very feasible.

The verdict is, if we could do it with a team of 2 students, game giants also can do it; the difference will probably be the scale of the game and the artworks, as we have very limited resources and time relatively.

Given the time constraint, we would not include a lot of scripted events and objects, and it would probably be a large single-map game, as it will be easier to manage.

Moreover, we will invite fellow students and postgraduate students in CUHK to try out the game, and ask for their feedbacks. Short questionnaires would be distributed. After the collection of their comments, we would summarize our project by evaluating how BCI affects the gameplay.

7. References

- 1) 2009 Neurosky, I. (2009, June 19). *mindset_instruction_manual.pdf*.
- 2) 2009 Neurosky, I. (2009). *quick_start.pdf*.
- 3) 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>
- 4) DevMaster.net. (2010, 11 29). *List All Engines*. Retrieved 11 29, 2010, from DevMaster.net: <http://www.devmaster.net/engines/list.php>
- 5) 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>
- 6) 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>
- 7) 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/>
- 8) Gregory, J., & Lander, J. (2009). Third-party SDKs and Middleware. In J. Gregory, & J. Lander, *Game Engine Architecture* (p. 31). A K Peters, Ltd.
- 9) Kenner, C. (2009, 4 1). *Emotivated vs Neurosky*. Retrieved 11 28, 2010, from Facebook: <http://www.facebook.com/topic.php?uid=9489703974&topic=8230>
- 10) 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)*.

- 11) 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*.
- 12) Moon, T. K., & Stirling, W. C. (1999). *Mathematical Methods and Algorithms for Signal Processing*. Prentice Hall.
- 13) NVIDIA. (2010, 1 17). *NVIDIA PhysX SDK Features*. Retrieved 11 30, 2010, from NVIDIA Developer Zone:
http://developer.nvidia.com/object/physx_features.html
- 14) OGRE. (2010). *Testimonials*. Retrieved 11 29, 2010, from OGRE 3D:
<http://www.ogre3d.org/about/testimonials>
- 15) 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
- 16) PixelMineGames. (2010, 11 11). *nFringe:Features*. Retrieved 11 30, 2010, from PixelMineGames:
<http://wiki.pixelminegames.com/index.php?title=Tools:nFringe:Features>
- 17) Porter, J. (2010). *UDN*. Retrieved 11 30, 2010, from Calling DLLs from UnrealScript (DLLBind): <http://udn.epicgames.com/Three/DLLBind.html>
- 18) 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.
- 19) Scaleform.com. (2010). *Scaleform GfX Core Technology*. Retrieved 11 30, 2010, from Scaleform Corporation: <http://www.scaleform.com/products/gfxtech>
- 20) Stevens, W. R., & Wright, G. R. (1994). *TCP/IP illustrated: The protocols*. Canada: Addison Wesley.

- 21) Tatum, W. O. (2008). *Handbook of EEG Interpretation*. Demos Medical Publishing.
- 22) 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.
- 23) 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
- 24) 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.
- 25) 陳國志, 楊文鎮, 張藝英, 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.

8. Appendix

8.1. Our Efforts in This Semester

We do not wish to explicitly record the number of hours every week, because this kind of recording does not add value to our progress.

If numbers are really required, we are sure that we have spent at least 20 hours working on this project every week on average.

We are glad that our supervisor Prof. Michael Lyu really cares about our team and has been spending his precious time to discuss with us, review and guide our progress on a **weekly** basis, during 1.5 hours long sessions.

We are confident about the recognition by Prof. Lyu on our efforts.

Anyway, the amount of work to be done in the next semester will also be huge; especially on the aspects of game design (even level editing would consume a large amount of time). Therefore, we will strive to push ourselves to our limits so that there won't be room for us to regret in the future.

8.2. Questionnaire

Brain Waves Sampling Feedback Form

*** Required**

For Clip#1 (the string orchestra), which of the followings best describes your general emotion? *

☐ Bored

☐ Concentrated

☐ Excited

☐ Fear

☐ Relaxed

☐ Other:

For Clip#1 (the string orchestra), a strike of loud sound was played, which of the followings best describes your emotion at that moment? *

☐ Bored

☐ Concentrated

☐ Excited

☐ Fear

☐ Relaxed

☐ Other:

Which of the followings best describes your emotion AFTER that? *

☐ Bored

☐ Concentrated

☐ Excited

☐ Fear

☐ Relaxed

☐ Other:

For Clip#2 (the Pope's speech), which of the followings best describes your general emotion? *

- ☐ Bored
- ☐ Concentrated
- ☐ Excited
- ☐ Fear
- ☐ Relaxed
- ☐ Other:

For Clip#2 (the Pope's speech), which of the followings best describes your general emotion for the first half of the clip? *

- ☐ Bored
- ☐ Concentrated
- ☐ Excited
- ☐ Fear
- ☐ Relaxed
- ☐ Other:

Which of the followings best describes your general emotion for the remaining part of the clip? *

- ☐ Bored
- ☐ Concentrated
- ☐ Excited
- ☐ Fear
- ☐ Relaxed
- ☐ Other:

For the first test of Clip#3 (the Bikini), which of the followings best describes your emotion BEFORE you got the answer? *

"Got the answer" = Spotting the difference by yourself OR reading the answer

- ☐ Bored
- ☐ Concentrated
- ☐ Excited
- ☐ Fear
- ☐ Relaxed
- ☐ Other:

Which of the followings best describes your emotion AFTER you got the answer? *

"Got the answer" = Spotting the difference by yourself OR reading the answer

- ☐ Bored
- ☐ Concentrated
- ☐ Excited
- ☐ Fear
- ☐ Relaxed
- ☐ Other:

On which attempt did you got the answer? *

"Got the answer" = Spotting the difference by yourself OR reading the answer

- ☐ First attempt
- ☐ Second attempt
- ☐ Third attempt
- ☐ Had to read the answer

For the second test of Clip#3 (the Bikini), which of the followings best describes your emotion BEFORE you got the answer? *

"Got the answer" = Spotting the difference by yourself OR reading the answer

- ☐ Bored
- ☐ Concentrated
- ☐ Excited
- ☐ Fear
- ☐ Relaxed
- ☐ Other:

Which of the followings best describes your emotion AFTER you got the answer? *

"Got the answer" = Spotting the difference by yourself OR reading the answer

- ☐ Bored
- ☐ Concentrated
- ☐ Excited
- ☐ Fear
- ☐ Relaxed
- ☐ Other:

On which attempt did you get the answer? *

"Got the answer" = Spotting the difference by yourself OR reading the answer

- ☐ First attempt
- ☐ Second attempt
- ☐ Third attempt
- ☐ Had to read the answer

For Clip#4 (Spot the ghost), which of the followings best describes your general emotion BEFORE the ghost popped out? *

- ☐ Bored
- ☐ Concentrated
- ☐ Excited
- ☐ Fear
- ☐ Relaxed
- ☐ Other:

Which of the followings best describes your general emotion AFTER the ghost popped out? *

- ☐ Bored
- ☐ Concentrated
- ☐ Excited
- ☐ Fear
- ☐ Relaxed
- ☐ Other:

8.3. Brainwaves sampling Feedback

	For Clip#1 (the string orchestra), a strike of loud sound was played, which of the followings best describes your general emotion?	For Clip#1 (the string orchestra), a strike of loud sound was played, which of the followings best describes your emotion at that moment?	Which of the followings best describes your emotion AFTER that?	For Clip#2 (the Pope's speech), which of the followings best describes your general emotion?	For Clip#2 (the Pope's speech), which of the followings best describes your general emotion for the first half of the clip?	Which of the followings best describes your general emotion for the remaining part of the clip?	For the first test of Clip#3 (the Bikini), which of the followings best describes your emotion BEFORE you got the answer?
Relaxed	Excited	Normal	Concentrated	Concentrated	Concentrated	Bored	anxious
Bored	Bored	Bored	Bored	Bored	Bored	Bored	Bored
Relaxed	Excited	Concentrated	Bored	Bored	Bored	Bored	Concentrated
Relaxed	Relaxed	Bored	Bored	Bored	Bored	Bored	Concentrated
Relaxed	Excited	Bored	Bored	Bored	Concentrated	Excited	Bored
Relaxed	Excited	Excited	Excited	Bored	Bored	Bored	Excited
Relaxed	Fear	Excited	Excited	Bored	Bored	Bored	Bored
Excited	Excited	Excited	Bored	Bored	Bored	Bored	Concentrated
Relaxed	shocked	Normal	Bored	Bored	Bored	Bored	Normal
Bored	Surprised	Relaxed	Bored	Bored	Bored	Bored	Bored
Fear	Excited	Fear	Bored	Bored	Fear	Fear	Bored
Relaxed	Excited	Relaxed	Bored	Bored	Bored	Concentrated	Concentrated
							have not reading the question
Relaxed	Fear	Concentrated	Bored	Bored	Concentrated	Bored	Bored
Bored	Bored	Bored	Bored	Concentrated	Bored	Relaxed	Relaxed
Relaxed	Excited	Relaxed	Bored	Bored	Bored	Bored	Excited
Relaxed	Excited	Concentrated	Relaxed	Relaxed	Relaxed	Fear	Concentrated
Bored	Excited	Concentrated	Bored	Bored	Bored	Bored	Concentrated
Bored	Concentrated	Concentrated	Bored	Bored	Bored	Concentrated	Concentrated
Bored	Bored	Bored	Bored	Bored	Bored	Bored	Concentrated

Which of the following best describes your emotion AFTER you got the answer?	On which attempt did you get the answer?	For the second test of Clip#3 (the Bikini), which of the following best describes your emotion BEFORE you got the answer?	Which of the following best describes your emotion AFTER you got the answer?	On which attempt did you get the answer?	For Clip#4 (Spot the ghost), which of the following best describes your general emotion BEFORE the ghost popped out?
Excited	Had to read the answer	anxious	Excited	Had to read the answer	Concentrated
Bored	First attempt	Bored	Bored	First attempt	Bored
Concentrated	Had to read the answer	Concentrated	Concentrated	Had to read the answer	Concentrated
Relaxed	Had to read the answer	Concentrated	Concentrated	Had to read the answer	Concentrated
Concentrated	Had to read the answer	Concentrated	Concentrated	Had to read the answer	Concentrated
Excited	Had to read the answer	Concentrated	Relaxed	Had to read the answer	Bored
Bored	Had to read the answer	Bored	Bored	Had to read the answer	Concentrated
Concentrated	Had to read the answer	Concentrated	Concentrated	Had to read the answer	Fear
Normal	Had to read the answer	Normal	Normal	Had to read the answer	Normal
Bored	Third attempt	Bored	Relaxed	Third attempt	Fear
Bored	Had to read the answer	Fear	Concentrated	Had to read the answer	Fear
Relaxed	Had to read the answer	Concentrated	Relaxed	Had to read the answer	Concentrated
have not reading the question	Had to read the answer	have not reading the question	have not reading the question	Had to read the answer	Concentrated
Concentrated	Had to read the answer	Concentrated	Concentrated	Had to read the answer	alerted
Concentrated	Had to read the answer	Relaxed	Suprised	Had to read the answer	Concentrated
Concentrated	Had to read the answer	Concentrated	Excited	Had to read the answer	Concentrated
Relaxed	Had to read the answer	Concentrated	Relaxed	Had to read the answer	Concentrated
Relaxed	Had to read the answer	Concentrated	Relaxed	Had to read the answer	Fear
Excited	Had to read the answer	Concentrated	Excited	Had to read the answer	Concentrated

Which of the followings best describes your general emotion AFTER the ghost popped out?	What is your gender?	What is your age?
Fear	Male	21
Bored	Male	100
Fear	Male	23
Concentrated	Male	24
Fear	Female	23
Fear	Male	21
Fear	Male	21
Fear	Male	21
Shocked	Male	23
Excited	Male	23
Bored	Male	23
Fear	Male	24
Fear	Male	21
alerted	Male	19
Fear	Male	21
Excited	Female	21
Excited	Male	23
Fear	Male	21
Excited	Male	22