Team AGEA

Saving Private Bryan

A natural and intuitive multimodal game

Edwin Keijl, Andreea Sburlea, Gijs Van Veen & Adnan Niazi

Coordinator: Mannes Poel

University of Twente, Human Media Interaction Department

3 July 2011

Table of Contents

Contents

Introduction

With the development of technology and popularization of electronic products, video games have become an important part of the life of adolescents and adults alike. Worldwide, the video games sale is expected to top $48.9 billion in 2011 and $68 billion in 2012, making it one of fastest-growing component of the media sector worldwide.

With video games revenues outstripping even those of the movie industry, developers are keen, more than ever before, to design the next bestselling video game and gadget. A recent trend in such an attempt is to make the player experience more immersive by engaging him physically in the gameplay. Gone are the days when people used to passively control game characters using WASD key combinations on the keyboard. With the introduction of gadgets like Wii, Kinect and Emotiv, video games will no longer be played in the confines of a chair.

Although most video games could be played with either Emotiv or Kinect, there is still no concept game available that could combine the features of both Emotiv and Kinect. More precisely, we would like to see a game where the gesture capturing capabilities of Kinect could be combined with brain computer interfacing features of Emotiv, and speech recognition, to provide an immersive, fun and intuitive way of controlling a game. This is the main goal of this project.

The whole game development process was done in two iterations. In the first iteration, two game ideas in the form of paper mock-ups, were tested on 10 participants. Based on the user feedback, the best game was selected, developed and then tested on 20 participants in the second iteration of the design process. The results of the second iteration were used to refine the game to its final version.

The rest of the report goes into the details and specifics of these two iterations, followed by conclusion and suggestions for future work and improvements.

Design Iteration 1

As the main aim of this project is to develop an intuitive and natural game for widest possible audience, so we identified the target group as being between 12 and 65 year old, with the restriction of being healthy and having the ability of speaking, so as to be able to use Kinect and Emotiv. We began the first iteration of the design process by defining the user and system requirements which are further elaborated below.

1. User Requirements

R1.1: Controlling the game should feel natural

Because a multimodal game uses different modalities to control different aspects of the game, it should be clear to the user of the game, which modality is controlling which game features. A player who is new to the game can be given a short explanation, but further than that, the player should know how to react to different situations encountered in game by using the proposed modalities. The mapping of modalities to game feature should feel natural and intuitive to the user.

Furthermore, it should be obvious when a control has been used. This means that the actions performed by the player should clearly have an effect on the game (e.g., the character on screen moves around if the player is moving his body in front of the Kinect in that specific direction). The player should not lose control over the system.

R1.2: Playing the game using natural interaction should have some advantage over traditional controls

There should be some kind of benefit of using body movement and voice commands instead of the traditional mouse and keyboard or joystick. It could be that certain tasks are performed more efficiently when done with one modality than the other. For example, if the player doesn't have to press multiple buttons at the same time to move and cast spells, but can simply move his body and pronounce those spells. Furthermore, natural interaction could introduce a whole new form of controlling a game that is not possible using traditional input methods.

R1.3: Training the system should be minimal to be able to play the game

Evaluation sessions will most likely last only 15 to 30 minutes (including training, playing the game and evaluation), which means that the system should not take long to adjust itself to the player. The game should feature a short tutorial on how to play.

R1.4: The game should be single player

The game should be controlled by only one player at a time, so the player can focus on using natural interaction (and not be distracted by the actions of a second player). This will also help us to observe the effects of using natural interaction in the game.

R1.5: The game should be attractive, fun and challenging to play

The game should offer some kind of challenge to the player, which makes him want to play the game. The key issue here is that the challenge should come from using natural and intuitive interaction to solve problems, not by simply spawning thousand enemies to increase the difficulty.

2. System Requirements

Given below are some of the system requirements for running the proposed game:

R2.1: The hardware used to control the game consists of Microsoft Kinect, Emotiv EPOC and a form of speech recognition

The game should be able to use these devices as input controllers.

R2.2: Complimentary methodology should be used to control the game

The multiple methods of the game should be used complementary. That is, specific actions should be mapped to specific types of input rather than making the input all influence the same action.

R2.3: The speech recognizer should use a closed grammar

To simplify the task of speech recognition, a predetermined set of commands should be used to control some elements of the game by voice. As an example, the player could pronounce “fire” to cast a fire spell (from the set of “fire”, “water”, “lightning”, etc).

R3.3: The game should be programmed in C#.

To allow for efficient game the design and optimal development for Kinect, Microsoft’s XNA environment will be used. All components of the system should therefore be programmed using C# for easy code integration.

3. Game Design Ideas

Various game design ideas were perceived during an initial brainstorm session. However, only two game design ideas were materialized. It was decided to make paper mock-ups of both games, test both gameplays, and select one game for the final development according to user preference. A description of both is given below.

3.1. The Pirate Game

The Pirate game is an obstacle puzzle game controlled by all three input modalities. In the game the player indirectly controls a pirate ship that has to reach the Treasure Island. To reach the island however, obstacles such as enemy pirate ships and whirlpools have to be passed. Unlike most games however, the ship is not controlled directly by the player, who instead controls the water around the ship.

A major gameplay element of the game is the current of the sea. The current will lead the ship if not steered by the player and will increase in strength based on the weather conditions present in the game.

3.1.1. Control Modalities

Kinect: For navigating towards the Treasure Island

Player is standing up straight and moves his arms to influence the current of the water. The player can move his arms in eight directions in front of him (Up, Down, Left, Right and the diagonal combinations: Up Right, Up Left, Down Right, Down Left). By pointing in one of the eight directions, the player "pushes" the ship in that direction on the screen from a projective top-down view.

Ideally, the direction of the ship changes fluently by pushing in different directions with your arms, instead of changing discretely from "moving right" to "moving upwards" when moving your arms. This difference is shown in Figure 1. The speed of the ship slowly decreases when the player does not move his arms.

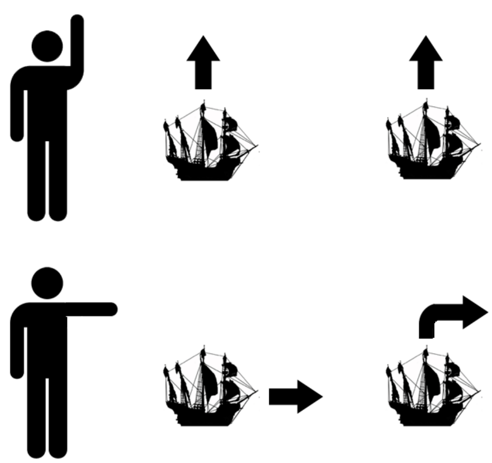


Figure 1 - Dynamic change instead of direct change based on arm position

When the ship collides with an object, such as an island or a ship, it stops moving. The player can change directions to make the ship move again. At high speeds, a crash might cause loss of health points.

Emotiv: For controlling the sea currents

The game screen is divided into a sort of grid. Each square of the grid has a certain vector assigned to it that represents the current of the water on that square. If the ship moves onto this square, the speed and direction of the ship itself is influenced by this vector. The length of the vector can vary based on the weather conditions. Figures 2 and 3 show the Sea as a vector grid and the game overlaying them.

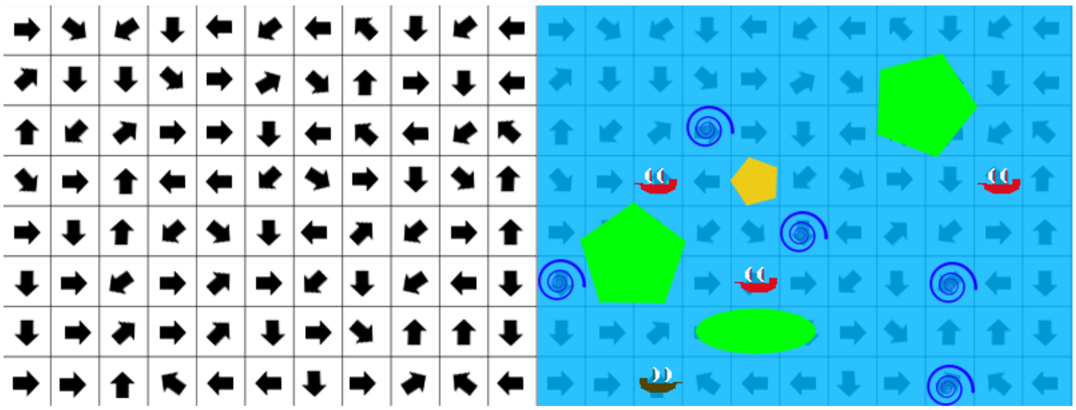


Figure 2 & 3- The sea as a vector grid (left) and a schematic representation of the game overlaying it (right)

BCI will be used to influence these weather conditions. The mental state of the player will be measured using the Emotiv device. The more relaxed a player is, the calmer the weather of the sea. When a player becomes stressed, the weather should change gradually, which should also be visible in the game (clouds come rolling in and windy noises can be heard, as well as a meter in the GUI changing value).

Speech: For the combat

Enemy pirate ships will be sailing the sea as well. When an enemy is in the vicinity of the player's ship, it will automatically start attacking the player's ship by shooting cannonballs at it. The player can dodge attacks by moving the ship and defend himself by using voice commands.

By shouting "FIRE!" the ship will start firing cannonballs at the nearest enemy and continue battle until one of the ships sinks. Aiming will be done automatically. The player should have a limited amount of ammunition, so he has to make a choice whether or not it is wise to engage.

3.1.2. Artificial intelligence

Enemy pirate ships have some form of intelligence, in the sense that they sail randomly across the sea and start attacking the player when his ship is near.

3.1.3. Game Characters

Ships

Ships are the only character-like elements of the game that move independently over the game screen. As described above, a ship's movement is influenced by the current of the sea and either the movement of the player or the game AI. Ships have a certain amount of hit points and sink if their hit points are depleted. A ship can receive damage by enemy attacks, other objects on the sea such as whirlpools or by colliding into other objects with high velocity.

Treasure Island

The goal of the game is to reach a treasure on an island with your ship. Once the ship collides with the Treasure Island, the current level is won and the player can continue to the next one.

Items

To add some extra challenge to the game than just reaching the Treasure Island, we can add items to the game that are randomly floating on the sea, waiting for a ship pick them up. These items can provide extra ammunition or some form of power-up to the player.

Objects/Mechanics

Other objects appearing in the game world are islands that cannot be traversed by the ship.

Whirlpools can hold the player in the center of it. The ship will receive damage over time while in the whirlpool. The player has to free the ship from the grip of the whirlpool by moving his arms in the right direction.

Story

The player is controlling the God of the sea, who was defeated by the crew of the ship. The God now has to do the captain's bidding. The captain (ab)uses the powers of the God (controlling the sea) to achieve the riches of the ocean.

3.2. Saving Private Bryan

Saving Private Bryan is a multimodal maze game. The maze for this game is shaped like a human brain with maze path defined by the constricted sulci of the brain.

The title of this game is synonymous to the movie *Saving Private Ryan* which depicts a few soldiers trying to bring Private Ryan back to the US because all four of his brothers have been killed in the Second World War. Our Private Bryan is a more recent one, who came back from Iraq after serving there for a year. During his short stay in Iraq, he contracted a malicious virus. The virus has now penetrated into his brain and causing tumors. The player's mission is to go inside Private Bryan's brain in a miniaturized capsule, remove all the existing tumors, and locate and kill this deadly virus. On his way to the target, the player will encounter various obstacles like antibodies that will try to block or quarantine the player. The player has to somehow neutralize the defenses offered by these antibodies without killing them.

3.2.1. Game Characters

The game consists of a few basic elements (depicted in Figure 4):

Virus: A virus is a small infectious agent that can replicate inside the living cells of organisms. The mission of the game is to kill this virus and to remove the tumors that were left behind the virus.

Capsule: The player navigates through the maze in a capsule, which also has a protective role.

Antibodies: The antibodies will occasionally try to block the path of the player. As antibodies are Bryan's friend, trying to prevent his brain from any further damage, the player should not harm these antibodies, but should find a way to go through them in order to get to the virus. The antibodies surround the player trying to slow him down.

Tumors: The virus will surround itself with a shield of tumors, which the player must first destroy/remove in order to cure Bryan and to reach the virus. Without removing the tumors first the player won't be able to kill the virus.

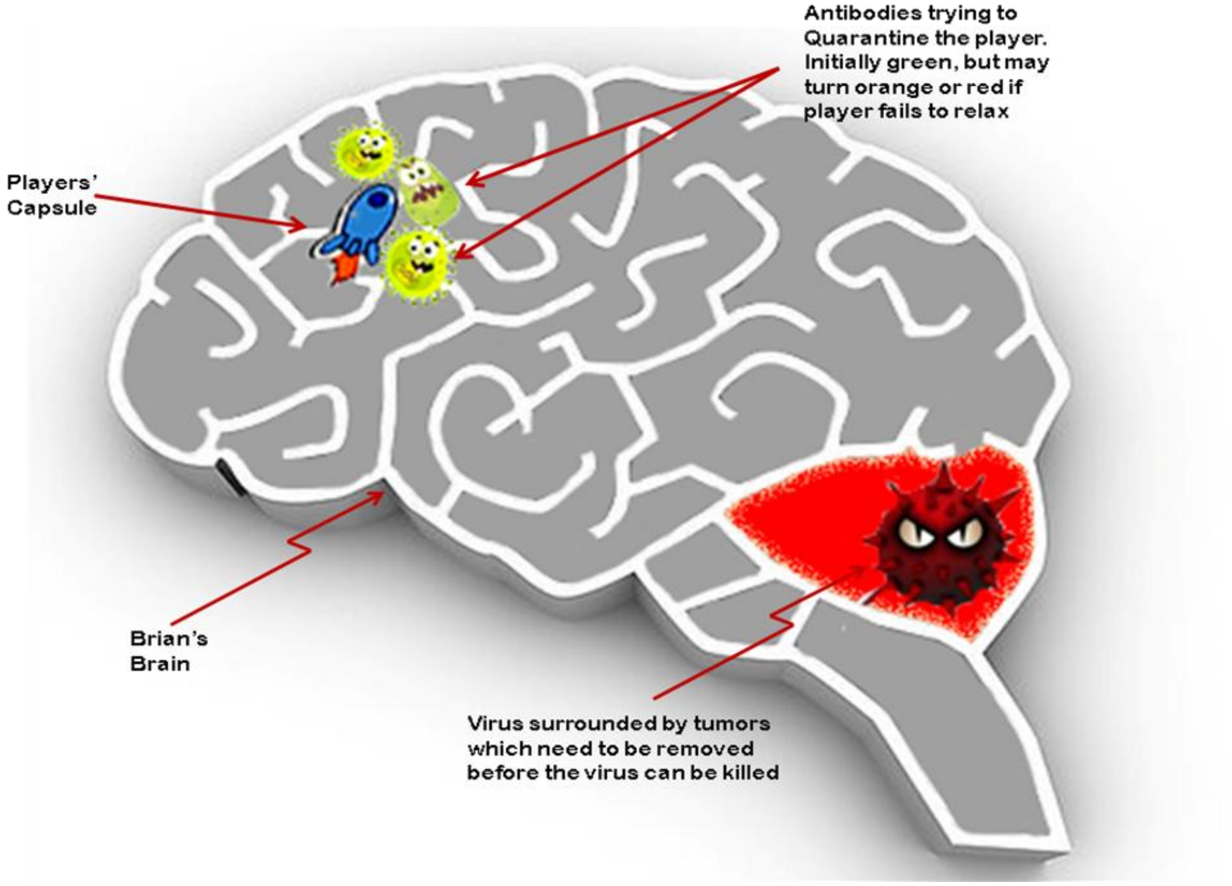


Figure 4 - Basic entities of the Saving Private Bryan game

3.2.2. Control Modalities

Following are the control modalities that would be used in this game:

Kinect: For navigating through the maze

To navigate through the maze, the Kinect's motion detection will be used. To indication motion, the upper body movement seems to be the most appropriate.

- When the player leans to left or right, the capsule will move to left or right respectively.
- When the player leans forward or backwards, the capsule will move forwards or backwards respectively as shown in Figure 5.

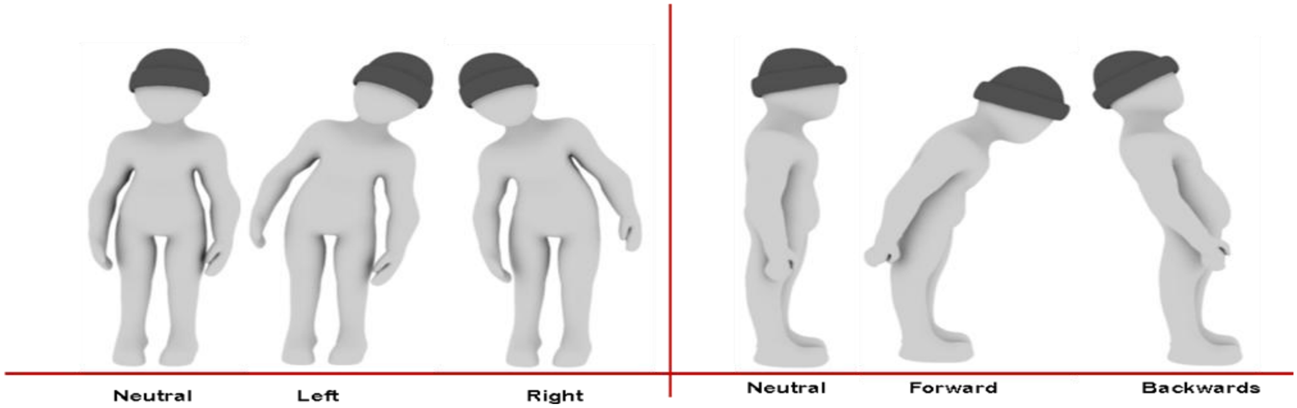


Figure 5 - Directions of body movement

Emotiv: For breaking through the quarantine of the antibodies

At some point in the game, the player's capsule will be completely surrounded by the antibodies that are trying to protect the brain from this foreign object. To break this quarantine the user will have to relax. This relaxed state will be detected in the alpha band using Emotiv. If the alpha band power goes higher than a specific threshold, the capsule will get a boost of acceleration and the antibodies will move out of the way. If the user fails to relax, the antibodies will become more and more aggressive. This will give feedback to the user in the form of change of antibody color from green to orange and then red. If the antibodies turn red, then the capsule is destroyed and the player is killed. It is "Game Over" state. The player has lost.

Speech: For removing the tumors and killing the virus

The virus will surround itself with a protective shield of tumors. Before the virus could be killed, these tumors have to be removed completely. To remove the tumors, the player will use a speech command like 'Extract' or 'Heal'. A hose will drop from the capsule and that will start sucking all the tumors. After all the tumors have been removed, the virus will become accessible and another speech command by the player like 'Fire' or 'Launch Missile' will launch a missile that will kill the virus.

4. Design of User Tests

This section describes specifically what we wanted to know from the users when evaluating both games. The goal was to find out whether these games are suited for a full fledged prototype and whether the proposed interactions feel natural to the users. In order to enhance the user centered design, we followed the advice of Anna Cox[1], who suggests using the outcomes of the user tests to reveal thoughts, opinions and experiences and afterwards formulate new hypotheses according to those.

During the evaluation, paper mock-ups were used instead of just explaining the concepts of the game, because that way the users asked more questions about the game and took more interest in its functionality. For each game, the evaluation consisted of a paper mock-up gameplay prototype and followed by a paper based questionnaire afterwards. Based on the results of the questionnaire we planned to select one of the games for final development.

One of the main goals of these preliminary user tests was to see if the game fulfilled the user's expectations in terms of interaction. According to their comments with regard to the whole functionality of the game, we could notice the missing elements or the features that could be added to the final game design. Furthermore, we wanted to see if the game commands were intuitive and that users felt comfortable while using them. Moreover, we were also interested in finding out new suggestions and idea for using complementary input modalities, because the whole purpose of the game is to have an interesting and fun gameplay.

4.1. Mock-up Procedure

At the start of the evaluation, the Emotiv headset was put on the participant and the gameplay was explained. However, instead of playing the actual game, the participants were presented a large paper representation of the game with magnetic game characters that could be moved on the game board. This allowed testing the game without actually developing it. Since a paper mock-up cannot be used to test the game in real time, the participant was explained that the designers will take his input every few seconds and process this as if it was done by the game. This not only includes in-game rules but also movement, speech and brainwave interaction that would be used for the game. Though this was not a full representation of the game the participant were able to get a decent insight into the functionality of the game. By asking questions, as well as our own observation, gave us a valuable insight as to what does and doesn't work in the game. Pictures of the mock-up game boards can be seen in figure 6.

1 Paul Cairns & Anna L. Cox: Research Methods for Human-Computer Interaction

Figure 6 - The two game concepts as a paper mock-up (Left: Saving Private Bryan, right: Pirate Game)

4.2. Pilot Evaluation

We started off by doing two pilot test evaluations, to test the flow of our evaluation before carrying out the actual procedure on real participants the very next day. The most important point made by the participants was that we should not guide the player through the game step by step, but rather explain each game (using a tutorial) and let the participant play through the game on his own. For the pirate game this was already fine, but the brain game needed more of a distinction between playing and introduction.

4.3. Evaluation Sessions

We conducted the user evaluations with 10 participants (8 male, 2 female), all students, in the range of 20-25 years old. Each participant played both games. First, the participant read a short instruction manual, then played one game, answered game specific questions. For building up a useful questionnaire for our purposes, we consulted Paul Cairns's suggestion[2], who mentions that the questionnaire should be used as a measurement instrument and questions like "Why don't you like this interface?" or "We made this. What do you think of it?", should be avoided. After completing the questionnaire, the participant played the second game, answered the same questions for this game. At the end a final round of comparative questions were presented. The order in which the two games were presented, alternated between participants to prevent any bias originating from the order of the games.

It took some time to propagate the interactions of the user to the mock-up (which of course will be smoother in the final design). Therefore, we allowed a five second delay between each 'game step'. For example, in the pirate game, the participant moved his arm up and next the evaluator moves the player's ship upwards and some of the enemy ships in a random direction.

The results of the user evaluations can be found in Appendix A.

[2] http://www-users.cs.york.ac.uk/~pcairns/RMHCI.html

4.4. Evaluation of User Tests

Before carrying out any type of analysis on the results, we carried out the Kolmogorov-Smirnov (KS) test on the dataset to test for normality of the data. The KS test reveals that all but Question 3 and 8 are normally distributed. This means that we can use Parameter Analysis like Pairwise T test on all but questions 3 and 8. The results of the parametric tests are given in the Table 1 below:

Question	Game	Mean	Standard deviation	2-tailed Statistical Sig	Winner	Probability of having backed the wrong winner
1.The control scheme was clear to me	Pirate	4,2000	,63246	1,000		
	Bryan	4,2000	1,03280			
2. The controls felt natural to me	Pirate	4,0000	,81650	,177	Pirate	17.7%
	Bryan	3,5000	,84984			
4. Controls added something to the gameplay when compared to traditional controls	Pirate	3,6000	1,07497	,591		
	Bryan	3,8000	1,03280			
5. I would prefer to play the game with traditional controls	Pirate	3,0000	1,33333	,213	Bryan	21.3%
	Bryan	3,5000	1,08012			
6. The game was fun to play	Pirate	4,0000	,66667	,758		
	Bryan	4,1000	,73786			
7. The game was challenging enough	Pirate	2,7000	,94868	,269	Bryan	26.9%
	Bryan	3,1000	,73786			

Table 1: Results of the Pair-wise T-Test for Normally Distributed Data

Question 1, 4 and 6 yielded a significance level greater than 0.5 which means that if we made a decision as to which game was the winner, then we would be wrong, no matter what winner we pick. This is because the probability of the distribution of answers to these questions is nearly the same and it would be unwise to use these three questions in the final decision. So this is the reason why we only used the results of Questions 2, 5 and 7.

To analyze questions 3 and 8 we used non-parametric Wilcoxon test. The results of the analysis are given in Table 2.

Question	Game	Median	2-tailed Statistical Sig
3. It was easy to learn the controls	Pirate	4	,564
	Bryan	4	
8. The game concept was interesting	Pirate	4	,157
	Bryan	4	

Table 2: Results of the Wilcoxon Test for Data with skewed distribution

The Wilcoxon test did not reveal any clear winner for the two games because the medians come out to be equal. So these two questions do not reveal any decisional information about the two games.

Based on the analysis carried out on all 8 questions the Bryan game has the majority vote.

5. Discussion

5.1. Game Selection criteria

5.1.1. Qualitative input

During the evaluations, the users commented on several aspects of both games. These can be categorized as follows:

Input modalities:

- Movement: The preference for either arm or body movements seems to be 50-50. In the pirate game, most users preferred to move only one arm. Some even stuck their arm in one direction and waited for the ship to start moving instead of making a wave motion. The point was made that a player might lose balance while leaning or that moving just the upper part of your body feels weird for controlling the ship (brain game). On the other hand, some participants found it a bit tiring to wave their arms for a long time.
- Speech: Participants noted that speech was more of a gimmick than a real addition for both games. Using a voice command did not provide much of a challenge in the current situation and needed to be improved.
- BCI: Not much input was given with respect to the brain control input. This was probably caused by the fact that the influence of the brain interface control was rather implicit during the evaluation. The effect could be seen in the game to some extent, but the player could not really influence it in any way.
- Game elements and interactions: Most of the game mechanics were well understood by the participants after reading a short tutorial. Some users were a bit unsure about

movement and orientation, especially during the brain game. A few players forgot they could shoot enemy pirate ships by shouting ‘Fire!’, but knew how to play through the game easily.

5.1.2. Quantitative results

As far as the statistical research can tell us, we can see that the users slightly preferred the interactions of the pirate game, but found the brain game a bit more challenging. It should be noted here that the results were very similar and both games score low with regard to challenge.

5.1.3. Complexity and challenge

Complexity could be viewed from various standpoints. First of all, we looked at complexity in terms of game design: a complex game usually provides more of a challenge to the user. This can stem from learning curve or the amount of ‘training’ needed to master the game. For this aspect, both game designs were nearly equal. This can be explained by the fact that the controls were pretty straightforward and the game mechanics (game elements, AI, story, etc.) were not incredibly advanced (at that stage).

Second, we looked at the complexity of the program code. The only thing we could say about this, at that moment, was that the pirate game would be more of a challenge to implement, which is caused by the dynamics of the game (influence of the sea, different objects colliding/interacting with each other), the simultaneous interactions, and the need for some kind of AI for the enemy ships. The brain game seemed simpler in that aspect, because the interactions with other game objects were much more simplistic (the ship could either move or not). The simultaneity might also cause problems during the implementation, caused by the artifacts that appear in the BCI measurements when trying to combine the BCI with speech or body movement.

5.2. Final decision

The goal of the project was to combine multiple input modalities for natural and intuitive control of a fun game and to examine how the three input modalities influence each other during the gameplay. But, as described in the previous section, combining other modalities with Emotiv might cause artifacts in BCI measurements. The Saving Private Bryan game was more preferred in this aspect, although the part of the game where BCI was not used needed to be made more interactive.

The results of the user evaluations seemed too inconclusive to form a solid basis for the decision which game to implement. There is only a slight preference by the users for Saving Private Bryan, as can be seen from the statistics. With respect to the last to criteria, we felt that both game designs could be improved or tweaked in such a way that they satisfied the user suggestions and met the requirements. We chose to implement Saving Private Bryan, because of the small preference by the users, as it fits the ‘BCI theme’ of the project a bit better than the Pirate game and because we believe we can incorporate enough changes to the game so it offers more fun experience.

Design Iteration 2

1. Developing 'Saving Private Bryan'

The first design iteration ended with the conclusion that Saving Private Bryan game idea needs to be further refined and developed into a full-fledged working prototype. So in this second design iteration we are going to incorporate the changes suggested by our users, in terms of interaction and gameplay. The details coming forth hence describe all the procedure for this second iteration of the game design process. This includes the final design of the game itself, evaluating the game and finally reviewing whether we succeeded in creating a multi modality game.

2. General Game Flow of 'Saving Private Bryan'

Much like with any software project, a game needs a proper design document to structure the design process. Due to the nature of games however, the design document differs from most software projects. A game should be designed around a game design document[3], which specifies all details on gameplay. It reflects on the visual design of the game, the internal game mechanics, how player input is processed and what objects have a role in the game itself. Through the game flow description of Saving Private Bryan, the entire structure of the gameplay should become clear.

The flow of the game was split up in two distinct sections. For the majority of the game, the player will be navigating through a maze, trying to reach specific points and dodging several obstacles. The second part consisted of the player fighting the virus in the brain. The player needed to finish part one to progress to the second part of the game.

2.1. Maze game part

Once the game was started the player would find him/herself in a capsule inside Bryan's brain. Somewhere in the brain the virus was located, and the ultimate goal of the player was to reach this virus. However, the path to the virus was blocked by the folds of the brain sulci, forcing the player to first complete some sub-missions to open the path.

To open the path leading to the virus, the player would have to navigate through the maze to reach several letters hidden within the maze. These letters formed a word that when pronounced by the player would open the path to the virus, if the player was standing close to the blocked path. The player could then progress towards the second part of the game. A full flowchart of this part of the game as well as the menu structure can be found in Appendix B.

---

[3] Rouse, Richard III "Game Design – Theory & Practice – Second Edition" 2004, Wordware Publishing

When navigating through the maze, there would however be some obstacles, called antibodies, for the player to either dodge or fight. These antibodies would appear near the player's ship and then chase the player. All of the antibodies would have short words written under them, which when pronounced by the player would make them disappear. If the player failed to evade the antibodies, they will stick to the player's ship and slow its movement. When attached to the ship, the antibodies will also start to destroy it. The longer an antibody was attached, the larger the chance will get for it to kill the player. To prevent them from doing so, the player could opt to stop moving entirely and concentrate which would be detected by Emotiv and remove the antibodies from the ship.

2.2. Boss fight

Once the player reached the virus, the game would show a different screen, the arena, in which the player was to fight the virus. The virus-fight consisted of two parts, one weakening the virus, the second to actually cure Bryan from the virus.

In the first part, the virus would shoot enemy virus-cells towards the player ship, which could be dodged by the player moving sideways. At the same time a group of different words will show up, which when pronounced would lower the viruses shield.

Once the shield was lowered, the virus would stop from attacking the player and becomes open to the heal process. To start this, the player would have to concentrate again, slowly killing the virus. This phase would only last for a short while however and if the virus was not killed within this time frame its shield will restore and the first phase of the boss fight would start all over again.

2.3. Visual Design

The game requires several visual elements to display the elements the player will encounter in the game. In general, the game features a cartoon like graphical representation, as this is relatively easy to make and will fit the gameplay as represented. Furthermore, cartoonish graphics makes the subject of the game, illness, less threatening. Since the game only requires movement along two axes (x, y) the graphic will be made in 2D. This also lowers the development time required for the game.

2.3.1. Graphical User Interface

Though the game wants to communicate some information to the player, such as the concentration level and whether the ship is in danger of being destroyed, it does not include a Graphical User Interface (GUI) for this purpose. Rather, visual representation on the objects in the game shows information that is relevant to their presence and needs to be known to the player.

2.3.2. Maze

Visually, the player finds himself in a maze that is shaped similar to the structure of a human brain. To ensure some complexity, the maze as a whole does not fit on the screen at a size in which the player's ship is visible enough. Instead, only a small part of the maze will be shown to the player through a viewport.

The viewport in the game however does not cover the whole screen, but rather creates a magnifying glass effect. This means that the entire maze is drawn on screen at a size too small for the player to have a good oversight. At the place the player's ship should be drawn in this maze, a circle with a larger representation of a part of the maze is shown, making it appear as if the player is watching this part of the brain through a magnifying glass.

2.3.3. Player Objects

The player is represented into the game as a microcapsule. Since the players heading is based on the direction this microcapsule is pointing, it is shown as a rocket-like object. When forward motion, the ship is animated indicating the movement.

Since the ship can be attacked by antibodies it also has an indication of the concentration level of the player. When not moving, a shield appears over the ship, increasing in size if the player is more concentrated.

2.3.4. Enemy Objects

The main enemy objects in the game are the antibodies, which are initially green colored. If attached to the ship, the antibodies can change their color from green to yellow and from yellow to red. A red antibody indicates that destruction of the ship is at hand.

Since antibodies can be stopped by saying specific words, moving antibodies will also show this word beneath them.

Another enemy in the game is the virus that is to be destroyed. This virus is represented in two ways. In the first one, the shielded version looks angry; the virus will have several words floating around for the player to say. The non-shielded version of the virus looks dizzy, indicating it is not able to defend itself and lights up according to the concentration level of the player. To display the health of the virus, its size decreases when the player is able to hurt him.

2.4. Controls

Instead of being controlled by the keyboard, the game supports a combination of different input methods, namely Kinect (for movement), The Emotiv EPOC for concentrating and a microphone for using speech commands.

2.4.1. Movement

Movement of the ship is done by leaning. Leaning forward makes the ship go into the direction it is pointing. By leaning sideward the player is able to turn the ship. This will require the player to steer the ship using his or her whole body. The game allows simultaneously moving forwards and steering, but the player is also able to steer the ship without moving forwards. This will prevent the player from getting stuck and allow for adjustments if body movement, which will most likely take some time to get used to,

Since the game takes place in a maze, movement is not entirely free. When a player collides with the walls of the maze, the ship will be unable to move forward. The player will thus have to navigate among these obstacles to complete the game.

2.4.2. BCI Use

In the game it will occasionally occur that an antibody-object gets attached to the ship. Since these antibodies eventually kill the player, it is essential that they are removed. To do this, the player has to alter his mental state to a form of concentration (short time excitement, to be more precise). If the player manages to stay concentrated, as measured by the Emotiv headset, for a short period of time the antibodies disappear.

In the end-game, the concentration of the player can also be used to kill the virus once it is in a vulnerable state.

2.4.3. Speech use

The last form of input is speech. Speech is implemented in several ways. First of all, there is the menu control, wherein the player can mention the word associated with a menu option to execute its function. For instance saying the word “start” will start the game, while “exit” will make the program return to Windows.

A second form of speech is used to kill the antibodies storming at the player. These antibodies have “kill words” associated to them, which when spoken by the player will kill the antibodies, if within hearing range. The same can be done for the virus, resulting in it becoming vulnerable to attacks.

Lastly, the game has one specific word that can only be used after all the letters spread through the game have been collected. This word is used to remove the blockade to the virus.

2.5. Game Objects

In the game some objects will appear that the player can interact with.

2.5.1. Player Ship

The player navigates through the maze in a capsule, which also has a protective role. The player can move this capsule in the way it is facing or rotate it. The ship has a maximum speed, which can be lowered when antibodies are attached to it. When not moving, the ship can generate a protective field, removing all the antibodies from it.

2.5.2. Antibodies

The antibodies in the maze will appear somewhere near the player and get closer to the player’s ship. When still chasing the player, the antibodies can be destroyed by the player by saying a word that is linked to the antibody. If the player fails to do so, the antibodies will attach to the ship and start to destroy it. Slowing down the ship and eventually killing the player.

2.5.3. Virus

A virus is a small infectious agent that can replicate inside the living cells of organisms. The mission of the game is to kill this virus and to remove the tumors that were left behind the virus. Initially, the virus is protected by a shield and able to fire smaller viruses towards the player. If specific words are spoken however, the shield will go down and the virus temporarily becomes vulnerable to attack, by the player’s concentration.

2.5.4. Letters & Blocked Path

To reach the virus, the player needs to open up the blocked path leading there. This can only be done by a speaking out a specific password, which can be obtained by collecting letters that are hidden within the maze.

2.6. Artificial Intelligence

Though the game features no real Artificial intelligence, the enemies are controlled by the computer. In the case of the antibodies, the game will automatically move them closer to the ship, if possible taking the maze into account (this is dependent on the time available). The appearance of the antibodies will be determined at random.

The virus will also require some computer interaction for attacking the player. Like the appearance of the antibodies, this will be done at random.

3. Design improvements in the second iteration

After discussing the results of the first iteration, we decided to make the following updates to the chosen game, Saving Private Bryan, which we planned to incorporate during the second iteration.

- While playing the game, in order to increase his power for the final fight with the virus, the user will have to collect certain items. The idea was to make the final boss fight more challenging by letting the user pronounce semi-random words (selected from a specified list) instead of blabbering just one command all the time. The amount of items collected during the game could influence how many letters are revealed, so the user has to guess the words.
- In order to be more natural and a bit less sequential, we decided to combine moving with speaking. For example, the final boss could fire projectiles back at the player or the player has to move to several regions around the final boss to find his weak spot (in combination with the previous point).
- Another challenge that we decided to add consisted of showing only a limited part of the game to the user. This meant that we would make the viewport smaller than the total area of the game world and the screen is centred on the player's ship. This would encourage the user to explore the whole game world.
- We believed that a time limit in which the player has to cure Bryan (i.e. destroy the virus) could make the game more competitive.

We consider that the feedback from our users enabled us to greatly improve the overall feeling of the gameplay.

4. Implementation

In this chapter, we describe the implementation of our system in C# using Microsoft's XNA framework. The class diagram of this system can be found in Appendix C. Only the important methods and attributes for understanding the implementation of the interactions are explained in this section.

XNA is a development framework created by Microsoft that allows the programmer to focus on building the game instead of low level technological details. Once the game object is constructed, the `Initialize()` function is called to set up non-graphic related content. `LoadContent()` is where the graphic content for drawing the game is loaded. Finally, an infinite loop is started where the game's `Update()` and `Draw()` methods are called about 60 times per second. All the important classes below, that influence the game world, will implement these methods, for example to implement movement and checks for victory conditions.

4.1. SPB

This class extends the `Microsoft.XNA.Framework.Game` class and initializes all major parts of the system: the input and output managers, the menu and the maze screen.

The `Update()` function executes all the logic for updating the game world, collision checking and input gathering. It calls the `Update()` function of both the BCI and SpeechManager in order to update the BCI value and update some timer values.

For each call to this function, a new `Frame` object is made that registers a selected set of variables that represent the user's data such as elapsed time, player position, BCI value, last word spoken and Kinect data. This data can be used after the session to analyze the performance of the user. The `StorageManager` is invoked to store the data in an XML file.

The `Draw()` function is responsible for rendering content on the screen and delegates this to either the `Menu` or the `Maze` object. Furthermore, this class contains methods to start, stop and reset the game.

4.2. Menu

This class simply represents the screen in between play sessions.

4.3. Maze

The `Maze` class controls all the objects related to the game world: `Player`, `Antibody`, `Letter` and `Virus`. Each of these objects is initialized in its own `SetUp()` method and updated in the `Update()` function. The `Update()` function also controls the switch between moving and concentrating and the spawning of antibodies.

The `HitsMazeWall()` and `CalculateBoundingRectangle()` functions are used for collision detection. The `Draw()` functions render the game world including magnifying glass and background and then delegates further `Draw()` calls to all objects in the world.

4.3.1. MazeObject

The `MazeObject` is an abstract class that contains only the position of the object and defines the `Update()` and `Draw()` functions. The following classes extend the `MazeObject`.

4.3.2. Player

The `Player` object needs some attributes for rotation and speed and methods to control the position of the player. In the `Update()` function, the state of the `MovementManager` is checked to see whether the player is currently leaning. If this is the case, then the player is either rotated or moved over some distance. The `Draw()` function determines which sprites have to draw for the player (player itself, shield or exhaust fume).

4.3.3. Virus

The `Virus` controls the win condition of the game. The game is reset if the player is less than 100 pixels away from the virus, collected all letters and said the word to kill the virus.

4.3.4. Antibody

Antibodies can move back and forth through 3 stages (green, yellow and red) that indicate how soon the player will be killed if he does not shield himself. As explained above, the antibodies will slowly turn red and try to kill the player. If the player manages to concentrate, the antibodies will turn back to green and disappear.

The antibodies always try to move in a direct path towards the player. The only exception is when it is stuck to the player, in which case it follows the player's movement. Collisions with the maze also prevent movement.

4.3.5. Letter

A Letter only needs to store a position and check whether the player is close enough to be picked up.

4.4. BCIManager

The `BCIManager` has attributes to store the concentration value and the amount of time a player is concentrated. In the constructor, a new event handler for the Emotiv engine (called EmoEngine) is added. Each time the state of the Emotiv is updated, such an event is generated and the data value for short term excitement can be read. Furthermore, this class only has methods to check for the user's concentration.

We conducted a small experiment to see which value we want to read from the Emotiv and how high the threshold should be. Please refer to Appendix D for further details. We chose to use the short term excitement value in our game because this was the only variable that clearly increased as the user started to move and decrease when he started concentrating. The value for long term excitement showed this behaviour as well, but the value changes too slow over time (as the name suggests). It seemed hard to influence the meditation, frustration and boredom values.

We have set a threshold as static variable to check whether the player is concentrated enough. This threshold was also chosen based on the results of this test.

4.5. MovementManager

Equivalent to the BCIManager, the MovementManager uses an event handler to catch updates from the Kinect. A third party library that makes use of the OpenNI library by Vangor Pterneas[4] was used to easily read the coordinates of relevant bodyparts.

In each `Update()` step, the x-, y- and z-value of all 15 bodyparts shown in Figure 7 can be used.

Figure 7 - Demo application for the Kinect body tracking API

For forward movement we check whether the distance over the z-axis (depth) between the head and the waist is greater than a certain boundary. For rotating (by leaning left and right), the distance on x-axis (horizontal) between head and waist is used.

4.6. SpeechManager

For the speech recognition we use the `SpeechRecognition` engine from Microsoft. During initialization the words defined in the `Keywords` class are added to the grammar (the speech recognizer uses a closed grammar). An event handler is added that stores the recognized words being said by the user for a short period of time.

5. Design of User Evaluation Test

To test the look and feel of the game, user tests were conducted on an incomplete version of Saving Private Bryan game. The goal was to find out whether the controllers are working properly and if they can satisfy the main requirements i.e. the natural and intuitive interaction during gameplay. The evaluation consisted of the actual gameplay followed by a questionnaire to evaluate interaction. More specifically, we wanted to find out whether the way we combined all three control modalities in the game, made the player feel both comfortable and challenged. The questions were structured for each modality (BCI, Kinect, speech) and were meant to help us in improving certain areas in the game. We also wanted to see by means of this questionnaire, if the transition between modalities was done naturally without breaking the game flow. Also the observations of the gameplay, helped us see how

[4] http://www.studentguru.gr/blogs/vangos/archive/2011/03/15/kinect-and-wpf-complete-body-tracking.aspx

natural is for the user to play this game. Another important aspect that we liked to take into account was the preference of the user for a certain control mechanism, and if one modality had a stronger or a weaker influence on the game. According to responses and comments from the participants, we planned to identify weak points of the game and improve them.

Another important part of the evaluation was the recording the observations, which were meant to summarize the spontaneous reactions of the users while playing the game. Because the main target of this project is to make the players feel natural while they are playing the game, we would adjust the controllers design according to our evaluation results.

5.2. Procedure

At the beginning of each evaluation session, the Emotiv headset was placed on the participant's head. Afterwards, he was briefed about the idea of the game and the gameplay by means of tutorial. During the tutorial the player learned how to use Kinect, for starting the game; which movements were required to move or rotate the ship. After that he would learn how to destroy the antibodies, by using Speech, pronouncing the word that appears beneath them, and BCI, by concentrating to shield him from the antibodies. The main purpose of the game was to kill the virus, in order to do that, the player had to collect all the letters that are scattered over the brain. The participant was told that after having all the letters, he had to get closer to the virus for the final fight and pronounce the new word obtained after getting all the letters, in order to kill the virus. After finishing the game, the player would be asked to fill a questionnaire regarding the entire gameplay.

Before starting evaluating the game on all participants, the evaluation itself was tested on one subject a day before, to refine the flow of evaluation and catch any caveats in the process. The subject was satisfied with how the game tutorial was presented and how the game was evaluated using the questionnaire afterwards. Three problem areas were identified during this preliminary evaluation session:

The Emotiv headset would stop working some time.

This was because the USB transceiver of Emotiv was located relatively far away from the subject because of which the signal was lost once the subject moved out of range. The headset then had to be removed from the subjects head, brought closer in range to the USB transceiver and mounted again.

An easier alternative was therefore devised by using a long USB cable and placing the USB transceiver very near to the subject in a non intrusive way. In case the Emotiv ever lost the signal again, the USB cable and the transceiver attached to it would be brought closer the headset, instead of dismounting the headset and bringing the headset closer to the USB transceiver. This procedure helped save a lot of time in case the Emotiv ever lost signal.

Speech Detection didn't work some time.

This was because we were using the laptops' built-in microphone which probably was too far away to reliably detect the speech signal from a subject, who was standing two meters away from it. The problem was solved by an external microphone located close the subject in a non-intrusive location.

Maneuvering the ship was difficult after a collision.

This happened because the collision detection system in the game was designed to report collision and hence stop the ship whenever it encountered a black pixel. However, some spurious black pixels were located in the wrong locations, causing a collision to be reported when in fact there should not be any collision. This made maneuvering the ship very frustrating for the subjects. However, redesigning the entire maze to rectify the problem seemed impossible at such short notice. So it was decided to leave the problem untouched and rectify it after evaluating the game on all subjects.

5.3. Evaluation Sessions

The game was evaluated on 20 participants (16 male, 4 female), all students in the age range of 20-25 years. One experimenter would walk the participants through an on-screen game tutorial, while at the same time another experimenter would mount the Emotiv headset on the participants head. The purpose of the tutorial is to show all the game elements one a time, in concise sentences and explain how the user can interact with them. In Figure 8 there is an example of such a screen.

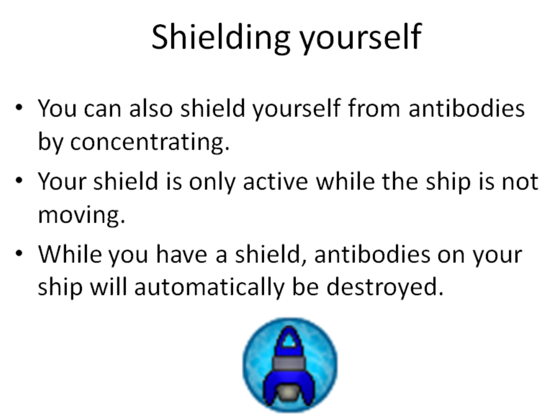


Figure 8 - Example of a tutorial screen

All electrode impedances were lowered to the green level in the Emotiv control panel. Once the cap was set and the subjects understood all the game instructions, the actual gameplay would start. At the start of the game, the user would have to stand in psi position to activate Kinect motion detection. Once the Kinect detected the subject, the game commenced. If the subjects' ship got destroyed before reaching the virus, the game would be started again to make sure that every subject completed the game at least once.

5.4. Questionnaire

After completing the game, the subjects were asked to fill in a questionnaire (see the Appendix E for a the entire Questionnaire) designed to evaluate various game features. The questionnaire was designed in Inquisit Millisecond software and conducted on a laptop computer.

The whole process from start to finish took on average about 25 minutes to complete.

All the evaluation results can be seen in the Appendix E. However, here we would only discuss the game features that the user either didn't like or felt a need for improvement.

S.No	Question	Yes [%]	No [%]
1.	Did it feel natural to concentrate in order to make the antibodies disappear?	55	45
2.	Could you successfully move and rotate the ship at the same time?	35	65
3.	Did the ship respond well enough to your movement (was the feedback good enough)?	50	50
4.	Would you prefer to use traditional keyboard/mouse controls for this game instead of speech, motion and BCI?	55	45
5.	To use the shield in game, you had to stop moving and start concentrating. Did you feel that this broke the game flow too much?	50	50

Table 3: First five questions

It seems from the user response in Table 3 , that the subjects found movement to be the most problematic part in the whole game. Furthermore, half of subjects felt that stopping to concentrate, in order to kill antibodies was a bit awkward. That's why we implemented another method of killing the antibodies by using a kill word written beneath the antibodies.

The question in Table 4 was presented on a 7 point Likert scale. We computed the average of the user response and present its interpretation below:

S.No	Question	Mean	SD	Interpretation
1	How difficult was it to maneuver the ship?	2.3	1.1	Moderately difficult

Table 4: Likert Scale Questions

Again it's obvious that maneuvering the ship was the most difficult part of the game, one that users found very troublesome to control.

An open-ended question was also presented at the end of the questionnaire where subjects were given the opportunity to inform us about their feedback and suggestion. We mention some of the most important critique in Table 5.

Subject ID	Question: What features of the game would you like to see changed?(you can also write any suggestions or comments here)
5.	Movement is hard, ship often sets off in a direction that was not intended some pixels are hard to miss, ship gets stuck
6.	Magnifier causes confusion; it's hard to see where the letters are. Also, adding more different words besides flower, tree cat and house would be nice.
16.	The font of the words was a little too small to read fast, the turning the ship could be slowed down
17.	Fix the collision detection :) Concentrating broke the flow of the game a bit Speech could maybe be used to select a target for a missile?

Table 5: Open Questions

5.5. Evaluation Results Summary

Based on the evaluation results, the following areas for improvements were identified:

1- The collision detection needs to be more accurate.

2- More exciting and game relevant words need to be used instead of cat, flower and boat.

3- The font for the words also needs to be bigger.

4- The rotation speed of the ship needs to be decreased.

These recommendations were incorporated in the final version of the game.

5.6. Objective Data

In this chapter we look at the data that was collected from all user sessions. As explained in the chapter on implementation, we store data such as the player's position, BCI value, game time and last word recognized for each frame of the play session. This data is stored in a large XML file that we can process and analyze to hopefully find interesting patterns in the play style of the participants. This data can tell us *how* the participants played our game and *how well* they did.

- BCI values
- Movement
- Player Deaths
- Antibody Kills
- Player Paths

5.6.1. BCI Values

Figure 9 from Appendix F shows us the average BCI value for each movement state that the user was in. This chart mainly shows us what the influence of movement is on BCI. As explained in Appendix D, we would expect that the BCI value would drop significantly while the player is standing still and concentrating. However, this hypothesis does not apply to this graph, since the player does not have to be concentrating while standing still. This explains why the BCI value for all the users is not significantly lower while standing still. What we can say is that a player's movement does not seem to influence the measured BCI value as we expected and that more factors that we could not measure influence this short time excitement value. The threshold value for the player's concentration (0.55) does seem pretty accurately chosen according to this chart, since the average concentration value for all users is about 0.57.

5.6.2. Movement

Figure 10 from Appendix F partitions the state of the user regarding movement over time. Ideally, the user would be able to control the movement by moving and rotating at the same time and only rotate now and then to correct his path through the maze. The fact that standing still takes up about 50% of a player's session can be explained by collisions with the maze wall and the user having to stand still to charge his shield. The relatively large rotating percentage is probably also caused by collisions.

5.6.3. Player Deaths

In order to determine how well players performed, it is of use to look at the amount of times players died during play. Death indicates failing to complete the games objectives, and thus could be argued to show that either the game is too easy/difficult or that a learning curve is in place. Looking at the objective data logs, it seems that the latter is the case for the Saving Private Bryan game.

As shown in Appendix G, the average play session had 2.2 deaths, which tended to occur around 28% of the session. This last figure indicates that players were actually improving their skills during play, for if the difficulty did not change for players, the figure would have probably been closer to the 50% mark. When not looking at players who did not die at all, the percentage is still at 34.48% of a play session. Players thus died more frequently at the first half of the game and less on the second half.

Looking at the amounts of deaths however, there seems to be a large differentiation between players. While some managed to finish the game without ever dying, it took others as much as nine deaths to complete the game. One of the reasons for this seems to be the average concentration value during play sessions. When looking at the sessions with a lot of deaths, they tended to have an average concentration level much higher than the average among all sessions. Likewise, players that rarely died often had concentration values below the universal average and below the threshold used to kill the antibodies. The concentration level thus predicts the amount of deaths, which follows from the fact an inability to get the concentration value below the threshold results in death as per the design of the game itself.

5.6.4. Antibody Kills

Another statistic indicating the skill of players is the amount of antibodies killed during playtime. Players managed to kill roughly two times as many antibodies using speech than with using their concentration level. This can probably be explained by the fact that antibodies can get killed using speech immediately, while concentration is only necessary for antibodies that were not killed using speech and managed to attach to the player's ship.

More interestingly is the fact that there seem no significant differences in kills between players that died a lot and players that did not die at all. Though there is a distinct variation between their average concentrations values, players that died a lot did not necessarily kill less antibodies using concentration. The relation between antibody kills and player deaths thus seems more complicated. Based on the multiple statistics in Appendix G a player that dies more either kills little antibodies per minute in total, likely resulting in more antibodies becoming attached, or has a bad ration in terms of antibodies killed by speech versus concentration. There is however not one specific variable that shows the players that die more and have a higher concentration value kill less antibodies by concentration than players who keep their concentration level low.

5.6.5. Player Paths

The stored XML data also allows us to draw the path each player traversed during the game, because we stored the player's position for each frame. Figure 11 (See Appendix H) is an example of such a user session. From these images we can see that the user needs some time at the start to figure out the controls, but after that moves more directly towards his goal. Another pattern we see is the appearance of squiggly lines in tight corridors of the maze. The user bounces from wall to wall to get through these narrow spaces, which can be caused by the controls being too sensitive or the collision detection being inaccurate. Sharp corners in the line indicate that the player stopped at that point and rotated before moving on (visible in the right part of Figure 11). Overall we can say that the players knew where they had to go, moving relatively directly towards the letters they have to collect. Some users even took an alternative route through the maze after they died, if that was faster for them. There is a wide variety in play styles and in how well the player could control the Kinect.

5.6.6. Observations

One of the main purposes of our evaluation was to see how natural all the controllers are to the user. During the playing session we noticed that some users prefer to move their entire body in order to either rotate or move the ship. There are also some that chose to move their hands according to the ship movements or their head to rotate it.

One of the most apparent observations however, was the ability of players to combine the several input methods. It was clearly noticeable that some players where real skilled at using movement but had trouble controlling the BCI or vice versa. During all sessions there often seemed to be one input method that players could handle form the start, while others proved more difficult. This was for example evidenced by the fact that quite some people stopped moving to say the words to kill antibodies. It would thus seem that not every player has the skills to combine multiple modalities at once, at least initially.

6. Conclusion

In designing Saving Private Bryan we tried to create a game that had to be played with multiple input methods. The goal of this was to review the possibilities of such game design. In the game created, Saving Private Bryan, we used controls with human body movements detected with Microsoft Kinect, Speech Recognition, and EEG signals collected with Emotiv EPOC headset. The body movements are used for the navigation interaction in the game. The speech recognition is used to perform the selection events in the game, but for real interaction with the opponents (the antibodies or the virus). The EEG signals are used for adjusting the shielding level of the player.

When evaluating the use of multi modality input, we should consider whether the actual game in use could be considered good for its application. Since a game relies on more than just its player input, a bad game could result in less precise information about the different modalities. According to the experimental results however, we confirmed that each module performed successfully and showed that the proposed approach provided a higher of immmersion immersion and interest to the user, compared to the traditional controllers. Answers of the questionnaire indicated that the majority of players liked the game, which we consider a good sign.

Based on observation however, most of the users preferred to use one or two of the three modalities at a time. This proved us that they enjoy using the presented modalities, but not combining them simultaneously. Based on objective data, which proved a learning curve, it is possible that players will adopt eventually to multiple inputs but as of now, it seemed that different users might have different skill sets.

According to the observations results during the gameplay, the preference of the users towards of a certain body movement, either hand movements or entire upper body movements, could make the game more natural for each of the players.

The results show that even if the flow of the game might be interrupted by speech and the needs of concentration, the entire application can be implemented with a commercial headset, such as Emotiv EPOC. A picture of a subject playing the game can be seen in Appendix I.

In general, we are confident that we succeeded in creating a multi modality input game, although there are still some challenges ahead.

Discussion & Future Work

Designing and evaluating the game showed some possibilities for both future research and improvement of the game itself.

Game Improvements

According to the evaluation results, we found some areas that would need future improvements:

1. When moving forward, sometimes the ship collides into a wall, case in which the player needs some time to establish the position of the ship, so a better collision detection would be needed.

The official Kinect SDK was released a few weeks ago, so many of the errors associated with user and collision detection should not be a problem anymore.

2. Our speech recognizer has a closed list of words, which seemed to some user a bit restrictive, as there are many antibodies and the words beneath them are mostly the same, so maybe a bigger list with a variety of words would solve this problem.
3. In order to improve the performances of the game, in terms of timing, the concentration level, should be represented as a balancing value. By this, we mean, that the concentration level should be more flexible, having a lower value and an upper value, being adjustable for every user.
4. As the required distance necessary for the Kinect to detect the player is approximately 2 meters, the dimension of the letters from the words that serve for killing the antibodies should be increased.
5. The magnifier glass was found by most of the users as being confusing, because they could seen anymore the position of the letters they had to collect for the final fight. In conclusion, for a future iteration, this feature can be removed or improved.
6. In order to keep the player in the flow of the gameplay, some proper background music could be played.
7. According to our users, the rotation speed of the ship is too high, and needs to be decreased, to avoid interruptions of the game.

Future Research

Besides the game in itself, further research should also be done on multi modality games in its entirely. Based on observation we indicated that players seem more skilled at one modality than the other and do not always succeed in combining them at the same time. It should thus be researched whether multiple modalities in games are either always difficult to combine for players or can be trained, much like muscle memory for standard game controllers. If the former is the case, it should also be researched what the optimal amount of multiple modalities is and what exact game design should be applied to get the optimal use out of multiple modalities.

Appendix A: Results of User Studies

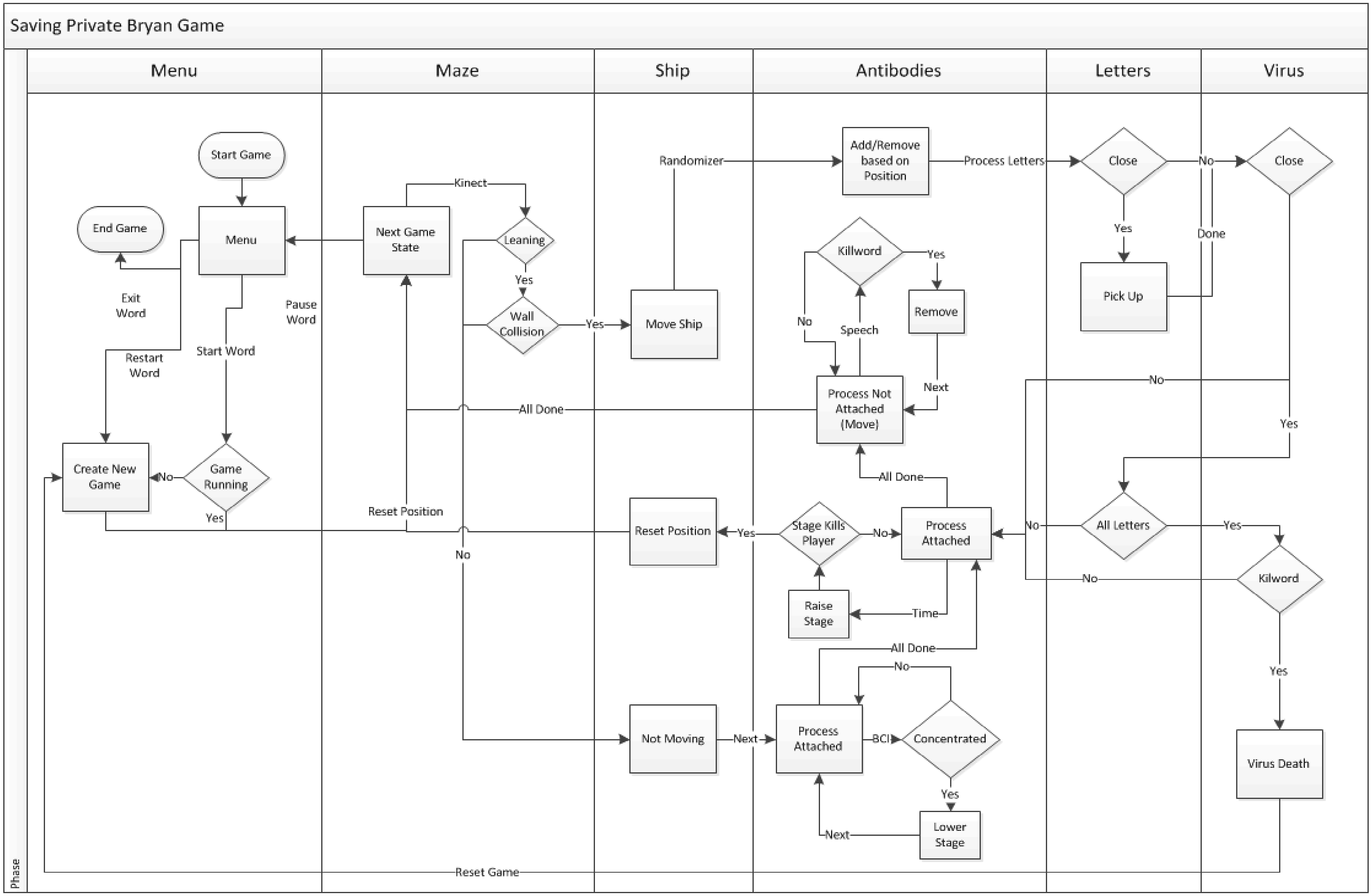
	Pirate	Bryan
1. The control scheme was clear to me	4.2	4.2
2. The controls felt natural to me	**4**	3.5
3. It was easy to learn the controls	4.3	**4.4**
4. The controls add something to the gameplay when compared to traditional controls	3.6	**3.8**
5. I would prefer to play the game with these controls instead of traditional controls	3	**3.5**
6. The game was fun to play	4	**4.1**
7. The game was challenging enough	2.7	**3.1**
8. The game concept was interesting	3.9	**4.1**

Table 1 Average scores of interview questions (all users)

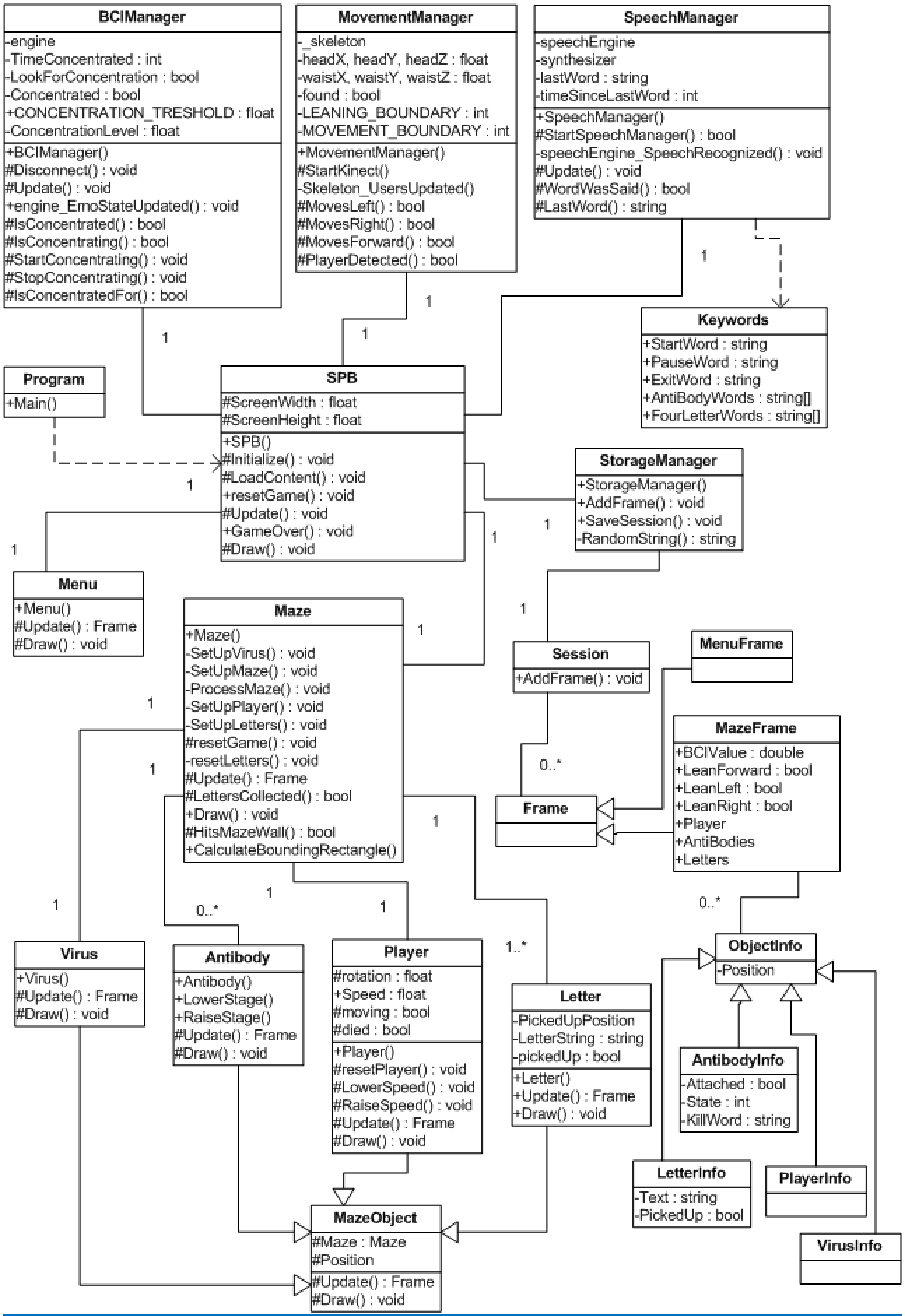
Open Questions	Brain Game	Pirate Game	Undecided
Best Controls	**5**	**5**	0
Most in Control	**5**	**5**	0
Best Speech	2	**5**	3
Best BCI	**5**	1	4
Best Kinect	3	**6**	1
Favorite Game	4	**5**	1

Table 2 Summation of users' preference (comparative questions)

Appendix B: Flowchart

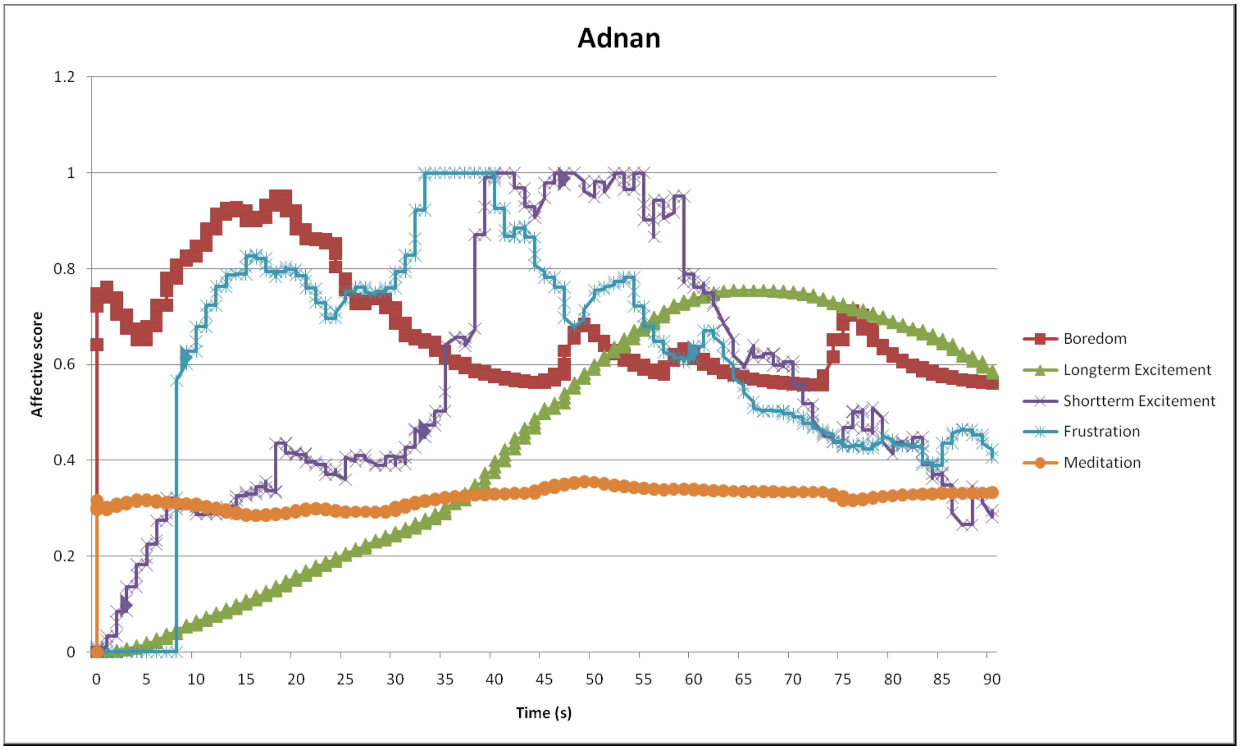


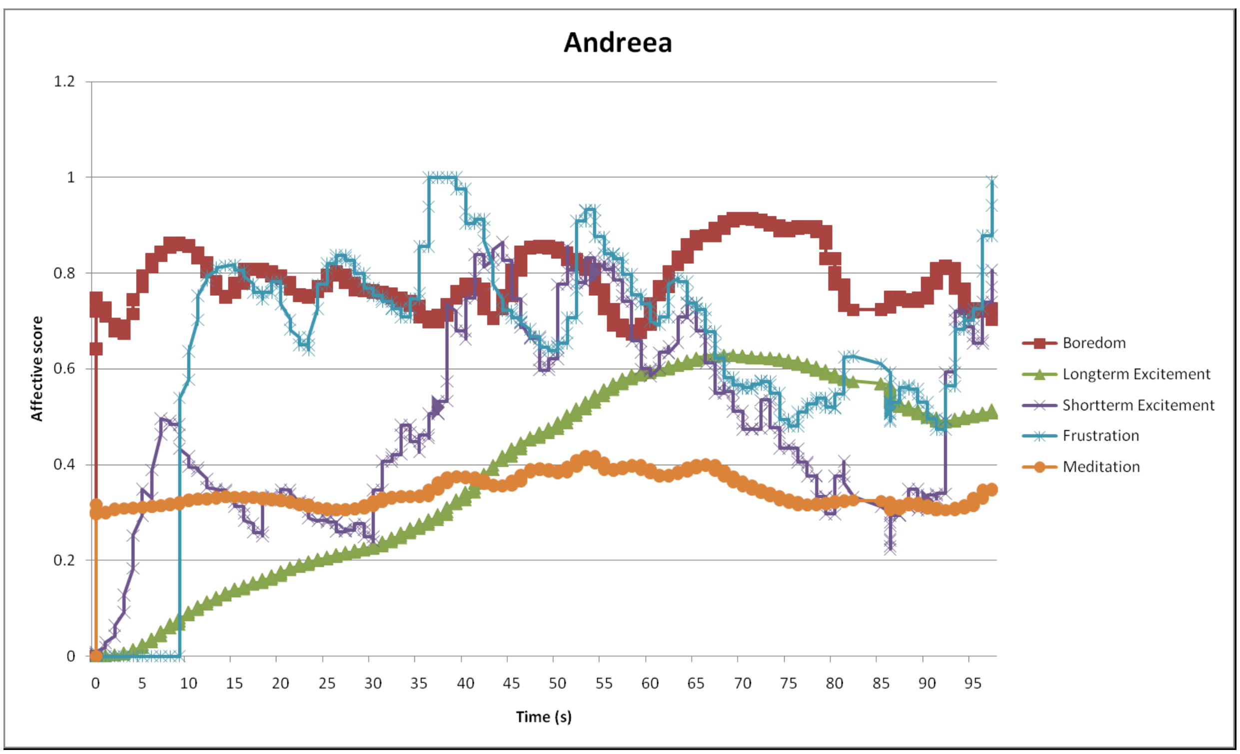
Appendix C: Class Diagram

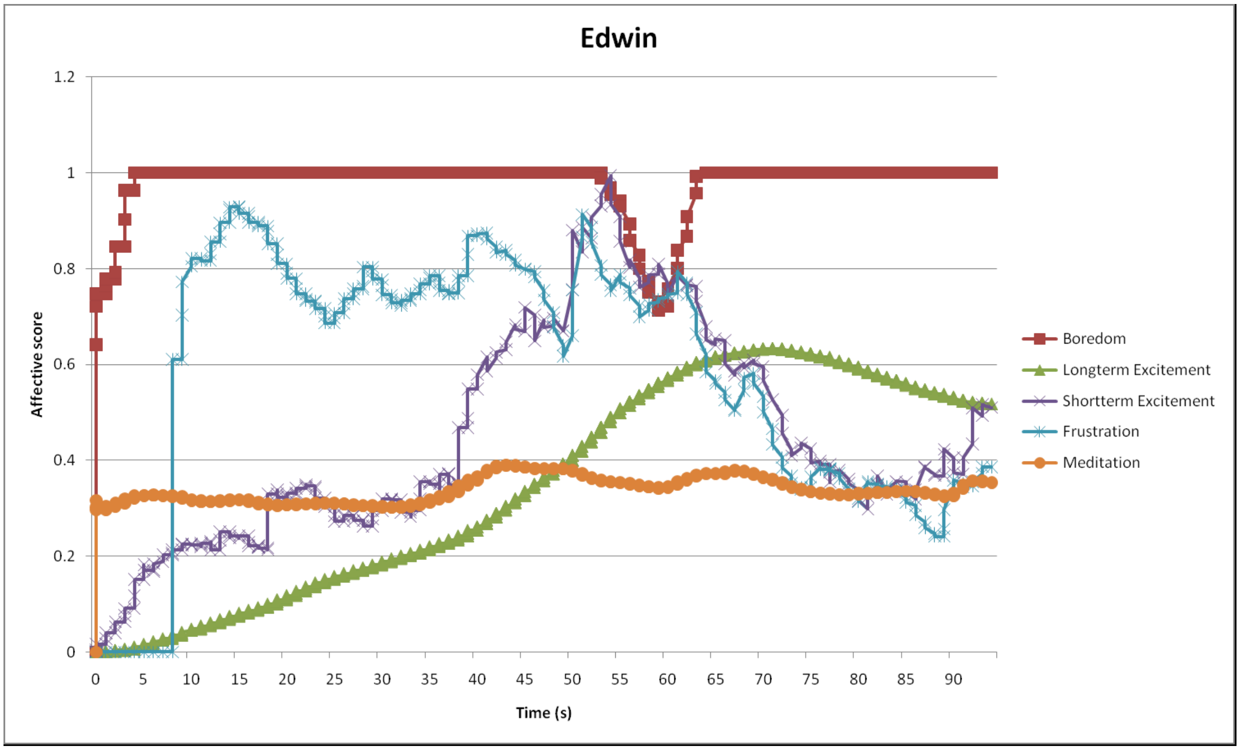


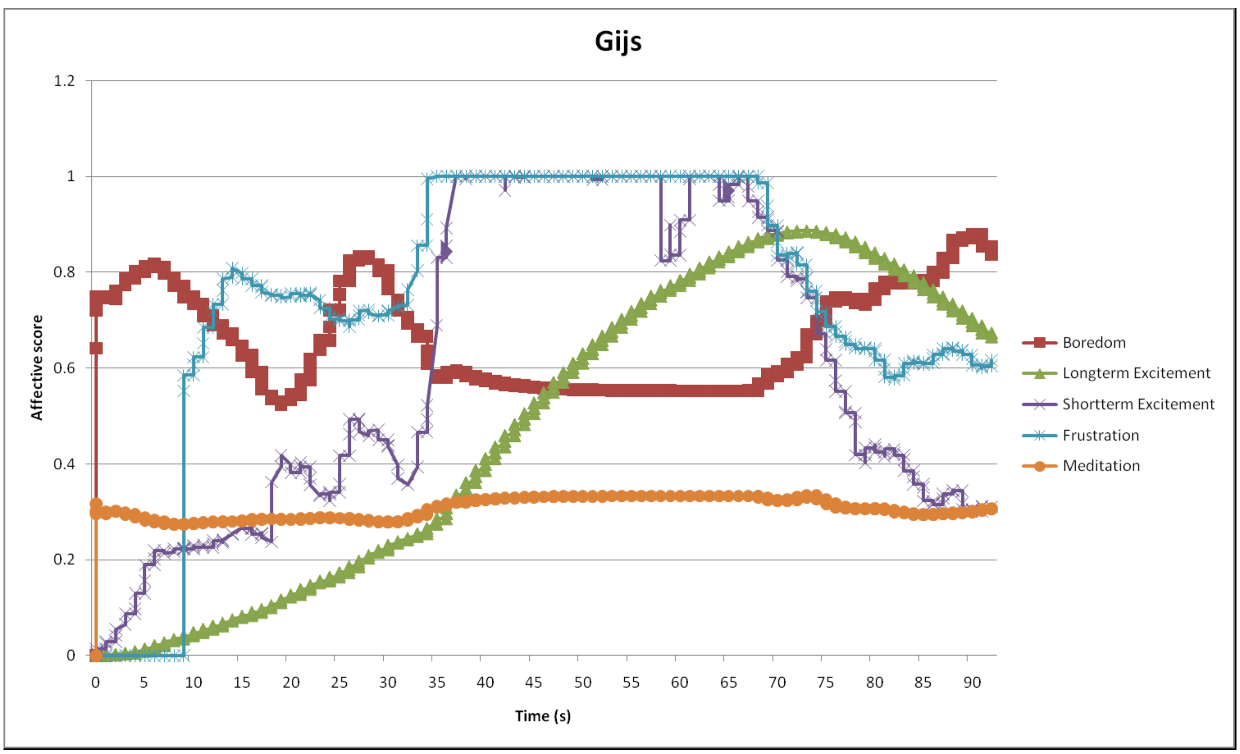
Appendix D: Emotiv test results

These graphs show the results of an experiment where we measured affective scores generated by Emotiv in rest and while actively moving. The period from 0 to 30 seconds is a state of concentration. During 30 to 60 seconds the person moved his body actively. He/she started to concentrate again after the 60 seconds mark.






Edwin
Affective score
Time (s)
Boredom
Longterm Excitement
Shortterm Excitement
Frustration
Meditation


Gijs
Affective score
Time (s)
Boredom
Longterm Excitement
Shortterm Excitement
Frustration
Meditation

Appendix E: Game Questionnaire Results

S.No	Question	Yes [%]	No [%]
1.	Did you know when you had to start concentrating?	80	20
2.	Did you know what you had to do to activate the shield?	95	05
3.	Were you able to activate your shield by concentrating when you wanted to?	80	20
4.	Did you get proper feedback about your concentration level while you were playing?	60	40
5.	Did you have the feeling that you could influence your concentration level while playing?	70	30
6.	Did it feel natural to concentrate in order to make the antibodies disappear?	55	45
7.	Was the movement of the ship clear to you?	100	00
8.	Could you successfully move and rotate the ship at the same time?	35	65
9.	Did the ship rotate too fast?	70	30
10.	Did the ship move too fast?	05	95
11.	Did the ship respond well enough to your movement (was the feedback good enough)?	50	50
12.	Was it tiring after a while?	40	60
13.	Did you ever lose your balance or felt dizzy while moving your body?	05	95
14.	Was it clear when you could use speech to destroy antibodies?	90	10
15	Did the speech detection respond fast enough to the words you spoke?	70	30
16.	Did you feel like you had enough time to read the words you have to say?	70	30
17.	Does the speech control (as it is implemented in the game) feel as an addition to the game?	80	20
18.	Did you like the fusion of the speech, motion and Brain Computer Interface, over traditional keyboard controllers?	75	25
19.	Would you prefer to use traditional keyboard/mouse controls for this game instead of speech, motion and BCI?	55	45
20.	Were you at any point confused because you had to combine speech with movement?	15	85
21.	To use the shield in game, you had to stop moving and start concentrating. Did you feel that this broke the game flow too much?	50	50
22.	Did you know what you had to do while playing the game (did you know what the goal was)?	100	00
23.	Were all the game elements visible enough to you from the distance you were standing at?	65	35
24.	Did you find the game challenging?	75	25
25.	Was the tutorial clear to you?	90	10
26.	Would you like to play the game again?	85	15

The questions below were obtained on a 7 point Likert scale. We compute the average of the user response and present its interpretation here:

S.No	Question	Mean	SD	Interpretation
1	How difficult was it to maneuver the ship?	2.3	1.1	Moderately difficult
2	How difficult was to collect all the letters in order to kill the virus?	3.7	1.5	No opinion
3	Would you like more or less influence of speech in the game?	4.1	1.1	its perfect as it is now
4	Would you like more or less influence of movement in the game?	3.7	0.9	its perfect as it is now
5	Would you like more or less influence of BCI (Brain Computer Interfacing) in the game? (BCI is the part when you were concentrating)	3.6	1.2	its perfect as it is now

The following is the response to a multiple choice question:

S.No	Did at any point you feel frustrated with the game? If yes, when?	While speaking [%]	While concentrating [%]	While moving [%]
		10	25	85

The following is the user response to an open ended question:

Subject ID	Question: What features of the game would you like to see changed?(you can also write any suggestions or comments here)
1.	better collision detection :)
2.	
3.	the rotation should be more segmented - the letters we are supposed to gather should be larger
4.	Better BCI integration, more movement options, high score, letting the ship slide over walls instead of getting stuck
5.	movement is hard, ship often sets off in a direction that was not intended some pixels are hard to miss, ship gets stuck
6.	Magnifier is confusion; it's hard to see where the letters are. Also, adding more different words besides flower, tree cat and house would be nice.
7.	Definitely has some interesting interface concepts going on, but the way you currently interact with the game feels a bit awkward to me; having to stand there bending your body and shouting random words.
8.	It's too easy to get stuck on the walls. Turning seems to respond somewhat slowly, the ship keeps turning longer than I intended to. That makes it hard to fly whilst steering. I'm personally not a big fan of speech control.
9.	Movement was a bit hard, moving just a bit to right or left is hard (you move a lot or you don't move). Also the concentration level is very low for me so it is almost impossible to relax so much during the game.
10.	The cat didn't listen xD. Also turning the ship sideways, I often went a little too far.
11.	-

12.	-
13.	I like the part where you have to concentrate because it put's you back on the ground if you're doing it on autopilot. You have to think for a moment „DIE VIRUS", then you think oh I trying to do this game. I think after training though you may even do that on autopilot.
14.	Walls were too obstructive, you should not be completely blocked when only partially moving into a wall (you should be able to slide along them, so to speak)
15	I got stuck to the walls a lot
16.	the font of the words was a little too small to read fast, the turning the ship could be slowed down
17.	Fix the collision detection :) Concentrating broke the flow of the game a bit Speech could maybe be used to select a target for a missile?
18.	The game isn't challenging because there isn't much reward, explosions when destroying enemies would've been nice (amongst other 'standard' reward/fancy/nice stuff), it gives a quite direct form of feedback and is nice to see (which is why the user likes destroying stuff for some part). Control scheme is difficult, moving your body takes time and corrections are hard to make that way. It might be better to move the ship in the direction the person is facing (e.g. person leans forward, ship goes up on the screen. Person leans left, ship goes left). Feedback on the brain interface wasn't direct, which makes it hard to guess whether you're doing it right. A line along the side of the screen or a health-bar like system would've given much more direct feedback. Something along the lines of how to control the strength of a throw or something like that in bowling or golfing-games would've been a better fit I think.
19.	Better / more flowing movement controls
20.	collision detection, movement: while standing still the rotation is good but while moving it reacts to quick to much (turning too quick), the magnifier is not making it more clear because you loose a part of the maze because it's not visible

Appendix F: Movement statistics

User ID 19 was left out of these figures because the average BCI value was 0, which means that the Emotiv headset was not properly working.

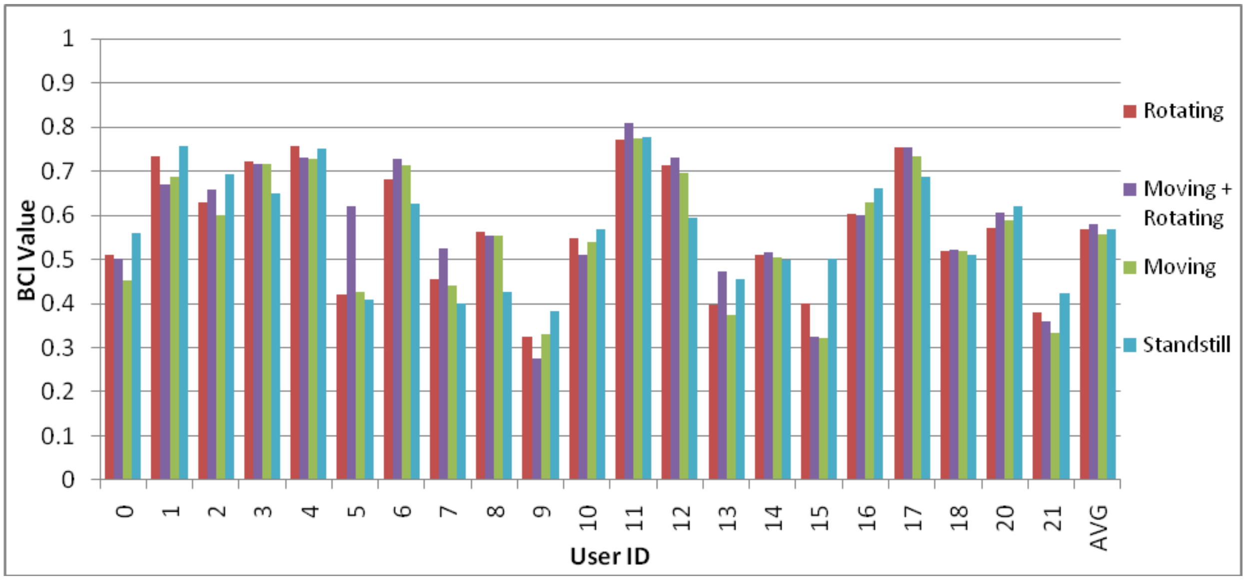


Figure 9 - User movement data

This chart shows the percentage of time the user spent in a certain state. These states are mutually exclusive.

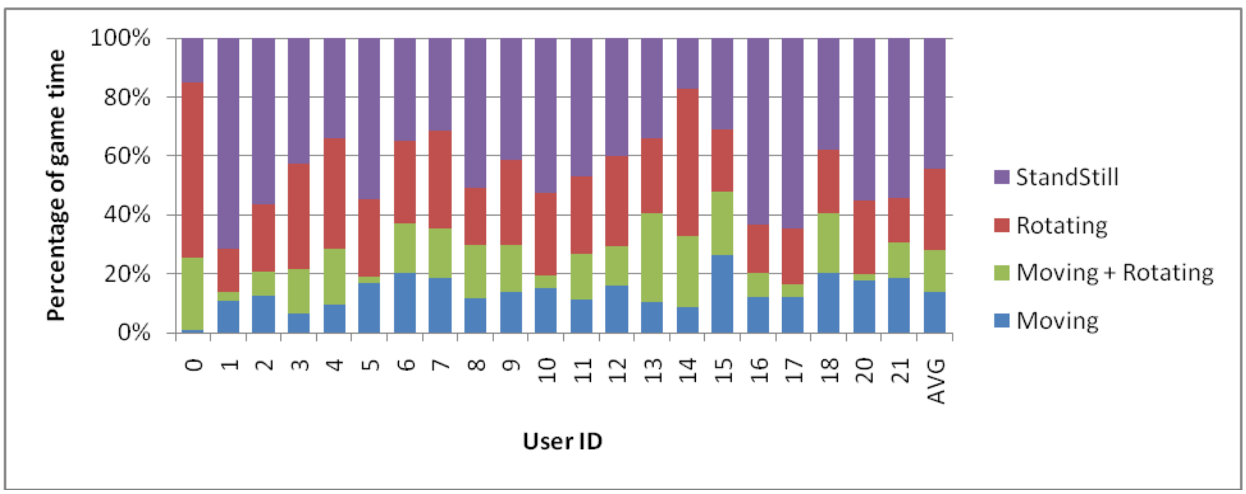


Figure 10 - Partition of user movement

The above chart shows the percentage of the total game time that the user spent in one of the four movement states.

Appendix G: User Statistics

The table below shows the playtime, deaths, antibody kills and BCI values for each play session. The table has marked color coding to indicate where players performed better than average and where they scored worse.

id	Time	Deaths	Avg Death Time	AB Deaths	Concentration	Speech	Average BCI	Death/Time	AB / Min	Con/Min	AB Ratio
0	0:16:35	1	09:58,00	182	56	126	0,515	60,10%	10,97	3,38	4:9
1	0:15:29	4	04:29,22	48	16	32	0,744	28,98%	3,10	1,03	1:2
2	0:12:56	9	06:39,57	56	27	29	0,663	51,49%	4,33	2,09	27:29
3	0:10:09	5	03:40,10	48	27	21	0,69	36,14%	4,73	2,66	9:7
4	0:14:33	5	06:41,97	123	14	107	0,748	46,04%	8,45	0,96	14:107
5	0:06:51	0	00:00,00	26	17	9	0,42	0,00%	3,80	2,48	17:9
6	0:10:19	1	00:50,55	93	26	67	0,677	8,17%	9,01	2,52	26:67
7	0:08:31	0	00:00,00	62	34	28	0,447	0,00%	7,28	3,99	17:14
8	0:11:08	1	02:15,47	59	20	39	0,491	20,28%	5,30	1,80	20:39
9	0:04:50	1	01:42,52	25	13	12	0,342	35,35%	5,17	2,69	13:12
10	0:12:39	2	06:29,85	55	36	19	0,555	51,36%	4,35	2,85	36:19
11	0:15:25	7	04:30,59	102	21	81	0,781	29,25%	6,62	1,36	7:27
12	0:05:53	1	02:07,86	46	12	34	0,665	36,22%	7,82	2,04	6:17
13	0:07:53	0	00:00,00	62	26	36	0,436	0,00%	7,86	3,30	13:18
14	0:07:42	0	00:00,00	91	26	65	0,509	0,00%	11,82	3,38	2:5
15	0:08:51	2	01:18,10	65	13	52	0,395	14,71%	7,34	1,47	1:4
16	0:10:12	3	02:52,89	40	12	28	0,644	28,25%	3,92	1,18	3:7
17	0:10:33	2	03:50,28	35	16	19	0,709	36,38%	3,32	1,52	16:19
18	0:05:07	0	00:00,00	47	6	41	0,516	0,00%	9,19	1,17	6:41
20	0:10:01	0	00:00,00	72	17	55	0,603	0,00%	7,19	1,70	17:55
AVG	0:10:17	2,2	02:52,35	66,85	21,75	45	0,5775	27,94%	6,58	2,18	29:60

Table 3 – user statistics on player and antibody death

Appendix H: Example play through

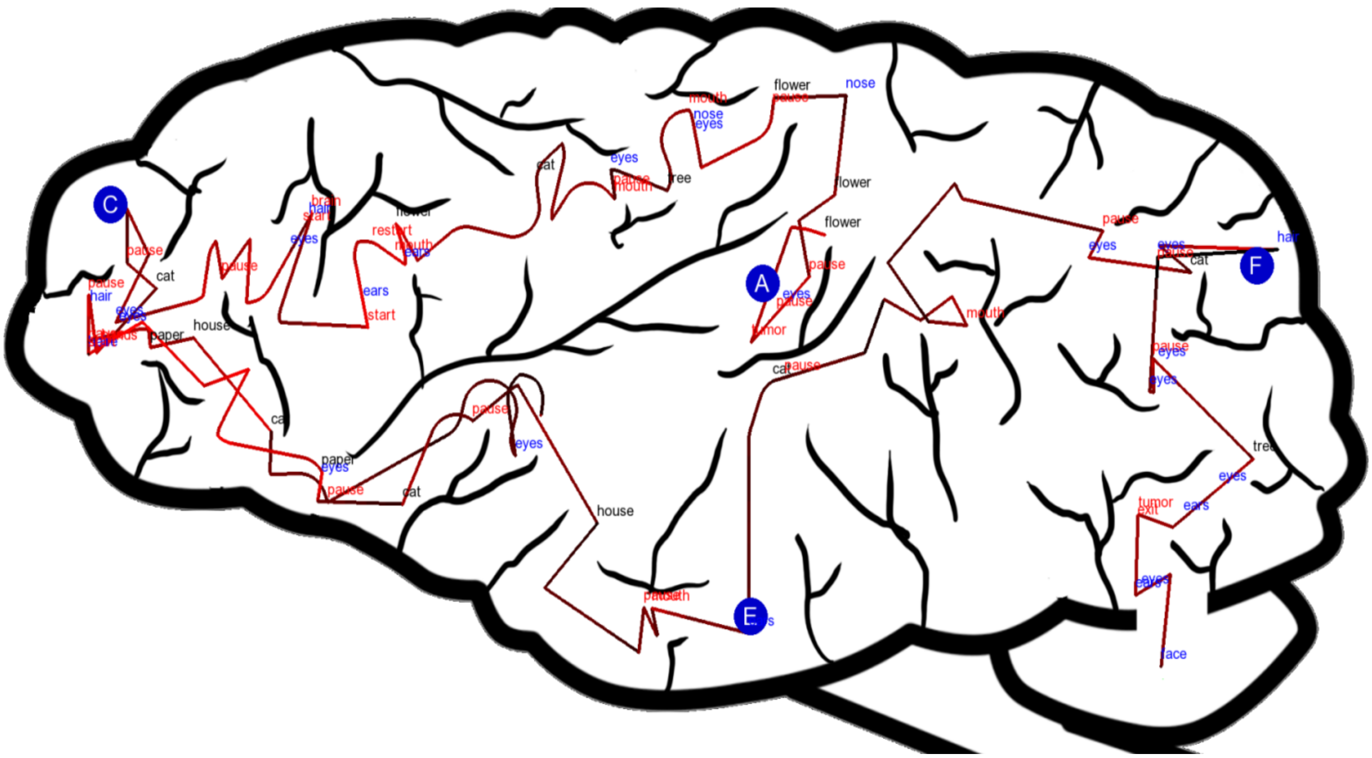


Figure 11 - A play through of one of the users

This image shows a typical game session by one of the participants. The line shows the path that the player followed over time. The color of the line indicates the BCI value on a scale from black (= low BCI value = concentrated) to red (= high BCI value = excited). A point where the line suddenly stops indicates that the player got killed. Some of the words that were recognized by the speech recognition software are displayed along the path. Black words are used to kill antibodies, blue to finish the game, red for game control. Not all words are displayed, to increase clarity of the image. The blue circles indicate the collectable letters.

Appendix I: Subject during gameplay