Bachelor Creative Technology

Influence performance of breathing exercises with game elements

Author:

Thomas Rebel

S1986082

Creative Technology

Supervised by:

Dr. ir. Dennis Reidsma (1st supervisor)

Dr. ir. Robby van Delden (Critical observer)

UNIVERSITY OF TWENTE.

Abstract

There are many domains that use breathing exercises to help achieve their goals. Singers to sing better, pregnant women to relax more, etcetera. These exercises are boring to do and do not give feedback about whether the execution is good or not. One solution is the use of serious games, to make the exercises more fun and to give the players feedback. The problem is that there is not much information on how to help the player execute the breathing exercises better. This is currently mostly done by explicit messages, explaining what the player needs to do. This is for most people not entertaining and brings them out of the flow of the game. The solution could be implicit stimulating elements: elements that are put in the game to increase performance of breathing exercises without telling the player outright, but with subtle sound cues and non-text visuals. The main point of this report is to see how these elements should be implemented. If game designers know more about this subject, the chance of the elements being effective without disrupting the flow of the game could potentially be much higher.

Previous research showed many different kinds of stimulating elements. Sound and visuals can help performance of certain breathing exercises, especially with the timing and duration. Other studies recommend not to add too many of these elements, because that can distract the player.

To get more insight of how to implement the elements, a serious game with breathing controls was made. There were two versions of the game, one with implicit stimulating elements and one without. Participants would play both games, and would fill in a questionnaire to see whether they noticed the elements and how useful they were. Furthermore, in-game data was collected to see how well certain breathing exercises of the game were executed. By comparing the results of the data and questionnaire between versions and participants a few observations were made on how to implement the breathing exercises. When implementing implicit stimulating elements, it is important to make them noticeable. This can be done by removing distracting game elements, or putting the elements on locations where the player mostly focuses on. Furthermore, it is also important that the stimulating elements are clear how they work and how to interpret them. Otherwise, the players can be confused, which decreases the effectiveness of the elements. Lastly, it is important to know the limits of the implicit stimulating elements. They are potentially very useful for easy breathing exercises, like just inhaling or exhaling. Nevertheless, for more complex exercises, where the player has to do multiple actions to complete it, it is sometimes better not to only use implicit stimulating elements. For example the use of explicit elements can sometimes be necessary. Game developers of affective games can use these observations to potentially make their implicit stimulating elements more effective, and their effects more predictable.

Acknowledgement

I first and foremost want to thank my supervisor Dennis Reidsma and my critical observer Robby van Delden for assisting me with this report. They gave me valuable feedback and helped me to set this report up. Furthermore, I want to thank Brolin Fernandez for making his software for breathing recognition free and let me use his sensor that he already programmed. Lastly I want to thank all the participants of the experiment for helping my research.

Table of contents

1 Introduction

The new trend in the entertainment business for decades is games. More and more they are used by people of all different age groups. The industry is even bigger than the television industry[1]. With the rise of video games, a new phenomenon is arising: serious games. While trying to keep the positive aspects of normal games, serious games try to teach something. This can be behavior, physical training, or just plain information. The only issue is that this new kind of learning material is very new, and people in many different industries try to decipher where it can be used for and what the best way is. What is known is the fact that serious games do have many upsides to them. They engage and motivate users, while also bringing fun and giving immediate feedback on their actions[1]. Furthermore, what mostly makes them engaging is that they are interactive. Because of that, they “create an active learning environment, support problem solving and learning through practice and enhance the educational curriculum”[1].

Serious games can be used for many different learning goals. This report is going to focus on breathing exercises, but why this subject? Breathing exercises are helpful for different kinds of domains. Domains that use and profit from these exercises among others.

- Singing, for one to increase the quality of the singing voice[2]
- Mindfulness, increases self-awareness, regulates the autonomic nervous system, and focuses the mind[3].
- Pregnancy, among other things, decreases anxiety and controls the duration of labor[4].
- Therapy, to slow down or halt the progress of conditions, such as ADHD and COPD.

The problem with breathing and similar affective exercises is that users lack the motivation to (continually) doing the exercises. Furthermore, they also get frustrated by the lack of clear benefits [5]. Serious games can be the solution to these problems, because they can give feedback on the performance, are engaging, and can show the user what their skill level is and if it is increasing.

To use serious games for breathing exercises, there are some aspects to take into account. The serious game should be intuitive to use, still be engaging and flexible so the game can adapt to the skill level of the player. By improving and applying current mapping and stimulating elements used by serious breathing games, the usefulness and engagement will rise thus which will improve the game. While there are already many papers about using breath inputs, only a small amount focuses on stimulating elements.

Stimulating elements are elements that positively affect the performance of breathing exercises.. These elements are divided into 2 categories: explicit and implicit. Explicit elements are obvious, and try to literally tell the players what they should do. Examples for this are typed explanations, examples, and feedback that literally tells the player about how to improve. Implicit stimulating elements are more subtle. These elements show the player how to perform exercises by non-text visuals and sound cues. Some examples are audio beats to stay into a rhythm and small game rewards when doing something correctly. Implicit stimulating elements can be very helpful. When used correctly, they do not break the flow of the game, while also

1:https://www.marketwatch.com/story/videogames-are-a-bigger-industry-than-sports-and-movies-combined-thanks-to-the-pandemic-11608654990

helping the user achieve their learning goals. So it is important to know how to correctly implement these elements. That is why the main research question is as follows.

Main RQ -- *How to implement implicit stimulating elements in a serious game on breathing techniques to increase the correctness of the execution of breathing exercises?*

To answer the main research question, more understanding is needed about breathing exercises and breathing games. To get enough knowledge about breathing exercises, breathing games, and stimulating elements, the following sub-research questions need to be answered.

1. What are the advantages and disadvantages of existing serious and non-serious breathing games?

It is very insightful to see what predecessors have done. To not make the same mistakes and use and combine what makes them great. To map the advantages and disadvantages, existing papers and user reviews of breathing games are used. Chapter 2.1 shows the discovered breathing games.

2. What are the advantages and disadvantages of existing sensors to measure breath?

It is the same story as the previous question, to learn from existing solutions. For the sensor, it is important that in the end, it can measure all the different breathing exercises with significant accuracy. By combining technical reports and user reviews, the reliability and accuracy of the sensors can be determined. The explored sensors are found in Chapter 2.2.

3. What breathing exercises exist for singers and pregnant women and how do they work?

To do the mapping of breathing inputs correctly, the first step is to know what the breathing inputs are. By listing all the exercises and seeing what actions are important to do them correctly, it is easier to implement them as breathing inputs. By using interviews of teachers of singers and pregnant women in other reports the exercises are easily found, without the need of doing an interview. The results are in Chapter 3.1.1.

4. Do certain breathing exercises of singers and pregnant women have similar actions to complete them?

Breathing games can help more people if they are flexible to use. To make it possible that a breathing game can be used in multiple domains, similarities of exercises need to be found. When there are similarities between domains, the same breathing inputs can be used, thus the same game can be used. By looking at what the exact actions are to do a breathing exercise and comparing it to other exercises of the other domain, similarities can be found. The similarities can be found in Chapter 3.1.2.

5. What flexible models and approaches are important for mapping game action controls to specific breathing exercises?

The breathing exercises are already determined, but how do you implement them in games? Without doing research beforehand, there is a high chance that the breathing inputs are implemented incorrectly and can negatively affect the game. By learning from existing models and approaches of implementing breathing controls, the chance that breathing inputs are implemented in the wrong way will diminish. See Chapter 3.2.1 for more information.

6. What are the effects of using a combination of traditional inputs and affective inputs as game action controls?

Knowing how to implement the breathing inputs is not enough. It is also insightful how the combination of traditional and affective inputs will influence the player. By knowing how to balance the use of both inputs, the chance of wrongly mapping the breathing inputs will diminish. By using different existing reports about the subject, guidelines can be created to implement the breathing controls. The conclusion is found in Chapter 3.2.2.

7. How do serious games on breathing exercises stimulate users to execute them correctly?

What is most important in a serious breathing game is that users learn from it. If the exercises are not done correctly, the user will not improve himself. By looking at different ways to influence the correctness of the execution then only sensors, the chance is higher than breathing exercises are done correctly. By looking at how existing games can influence physical behavior through reading reports about the matter, the stimulating elements can be discovered and thus used. The answer is explained in Chapter 3.3.1

8. How do teachers of breathing techniques stimulate their students to execute the breathing exercises correctly?

In the last question only existing serious games were researched, but games are not the only material that teaches breathing exercises. Teachers of breathing techniques also teach people how to do breathing exercises. The teachers use also stimulating elements to motivate and guide the students to do the exercises correctly. By interviewing some experts, these elements are identified and converted to game elements. The conclusion is found in the last chapters 3.3.2 to 3.3.4.

To see whether the implicit stimulating elements have effects on the execution, two almost identical games need to be made. The key and only difference between these games is that one has the implicit stimulating elements implemented, while the other does not. The next step is to compare how well the breathing exercises are executed per game. If the execution of breathing exercises of the game with implicit stimulating elements is better than the game without, the elements work. When the implicit stimulating elements seem to not work, the reason why needs to be determined by using the interviews of the users.

2 State of the Art

The first step of the research is to use the information from existing breathing games. For one, information about stimulating elements that work or does not work. In addition, other factors hat need to be kept in mind, like how to design the control scheme. Another part that is important to find information about for later is sensors. For the breathing game, breath is going to be measured, and it is vital to do that as reliably as possible. Furthermore, the sensor should be able to measure certain breathing exercises, and by learning about the limits, it is possible to check which exercises each sensor can handle. The following sections will answer research questions 1 and 2.

2.1 Breathing games

It is important to learn from other serious breathing games. These can give insight into how users play these games and what makes them work. Serious breathing games that have a paper about them have the most reliable information, so it is fruitful to find as many of these as possible. Nevertheless, breathing games are not researched and still can give helpful information. This information is found mostly on their websites. Nevertheless, using only the information that is provided by the creator is not reliable, because there is a high potential it is biased. A solution is to read user reviews, to see what they think and if it aligns with what the creator had written down.

2.1.1 SilverFit Flow[2]

SilverFit Flow is a commercial serious game on breathing exercises that helps people with lung problems, like Dyspnea, COPD, and asthma. SilverFit admits that breathing exercises are boring and repetitive, so they see this game as the solution. The game first has to be played under the supervision of a therapist (see figure 1), but later when the patient displays enough skill the exercises can be done independently. SilverFlow has 3 minigames that the user can choose from, shown below.

- Javelin throw: Throw a javelin with the power of your exhalation.
- Star path: Collect stars by breathing in a certain rhythm
- The diamond mine: Breath in sucks diamonds into a vacuum cleaner.

SilverFit Flow is a new innovation, so there are not many user reviews to be found. SilverFlow has been tested by patients. Nevertheless, the opinions of the user were positive. The participants lost track of time while playing the game, which could mean they were in flow thus enjoying playing the game. Using serious games for breathing exercises potentially works.

2:https://www.zorginnovatie.nl/innovaties/silverfit-flow

Figure 1: User with therapist playing SilverFit Flow. Picture from SilverFit. From https://silverfit.com/nl/producten/silverfit-flow-ademhaling/710-ervaring

2.1.2 Breathing Games[3]

Breathing Games is, surprisingly, a commercial serious game on breathing exercises that helps children with breathing or stress-related disorders (see figure 2). They focus on teaching Pursed Lip Breathing, a breathing exercise that will give the participant more oxygen, energy or help him to relax. Breathing games have a large number of minigames, more than 30, that is still being increased to this day. They use a headset to measure breathing, and also give a breathing diary to track progress and compete with other users by comparing scores.

Most user reviews were positive about the games. A few users did complain about the controls being unreliable. This is probably caused by the fact that they did not use the special sensor of Breathing Games, but this is not for sure. What can be concluded is that it is important to use the right sensors to measure breath. In addition, Breathing Games also shows that breathing games are potentially a good solution for breathing exercises. Lastly, in contrast to other breathing games, Breathing Games brings a large amount of content, which seems to be liked by the users. Breathing Games shows that it is important to be clear how a game needs to be played, otherwise misconceptions can be created. Furthermore, they also show that serious games can potentially be a good way to do breathing exercises.

Figure 2: User playing Breathing Games with their controller. From https://www.fitness-gaming.com/news/health-and-rehab/breathing-games-and-toys-help-manage-asthma-and-stress-with-fun-gaming-experiences.html

2.1.3 ChillFish[4]

ChillFish is a breathing game for children with ADHD[6]. They combine medical knowledge with game design principles, to make an engaging game that calms the children down. The game uses a respiration sensor, made out of Lego (see figure 3). The goal of the game is to collect starfish as a fish in two minutes. By inhaling and exhaling, the fish goes up and down. The game can be relaxing for users. This is done by removing a way to lose, a calm theme, and aesthetics. ChillFish shows that to achieve the learning goal, it is good to put attention to every

3:https://www.breathinglabs.com/breathing-games/
4:http://www.chillfish.dk/

detail, even how the sensor looks.

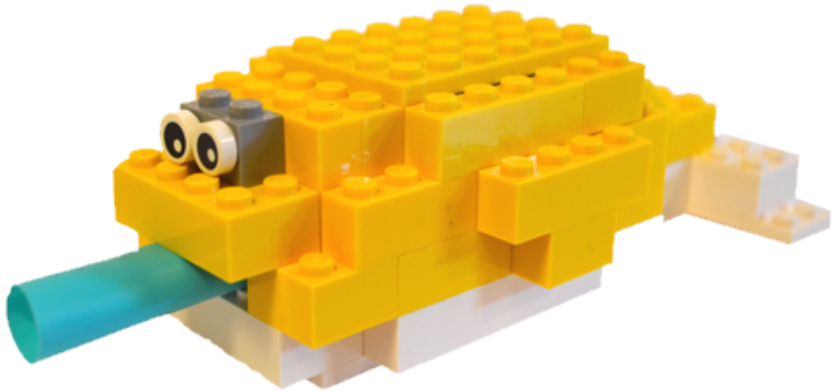


Figure 3: ChillFish respiration sensor. From http://www.chillfish.dk/.

2.1.4 Rakesh Patibanda et al. games

A group of researchers wanted to gain more insight into how to design better breathing games[7]. The intention of the researchers was not only to use serious games to relax users but also how to use breathing inputs in an engaging way. Like Breathing Games, they focus on teaching Pursed Lip Breathing. The researchers made three games to get more insight into the mindset of the players that are described below.

- Space Gaze: Collecting oxygen to stay alive by moving. Going up and down is done by inhalation and exhalation. What is interesting is that Space Gaze slows the game down, so it looks like everything is in slow motion. They do this to relax the player more. It is a very original idea. The takeaway is that it is useful to think outside the box, especially for stimulating elements.
- Focus Tree: The goal is to grow as many trees as possible by breathing in rhythm. Clouds will appear and disappear in front of the player when they inhale/exhale. The players liked the nature theme, and that their progress can be easily tracked by the number of trees that were planted. Focus Tree shows a good way to give implicit feedback.
- Outlandish Whisper: It is an ambiguous particle game. By breathing in gold particles will go into your mouth. By breathing out red, when you breathe out of rhythm, or blue, when your breath into the rhythm, will come out of your mouth. A pre-recorded voice will speak to explain the game and help the player to get in a rhythm. Outlandish Whisper implicit particle feedback was useful for the players. The game shows that it does not matter how vague a stimulating element may look, if it is implemented correctly it still can be highly useful.

2.1.5 Fernandes' games

Fernandes is also a researcher who made a few breathing games to get more knowledge about serious breathing games[5]. For the games, he focussed on breathing exercises for singers. The games were connected to an app, so the teacher could see what the students were doing. The advantage of the app is that the teacher had much influence on the progress of the students. He could make exercises easier or more difficult. Due to time limitations, the game works with minimum functionality but is sufficient to see the advantages.

- Breath Slash: A game inspired by Beat Saber. The goal is to destroy upcoming blocks with the right timing of inhalations and exhalations. The game could be made more difficult or easier when needed. The game was enjoyable for the participants, and they liked that they could put their own songs in the game. Breath Slash proves that rhythm games have a high potential to be breathing games, in addition to that implicit audio cues can help the performance of the exercise.
- Road Tripper: The player controls a moving car by letting him move up and down by inhaling and exhaling. The car needs to collect collectible items and avoid any obstacles. The users liked the collectibles, it increased their engagement. Interesting was that the timing of beats of the background sound influenced the players' actions. It gave

feedback when a player's timing was off but confirmed when the timing was right. It is a good way of using implicit stimulating elements.

2.1.6 Robby van Delden et al. games, SpiroPlay

Another group of researchers wanted to "prevent asthma exacerbation via regular monitoring of children with asthma through spirometry at home"[8]. They made and tested three mini-games for their target group, built into a game called SpiroPlay. The three mini-games worked as follows.

- Car metaphor: The player controls the car by breathing. Different aspects of the breath have different effects on the speed. The goal is to go over the finish, but it is possible to go further.
- Popping balloons metaphor: In this game, the player controls a bow with his breath. The goal is to shoot through as many balloons as possible. Different aspects of breathing have an influence on whether the arrow will be fired or not, and how many balloons the arrow will pop.
- Diving metaphor: Here the goal is to let the diver jump as high as possible using breath.

The first learning point of SpiroPlay is that they discovered a difference between *Fun* and *Prefer to Play*. What looks fun, does not mean the user will keep playing. While this concept should still be researched more, it is still useful to know to look for long-term enjoyment of the game. The second learning point is to make the breathing inputs reliable by connecting them to real-life actions. Some game actions can feel intuitive to real-life action. For example, letting an in-game dragon spit fire is breathing out a more intuitive action than moving their arms for the player. When done correctly, the spirometry results would increase. This report shows the use of metaphors as stimulating elements could be useful.

2.2 Sensors

There are multiple sensors that can measure some aspects of breath. It is useful to know what exactly a certain sensor can measure, and what other advantages and shortcomings the sensors have. By reading explanations of the websites these sensors are sold on, these specifics of the sensors can be mapped.

2.2.1 Microphone of phone or laptop[5, 6]

The most obvious sensor to measure breath is a microphone. Using microphones has many advantages. For one, almost everybody has one. Phones, laptops, headsets, all contain microphones. Microphones do not need physical contact with the user to work, which makes them comfortable to work with. A microphone can measure tone, the strongness of breaths, and the pace of breaths. Some researchers even managed to recognize breathing disorders like asthma[7]. The limits are that a microphone cannot measure whether you breathe from your chest or abdomen, and cannot distinguish whether you inhale or exhale.

2.2.2 Respiration Sensor[8,9]

A respiration sensor is similar to a strain sensor. It will measure the pressure of the abdomen or chest. This is done by a very sensitive girth sensor. Just like the strain sensor, it will measure how deep the breaths are, the pace of breathing, and where it is measured. The difference between the respiration and strain sensor is that it is more comfortable to wear. It is also made for breathing, so most have already built-in software to recognize breath.

2.2.3 Spirometer[10]

A spirometer is a device that measures long functions. Its main function is to recognize breathing disorders, like asthma and COPD, but it can also be used as a normal sensor. This is done by breathing into a tube. Then it measures, depending on how advanced the device is, the strength, flow, and irregularities of the breath. Some spirometers have a difficulty setting, which influences how hard somebody has to breathe. It can also sense inhalations and exhalations. An advantage compared to the other solutions is that the spirometer can be more flexible to use because some of them have controlled resistance. A disadvantage is that it cannot measure if the user is breathing from the stomach or chest.

5:https://www.washington.edu/news/2012/09/18/app-lets-you-monitor-lung-health-using-only-a-smartphone/
6:https://asa.scitation.org/doi/10.1121/1.4865269
7:https://www.oulu.fi/university/news/mobile-respiratory-measurement-tool
8:https://www.nature.com/articles/s41746-019-0083-3
9:https://www.mindmedia.com/en/products/sensors/respiration-sensor/
10:https://www.mottchildren.org/health-library/abj5949

3 Background and related work

Analyzing existing breathing games and sensors is the first step of getting a better insight into the subject. The next step is to get information from other sources and see what their findings are. These sources are literature reports, articles, books, youtube videos, and an interview with teachers of breathing exercises. By using all these sources to answer the research questions, the answers will be more reliable and have more content.

3.1 Analyzing Breathing Exercises

It is important to know as much as possible about breathing exercises. These exercises will be implemented in the game, thus it is vital to know how they are performed, what the advantages are of doing them, and what the most important actions are that need to be monitored. Furthermore, similarities between the exercises can be discovered. Similar exercises have the advantage that the design choices of one exercise, stimulating elements, controls, game elements, etc, apply to the others, which saves time. Having information about breathing exercises will help with implementing the breathing exercises in the game and combine it with the sensor and stimulating elements. The following sections will answer research questions 3 and 4.

3.1.1 Existing Breathing Exercises per Domain

3.1.1.1 Singer

Breath is one of the fundamental factors of having a good singing voice. It is essential to practice breathing exercises to be a better singer[2]. By improving breathing techniques, the quality of singing voice intonations will also increase[2]. In [9], the researcher White agrees that training breath is important to develop a singing voice. He also points out the importance that the breathing exercises are done correctly. On the other hand, misuse of breathing can have negative effects on singing, like on vocal range, dynamics, and singing long notes[10]. Fernandes[11] studied typical problems related to breathing exercises by interviewing children from age six to eight. The results are as follows.

- “Shallow breathing
- Breathing that is not full/deep enough
- Raising shoulders, wrong posture
- Absence of physical sensation of breathing
- Lack of breath support as the position of inhalation is not maintained while singing
- Absence of the sensation of fixed inhalation and of economic, calm exhalation
- Absence of the skill of a noiseless inhalation through the nose
- Inability to let the breath out gradually till the end of a singing phrase
- Lack of breath energy“[11]

There are 3 components of breath that affect the singer: “Breath support, the formation of the sensation of inhalation and exhalation, and breath energy”[11]. Breath support is about the control of breathing muscles. This contains maintaining the right body posture and having a

smooth breathing rhythm. The next component is to know and feel how to do the breathing exercises correctly. It is important to sense if every aspect of the breathing exercise is done correctly, i.e. breath from chest or abdomen. The last component, breath energy, refers to the loudness and maintaining a tone or note for a long period of time.

In figure 4, most singing exercises are shown, with instruction and what the benefit is by doing it.[2]

Exercise	**Instruction**	**Goal**
Instruction	Explain: a) which specific sound is to be produced; b) what is needed to do that; c) what they must carefully watch during the formation of voice	Understanding
Training inhalation	a) Visualize a bunch of flowers; b) calmly inhale through the nose, while thinking of pleasant aroma; c) a short hold; d) exhale	Understanding, noiseless breathing
Push-up	a) A push-up from a (school) desk: while doing this, pay attention to the character of inhalation and immediate holding of breath; b) repeat the inhalation in a standing position, fixing with hands the lower costal area and the front abdominal wall	Developing stable sensation of lower costal breathing
Training exhalation	“Let’s blow the candle”, i. e. try to keep the smoothness of exhalation and concentrate the flow of the exhaled air	control of air consumption
Constant consonant	a) an active inhalation, b) a hold, c) a long active exhalation on sound “s”/“f”/“v”, d) expelling of the residual air	Developing right breathing technique for singing
Visualize	a) children are asked to visualize themselves as big colorful air-balls; b) after the game, the ball is put on the grass to rest for a while and the valve is taken out: the air gradually goes out (children let the air out very slowly)	Understanding
Pant	Speeding up the breathing and making it shallower is sometimes, but not always, more effective.	Strengthen diaphragm

Figure 4: Table of breathing exercises for singers, how to perform them, and what the goal is, used from Fernandes’ report[5].

3.1.1.2 Pregnant women

Breathing exercises are also useful for pregnant women. They “can decrease the anxiety and also control the duration of labor, along with several other benefits for pregnant women and can be practiced at any stage without side effects[4]” [11]. Breathing exercises for pregnant women are mostly done to relax the user (see figure 5). Breathing is distracting and enhances mindfulness. If done correctly, breathing exercises can even replace drugs against pain when in labor [4][12]. By relaxing, not only pregnant women feel better, but also the child will be healthier[12].

Exercise	**Instruction**	**Goal**
Just Breath	Slow, deep breathing is particularly effective. The “right” way to breathe is whatever feels right. Keep breathing conscious, not automatic	Relaxation
Pant	Speeding up the breathing and making it shallower is sometimes, but not always, more effective.	During stronger labor contractions
Comfort	In addition to breathing, do comforting activities like moving, changing position, slow dance, massage	Comfort
Focus	Focusing on something, either with eyes closed or open, can help maintain the rhythm of the breathing	Focus, mindful, relaxation

Figure 5: Table of breathing exercises for pregnant women, how to perform them, and what the goal is, made by Fernandes[5].

3.1.2 Similarities of Exercises between Domains

Singing and being in labor sound like very different tasks at first, but when analyzing both figures 4 and 5, some similarities show. The first one is panting. Both singing and labor have exactly the same exercise, only the goal is different per domain. Another similarity is that both domains have to pay attention to their breath by breathing slowly. “Just Breath” is the most similar to “Training Inhalation”, but also is comparable to “Training exhalation”, “Constant consonant”, and “Visualize”. All these exercises focus on long breaths, to get more understanding or just to relax. While the goal is different, the exercises themselves can be done by people from both domains, when implemented correctly.

3.2 Combining Traditional and Breathing Inputs

After knowing more about breathing exercises, the next step is to translate them into game control actions. The first step is to determine the different kinds of inputs. When key aspects of the controls are determined, correlation can be found with the breathing exercises. By making the right combination of breathing exercises and inputs, the gameplay of breathing games will be more engaging and intuitive. The following sections will answer research question 5.

3.2.1 Configuration Of Controls

Implementing breathing inputs into a game with only traditional inputs is more tricky than it looks. There are many different aspects that need to be taken into account. There are 6 prominent ones. The first aspect is that the breathing inputs should not break the immersion of the game[13][15]. In the experiment of [15], they tested breathing inputs on different traditional inputs of the games. They found out that some inputs, like blowing up buildings, were more intuitive to use and increased immersion, while other inputs, like attacking, had the opposite effect. In the game made by [14], the breathing input was used to use and control a flamethrower. The participants enjoyed this feature, because “they most enjoyed using the sensors when they felt their physiological actions mapped naturally to the in-game reaction”[14]. So breathing inputs should be intuitive to use[16]. The disadvantage of intuitive inputs is the lack of flexibility[14].

What also is an important aspect, is to analyze general traditional inputs to see which breathing inputs can replace them. Following research [5], inputs can be divided into two categories:

- “Single Action: These are events that trigger an in-game action with a single gamer action, such as a button press or a screen swipe. For example, shooting bullets in ‘CS:GO’, jumping in ‘Mario’, etc
- Continuous Action: These are events that require continuous action from the player, requiring sustained action game-controllers. For example, player movement in various games, motion sensor, mouse pointer action, etc. “[5].

Game actions can also be a combination of the two. Additionally, the game actions can also require directional control, like walking, or amplitude control, like charging a shot in FIFA (see figure 6). By mapping traditional inputs with GACE (Game Action Control Event), the process of replacing them will be easier and more reliable. Nevertheless, Fernandes [5] remarks that GACE alone is not sufficient alone. Fernandes made a list of aspects he concluded are important for using breathing inputs. He introduces the third aspect, “these games should not be too complicated, involving simultaneous control of several elements of the game”. A person can only do a single breathing exercise at the same time. What this means is if certain breathing inputs, like walking and aiming, need to be controlled simultaneously, the player is unable to play the game correctly. The fourth aspect warns that making a game too complicated will harm the focus of the participant. If the game is too complicated, the user can not pay full attention to the execution of the exercises. The fifth important aspect is the progress of the player. A key aspect of breathing exercises is that the difficulty of the exercise should correspond with the skill of the player[15]. This means that the breathing inputs should be scalable. The last useful aspect of Fernandes is that the players need time to breathe normally. Players find it hard to

continually do breathing exercises[13], so having breaks between exercises is necessary to relax the player and prevent shortness of breath.

Lastly, not all traditional inputs can be replaced by breathing inputs[14]. This can be if the input conflicts with one of the guidelines of Fernandes. For example being too complicated to be replaced, like aiming in a shooter, or needed to be continuously used throughout the whole game, like running in Mario, which does not give the player a chance to take a break or it conflicts with other breathing inputs.

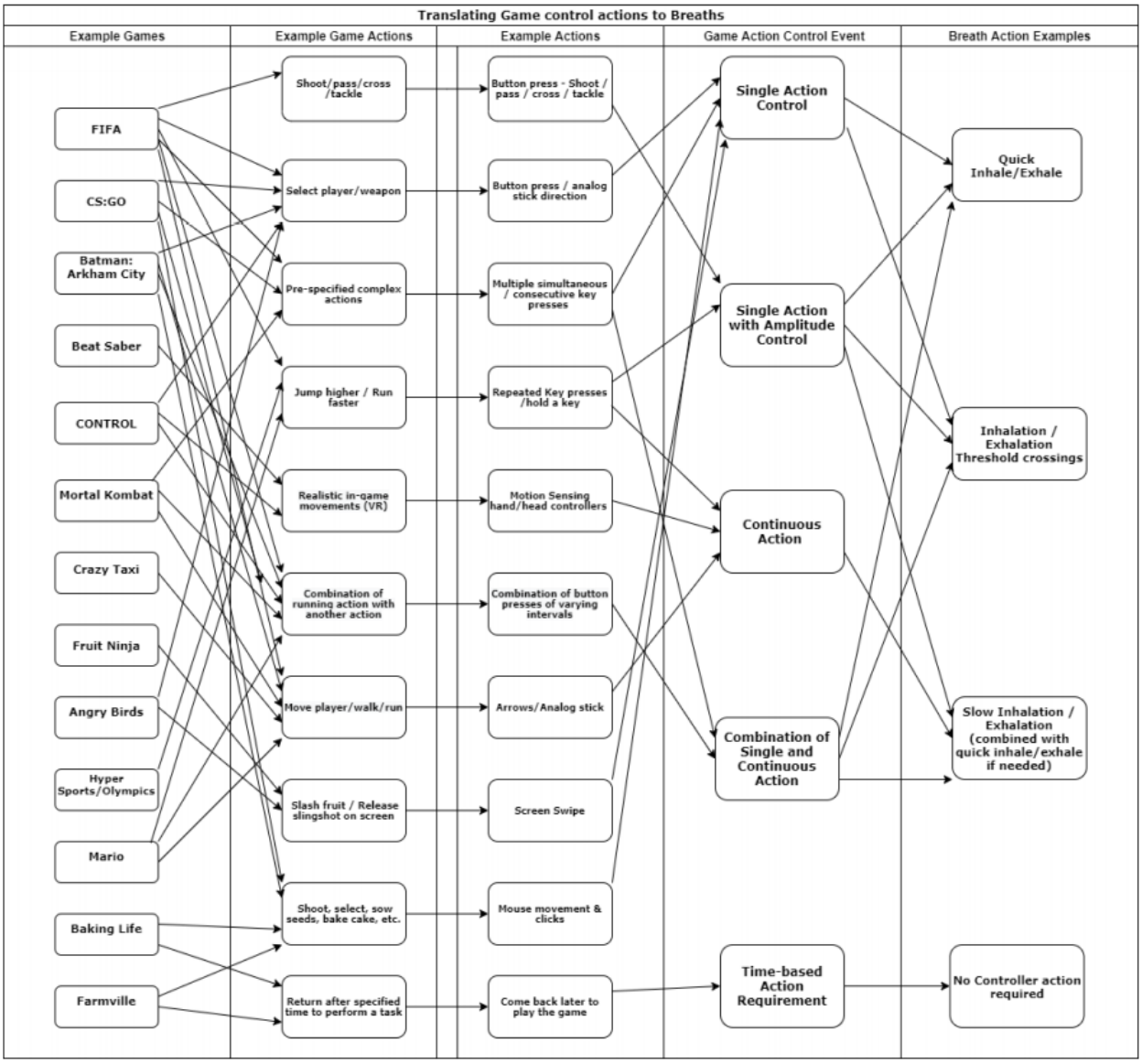


Figure 6: Translating game controls to breaths, made by Fernandes[5].

3.2.2 Combination Of Controls

The next phase is to see what is important to pay attention to when combining affective with traditional inputs. The first step is to see what exactly are affective inputs and how do they influence the player. After that, the last step is to see what is important when trying to combine the two. The information needed is found in papers of other researchers. The following sections will answer research question 6.

3.2.2.1 Affective inputs

Next to understanding specific traditional and breathing inputs, it is also important to know the use of affective inputs influence the user. Affective inputs are inputs done by a player's own cognitive and emotional state. For example, breath, heartbeat, and eye movement. While not that many games use affective controls, users do enjoy games with affective inputs[15]. They feel more involved[14]. While Kuikkanieme [17] only focussed on an FPS game, he had the same conclusion. Although the research question is focused solely on breathing inputs, it is still profitable to focus on other affective inputs. There are aspects of one affective input that can be applied to another.

Affective inputs can be divided into two sections: direct and indirect. Indirect affective inputs are inputs the player can barely control, like heartbeat and brain waves, while direct affective inputs are mostly controllable by the player, like breath and eye movement. Indirect affective inputs sound as useful as direct inputs, but in reality, it is not the case. While these inputs can be used passively, like changing the atmosphere in a game slowly, the use is too slow and inaccurate to implement it in a game[15]. Nevertheless, Nacke also concluded that there is the potential use for indirect inputs for the use of relaxing the player, provided the inputs were used for passive changes, like the weather, in a game. On the other hand, direct affective inputs are proved to be fun for the player. Participants of the experiment of [15] said they liked the input, because of the visible responsiveness and if the controls were done intuitively, it will increase the engagement of the players. Participants of the experiment [16] had a similar response. They also liked direct affective controls.

3.2.2.2 Combination traditional and affective inputs

Traditional controls are not easy to replace. They are great in certain game-related tasks, like pointing with the mouse. Furthermore, traditional inputs can be good in terms of preference, user experience, and performance[14]. That is why replacing traditional inputs with affective ones is not always doable[14]. Nevertheless, combining the inputs has many advantages, thus is recommendable to correctly implement in a game. For one, when affective controls are implemented, it increases the engagement and fun of the users significantly[14][15][16]. Zafar[18] did an experiment to test the effects of affective controls by designing two games, one with and one without affective inputs, to compare the user experience. The first thing, that aligns with [14][15][16], is that users experienced more fun when playing with affective controls. This can be caused by a couple of factors. One is that most users are not used to playing games with affective controls. It is new for them. Also, because playing affective games is new, it

becomes more interesting[19]. Additionally, players also have to get used to the new control scheme. It is an extra challenge, but it is one that provides enjoyment for the player[14]. The second factor is that the interaction is more natural between humans and machines[15]. The last big advantage is that by using affective inputs, a serious game is more effective in teaching its users. Nacke [14] “feels that augmenting traditional game controllers with physiological input will allow for a gentle learning curve as players become used to physiological control”. This statement corresponds with [18]. Thus games with affective controls can potentially be more effective in teaching users than games without.

3.3 Stimulate the Physical Execution

After getting more insight into the breathing exercises and game inputs, it is possible to tackle the key topic of this report. The main part of a breathing game is to make sure the players are able to learn how to do the exercises correctly. The player needs to have some kind of feedback that helps him to achieve the right execution of the breathing exercises. This can be done in multiple ways, but all have advantages and disadvantages. Before designing the breathing game and stimulating elements, it is important to evaluate existing stimulating elements and their effects. There are three main platforms of teaching breathing exercises: Games, Youtube, and physical teaching sessions done by a teacher. Games can influence many different senses of the players, so it is important to investigate how the senses are influenced and in what way. For youtube and teachers, it is meaningful how the teachers transfer their knowledge and what effects it has. Even when they use mostly explicit stimulating elements, they still can have some hidden, implicit tricks to help their students or viewers. The following sections will answer research questions 7 and 8.

3.3.1 Stimulate Through Game Elements

One of the biggest strengths of serious games is that they are engaging. Players are motivated to keep playing them. The problem is to check whether serious games help users to perform better in certain tasks. There are ways serious games solve this problem by using elements that are used to influence the execution of the exercise; stimulating elements. There are many different stimulating elements that can be used to motivate the right execution. The first element has to do with engagement. Engagement motivates the player to do the actions more seriously. Stimulating elements that are easily implemented and effective are a goal and a reward when the goal is reached [20]. To check whether the goal is achieved, games mostly use sensors to check. Nevertheless, stimulating elements can also be used to bypass the use of a sensor. Faust-Christmann[21] showed in her experiment that when the stimulating elements are used correctly, sensors are not necessarily needed. The stimulating elements worked assumingly well enough that the execution of the users was mostly sufficient. When users execute the breathing exercises most of the time correctly, they do not need feedback, thus do not need a sensor who controls the feedback. Nevertheless, using sensors makes sure the players make fewer mistakes with the exercises and know and learn from these mistakes.

3.3.1.1 Audio elements

The second kind of stimulating element has to do with audio. The useful role of audio is already well-known. It increases engagement and emotional investment[22]. The audio stimulating elements can also have an influence on the physical execution of exercises. Audio can be either used explicitly or implicitly. Good examples of how to use implicit stimulating audio elements can be found in [23]. Tajadura-Jiménez[23] saw with her experiment that by just changing the frequency of the audio the participants would feel lighter/heavier, and would walk differently because of that. Audio in serious breathing games also helps by giving feedback on the actions of the player. This is done by either helping to stay in rhythm, time how long a user needs to

inhale or exhale and notify the players when they do something wrong or correct. Lastly, audio also has an influence on the mental state of the player. It can relax or energize users, depending on the tempo of the audio[24]. To conclude, audio can be very helpful to be used as an implicit stimulating element.

3.3.1.2 Visual elements

The last part of stimulating elements is about visual components. Visual components also help with improving engagement, but also with timing. Same as the audio stimulating elements, visuals can show the amount of time for exhalations/ inhalations. In addition, it can also influence how hard the user has to breathe by showing for example a mouse or an elephant breathing[8]. Visual stimulating elements can also stimulate if the user needs to breathe from the abdomen, and breathe in a certain rhythm. The visual components are mostly implicit and hidden in engaging elements, like what is done in the games of Fernandes[5] and Patibanda[7]. Fernandes used blocks that needed to be cut down on the right timing by inhaling and exhaling, to enforce breathing in a certain rhythm. Patibanda used multiple elements to enforce the right execution. The first one was the most direct, a game where you must survive by dodging obstacles. The second is more ambitious, here you control the world by breathing. Clouds will move with breathing, and trees will spawn when done in a certain rhythm. The last game uses the most implicit elements. Here colors and particles will give feedback about the breathing of the player. There are also explicit stimulating elements that can be used. These are mostly examples and clear and detailed explanations. While they are useful, they diminish the engagement of players.

3.3.1.3 Implementation stimulating elements

Knowing the stimulating elements is not all that is needed to implement them. There are some aspects that need to be accounted for. Mainly, stimulating elements need to have a reason added and make sure there are not too many of them[26]. Adding seductive elements can boost performance when the user becomes more motivated because of them and pay more attention[25]. On the contrary, some studies suggest that motivating elements should be used sparsely, or even not at all. They say that seductive elements can be distracting, and are an inefficient use of cognitive ability[21][27][28]. In addition, The studies advise that a serious game should mainly focus on reaching the instructional goal, and unnecessary information should be avoided to lessen the cognitive load. Nevertheless, the key is to have the right balance between seductive and instructional game elements[26]. While adding too many seductive elements can be indeed distracting, having the right amount can be positive. The important part is that “the core game mechanics should be well integrated with learning content, learning mechanics, and instructional aspects”[29]. Only when the base game is worked out, seductive elements can be added.

3.3.2 Youtube Analysis

Breathing exercises can be taught in multiple ways. One of which is the use of online media. To be precise, youtube videos. YouTube is the biggest online sharing platform in the world. It contains videos about almost every subject. This includes breathing exercises. When analyzing these videos (see figure 7), they all seem similar. Firstly, they all explain the advantages and disadvantages of the exercises. There is a difference in the amount and details of the explanation, but it mostly comes down to the same things. This is probably done to motivate the viewer to show them a goal where they can work towards. Secondly, what also is similar is that never only one exercise is explained. The videos always explain multiple breathing exercises, in addition to explaining when to use them and why. Thirdly and lastly, in their commentaries, they mostly explain the exercise as detailed as possible, while using metaphors and examples to make it even more clear. This is probably done because the user cannot ask questions, so it needs to be as clear as possible. To also ensure that the breathing exercises are done correctly, some videos ask to put hands on the abdomen and chest to see if the breathing comes from the right place. Furthermore, some videos also use encouraging messages to probably keep the viewer happy and engaged. Lastly, what is confusing is that some videos emphasize the importance of body posture. Some videos say it is important while others do not even mention it.

Name Video	7 Best Breathing Exercises for Singing	Top Breathing Exercises to Improve Your Singing Voice | #DrDan	How to Breathe from Your Diaphragm While Singing	A Breathing Exercise for Pregnant Women	BREATHING Techniques for an EASIER LABOR | How To Breathe During Labor | Birth Doula | Lamaze	Breathing Exercise For Pregnancy
Upload Date	13/10/2018	31/10/2019	24/10/2019	7/6/2016	5/3/2019	20/8/2014
Length	7:27	18:13	7:40	5:23	10:02	2:42
Topic	Breathing exercises for singing	Breathing exercises for singing	Breathing exercises for singing	Breathing exercises for pregnant women	Breathing exercises for pregnant women	Breathing exercises for pregnant women
Author	TakeLessons	Dr Dan's Voice Essentials	Katarina H.	Live Sonima	Bridget Teyler	Homeveda Parenting
Link	https://www.youtube.com/watch?v=GfuJAb62Uek&t=322s	https://www.youtube.com/watch?v=SEdCYhnStrk&t=977s	https://www.youtube.com/watch?v=2PnzaCn-Lis	https://www.youtube.com/watch?v=1eW-T6vtO7k	https://www.youtube.com/watch?v=eK9BrVX8RhM	https://www.youtube.com/watch?v=LCqxcKNImYw

Figure 7: Sources Youtube analysis

4 Game Idea Formulation

After gathering much information in the previous 2 chapters, it is time to use it. The next step is to make a design, has stimulating elements and breathing exercises correctly implemented. To do this, the first phase is to come up with ideas and choose the best one. When an idea is chosen, the next phase is to see which breathing exercises and stimulating elements can be implemented. The last phase is to compare multiple concepts of the final idea and see which can use them the best.

4.1 Game genres for breathing games

To start the ideation, the first step is to define the suitable game genres. There are many game genres (see figure 8), some work with breathing exercises and some will not. To not waste time by generating ideas that will not work from the start, it is useful to already have limits. Only the top 10 most popular game genres will be looked at. These seemed to be proven to be enjoyable to play, and because of their popularity learning how to play them should be easier. The first difficult part is defining popular game genres. There is not an official list with game genres. There are many confusions between lists, for example, what falls into an action game and what is a genre on its own. Fighting games and shooters mostly have this problem. To be concise, the work of Nicolas Besombes[9] is used to make and define the genre list. He used multiple sources to come up with his model, which makes it reliable. Each genre will be judged by three things. First is the difficulty of making the game. For this research, not more than a month should be needed to make the game. Second, breathing exercises should be easy to implement. Lastly, it should also be possible to use multiple breathing exercises in the same game genre. The list is as follows, but it is not in a particular order.

4.1.1 Shooting game

Shooting games are games where shooting a ranged weapon, a gun, or a bow, for example, is the main gameplay. Think about Call of Duty, Star Wars Battlefront 2, or Resident Evil. What the targets are or the goal will differentiate between games. The core gameplay, shooting, is easy to build. Furthermore, because this genre is built around a single gameplay element, there is much freedom in designing the game. Putting breathing exercises in there should not be difficult.

4.1.2 Strategy game

Strategy games are about thinking strategically, logically and tactical to beat obstacles and achieve victory. Many games in this genre emphasize thinking ahead and manage multiple elements, like resources, troops, and workers. Examples are Civilisation and Age of Empires. Nevertheless, there are also strategy games that are a bit simpler, like checkers and Teamfight Tactics. Sadly strategy games are not well adaptable for breathing games. For breath exercises, timing and rhythm are necessary, which cannot be found in strategy games. Furthermore,

strategy games stay complicated compared to other games and thus can distract the player. Lastly, designing and making a good strategy game is also difficult.

4.1.3 MOBA

MOBA stands for Multiplayer Online Battle Arena. In this game genre, players each have their own character, which they use to defeat the enemy characters. Popular examples are League of Legends, Dota 2, and Smite. MOBAs are hard to make. They require a good net code and balanced characters. Especially a good net code is hard to make. Nevertheless, breathing

9:https://medium.com/@nicolas.besombes/esports-competitive-games-by-genre-61fcaf9c6a8f

exercises can easily be implemented. The playable characters have abilities, and they need to be used at the right time to be efficiently used. This is not enough to solve the net code issue. The only solution can be to make the game either local multiplayer, so from 1 computer, or make it a simple player, but then the game is a SOBA instead of a MOBA.

4.1.4 Card game

Card games, as the name implies, are games where the main element is playing cards. Examples are Magic the Gathering, Hearthstone, and Yu-Gi-Oh. The goal is most of the time to defeat an opponent, a real person, or a computer, by using the cards in an efficient and better way than your opponent. It has characteristics of a strategy game because to play a card game requires the player the same way of thinking. Building a card game is not difficult, the obstacle is implementing breathing exercises. While it is doable when being creative and designing a whole new card game, in an average card game it is hard to implement breathing exercises.

4.1.5 Sport game

Sport games are mostly game simulations or adaptations of real-life sports. Think about FIFA, MADDEN, and NBA. The goal is mostly to score goals or points, like in the real-life sport it is adapted from. Building a sport game can be hard, depending on the sport. Bowling for example is much easier to make than a game with many rules like soccer. Breathing exercises can be implemented, depending on the sport, but implementing different ones can be a struggle. Most sport games are built around 1 game element, like for darting only throwing a dart. If not, the game becomes far more complicated with not simple game action. For example basketball. It is hard to translate running, shooting, or passing to breathing exercises.

4.1.6 Fighting game

Fighting games are centered around two players, who both control a character, fight each other and try to defeat the other person by landing more hits. Examples are Street Fighter, Super Smash Bros, and Tekken. Fighting games are mostly based on reaction, timing, and tactical thinking. Most of them are complicated to play, but also to build. There are many balancing and gameplay elements that need to be almost perfect to make the gameplay enjoyable. Furthermore, the fast pace of fighting games, and the complexity, make it hard to focus on breathing exercises.

4.1.7 Puzzle game

Puzzle games are similar to strategy games but simpler. It has mostly simple rules and gameplay elements. The player still has to think logically, but the goal is much clearer. Candy Crush and Tetris are examples of puzzle games. While a puzzle game is easy to build, because of the few rules and the easy gameplay elements, incorporating breathing exercises is hard. Puzzle games are slow-paced and are not based on mental skills. Adding physical skills to it will be hard to justify and could be destructive for enjoyment and engagement.

4.1.8 Racing game

In racing games, multiple players, real and computer, try to be the first one to get over a finish as fast as possible. This is mostly done by cars but can be done in any number of ways, like spaceships and running. Some examples are Mario Kart, Forza, and Trails. Because there are only a few gameplay elements, programming a racing game is not that time-consuming. Furthermore, this makes it easier to implement breathing exercises. Having only a few gameplay elements gives more freedom to change the design. Furthermore, in normal driving breathing exercises can be incorporated.

4.1.9 Rhythm

The goal of rhythm games is to do multiple sets of actions at precise timing. These can be traditional controls, but there are also multiple examples of affective. Examples of both are Just Dance, Beatmania, and Guitar Freaks. The pace can be adjustable, which is useful when implementing scalable breathing exercises. There is also much room for creativity because the core gameplay is simple and can be a wide variety of things. Furthermore, other affective inputs already seem to work with this genre. Lastly, because the gameplay is simple, it is not hard to make.

4.1.10 Role-play game

A role-play game is based around a player controlling a character in a fictional setting. There are many different games in this genre, but it comes down to being the main character in a storyline the player needs to complete. This can be done alone or with others. Examples are The Witcher 3, Minecraft, and World of Warcraft. Role-play games have mostly much content to increase engagement. This is hard to make because there is much that needs to be built in a Role-Play game. Breathing exercises can be easily implemented, because of the many gameplay elements this genre can contain, but only when the game is not too complicated.

4.1.11 Conclusion game genres

Shooting, racing, and rhythm games are the easiest to make breathing games out of. There is much freedom and room for creativity to design and implement breathing exercises. This is because these game genres are based around one gameplay element, which can be put in easily in different scenarios.

For sport, MOBA, and card games it is possible to add breathing exercises, but it will be harder. Either because the game is hard to make, or that the gameplay element does not lend itself to breathing inputs. By using a creative approach and keeping in mind the flaws, it should be possible to create breathing games with these genres.

The remaining genres, strategy, fighting, puzzle, and role-play, are not applicable to breathing games, or incorporating breathing exercises would be at least very difficult. The core gameplay of these genres is either too focused on thinking, too complicated, too difficult to make, or a combination of the previously mentioned problems.

So the chosen six genres are shooting, racing, rhythm, sport, MOBA, and card games.

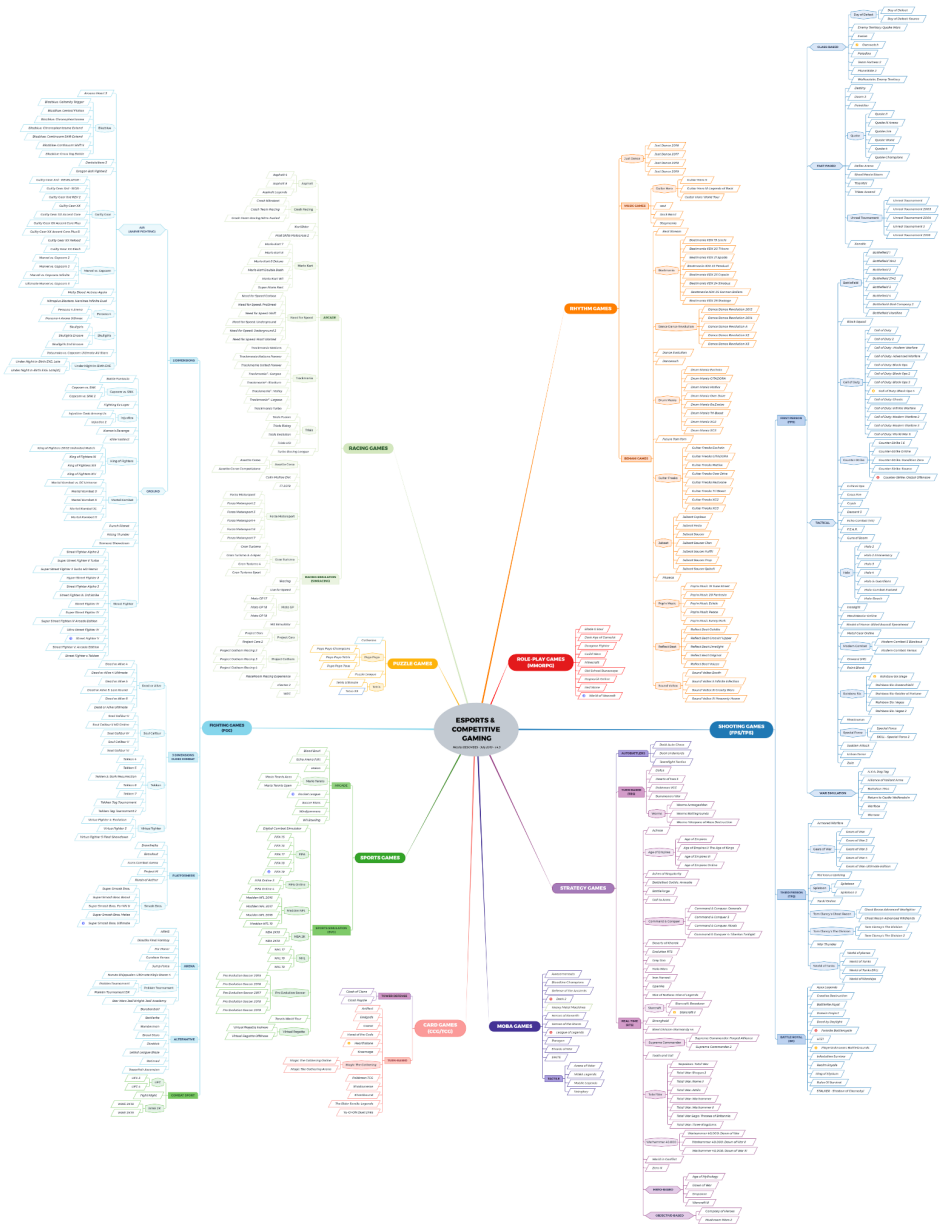


Figure 8: Game genres and subgenres, made by Angelika Mader and Wouter Eggink[33]

4.2 Brainstorming idea

After knowing what genres are compatible with breathing exercises, the next phase is brainstorming to come up with the final concept. This will be done by firstly brainstorming about new ideas, on the basis of the chosen six genres. After that, by comparing all the ideas the best one will be chosen and will be the final concept.

4.2.1 Outcome Brainstorm

The first step is to come up with at least 20 ideas that are from the six chosen genres. To come up with ideas, the game genres will be the inspiration source. For every genre as many ideas that can be thought off will be written down. There is no limit, but also no minimum. This is because it is not useful and efficient to think too hard about an idea[31]. To enhance inspiration further, multiple genres can be combined to see if that combination has potential. Lastly, the brainstorm session will be done over multiple days, to not try to force ideas out.

Genre	**Description**
MOBA 1	A MOBA, but then for single player, with a simpler map, and use breathing exercises to do abilities.
MOBA 2	Go from point A to B, but solve the obstacles and defeat the enemies along the way. Do abilities with breathing exercises.
MOBA 3	Influence the environment with breathing, like slow-mo with exhalation, panting will create an earthquake, etc.
MOBA 4	Normal character, but he levels up by doing breathing exercises
MOBA + card game	Use cards to use abilities, but need to do breathing exercises to let the cards work/ improve the effect of the cards
Card game 1	Normal card game, but improve the card by doing breathing exercises. The better the player executes them, the better the improvements of the card.
Card game 2	Do general effects on the game with breathing exercise, like stealing cards from opponents.
Sport 1	Athletics competition, every different activity is a different breathing exercises (running, spear throwing, etc)
Sport 2	Penalties in football, but things like aiming and power is done by breathing
Sport 3	Pulling rope by doing breathing exercises
Shooting game 1	Shooting game, but basic actions can only be performed by breathing exercises, like reloading, using a grenade, heal.
Shooting game 2	Shooting game, but the player has abilities that can only be activated by breathing exercises
Shooting game 3	Influence the environment with breathing, like slow-mo with exhalation, panting will create an earthquake, etc.
Race game 1	Driving a vehicle for multi terrains. Per terrain, a different breathing exercise is needed to boost the speed.
Race game 2	Racing game but with abilities, activated by breathing exercises. Like Mario kart

Race game 3	Influence the environment with breathing, like slow-mo with exhalation, panting will create an earthquake, etc.
Race game 4	Dragons racing, need to breathe fire against other dragons races and burn villages
Rhythm game 1	Guitar hero kinda game, but then with breath.
Rhythm game 2	Geometry Dash, but by inhalations and exhalations control
Rhythm game 3	Flappy bird, but by inhalations and exhalations control
Shooter	Shoot fire attacks at snowman to survive
Other	Shoot the spaceship as far as possible in space by charging the thrusters and dodging obstacles.
Other	Help helicopter land by manipulating the weather, like pushing it left and right with inhalations and exhalations
Race	Race with sailboats, use breath to control boat
Shooter	You are a volcano, shoot lava, gas, and stone to kill certain enemies
Race	Simulation of car driving, need to do breathing exercises to get rid of rain, mist, etc.
Sport	Bowling, but control the ball with breathing exercises
Sport	Golf, use breathing exercises to determine the swing
Sport	Golf, but get rid of obstacles and control external influences like wind by doing breathing exercises
Rhythm	Move on the beat of the music, attack enemies with breathing exercises. Defeat all obstacles and go to the exit to win
Other	Control air balloon, try to dodge birds and planes with breathing exercises
Other	Fruit ninja but with breathing exercises, slice fruit to get points

Figure 9: Game ideas per game genre

4.2.2 Comparing Ideas

The next step is to find objectively the best ideas from figure 9. This will be done by comparing every idea on 4 different key elements (see figure 10). These aspects are determined by production restrictions and the implementation and effect of breathing exercises. Each idea will get a grade between -3 and +2. The more positive the score, the better. Enjoyment is not factored in, because it is hard to determine if a game is fun without playtesting it. The ideas with the highest overall grade will be compared to each other to choose the best one.

4.2.2.1 Time needed to make game

There is only a limited time to make the game. While some normal games can take up over 5 years to make, the time limit for this game is a month. To ensure the deadline is reached, the scope should not be too big or too complicated. Game ideas score high when there are not many gameplay elements and are not complicated to make. For example, a MOBA takes much

time to make, because the character movement and abilities and the enemy movement and abilities need to be programmed and balanced. On the opposite side, rhythm games are mostly built around a few, simple gameplay elements, like for Flappy Bird it is just jumping and dodging pipes. A MOBA will score generally low, while games like Flappy Bird score high.

4.2.2.2 Amount of content

An important aspect of the game concluded in chapter 3 is that a game should have enough content. Content means in this context the number of hours of enjoyable gameplay. At a certain moment, almost every game is out of combat, and when this happens, some people will stop playing the game. Also, more content means it is easier to make a gradually increasing difficulty that corresponds with the skill level of the player. Scoring high is not only determined by the number of different levels and the size, but also the replayability. Games like MOBAs do not have that much content, only many different playable characters. The rest of the game is the same, but still, people enjoy it for thousands of hours.

4.2.2.3 Complexity of gameplay

Another conclusion in chapter 3 is that the game should not be too complicated. This can distract users from doing their breathing exercises correctly. It also helps players to learn the game more quickly. The gameplay thus should be simple. There should not be too much going on at the same time, and the gameplay elements should be easy to do and straightforward. This can contradict with the amount of content. More content can mean more things the player needs to learn and/or remember. Games like MOBAs score low, because to play them correctly a player needs to learn their own character, the enemies, and most of the time all the items.

4.2.2.4 Usability for multiple exercises

The last element to compare is usability for one and multiple exercises. This is graded upon multiple aspects. First, how well single breathing exercises can be implemented in the game. Breathing exercises can be implemented correctly when the game action is not too complicated and can not overlap with other breathing inputs. Additionally, it is important to look at how intuitive the breathing exercises are in the game. Lastly, games will score high when using multiple, different breathing exercises is easy to implement. This has to do with the amount of freedom. It is harder to implement breathing exercises for a game with only a few specific gameplay elements, like the Pulling rope game example.

Game Idea	**Game idea explanation**	**Time needed to make game**	**Amount of content**	**Complexity of gameplay**	**Usability for one and multiple exercises**	**Total**
MOBA 1	A MOBA, but then for single player, with a simpler map, and use breathing exercises	-2	+2	-1	+2	+1

	to do abilities.					
MOBA 2 + adventure	Go from point A to B, but solve the obstacles and defeat the enemies along the way. Do abilities with breathing exercises.	-2	+1	-1	+2	0
MOBA 3	Beat levels by beating monsters. Influence the environment with breathing, like slow-mo with exhalation, panting will create an earthquake, etc.	-2	+2	-1	+2	+1
MOBA 4	Beat levels by beating monsters. Normal character, but he levels up by doing breathing exercises	-2	+2	-1	+2	+1
MOBA + card game	Use cards to use abilities, but need to do breathing exercises to let the cards work/ improve the effect of the cards	-3	+2	-2	+2	-1
Card game 1	Card game as Hearthstone, but improve the card by doing breathing exercises. The better the player executes them, the better the improvements of the card.	-2	+1	+1	+2	+2
Card game 2	Game like Uno, Do general effects on the game with breathing exercise, like stealing cards from opponents.	-2	+2	+2	+1	+3
Sport 1	Athletics competition, every different activity is a different breathing exercises (running, spear throwing, etc)	-2	+2	+1	+1	+3
Sport 2	Penalties in football, but things like aiming and power is done by breathing	0	-1	+2	0	+1
Sport 3	Pulling rope by doing breathing exercises	+1	-1	+2	-1	+1
Shooting game 1	Shooting enemies to reach the goal, but basic actions can only be performed by breathing exercises, like reloading, using a grenade, heal.	-1	0	0	0	-1
Shooting game 2	Shooting enemies to reach the goal, but the player has abilities that can only be activated by breathing exercises	-2	+1	-1	+1	+1

Shooting game 3	Shooting enemies to reach the goal, Influence the environment with breathing, like slow mo with exhalation, panting will create an earthquake, ect.	-1	+1	-1	+1	0
Race game 1	Driving a vehicle for multi terrains. Per terrain, a different breathing exercise is needed to boost the speed.	-1	+2	-1	0	0
Race game 2	Racing game with cars, but with abilities, activated by breathing exercises. Like mario kart	-2	+1	0	+1	0
Race game 3	Normal race game with cars, Influence the environment with breathing, like slow mo with exhalation, panting will create an earthquake, ect.	-1	0	0	+1	0
Race game 4	Dragons racing, need to breathe fire against other dragons races and burn villages	-2	+1	-1	+2	0
Rhythm game 1	Guitar hero kinda game, but then with breath.	0	+1	+2	0	+3
Rhythm game 2	Geometry Dash, but by inhalations and exhalations control	0	+1	+2	+1	+4
Rhythm game 3	Flappy bird, but by inhalations and exhalations control	0	0	+2	-1	1
Shooter	Shoot fire attacks at snowman to survive	-1	-1	+1	+1	0
Other	Shoot the spaceship as far as possible in space by charging the thrusters and dodging obstacles.	0	0	0	-1	-1
Other	Help helicopter land by manipulating weather, like pushing it left and right with inhalations and exhalations	+1	-1	+1	0	+1
Race	Race with sailboats, use breath to control boat	+1	0	0	0	+1
Shooter	You are a volcano, shoot lava, gas, and stone to kill certain enemies	0	0	0	+1	+1
Race	Simulation of car driving, need to do breathing exercises to get rid of rain, mist, etc.	-1	0	0	+1	0

Sport	Bowling, but control the ball with breathing exercises	+1	0	+1	-1	+1
Sport	Golf, use breathing exercises to determine the swing	0	0	0	-1	-1
Sport	Golf, but get rid of obstacles and control external influences like wind by doing breathing exercises	-1	+1	-1	0	-1
Rhythm	Move on the beat of the music, attack enemies with breathing exercises. Defeat all obstacles and go to the exit to win	-1	+1	-1	+2	+1
Other	Control air balloon, try to dodge birds and planes with breathing exercises	0	0	+1	-1	0
Other	Fruit ninja but with breathing exercises, slice fruit to get points	0	+1	0	0	+1

Figure 10: Grading table to determine the best game idea

4.2.3 Final Idea

After comparing all the ideas with others in figure 10, the top three ideas were card game 2, and both rhythm games 1 and 2 (see figure 11). While the card game scored high in almost all categories, its main difficulty lies with making the game. The gameplay elements are simple, but the problem lies with Uno being a game that is played with others. In these times, physical contact is not possible because of Covid. This means the game needs to have online functionality, which is difficult and time-consuming to make correctly.

Rhythm game 1 is similar to rhythm game 2, but the last one has a slight advantage over the first one: freedom. It is important for *Usability for one and multiple exercises* that there is enough freedom and room in the game to add multiple, intuitive breathing exercises. The more freedom, the higher the chance the breathing exercises can be implemented correctly. The problem with the Guitar hero game is that there are only a few, simple gameplay elements. Not much can be added to it without changing the whole game concept. Rhythm game 2 has more room to implement different gameplay elements, which makes that game the idea for the final concept.

Game Idea	**Description**	**Score**
Card game 2	Game like Uno, Do general effects on the game with breathing exercise, like stealing cards from opponents.	+3
Rhythm game 1	Guitar hero kinda game, but then with breath.	+3
Rhythm game 2	Geometry Dash, but by inhalations and exhalations control	+4

Figure 11: The three best game ideas from figure 10

4.3 Concepts

While the final idea is determined, there is still much freedom on the specifics of the game. The next step is to determine which breathing exercises can be used in the game and how they can be implemented. The same goes for the stimulating elements. Lastly, the specific gameplay has to be determined by comparing concepts to each other to see which one synergizes the best with the breathing exercises and stimulating elements.

4.3.1 Further Explanation Final Idea

4.3.1.1 Description

The final concept is the dash game. The game is inspired by Geometry Dash, and thus has similar gameplay. For one, levels, just like Geometry Dash, will move automatically from left to right when the game starts. The goal of the game is to dodge all the obstacles and reach the finish line. The obstacles contain walls, flying triangles and spikes. Hitting a wall at the side will kill the player, while jumping on it does nothing. On the contrary for spikes and other obstacles, hitting it with the character is always killing the character immediately. When killed, the player has to start either at the start of the game or at a checkpoint.

4.3.1.2 Mapping breathing exercises

The character's main way to dodge obstacles is jumping, but it needs to have more abilities to accommodate breathing exercises. These abilities should be a mix of continuous and simple actions. The reason for this is that breathing exercises can mostly also be divided into two categories. The breathing exercises “Training exhalation” and “Just Breath” are examples of continuous breathing exercises. For them, the user needs to do long breaths to complete. For single breathing exercises, this is different. Here the user only needs to do a single or multiple short breaths to complete the exercises. An example is “Panting”.

To accommodate the breathing exercises, 7 potential abilities can be implemented in the game: duck, boost, mirror gravity, shoot laser, shoot rockets, levitate, and slow-down-time (see figure 12). These abilities do not have to be all implemented. There is also the jump action that definitely will be in the game. While this is a single action and can be used for breathing exercises like panting, it is better for the game to control it via traditional inputs. The reason for this is that jumping is going to be used all the time in the game. This could cause a few problems when breathing controls this input. First, players have to continuously do breathing exercises over the whole game, which makes them out of breath. The only way to give players a breathing break is to remove all the obstacles for a short time periodically, but that could cause the player to lose engagement. The second problem is that the input can overlap with most of the other abilities. For example, jumping and shooting lasers or rockets. By using traditional

inputs to control jumping, breathing breaks can be better managed and the chance of overlap between breathing controls is smaller.

Ability	**Description**	**Game Action**
Duck	Ducking makes the character length smaller for an amount of time.	Single or Continuous
Boosting	Boosting increases the speed of the level moving from left to right. By boosting, the character can make bigger jumps.	Continuous
Mirror Gravity	By changing gravity, the character will mirror his own gravity, and thus will land on the roof of the level if he started on the ground, vise versa if he starts on the roof.	Single
Shoot Laser	Shoot a laser for a certain amount of time that damage certain obstacles	Continuous
Shoot rocket	Shoot rocket to damage obstacles	Single
Levitate	Character hangs in the air in the middle of the levels for a certain amount of time	Continuous
Slow-down-time	Slow-down-time meant that the speed of the level going from left to right will diminish for a certain amount of time	Continuous

Figure 12: Potential game abilities for the playable character and what kind of game action they are

Breathing exercise	Breath Action
Singing	
Instruction	Other
Training inhalation	Continuous
Push-up	Other
Training exhalation	Continuous
Constant consonant	Continuous
Visualize	Other/Continuous
Pant	Single

Pregnant women	
Just Breath	Continuous
Pant	Single
Comfort	Other
Focus	Other

Figure 13: Breathing exercises per domain and what kind of breath action they are

When looking at figure 13, for the target groups there is only one Single Breath Action, panting. This can be used either on the game actions duck, mirror gravity, or shoot rocket. On the other hand, there are many breathing exercises that are continuous. There are many different combinations that can be made with the game actions that are also continuous. What is important with implementing these breathing exercises is that there is room for breathing breaks and that exercises do not overlap. If that is taken into consideration, then most kinds of continuous breathing exercises can be implemented into the game, also from other domains.

The breath action “Other” is used when the user needs to do physical activities or needs external help when doing the exercise. These exercises will be harder to measure and can be distracting when playing the game. For this reason, breathing exercises with “Other” will not be implemented into the game.

4.3.1.2 Stimulating element

Controls help the functionality of playing the game, but it does not address how to reach the users’ actual goal. For singing, it is a mixture of breathing control and power, while for pregnant women it is to relax. There are many in-game ways to help to reach those goals without changing the gameplay and goal of the game. These stimulating elements help to improve the execution of the exercises and help users to reach their goals more easily. This can be either by changing gameplay elements, audio, and visuals.

For pregnant women, stress should be minimized. This means the game should be easy to play. A way to do that is to slow the game down compared to the singer's game. By making the game less difficult, it should help the player to relax more. Additionally, fewer obstacles can be used to also lower the difficulty. Lastly, the number of abilities could be decreased compared to the singers. This makes the game less complicated, and thus easier to play. Singers do breathing exercises to improve breathing skills. This means the game’s difficulty should scale with the level of the user's breathing skills. So the game can start as easy as for the pregnant women, the difficulty should go up over time, by adding obstacles, new breathing exercises, and faster pacing of levels.

Audio can also have an important role in either reaching the player's goal or helping with the execution of the exercise. For one, sound can help with timing. This can help with using single breathing actions at the right time, knowing how long a continuous breathing exercise should be continued or when it is time for the player to have a breathing break and breathe normally. The background music can also have an effect on the players. Music can help relax people, but can also let people focus and memorize better. So it can be fruitful to use the right background music based on the player.

Visuals can help users the same way as audio. Like audio cues, visual cues can help with the timing of the exercises the same way. The overall look of the game can also influence the players. Colors and themes used in the game can also have an influence on the player's performance[32]. [32] is suggesting that colors like white and blue help to relax, while colors like yellow and orange help to stimulate.

4.3.2 Concepts

After establishing potential game actions, it is now time to see which ones make it into the final game. This will be done by making three, distinctive concepts, each with other combinations of game actions and breathing exercises. When the best concept is chosen, the next step is to see how to apply stimulating elements to it.

4.3.2.1 Concepts without stimulating elements

To design the concept, some criteria must be taken into account. To make sure the player learns enough, at least 3 breathing exercises should be implemented in the game. At least one for both single and continuous exercises should be present in the game, to have varied breathing exercises. Furthermore, the breathing exercises should be implemented for different purposes. Having only breathing exercises that help understand breathing but not exercises that strengthen the breath will potentially not be that useful for singers.

While the requirement is that at least three breathing exercises should be implemented, there are only two exercises for pregnant women to be used, and one is literally just breathing to understand the feeling more. Breathing exercises for singers have also many breathing exercises to get more understanding, and both singers and pregnant women have one exercise that is completely the same, panting. For this reason, the concepts will be designed to help both domains. This is also done to shorten development time, so there is more time to optimize the game. To do that, the panting exercise is required to be in the concept. The other breathing exercises should have the purpose to understand, so it is useful for both domains. While singers have breathing exercises that focus on other points, for the sake of this research it is not important. The important part is to see the influence of stimulating elements on the implemented breathing exercises, but it is not important for this research to train singers in every way.

When using these last criteria, only two breathing exercises are applicable next to panting: Training Inhalations and Training Exhalations. Only these exercises help to understand

breathing more and do not need physical activities or external help to execute them. So for the concepts, the breathing exercises are already determined, but not with which game action they can be combined.

The core game is the same for every concept. The player is a cube that can jump with the keyboard. The cube has to dodge and jump over traps like spikes. Furthermore, hitting the side of the wall also kills the player, so the player needs to jump on the walls to progress and survive. The level and cube move automatically from left to right. The player wins when they dodge all obstacles of the level and reaches the finish.

4.3.2.1.1 Concept 1

This concept will be focused on making the core gameplay as simple as possible. To do this, only two extra game actions next to jumping will be implemented: Duck and Levitate. Duck is a game action that can be either single or continuously used. By manipulating the levels players can be forced to use one of them. To make the use straightforward, the exercises panting and training exhalation will be implemented. The reason for this is that both exercises use exhalation. Furthermore, exhalation can be intuitive to use for ducking, because by exhalation things get smaller. Levitation will be then combined with training inhalation, but the levitation game action will be not as simple as just falling slower. To make it more intuitive to use with training inhalation, the whole process of the exercise will have an influence on the levitation action. For the first part, breathing in, the cube (playable character) will get wings and start going higher than normally jumping. Next, the cube will stay at a certain height when the user needs to hold his breath. This will have a time limit, to not force the user to keep holding his breath. Lastly, when the user exhales, the cube will go down slowly until it reaches the ground or the user stops exhaling.

4.3.2.1.2 Concept 2

The next concept will be more similar to the game it was inspired from, Geometry Dash. This concept will have as single action mirror gravity, controlled by the breathing exercise panting. While it is questionable how intuitive to use the combination is, Geometry Dash proved that this game mechanic is fun to use. For the continuous game actions, boost and slow-down-time will be implemented. The reason for this is that “changing time” and “changing gravity” can be used in the same theme. To boost the breathing exercise training inhalation will be used. The boost action duration will be determined by the amount of air that is inhaled, measured by the respiratory sensor. The duration will be slightly lengthened when the breath is held between two and six seconds, to stimulate the player to not breathe out immediately, but also not to hold it in too long. When the player starts exhaling, the boost begins. The boost ends when the duration that is determined by the inhalation and holding is over. Slow-down-time is combined with training exhalation, and the duration is the same as the duration of the exhalation.

The boost and slow-down-time action can only be used when the player is in a certain area of the level. This is to ensure that the player is not exerting himself, and can take breathing breaks.

4.3.2.1.3 Concept 3

The last concept will be war-themed. The playable character, the cube, will have 2 abilities next to jumping: Shooting rockets and shooting a laser. In this game, enemies will spawn that need to be shot down to progress in the level. Shooting rockets is single action, so panting will be used for it. When panting, a rocket will come out, flying is a horizontal line destroying the first enemy it hits. The rocket also gets destroyed when it hits another obstacle or leaves the screen. The laser will be controlled with a combination of training inhalation and training exhalation. The beam first gets charged by the amount of inhalation, sensed by the respiration sensor, and by holding the breathing between 2 to 6 seconds. The more the beam is charged, the bigger the laser is. The laser is starting to shoot when the player starts exhilarating. The duration of the laser is the same as the duration of the exhalation. The laser can kill the same enemies as the rocket, but certain enemies can only be killed by the laser. Both laser and rockets are abilities with a cooldown timer, so players are forced to take breathing breaks.

4.3.2.2 Final Concept

While all 3 of them have much potential, concept 2 is the best one to make. For one, the game Geometry Dash uses similar abilities, and that game is played and enjoyed by many players. The game proves that the concept can work. Furthermore, concept 2 is not complex, but also has some interesting mechanics. Lastly, making the abilities is not difficult. It is mostly slowing down or fastening the speed of the cube. This leaves more time for the optimization of the game.

5 Specification

Before building the game, it is important to distinguish which elements are more important than others to implement. This is to make sure that the experiment can be finished in time, even with setbacks. The first step is to determine what the results of the experiment should make clear. The next step is the method to distinguish the elements used in this report is MoSCoW. This method is made for software engineering, but can be applied to gaming. MoSCoW makes a difference between Must Haves elements, Should Haves, Could Haves and Won't Haves. Must Haves are elements that the game must-have. Without it the game cannot be used and/or the experiment cannot start. Should Haves are elements that make the gameplay much better, and help to have better results for the experiment. The experiment can still continue without them, but there is much merit in adding them. Could Haves are elements that make the game and results a little better, but not significantly. They only add a little, so only add them when there is time to spare. The last one, Won't Haves, is elements that will not be added, even if it could make the game and experiment a little better. The time needed to make these elements outweigh the benefits gained. Won't Haves determine the boundaries of the game.

5.1 Practical research questions

The first step is to specify what the experiment needs to accomplish: data about stimulating elements. For the game, the most important part is that it can test the effects of the implemented stimulating elements. The game needs to be able to answer the following questions.

1. *Do the implemented stimulating elements improve the execution of the breathing exercises and why? If not, do the implemented stimulating elements affect the players in another way, and how?*
    - Do the implemented stimulating elements for "training inhalation" improve the execution of the breathing exercise?
    - Do the implemented stimulating elements for "training exhalation" improve the execution of the breathing exercise?
    - Do the implemented stimulating elements for "panting" improve the execution of the breathing exercise?

The hypothesis is that the stimulating elements will improve the execution of the breathing exercises. If they do, it is also important to know why. Knowing why means the effects can be more easily repeated with other stimulating elements. Furthermore, it is also important to look at breathing exercises whether the stimulating elements work. There is a possibility that certain stimulating elements work better for specific breathing exercises, which would be useful to know.

When the hypothesis is proven wrong, the next step is to find the reason. The stimulating elements can have an unpredicted effect on the player. It is important to know what the effect is, and what caused the effect. The reason for this is to get more understanding of how elements can affect the player. With more understanding, it is possible to make better predictions of the effects of the stimulating elements.

5.2 Must haves

The experiment is about to determine whether the stimulating elements work for the breathing exercises. So to make the experiment possible, the game needs at least one stimulating element and one breathing exercise that influences the game. To implement the breathing exercises in the game, a breathing sensor should be connected to the computer. This sensor should also be able to be used on many different people. To achieve that, there should be a calibration function, which maps the breathing pattern of users.

5.3 Should haves

While one breathing exercise is enough to do the experiment, the data gained is less than desirable. It is sufficient, but having more exercises will not only increase the amount of data, but also the reliability. For one, by using more different breathing exercises, it can be concluded whether the stimulating elements work on all breathing exercises. When using only one, it can only be assumed the stimulating elements work for those exercises. Up to a total of three, one that tests exhalation, one inhalation, and one for panting. For the stimulating elements, it would be fruitful to use one that uses visuals, and one that uses sound. By using elements that trigger different senses it can be determined whether the elements have different effects.

The game should also have fundamental game elements. The most important is a tutorial that teaches the user the game. Next to that, some decently looking graphics and a background sound, to engage the player more. Lastly, the game should have a pause and skip button. The first one is to take more breathing breaks, the latter to help the players progress when they are stuck at a certain point.

5.4 Could haves

If there is time left, more stimulating elements could be added. The more data, the better. Next to that, being able to change the difficulty could also help the players to be more engaged and help them to not overexert themself. The difficulty can mean either how hard the level itself is too bad or how hard the breathing exercises are. For example, in the game, there can be fewer obstacles, and for the breathing exercises, the time of inhaling and exhaling can be lower.

The game could also have 2 versions, one for each target group. This could improve the engagement and effectiveness of the game and its stimulating elements. Other elements that could improve the engagement would be dynamic graphics and the use of sound. The more users are engaged, assumingly the more they will pay more attention.

Lastly, what could help is to make the pause and skip function automated, so the chance of the player exerting his breathing is smaller and it makes sure the player is not stuck at a part of a level for too long.

5.5 Won’t haves

There are limits to the size of the game. This is for one not to put too much time in it, but also to not make the game too complicated. Furthermore, it makes sure the experiment will go more smoothly. For one. Not more than 4 stimulated elements can be in the game. By having too many it is hard to determine which stimulating elements cause what effect, and it can become distracting for the player. Next to that, the game should not have more than 3 breathing exercises implemented. This is also to make sure the game is not too complicated. To make sure the experiment will not take too long, the levels should not take too long. If a player beats a level perfectly, the maximum amount of time that should take is 2.5 minutes. There should be limited new data when the levels become longer than that. Lastly, except for the difficulty levels, there should not be more levels added. For example, each difficulty level has 3 other levels a player can choose from. This takes too much time, and will not add anything positive to the experiment and the data, except that the players could enjoy more content and more variety.

6 Realisation

After determining what the game should have set the boundaries, the next step is to make it by combining the ideation chapter and the specification. The building time is two weeks, and the goal is to implement as many functionalities as possible in that time. The more functionalities described in specifications are implemented, the better the results are. The code and art can be found in Appendix A.

6.1 Base Game

The name of the game is Breath Dash. It is made in UnityThe basis of the game is to dodge obstacles, either walls or spikes, to reach the finish. Most obstacles can be dodged by jumping with the spacebar. When the player hits an obstacle, they will be set back to the start point, or a checkpoint that they passed. This makes the core of the game simple, which means it is easier for the user to focus on the breathing exercises. The theme of the art and background color are mostly derived from the game “Geometry Dash”. It is assumed that the developers of that game had a reason for why they used their style, and it is proven that it works. The game is split into zones for the breathing exercises, each color-coded so it is easier to remember when to do which exercise.

Blue, normal zones: Here the player does not have to do breathing exercises and just uses jump to dodge obstacles (see figure 14). These areas are implemented to make sure the player can take breathing breaks.

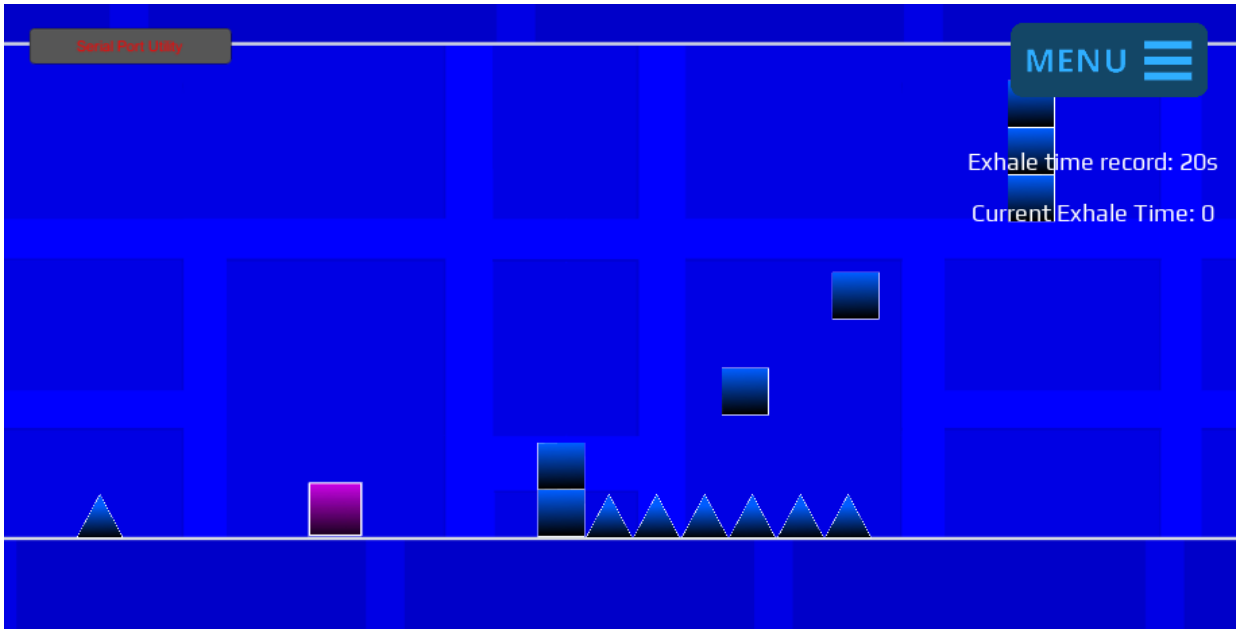


Figure 14: Blue zone of the game. The purple cube is the playable character

Green zones: In these zones, the player can start levitating by inhaling (see figure 15). The longer they inhale, the longer the levitation will go on. The levitation can only be done in the green zone; it is impossible to levitate before or after the zone. The player has to inhale to go

past the zone, because the obstacles in the green zone cannot be dodged by simply jumping. The player has to levitate for a minimum of four seconds, which is a little under what “training inhalation” requires.

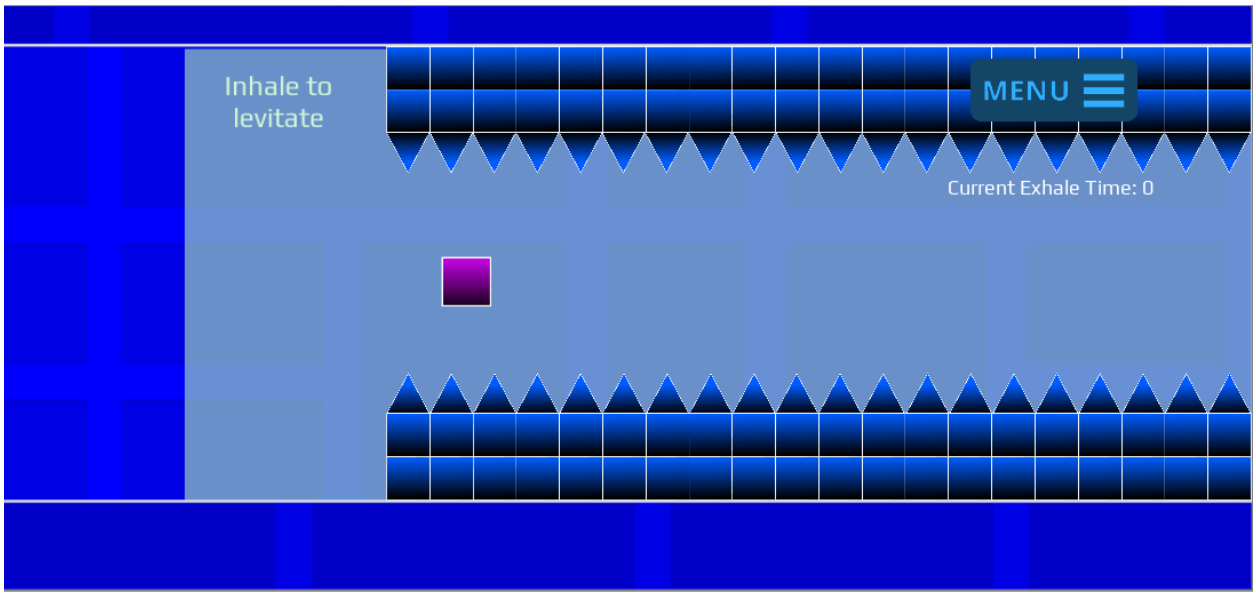


Figure 15: Green, inhaling zone of the game. The purple cube is the playable character

Red zones: These zones are always after the green zones. These zones are designed to do the “training exhalation” exercise. The red zones are after the green zones, because it forces the player to already have enough air to do the exercise. The zone has red instead of blue obstacles (see figure 16). When the player exhales in the zone, the character can destroy these obstacles. This is necessary, because for the first 10 seconds there is a wall that cannot be jumped. This means that the player has to exhale for at least 10 seconds if they want to pass the obstacle at one time. After the wall, the zone will continue for 20 seconds. In this part, the player can choose whether they want to pass the obstacles by exhaling or just by jumping. The reason for this is that on average, people who are novices in doing breathing exercises can exhale for 10 seconds, but experts can even go to 30. To make sure people from all skill levels can improve, the zone is in total 30 seconds. Nevertheless, it is designed so that the novice players can still complete the level. Even When a player cannot breathe for more than 10 seconds, it is still possible to go past the zone. This is because the red obstacles will stay destroyed, even when the player resets his position. So if someone only can breathe for six seconds straight, it is still possible to eventually beat the red zone. Lastly, there is an exhale timer, to show when the player is exhaling.

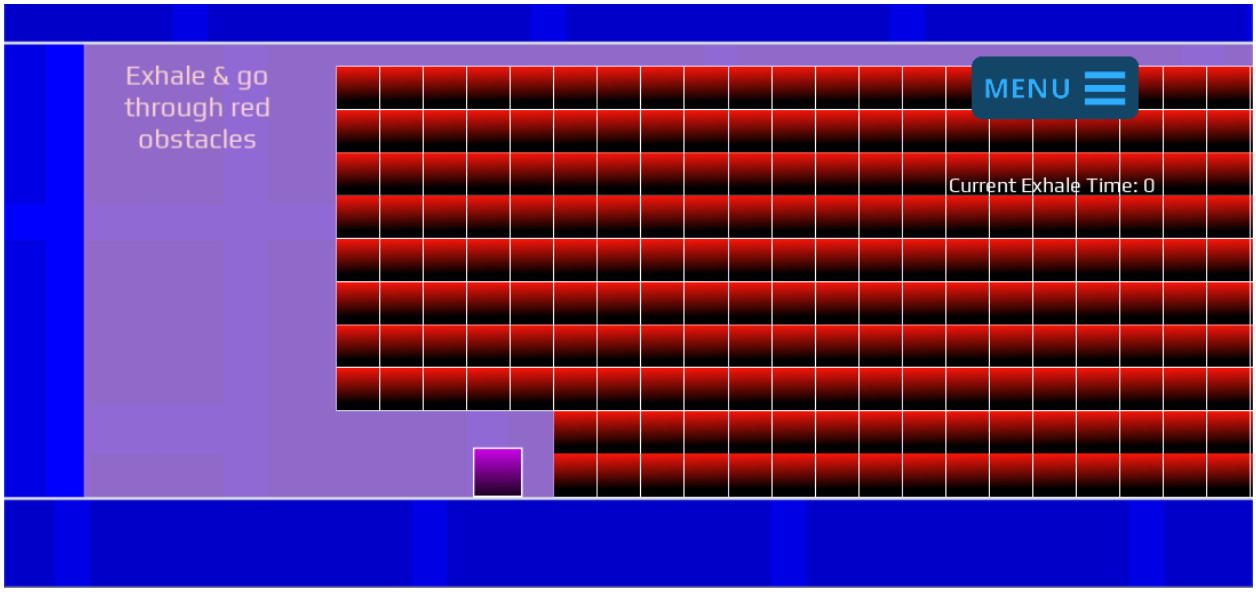


Figure 16: Red, exhaling zone of the game. The purple cube is the playable character

Purple zones: In these zones, the player has to practice the “panting” breathing exercise (see figure 17). By simply panting once, the player will change the gravity switch from touching the floor to touching the roof, or vice versa. Just like the other zones, this can only be performed in this zone. The zone is pretty big, so people have time to prepare to pant. The user has to do it six times.

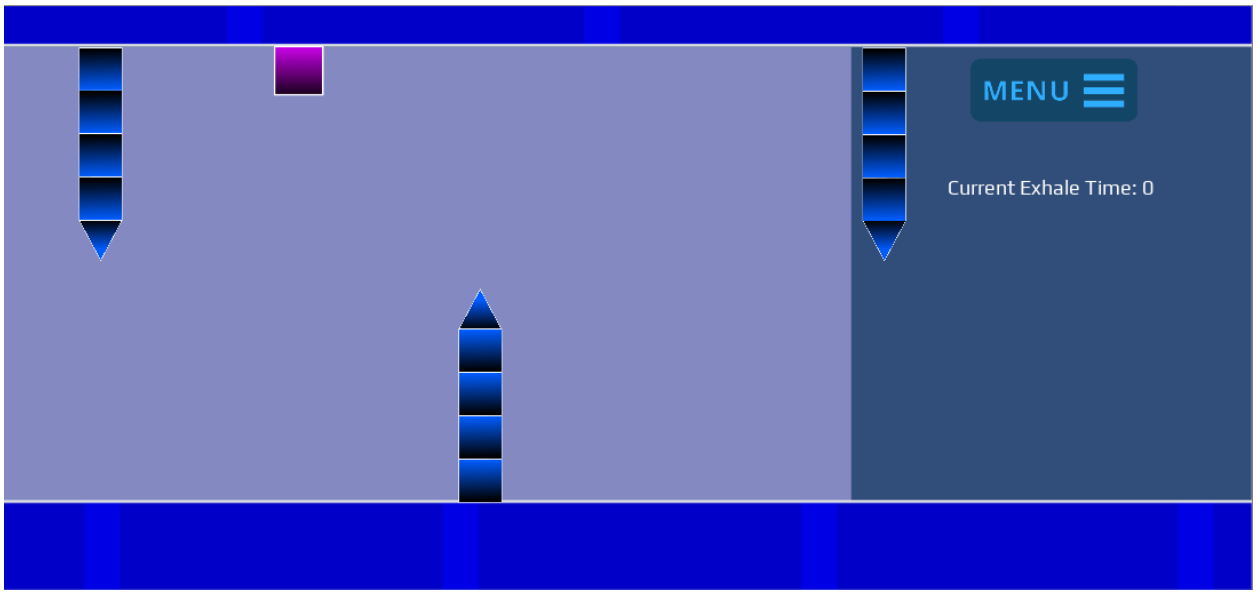


Figure 17: Purple, panting zone of the game. The purple cube is the playable character

Another functionality the game has is the tutorial level. Here, every zone is shown and explained, and there is an easier version implemented. For example, users only have to exhale for two seconds to go through the wall instead of 10. The tutorial also explains how the basic gameplay works, like what checkpoints do and how to jump. Another functionality the game has is the menu button, which pauses the game, so the player can have a break if needed, and

shows a button to skip the rest of the level if a player gets stuck. The last functionality is a function that emails all the in-game data to a specific email address. This way it is easy to see what the player did and how well they execute the breathing exercises.

The art is mostly self-made. The videos of the metaphors are used from Youtube, while the pictures of the metaphor are free-to-use pictures that were adjusted to fit the theme. The background music was also used from the internet, but free-to-use. For the rest, everything is uniquely made. The remaining stimulating elements, both audio and visual, the background, the floor and roof, the basic player and the obstacles were all uniquely made. The visuals were made in Paint.net, while the sound cues were made in CHIRP. The art is made in Paint.net.

6.2 Sensor

To make sure all the breathing exercises can be recognized, a sensor is needed that can at least measure the difference between exhalation and inhalation. A bonus would also be whether it can differentiate between breathing through the abdomen or chest, but it is not necessary for measuring the exercises. The choice between the sensors is either a microphone, a spirometer and a strain/respiration sensor (a respiration sensor is a strain sensor, but programmed to measure breath). A microphone would be the most convenient to use. When using a microphone as a sensor, there is no need for external components, because most devices already have one built-in. This can make testing the game easier by not having to be near the participant. With an external sensor, the sensor has to be brought to the participant. Sadly, after continuous testing, it was concluded that a microphone is too unreliable and cannot distinguish inhalation and exhalation. A microphone can measure amplitude (how loud a sound is) and frequency (how high or low a tone is), but for measuring breathing it is not enough. Both the other sensors are known to differentiate between inhalation and exhalation, but the better one for the game is the respiration sensor (see figure 18). The first reason is that the respiration sensor can measure whether the user uses their abdomen or chest for breathing, while that is not possible for the spirometer. The second reason is that Fernandez[5][12] has already found a way to connect a respiration sensor to Unity. He even wrote a code that would calibrate the sensor to the person who uses it. This would make the readings of the sensor far more reliable. With this calibration, the sensor can even measure how hard somebody is breathing. For all these reasons, a respiration sensor is used for the game.

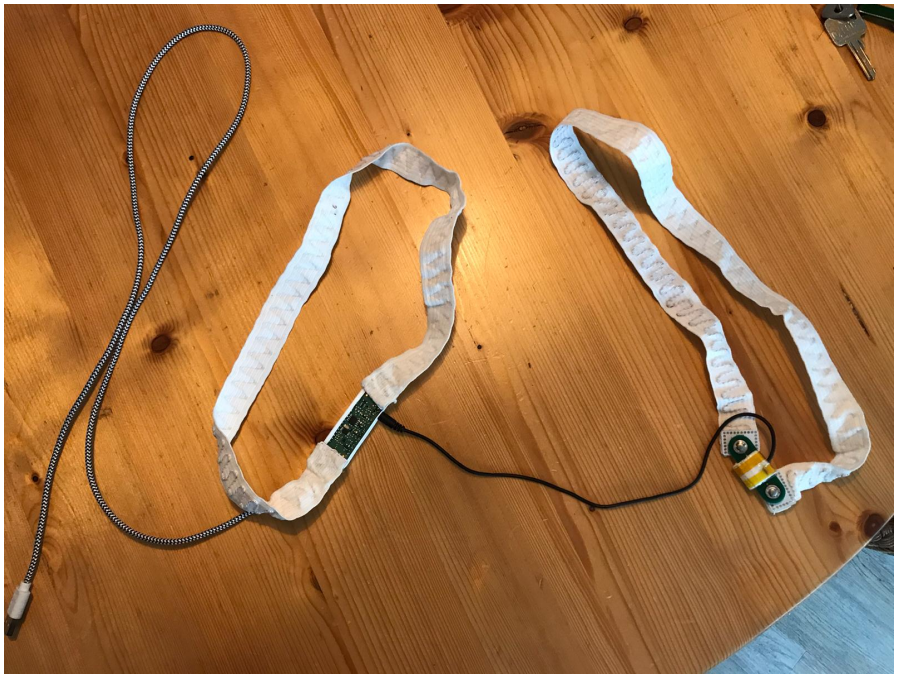


Figure 18: Respiration sensor. The left band is to measure breath for the abdomen and the right one for the chest

The respiration sensor used for the game is borrowed from Fernandez. The sensor is two strain sensors that are attached to a SparkFun. This one will send the data of the two strain sensors to the computer, where it will be deciphered in Unity. The deciphering will be done by a script also written by Fernandez.

6.3 Stimulating Elements

For the stimulating elements, a total of four different implicit elements are implemented (see figure 19). These can be divided into three categories: Visuals, Audio and Game. Visuals are elements that influence the player through their eyes. The first stimulating element that was used in this category is metaphors. The metaphors show real-life examples that have a similar execution as the breathing exercises. For exhaling, it is to slowly let the air out of a balloon, and for panting, it is a dog who breathes. The metaphors are shown with a small video in the background and the character will change into a balloon or dog for the corresponding zone. For the inhaling zone there is a bike pump that pumps in, and the character would change to a bike pump. The other visual stimulating element is circles that animate the time the breath for the specific exercise. For inhaling and exhaling, the middle of the circles will either go from small to big or vice versa respectfully. For panting, the circle will also animate how the player must breathe.

The Audio stimulating elements affect the player through sound. The sound elements used in the game behave similarly to how the circles do. For inhaling and exhaling, there is a tone that will go from soft and low to hard and high, or vice versa again respectfully. This will also help

with how the player has to time their breath. For panting, the sound of panting is mimicked by basic tones, to help the player know how to execute the exercise.

The last stimulating element is from the Game category, which focuses on motivating and engaging players. Players can try to beat an exhalation high score of 20 seconds in the red exhale zone. By beating the 20 seconds, they beat the high score, which updates it immediately. The element tries to motivate players to breathe longer than necessary, so they improve more. Exhale record to motivate to exhale longer than normal.

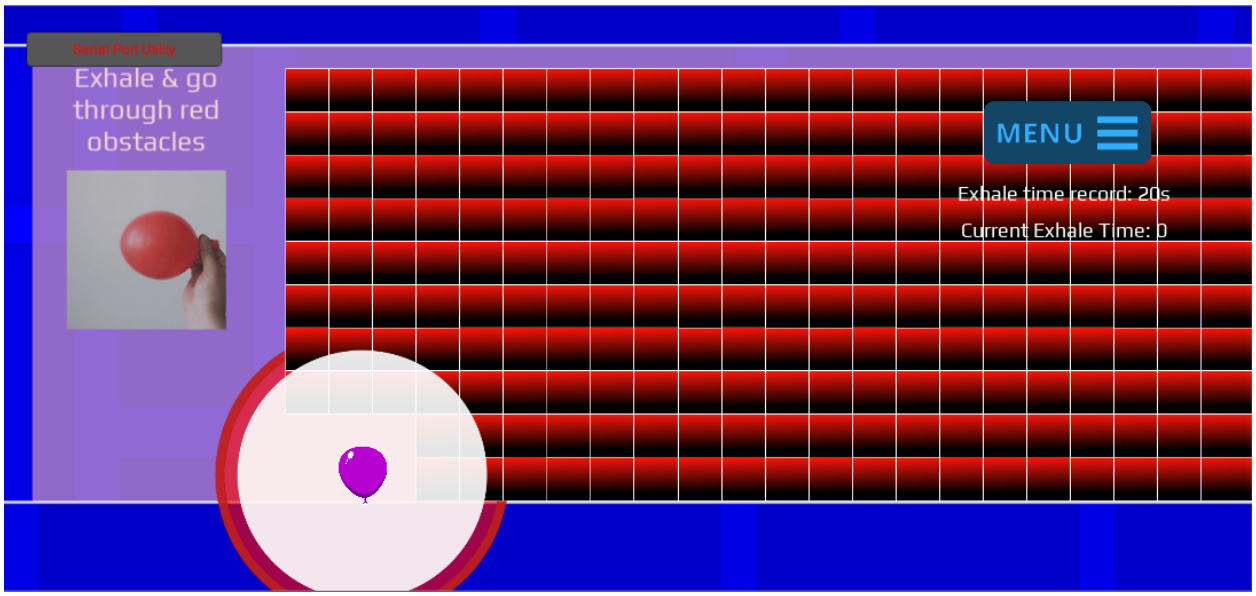


Figure 19: Game with stimulating elements. This version has metaphors, dynamic circles, sound cues and a exhale time record

6.4 Pilot Tests

Before the game gets played for the experiment, it is fruitful to first test it with random people with the intention to find mistakes. In total there were two pilot tests with different people. They only needed to play the game while the researcher observed them. A few problems were found after the testing.

- The game itself was clear, because of the tutorial level. The problem was that people eventually stopped reading the explanation and just continued with the game. Sometimes they lost important bits of information, which made painting especially panting hard to do. To solve this problem, the test was at some places shortened, and some zones now have a more in-depth explanation that is always visible.
- There were also some small problems, bugs, found that was easily fixed. For example, the background music was too loud, and sometimes the levitate ability would continue after leaving the green inhale zone.
- Panting was by far the most difficult exercise to execute. For people with or without breathing exercise experience, it was hard to do the exercise correctly. This was for one how it was coded: If someone was exhaling before panting, the exercise would work. While it is explained in the tutorial, users read over it. The other problem is that panting

is very strict about the execution. If a player did only one aspect wrong, the whole exercise would not register. To make sure players understood panting more, the stimulating elements were applied to the tutorial, in the hope it would help to make panting more clear.

- The bike pump metaphor was confusing. The test people only linked the pump with exhalation, not inhalation. As a solution, the metaphor was removed. A replacement was not found in time, so for the green inhalation zone there is no metaphor anymore.
- The biggest problem was the sensor. While it was very reliable, it had apparently a one-second delay in measuring the breathing. One-second does not sound like much, but for a game like this where timing is crucial, it was really distracting for the players. People got mostly used to it because of the green inhalation zone and the red exhalation zone, the panting exercise was almost impossible to do. To try to solve this problem as best as possible, all zones' start area, the area in a zone that the player has time to start doing the corresponding exercise, got widened, so players had more time to calculate the delay. The panting zone as a whole was also widened to give players more time for each pant.

7 Evaluation

The next step is to see what the effects are of the stimulating elements. This is done by an experiment, which lets participants test the game. The goal is to see what the effects are on the participants. This aligns with the hypothesis that stimulating elements make the execution of breathing exercises better, or it can even be that the stimulating elements have no effect.

7.1 Method

The experiment will be a combination of a questionnaire and a game that the participants have to fill in and play. Before the users could participate, they had to sign the consent form which can be found in Appendix B, Consent form participants. When the consent is filled in, the participants can start doing the experiment. The first step is to fill in the questionnaire. This questionnaire had multiple parts, but the first part is about the experiences of the user with breathing exercises. This was to see if the experience would affect the results. When this part was filled in, the users could start playing the game.

The game itself has 2 versions, one with and one without stimulating elements. The players played both versions, but the order is different for half of the players. This is to compare the results for each version per player, and to be able to compare both versions without the influence of the other. Before the user can play the first version of the game, they have to strap on the respiration sensor. This is explained in detail in the game and questionnaire. After that, the player starts playing one of the versions of the game. The participants will play the game until they execute all breathing exercises at least 3 times. If the player would be stuck after that for more than 2 minutes, the level would be skipped. At that point, there should be enough data and it diminishes the risk of the participant overexerting himself. The next step for the participant is to fill the second part of the questionnaire. This part is about the experience the player had with the first version of the game. It also contains questions about the stimulating elements, to see if players noticed them and if they were effective. After the participants finish that part of the questionnaire, they will play the version of the game they did not already play. The same rules for playtime apply to it. When the player finished playing the second version of the game, the participants were directed to the last part of the questionnaire. This part is also about the experiences the player has with the version played, but also has questions about comparing the two versions. This is to see if users notice the stimulating elements.

The questionnaire is not the only way data is collected in the experiment. The game itself will also collect gameplay data for each version of the game. After the players finish the whole experiment, all the data will be sent via email. This data contains the following information, for each version.

1. How many times the player fails the green inhale zone. This is to see how much difficulty the participants had to do this exercise and pass this zone.
2. How many times the player fails the red exhale zone. This is also to see how much difficulty the participants had to do this exercise and pass this zone.

3. What the longest single exhaling time was. For the breathing exercise “training exhalation”, this is what determines how well the exercise is executed.
4. How far do players go in the purple panting zone? In the pilot tests, participants had difficulty executing the panting exercise and would stop playing the level in this area. To take into account that for the experiment it is the same, not the number of fails is counted. This is because there is a chance that the players will not pass this zone, and stop at a random time which means a random amount of fails. To see if participants improved, it is better to see how far they go, instead of the failures.
5. What is the average use of the chest and abdomen? This is to see if participants breathe more through their stomach, which means they execute the exercises better.

The participants will be people who train singing from a singing teacher or pregnant women between being 5 months pregnant and a year after the birth of the child. Singers that have professional help should also have experience with breathing exercises. For pregnant women, they should also have experience with the exercises between this time, or still remember them. If it is not possible to recruit at least two of each group, regular people will ask to be participants. It does not matter how old they are, which gender, or how much experience they have will breathing exercises. The assumption is that if regular people can show an improvement in the execution of the breathing exercises, the same effect should also apply to the target group.

7.2 Results

The experiment had in total four participants, all regular people with no experience with singing and being pregnant. Two started playing the version of the game with stimulating elements, and the other two were without. The game was sometimes a bit difficult for the participants due to the lack of knowledge about breathing exercises, but all finished the experiment. All the participants got stuck in the purple panting zone and skipped the level around that point, but that was expected. The following sections give the outcome of the gameplay data that was collected and the questionnaire that was filled in.

7.2.1 Data

There can be 3 comparisons made in the experiment. The first one is that the participant starts with the game version without the stimulating elements, and see if there are improvements in the second version of the game with stimulating elements. The second comparison is the previous one, but then vice versa. The last one is comparing the first played version to each other, to being able to see the results without the interference of experience of playing the game, like what can influence the results of the previous two comparisons.

There are two results that are the same for each comparison when looking at figure 20. The first one is that the breathing exercise panting is slightly executed better, but not significantly. There is only a small difference between the versions with and without stimulating elements. The

second one is that abdomen and chest usage almost stays the same in all comparisons. For the other data collected, there are some differences

1st game	Start game without stimulating elements	start game with stimulating elements
Amount levitate failures	2.5	3
Amount boost failures	7.5	8
How far panting average	0.9	1.2
Best single exhalation time (seconds)	5.35	9.95
Average abdomen	0.56	0.6
Average chest	0.73	0.68
	2nd game without stimulating elements	2nd game with stimulating elements
Amount levitate fail	2.5	1
Amount boost fail	3.5	4.5
How far panting	1.2	1.5
Best single exhalation time (seconds)	5.9	8.75
Average abdomen	0.5	0.53
Average chest	0.7	0.67

Figure 20: Shows average results of the experiment for each group. The initial data is found in Appendix B, Result gameplay data.

7.2.1.1 Start game without stimulating elements

When comparing the first and last game the participant played with figure 20, the second game had fewer mistakes in training exhalations. Furthermore, the longest time of a single exhalation is slightly longer in the second game. For the number of inhalation failures, a conclusion cannot be made. This is because one of the users, user 1, took advantage of a bug. The bug was that the game registered holding breath as inhaling, so the user did that instead. There is then only one result left to measure inhalation failures, which is of user 2. While the data shows user 2 improved, using only one result is not enough to make a reliable observation. Overall, the

version with stimulating elements scored better for inhalation and exhalation, but for inhalation it is unreliable

7.2.1.2 Start game with stimulating elements

The results for this comparison are almost the same as the previous comparison. Exhalation failures are lower and the longest single exhale time is higher in the second game compared to the first one played. Furthermore, inhalation failures are also lower in the second game. This time the result is more reliable, because no participants made use of the bug. So the version without stimulating elements scored better than the one without.

7.2.1.3 Comparing both versions

When comparing the first game played of each participant in figure 20, the first observation is that inhalation and exhalation failures are almost the same. The version without stimulating elements is slightly, but not significantly better. Nevertheless, what does have a big difference between the versions is the longest single exhaling time, it is almost twice as high for the version with stimulating elements compared to the one without.

7.2.2 Questionnaire

The other data that was collected for the experiment was the questionnaire. This data is about the participants' experience with breathing exercises, and their experience with both versions of the game. The questions and answers can be found in Appendix B, Result questionnaire.

7.2.2.1 Experience breathing exercises

Coincidently, everybody that participated had some kind of experience with breathing exercises. For most of the participants, it was something that they did in the past for a while. Others still do it regularly, but not to improve singing or being pregnant. Most of them did not have problems with executing the breathing exercises, and felt they knew how to do them correctly.

7.2.2.2 Experience game versions

Most participants liked playing the games. Nevertheless, they are divided about whether it is a better replacement than doing breathing exercises without help. This is most probably because of the difficulty of the levels. The most prominent factor for this was the sensor. All participants had complaints about it. For one, it had a delay on it. This means when a participant would for example exhale, it took the sensor around one to two seconds to recognize. In games, responsiveness is important, even more in a Dash game like the participants played. To pass obstacles, the most important aspect is timing in the game. The delay was not the biggest problem. After a while, participants would get used to the delay. When they did, they could pass the inhale and exhale zones. The main problem was the panting zone. It was for one hard to execute the panting in such a way the sensor would recognize it. The reason for this is that panting is a very specific action with inhaling and exhaling attributes. Furthermore, even if the

participant executed the panting correctly, they would be too late to see whether they did it correctly because of the delay. Another problem the participants felt with the sensor was that it would work less reliably over time. While this cannot be proven, it could be that over time the bands of the sensor would move because of the breathing, which would mess up the calibration.

While they felt the game was sometimes unresponsive and difficult, the participants did say they felt it was useful to do breathing exercises. Especially the version with the stimulating elements. The stimulating elements made the game easier. The stimulating elements circles were most noticed. It helped the participants to time their breathing better. Nevertheless, not all circles worked on one person. Some users felt that only the red circle helped, while others used it more. The only problem with the circles is that some participants did not know whether the circles were animated or would change size controlled by breath. This was confusing for the participants. The rest of the elements had less influence. Only one participant used the sound cues, but nobody noticed the metaphors.

8 Discussion and Future Work

The goal of this report is to see what the effects are of stimulating elements for breathing exercises. With the State of the Art, Background Literature and Evaluation completed, these effects can now be determined, thus the research questions can be answered. After answering the research questions, it is also possible to see the limitations of the experiment, and what could be elaborated on in future works.

8.1 Research Questions

The main research question of this report was *How to implement implicit stimulating elements in a serious game on breathing techniques to increase the correctness of the execution of breathing exercises?* To answer this question, first the following subquestions, determined in chapter 5 Specifications, need to be answered. After the evaluation phase, it was observed that some stimulating elements worked, and others did not. The following subquestion focuses on the parts that do.

- *Do the implemented stimulating elements improve the execution of the breathing exercises? If not, do the implemented stimulating elements affect the players in another way and how?*

In the evaluation phase, some stimulating elements of the game had a positive influence on the execution of the breathing exercise. Mainly the circles that showed how the users should breathe had a positive effect, following the data of the questionnaire. This is probably because the circles were the most obvious. They are big, dynamic, and are exactly in the area users look the most, on the player. Furthermore, also some sound cues seemed to help the performance of the exercises of the participants. Nevertheless, most were unnoticed. This could be for one that the background music was too distracting. The sound cues had a much higher volume than the background music, but maybe it was not enough. Either the sound cues should even be louder, or the background music should be either removed or replaced by something less obvious. One stimulating element did not work at all. The metaphors were not used at all, the same for the single exhale record. One reason why could be that they were not noticed. This is a slight chance, because the player's character changes into a metaphor and the animations of the metaphors are big and thus stand out. It is more possible for the exhale record. It was small and did not add anything to the game. Furthermore, it was at a location the player would not look at much. The reason why is probably that they just do not work. Probably the users could not make a link of one of the metaphors to how to execute a breathing exercise. One solution could be to explain the metaphors better, to give them some explicit elements. The metaphors could work when it is more clear what they do. Nevertheless, it can also just be that metaphors do not work for breathing exercises. This has to be researched further.

When looking at the data, one observation is that the second-played version of the game has almost always better results. This stems from practice makes perfect. The participants learn in the first version played how to deal with the sensor and how to execute the breathing exercises. For the second played version, they already know what to do. This makes the comparison

between the two versions played by one participant unclear and unreliable. The comparison that compares the first game version played by different players, does not have this problem, thus can be used reliably. When observing this comparison, there is only a big difference in the longest single exhaling time. The other data have similar results, which means the stimulating elements did not work for those parts. This contradicts the results of the questionnaire. This either means the data measured is unreliable, which is possible because of the small sample pool, or something else should have been measured. Something that can also influence the data is the sensor. The sensor had a delay in recognizing breath actions, around 1 to 2 seconds. This was distracting for the players, and they needed time to adjust to it. Still, even when adjusted, it still distracted the players greatly. This can influence the scores, but not the participant's opinion on the stimulating elements.

Specific stimulating elements worked on some breathing exercises better than others. The following subquestions will give more detailed information about the effects of specific exercises.

- *Do the implemented stimulating elements for "training inhalation" improve the execution of the breathing exercise?*

The circles worked the best on the "training inhalation" part. One reason could be that people normally do not inhale slowly. With the circles, the participants could see exactly how fast they needed to breathe in. The sound cure for inhaling did not work. A possible reason for this could be that the tone and sound started low in the sound cue. This could make it harder for the player to notice the sound until the end when it is loud and high.

- *Do the implemented stimulating elements for "training exhalation" improve the execution of the breathing exercise?*

In general, the circles also helped with the exhaling breathing exercise. Just as for the inhaling breathing exercise, it helped users time their breath better. What is more, the sound cue was also recognized and was helpful for some participants. It can be assumed that the reason why the sound cue of exhaling worked, on the contrary of the sound cue of inhaling, was that it started loud and with a high tone. This made the tone more recognizable. When the sound cues are recognized, they apparently have the potential to be helpful.

- *Do the implemented stimulating elements for "panting" improve the execution of the breathing exercise?*

The circles also helped with the execution of panting, but it was not enough to let the participant execute the breathing exercise reliably. This is because of the delay of the sensor, and the complexity of the exercise. The stimulating elements did not have enough influence to overcome these problems. The player needs more assistance, and the question is if implicit stimulating elements can overcome these problems. Nevertheless, the scores of the average distance in panting zone are higher for the stimulating elements. The difference is not that big, and the sample size of the experiment is small, so the results are not totally reliable.

With the subquestions answered, there is enough information to answer the main question.

- *How to implement implicit stimulating elements in a serious game on breathing techniques to increase the correctness of the execution of breathing exercises?*

Based on the previous chapters, a few conclusions can be made on how to implement implicit stimulating elements. The most important rule is to make sure the stimulating elements are noticeable. There can be many reasons why this can fail. The first aspect is that the game itself masks the stimulating elements. Players can be distracted by the game, and not pay attention to the elements. For example, background music can distract players from audio cues. One solution is to make either the stimulating elements extremely noticeable. For one, sound cues can be three times as loud as the background music. For visual elements, there are more ways to let them stand out. Enlarging the size is not enough. The animations of the metaphors were big, but that did not help them to get recognized by the player. What is important is to put the stimulating elements in places in the game the player will pay much attention to. In Breath Dash, the most successful stimulating element, circles, was directly on the player. This element was the most noticeable of all the stimulating elements following the questionnaire, looking at how much is talked about each element. Another way to make the stimulating elements stand out more is to remove game elements that distract. For the sound cues, the background music can be deleted. For visuals, some obstacles can be removed to make sure the player can only focus on the stimulating elements. Furthermore, not only the game itself can be distracting for the players. A sensor with many shortcomings can also shift the focus of players. This happened in the experiment in chapter 7, Evaluation. An easy fix is to make sure the sensor is reliable and responsive. Thus, it is important to make sure the stimulating elements are obvious and get noticed, and that game elements are not too distracting.

The second aspect that is important for implementing stimulating elements is to make it clear what they do and how to interpret them. In chapter 7 Evaluation two stimulating elements had the problem that players did not understand them sufficiently. The first one was the circles, in which players did not whatever they were animations or would react to their breath. While this issue was not making the elements useless, it still confused thus distracted the player. It would be better if the game made it more obvious how the circles worked, for example showing the circles in an area where the player does not need to do a breathing exercise or to explicitly explain the element. The other stimulating element that was not clear was the metaphors. People could not make a connection between them and how to perform a breathing exercise. So, it is important to make sure there are no misconceptions about the stimulating elements, and that people recognize what the stimulating element is trying to do.

The last aspect is complexity. While implicit stimulating elements are useful, they have their limits. Some exercises can be too complicated for the implicit stimulating elements to work. The breathing exercise “panting” potentially had that problem. This exercise needs very specific actions to be executed correctly. This was difficult to do with only implicit stimulating elements. The use of also explicit stimulating elements could potentially help the performance more. To conclude, not all breathing exercises are compatible with using only implicit stimulating

elements. If the exercise is too complicated, it is potentially fruitful to use other kinds of elements to increase performance. For example explicit stimulating elements.

When implementing implicit stimulating elements, these three aspects should be taken into account. This way, there is a higher potential that the elements have a higher chance of working. Implementing the elements can be hard, because there is much freedom in how to do it. So it is helpful to have some guidelines on how to do it correctly.

8.2 Limitations

The report is not completely perfect, it has its limitations. For one there were some operational difficulties, which impacted testing. The first problem was to find the target group. People from the target group were not eager to join the experiment. This could be that the experiment did not interest them, or that they were afraid to get corona. The reason is not clear. Nevertheless, to solve this problem, the target group had to change. This change does not make the results much less reliable, but it would be interesting to see how the original target group would react to the stimulating elements. Furthermore, waiting for the responses of the original target group and switching to the new cost much more valuable time. Another operation problem was the sensor. It would randomly stop connecting to a computer. This happened multiple times while testing. When this happened, it meant that for the rest of the day no testing could happen. This and the previous problem made the number of test results not as high as planned. The number of test results would not be high anyway, because it is a qualitative experiment. Nevertheless, more results would increase the reliability of the conclusions.

The sensor had more problems than not wanting to connect to a computer. It had a long delay in recognizing breath. This distracted the players, and made the game far more difficult to play. Especially players had problems with the breathing exercise “panting”. It is not the sole reason, but the sensor contributed to it heavily. Furthermore, if the delay of the sensor was noticed earlier, it would have impacted the ideation phase. A dash game's main gameplay is to time the abilities of the character correctly. This is hard to do with a slow sensor. Either the game idea needed to change or a faster sensor should be implemented, to make the game less difficult.

8.3 Conclusion

The goal of this report was to see how to implement implicit stimulating elements, so that serious game designers could have an easier time with implementing their elements. Furthermore, it also would make the elements potentially more effective. This goal was written into the following research question.

- *How to implement implicit stimulating elements in a serious game on breathing techniques to increase the correctness of the execution of breathing exercises?*

After doing preliminary research and testing, three aspects of how to implement implicit stimulating elements were found. The first aspect is that the stimulating elements should be noticeable. There are many ways to do that, from removing distracting elements in the game to

putting the elements in locations where the attention of the player is, for example near the player character. The second aspect is to make the stimulating elements clear. When there is confusion about how they work or how to relate to it, the effectiveness will decrease. This can even be so low that the elements do not even have any effect on the player. The last aspect is about complexity and the limits of implicit stimulating elements. The more complex a breathing exercise, the harder it is to have an effective implicit stimulating element. It is useful to be open-minded for other ways to help the performance for complex breathing exercises, next to implicit stimulating elements.

Other findings in the report were that implicit stimulating elements potentially have a positive influence on the execution of breathing exercises. These elements were the most effective when all the aspects were used. Lastly, when designing an affective game, it is important to have a reliable and responsive sensor. When the sensor has big shortcomings, it heavily distracts the player from the game and from the stimulating elements.

8.4 Future Work

This report gives much information on how to implement stimulating elements, so they can be effective. This is useful information, but it is only the start. One thing observed in the chapter Background and Related Work, is that there are many different kinds of stimulating elements. There are many ways to influence people's behavior: Visuals, sound, even smell. Stimulating elements can also be implicit or explicitly implemented. All these different kinds of stimulating elements can affect the players differently. It would be fruitful to have more information on how each different element affects the player, so they can be implemented in the correct situation.

Another research that could help the implementation of stimulating elements is to see how game elements get noticed. In the experiment of this report, participants did not notice most of the stimulating elements. While there are theories why, it would be better to have something more reliable. Implemented stimulating elements that are not noticed, are not effective. There already are suggestions in this report on how to do it, but they can be elaborated on. So it is important to make sure they do.

References

[1] Malliarakis, Christos & Tomos, Florica & Shabalina, Olga & Mozelius, Peter. (2018). “Andragogy and E.M.O.T.I.O.N.: 7 key factors of successful serious games”. ECGBL 2018, 12th European Conference on Games Based Learning. At: Sophia Antipolis, France. Vol. 12. pp. 371-378.

[2] Davidova, Jelena & Zavadska, Galina & Rauduvaitė, Asta & Chuang, Ming. (2017). “Strategies for the Development of 6–8-Year- Old Children’s Breathing for Singing”. Pedagogika. Vol. 125(1). pp. 111-121. DIO: 10.15823/p.2017.08.

[3] Napoli, Maria & Krech, Paul & Holley, Lynn. (2005). “Mindfulness Training for Elementary School Students: The Attention Academy”. Journal of Applied School Psychology. Vol. 21. pp. 99-125. DIO: 10.1300/J370v21n01_05.

[4] Gilbert TB. (1998). “Breathing difficulties in wind instrument players”. Maryland Medical Journal. Vol. 47(1). pp. 23-27.

[5] Fernandes, Brolin. (2020). “Exploring possibilities in interactive technology to create Cross-Domain Interactive Breathing Exercise Games”. M.Sc thesis. University of Twente.

[6] Sonne, Tobias & Jensen, Mads M. (2016). ChillFish: “A Respiration Game for Children with ADHD”. the TEI '16: Tenth International Conference. DOI:10.1145/2839462.2839480

[7] Patibanda, Rakesh & Mueller, Florian & Leskovsek, Matevz & Duckworth, Jonathan. (2017). “Life Tree: Understanding the Design of Breathing Exercise Games”. the Annual Symposium. pp. 19-31. DIO: 10.1145/3116595.3116621.

[8] Van Delden, R. W. & Bos, D. P. & de With, A. J. V. & Vogel, K & Klaasen, R. & Zwart, N. & Faber, J. & Thio, B. & van der Kamp, M. (2020). SpiroPlay, a Suite of Breathing Games for Spirometryby Kids & Experts. DIO: 10.1145/3410404.3414223.

[9] Sharma, Pragya & Hui, Xiaonan & Zhou, Jianlin & Conroy, Thomas & Kan, Edwin. (2020). “Wearable radio-frequency sensing of respiratory rate, respiratory volume, and heart rate”. Vol. 3(1). DIO: 10.1038/s41746-020-0307-6.

[10] White Jr, Robert C. (1988). “On the teaching of breathing for the singing voice”. Journal of Voice. Vol. 2(1). pp. 26–29. DIO: 10.1016/S0892-1997(88)80054-7.

[11] Sataloff, Robert. (2000). “Evaluation of professional singers”. Otolaryngologic Clinics of North America. Vol. 33(5). 923-56. DIO: 10.1016/S0030-6665(05)70256-5.

[12] Fernandes, Brolin. (2020). “Music, Media to Breathe”. Research Topics of M.Sc thesis. University of Twente.

[13] Haddadi, Raziyeh & Ravanbakhsh, Majid & adati, Najmiyeh & Mohammadi, Masumeh & Nargesi, Moslem. (2014). “The Effect of Breathing Exercises on Breathing Pattern of Pregnant Women”. Physical Treatments: Specific Physical Therapy Journal. Vol. 4. pp. 57-61.

[14] Kim, Jong-Hyun & Lee, Jung. (2020). “Controlling your contents with the breath: Interactive breath interface for VR, games, and animations”. Vol. 15(10): DIO: 10.1371/journal.pone.0241498

[15] Nacke, Lennart & Kalyn, Michael & Lough, Calvin & Mandryk, Regan. (2011). “Biofeedback Game Design: Using Direct and Indirect Physiological Control to Enhance Game Interaction”. Proceedings of the International Conference on Human Factors in Computing Systems, CHI 2011, Vancouver, BC, Canada. pp. 103-112. DOI: 10.1145/1978942.1978958.

[16] Kuikkaniemi, Kai & Laitinen, Toni & Turpeinen, Marko & Saari, Timo & Kosunen, Ilkka & Ravaja, Niklas. (2010). “Influence of implicit and explicit biofeedback in first-person shooter games”. Proceedings of the 28th International Conference on Human Factors in Computing Systems, CHI 2010, Atlanta, Georgia, USA. Vol. 2. pp. 859-868. DIO: 10.1145/1753326.1753453.

[17] Zander, Thorsten & Kothe, Christian & Leske, Sabine & Gärtner, Matti. (2010). “Enhancing Human-Computer Interaction with Input from Active and Passive Brain-Computer Interfaces”. DIO: 10.1007/978-1-84996-272-8_11.

[18] Nacke, Lennart E. & Mandryk, Regan L. (2021). “Designing Affective Games with Physiological Input”.

[19] Zafar, M. & Ahmed, Beena & Gutierrez-Osuna, Ricardo. (2017). “Playing with and without Biofeedback”. IEEE International Conference on Serious Games and Applications for Health. pp. 1-7. DIO: 10.1109/SeGAH.2017.7939272.

[20] Silvia, Paul & Kashdan, Todd. (2009). “Interesting Things and Curious People: Exploration and Engagement as Transient States and Enduring Strengths”. Social and Personality Psychology Compass. Vol. 3(5). pp. 785-797. DIO: 10.1111/j.1751-9004.2009.00210.x.

[21] Landry, Pascal & Minsky, Joseph & Castañer, Marta & Camerino, Oleguer & Rodriguez-Arregui, Rosa & Ormo, Enric & Pares, Narcis. (2013). “Design Strategy to Stimulate a Diversity of Motor Skills for an Exergame Addressed to Children”. DIO: 10.1145/2485760.2485781.

[22] Mayer, Richard. (2005). “Cognitive theory of multimedia learning”. The Cambridge handbook of multimedia learning”. pp.31-48. DIO: 10.1017/CBO9780511816819.004.

[23] Gaina, Raluca & Stephenson, Matthew. (2019). ““Did You Hear That?” Learning to Play Video Games from Audio Cues”. IEEE Conference on Games 2019. pp. 1-4. DIO: 10.1109/CIG.2019.8848088.

[24] Tajadura-Jiménez, Ana & Basia, Maria & Deroy, Ophelia & Fairhurst, Merle & Marquardt, Nicolai & Bianchi-Berthouze, Nadia. (2015). “As Light as your Footsteps: Altering Walking Sounds to Change Perceived Body Weight, Emotional State and Gait”. CHI '15: Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems. pp. 2943-2952. DIO: 10.1145/2702123.2702374.

[25] Hogue, Hannah. (2021). “The Effects of Music Genre on Rate of Perceived Exertion in Aerobics Classes”. Honors College thesis. University of Southern Mississippi.

[26] Um, Eunjoon & Plass, Jan & Hayward, Elizabeth & Homer, Bruce. (2012). “Emotional Design in Multimedia Learning”. Journal of Educational Psychology. Vol. 104(2). pp. 485-498. DIO: 10.1037/a0026609.

[27] Ninaus, Manuel & Kiili, Kristian & Wood, Guilherme & Moeller, Korbinian & Kober, Silvia. (2020). “To Add or Not to Add Game Elements? Exploring the Effects of Different Cognitive Task Designs Using Eye Tracking”. IEEE Transactions on Learning Technologies. Vol. 13(4). pp. 847-60. DIO: 10.1109/TLT.2020.3031644.

[28] Harp, Shannon & Mayer, Richard. (1998). “How Seductive Details Do Their Damage: A Theory of Cognitive Interest in Science Learning”. Journal of Educational Psychology. Vol. 90(3). pp. 414-434. DIO: 10.1037/0022-0663.90.3.414.

[29] Mayer, Richard & Moreno, Roxana. (2003). “Nine Ways to Reduce Cognitive Load in Multimedia Learning”. Educational Psychologist. Vol. 38(1). 43-52. DIO: 10.1207/S15326985EP3801_6.

[30] Plass, Jan L & Mayer, Richard E. & Homer, Bruce D. (2020).“Emerging design factors in game-based learning: Emotional design, musical score, and game mechanics design”. Handbook of Game-Based Learning. pp. 347–365.

[31] Schell, Jesse. (2008). “The Art of Game Design Book of Lenses”. London, England. Taylor & Francis.

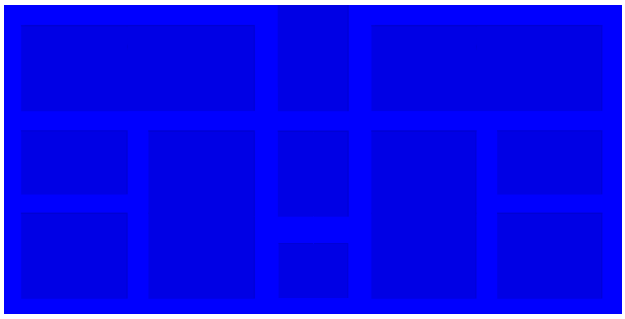
[32] Schaie, K. W. (1961). “Scaling the Association between Colors and Mood-Tones”. The American Journal of Psychology. Vol 74(2). pp. 266-273. DOI:10.2307/1419412

[33] Mader, Angelika H., & Eggink, Wouter. (2014). “A Design Process for Creative Technology”, Proceedings of the 16th International conference on Engineering and Product Design, E&PDE 2014. pp. 568-573.

Appendix A

Artwork game

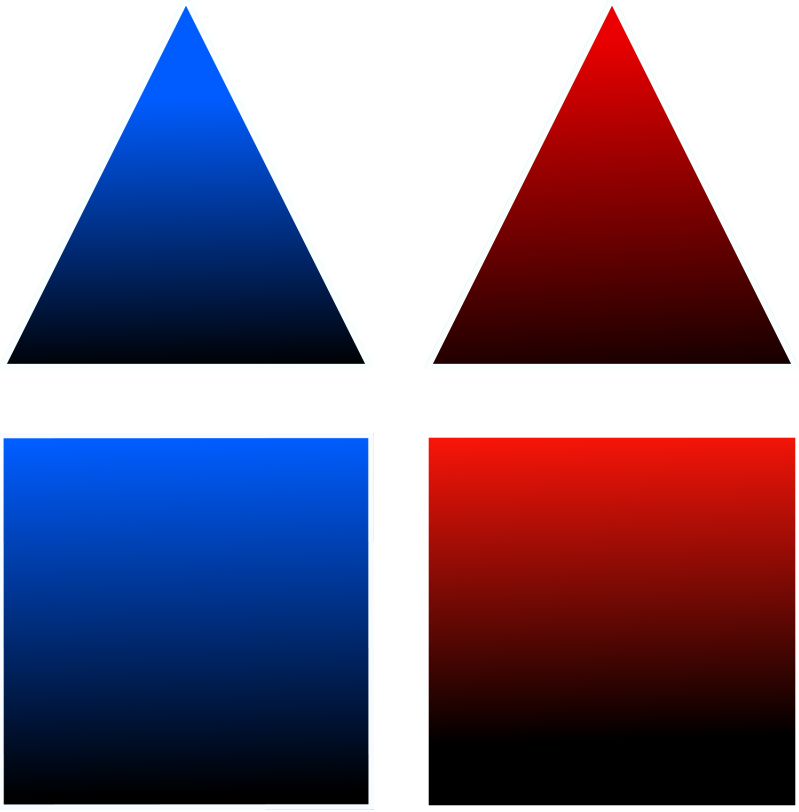
Background and Ground



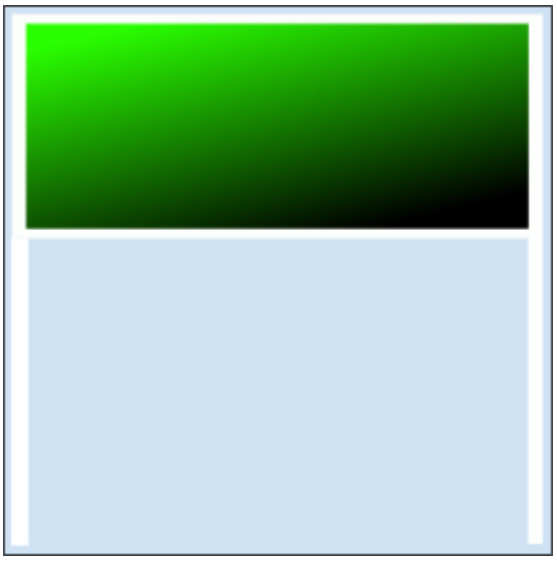
Menu

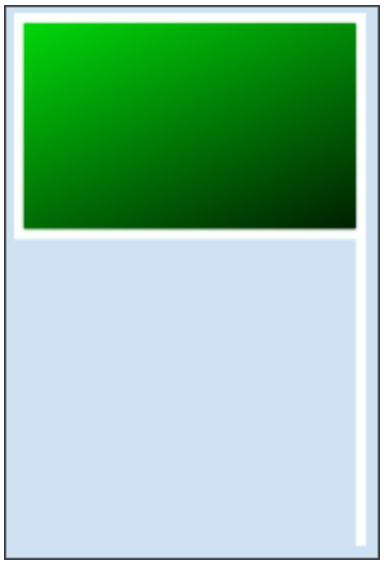
Playable Character

Obstacles

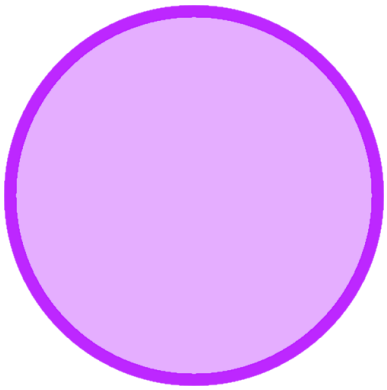
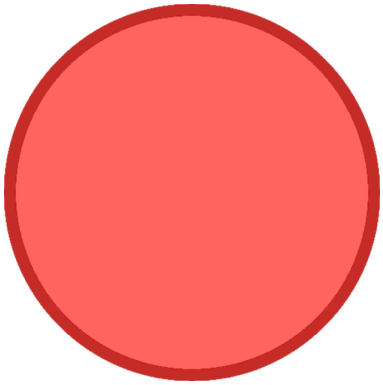
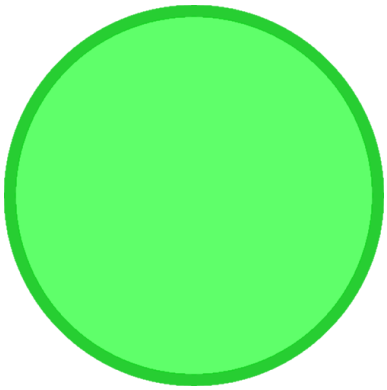


Checkpoint and finish



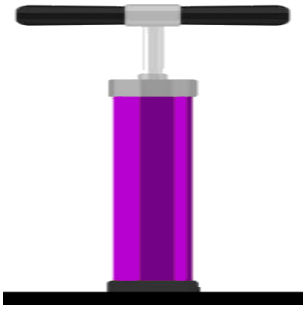


Stimulating elements circles

Stimulating elements metaphor player character





Code game

Brolin Fernandez code

Get Breath Packet

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using System.Linq;

public class GetBreathPacket : MonoBehaviour
{
	// To get data from script that handles serial input
	private SerialPortUtility.SerialPortUtilityPro serialScript;

	// Final integer breath values
	public int abdBreathValue;
	public int chBreathValue;

	// Data of relevance
	public int abdHighByte;
	public int abdLowByte;
	public int chestHighByte;
	public int chestLowByte;

	// The entire raw incoming packet as integer values
	public byte[] packet;

	public float timer;
	private string temp;

	private Callibration callibration;
	ButtonsManager buttonsManagerScript;

	private void Awake()
	{
	serialScript =
GameObject.FindObjectOfType<SerialPortUtility.SerialPortUtilityPro>();
	callibration = GameObject.FindObjectOfType<Callibration>();
	buttonsManagerScript = GameObject.FindObjectOfType<ButtonsManager>();
```

```
	}


	// All input byte data cleaning, escape sequence addition, etc.
	public void GetThePacket(byte[] dataPacket)
	{
	packet = dataPacket;
	timer = Mathf.Round(Time.time * 1000f) / 1000f;
	//Debug.Log(timer);

	// According to relevant bytes as ordered in the incoming packet
	// If no escape sequence involved, byte number 2,3,4,5 are required
to be extracted in my case, could be different per device/byte sequence
	if (!(packet[2] == 125 | packet[3] == 125 | packet[4] == 125 |
packet[5] == 125))
	{
		abdHighByte = packet[2];
		abdLowByte = packet[3];
		chestHighByte = packet[4];
		chestLowByte = packet[5];
	}

	// Next 4 if conditions are for escape sequence handling. Comments in
the first if loop are relevant to all others.
	if (packet[2] == 125)
	{

		temp = packet[3].ToString("X"); // Hex value of the byte from
which the required data needs to be retrieved
		abdHighByte = packet[3] ^ 32;   // To get back the value that
caused the escape sequence to occur. Byte 2 is replaced by an escape value,
which shifts the corresponding values one place ahead.
		/* Debug.Log("TEMP: " + temp);
		Debug.Log("ABD HIGH BYTE:" + abdHighByte);*/
		abdLowByte = packet[4];       // Abd low byte then becomes
packet[4] and so on..
		chestHighByte = packet[5];
		chestLowByte = packet[6];
	}

	if (packet[3] == 125)
	{
		abdHighByte = packet[2];
```

```
            temp = packet[4].ToString("X");
            abdLowByte = packet[4] ^ 32;
            /*Debug.Log("TEMP: " + temp);
            Debug.Log("ABD LOW BYTE:" + abdLowByte);*/
            chestHighByte = packet[5];
            chestLowByte = packet[6];
      }

      if (packet[4] == 125)
      {
            abdHighByte = packet[2];
            abdLowByte = packet[3];
            chestHighByte = packet[5] ^ 32;
            temp = packet[5].ToString("X");
            /*Debug.Log("TEMP: " + temp);
            Debug.Log("CHEST HIGH BYTE:" + chestHighByte);*/
            chestLowByte = packet[6];
      }

      if (packet[5] == 125)
      {
            abdHighByte = packet[2];
            abdLowByte = packet[3];
            chestHighByte = packet[4];
            chestLowByte = packet[6] ^ 32;
            temp = packet[6].ToString("X");
            /*Debug.Log("TEMP: " + temp);
            Debug.Log("CHEST LOW BYTE:" + chestLowByte);*/
      }

      // Once escape sequence handling is done, set the integer values by
combining the bytes:
      if(abdHighByte > 0 & chestHighByte > 0)
      {
            abdBreathValue = abdLowByte + (abdHighByte * 256);
            chBreathValue = chestLowByte + (chestHighByte * 256);
      }

      // Send relevant data to csv file handler
      CSVManager.AppendToReport(GetReportLine());

      }
```

```
      // Sends data to csv tool created
      string[] GetReportLine()
      {
      string[] returnable = new string[3];      // Adjust according to how
much data you need to send to csv files
      returnable[0] = timer.ToString();
      returnable[1] = abdBreathValue.ToString();
      returnable[2] = chBreathValue.ToString();

      return returnable;
      }

}
```

Button Manager

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class ButtonsManager : MonoBehaviour
{

      public bool startCallibration = false;
      public bool doTheCallibration = false;
      public bool finalInputsIncoming = false;

      public GameObject canvas;
      public ScreenControl screenScript;

      public void StartCalibrationButtonFunctionality()
      {
      startCallibration = true;
      doTheCallibration = false;
      finalInputsIncoming = false;
      }

      public void DoTheCallibrationButtonFunctionality()
      {
      doTheCallibration = true;
      startCallibration = false;
      finalInputsIncoming = false;
```

```
	}

	public void StartFinalGameInputButtonFunctionality()
	{
	doTheCallibration = false;
	startCallibration = false;
	finalInputsIncoming = true;
	screenScript.ActivateTutorialScreen();
	}
}
```

Calibration

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using System.Linq;
using UnityEngine.UI;
using System.Linq;

/// <summary>
/// All input comes from here
/// </summary>

public class Callibration : MonoBehaviour
{
	public List<float> abdList = new List<float>();
	private List<float> chList = new List<float>();
	private List<float> tList = new List<float>();

	public List<float> abdCallibrationList = new List<float>();
	private List<float> chCallibrationList = new List<float>();
	private List<float> timeList = new List<float>();

	private List<float> mov_aves_abd = new List<float>();
	private List<float> mov_aves_ch = new List<float>();

	public bool sendToNorm = false;

	private GetBreathPacket getBreathPacketScript;
	private ButtonsManager buttonsManagerScript;

	float sum_abd_std;
```

```
	float sum_ch_std;
	public float mean_abd;
	public float mean_ch;
	public float std_abd;
	public float std_ch;
	public float max_abd;
	public float max_ch;
	public float min_abd;
	public float min_ch;

	public int std_multiplier = 3;

	public int mov_avg_N = 4;
	float mov_avg_abd_val = 0;
	float mov_avg_ch_val = 0;


	private void Awake()
	{
	getBreathPacketScript =
GameObject.FindObjectOfType<GetBreathPacket>();
	buttonsManagerScript = GameObject.FindObjectOfType<ButtonsManager>();
	}
	// Start is called before the first frame update
	void Start()
	{

	}
	public float temp_time = 0;
	// Update is called once per frame
	void Update()
	{
	if(getBreathPacketScript.timer > temp_time)
	{
		if (buttonsManagerScript.startCallibration)
		{
			CallibrationStart();
		}

		if (buttonsManagerScript.doTheCallibration)
		{
			CallibrationStop();
			sendToNorm = true;
```

```
        }

        if (buttonsManagerScript.finalInputsIncoming)
        {
            FinalInputs();
        }
    }
    temp_time = getBreathPacketScript.timer;
    }


    // When the 'Start Calibration' UI button is clicked

    public void CallibrationStart()
    {
    abdList.Add(getBreathPacketScript.abdBreathValue);
    chList.Add(getBreathPacketScript.chBreathValue);
    tList.Add(getBreathPacketScript.timer);
    }


    // When 'Stop' button gets clicked, perform the following operations
    public void CallibrationStop() {

    Debug.Log("stop flag");
    // Calculate Means and Standard Deviations
    abdCallibrationList = abdList;
    chCallibrationList = chList;
    timeList = tList;

    print(abdCallibrationList.Count);
    for (int i = 0; i < abdCallibrationList.Count; i++)
    {

    }

        mean_abd = abdCallibrationList.Average();
    foreach (float value_abd in abdCallibrationList)
    {
        sum_abd_std += Mathf.Pow((value_abd - mean_abd), 2);
    }
    std_abd = Mathf.Sqrt(sum_abd_std / abdCallibrationList.Count);
```

```
      mean_ch = chCallibrationList.Average();
      foreach (float value_ch in chCallibrationList)
      {
            sum_ch_std += Mathf.Pow((value_ch - mean_ch), 2);
      }
      std_ch = Mathf.Sqrt(sum_ch_std / chCallibrationList.Count);


      // Detect outliers with Means and STDs
      foreach (float val in abdCallibrationList.ToList())
      {
            //Debug.Log(abdCallibrationList.IndexOf(val) + ":" + val);
            if (val > (mean_abd + (std_multiplier * std_abd)) || val <
(mean_abd - (std_multiplier * std_abd)))
            {
                  int inda = abdCallibrationList.IndexOf(val);
                  abdCallibrationList.RemoveAt(inda);
                  chCallibrationList.RemoveAt(inda);
                  timeList.RemoveAt(inda);
            }
      }

      foreach (float val in chCallibrationList.ToList())
      {
            if (val > (mean_ch + (std_multiplier * std_ch)) || val <
(mean_ch - (std_multiplier * std_ch)))
            {
                  int indc = chCallibrationList.IndexOf(val);
                  abdCallibrationList.RemoveAt(indc);
                  chCallibrationList.RemoveAt(indc);
                  timeList.RemoveAt(indc);
            }
      }

      // Perform Moving Average Smoothing

      float sum_abd;
      float sum_ch;

      for (int window_start_index = 0; window_start_index <
(abdCallibrationList.Count - mov_avg_N); window_start_index++)
      {
            float temp_abd = 0;
```

```
            float temp_ch = 0;

            for (int i = 0; i < (mov_avg_N); i++)
            {
                  sum_abd = abdCallibrationList[i + window_start_index] +
temp_abd;
                  temp_abd = sum_abd;
                  sum_ch = chCallibrationList[i + window_start_index] +
temp_ch;
                  temp_ch = sum_ch;
            }
            mov_avg_abd_val = temp_abd / mov_avg_N;
            mov_avg_ch_val = temp_ch / mov_avg_N;
            mov_aves_abd.Add(mov_avg_abd_val);
            mov_aves_ch.Add(mov_avg_ch_val);
      }

      abdCallibrationList = mov_aves_abd;
      chCallibrationList = mov_aves_ch;

      int len_abd = abdCallibrationList.Count;
      int len_ch = chCallibrationList.Count;
      int len_t = timeList.Count;

      int diff = len_t - len_abd;

      for (int popper = 0; popper <= diff; popper++)
      {
            timeList.RemoveAt(0);
      }

      // Calculate Maximum, Minimum, Mean and Standard deviation for the
final smoothed lists
      max_abd = abdCallibrationList.Max() + 75;
      max_ch = chCallibrationList.Max() + 75;
      min_abd = abdCallibrationList.Min() - 25;
      min_ch = chCallibrationList.Min() - 25;

      mean_abd = abdCallibrationList.Average();

      foreach (float value_abd in abdCallibrationList)
      {
            sum_abd_std += Mathf.Pow((value_abd - mean_abd), 2);
```

```
        }
        std_abd = Mathf.Sqrt(sum_abd_std / abdCallibrationList.Count);

        mean_ch = chCallibrationList.Average();
        foreach (float value_ch in chCallibrationList)
        {
              sum_ch_std += Mathf.Pow((value_ch - mean_ch), 2);
        }
        std_ch = Mathf.Sqrt(sum_ch_std / chCallibrationList.Count);

        Debug.Log("A mean --------------------------- : " + mean_abd);

        buttonsManagerScript.doTheCallibration = false;

        }


        public float abdValFromGBP;
        public float chValFromGBP;
        public float tValFromGBP;

        private List<float> abdSmoothingWindow = new List<float>();
        private List<float> chSmoothingWindow = new List<float>();

        public float smoothedAbd = 0;
        public float smoothedCh = 0;

        public float normedAbd = 0;
        public float normedCh = 0;

        public bool isInhaling = false;
        public bool isExhaling = false;
        public bool isConstant = false;

        public void FinalInputs()
        {
        // Perform Outlier Removal with means and standard deviations and
then get the cleaned input without outliers

        if (getBreathPacketScript.abdBreathValue > (mean_abd +
(std_multiplier * std_abd)) || getBreathPacketScript.abdBreathValue <
(mean_abd - (std_multiplier * std_abd)) ||
getBreathPacketScript.chBreathValue > (mean_ch + (std_multiplier * std_ch))
```

```
|| getBreathPacketScript.chBreathValue < (mean_ch - (std_multiplier *
std_ch)))
      {
            //Debug.Log("Outlier Eliminated: ABD: " +
getBreathPacketScript.abdBreathValue + "  CH: " +
getBreathPacketScript.chBreathValue);
      }
      else
      {
            abdValFromGBP = getBreathPacketScript.abdBreathValue;
            chValFromGBP = getBreathPacketScript.chBreathValue;
            tValFromGBP = getBreathPacketScript.timer;
      }

      // Moving Average Smoothing

      if (abdSmoothingWindow.Count < 5)
      {
            abdSmoothingWindow.Add(abdValFromGBP);
            chSmoothingWindow.Add(chValFromGBP);
      }

      float sum_abd;
      float sum_ch;

      for (int window_start_index = 0; window_start_index <
(abdSmoothingWindow.Count - mov_avg_N); window_start_index++)
      {
            float temp_abd = 0;
            float temp_ch = 0;

            for (int i = 0; i < (mov_avg_N); i++)
            {
                  sum_abd = abdSmoothingWindow[i + window_start_index] +
temp_abd;
                  temp_abd = sum_abd;
                  sum_ch = chSmoothingWindow[i + window_start_index] +
temp_ch;
                  temp_ch = sum_ch;
            }
            smoothedAbd = temp_abd / mov_avg_N;
            smoothedCh = temp_ch / mov_avg_N;
```

```
            float temp_prev_window_value = abdSmoothingWindow[0];
            int posCount = 0;
            int negCount = 0;
            foreach(float each_val in abdSmoothingWindow.GetRange(1,
mov_avg_N-1))
            {
                  if(each_val > temp_prev_window_value)
                  {
                  posCount++;
                  }
                  else if(each_val < temp_prev_window_value)
                  {
                  negCount++;
                  }
                  else
                  {

                  }

                  temp_prev_window_value = each_val;
            }
            /*if(posCount > negCount)
            {
                  Debug.Log("Decreasing");
            }
            else if(posCount < negCount)
            {
                  Debug.Log("Increasing");
            }
            else
            {
                  Debug.Log("Equal");
            }*/

            // CheckDirection(abdSmoothingWindow, "abdomen");
            CheckDirection(chSmoothingWindow, "chest");

            //ImmediateAccelerationCheck(abdSmoothingWindow, mean_abd);
            ImmediateAccelerationCheck(chSmoothingWindow, mean_ch);

            abdSmoothingWindow.RemoveAt(0);
            chSmoothingWindow.RemoveAt(0);
      }
```

```
// Normalize values using Max, min

normedAbd = 1 - ((smoothedAbd - min_abd) / (max_abd - min_abd));
normedCh = 1 - ((smoothedCh - min_ch) / (max_ch - min_ch));

if (normedAbd > 0.99)
{
      normedAbd = 0.99f;
}
if (normedAbd < 0.01)
{
      normedAbd = 0.01f;
}

if (normedCh > 0.99)
{
      normedCh = 0.99f;
}
if (normedCh < 0.01)
{
      normedCh = 0.01f;
}

}

//public float lowThreshold = 0.2f;
//public float highThreshold = 0.8f;
public void CheckDirection(List<float> window, string whichPart )
{
float temp_prev_window_value = window[0];
int posCount = 0;
int negCount = 0;
foreach (float each_val in window.GetRange(1, mov_avg_N - 1))
{
      if (each_val > temp_prev_window_value)
      {
            posCount++;
      }
      else if (each_val < temp_prev_window_value)
      {
            negCount++;
      }
```

```
            else
            {

            }

            temp_prev_window_value = each_val;
      }
      if (posCount > negCount)
      {
            isExhaling = true;
            isInhaling = false;
            isConstant = false;
            //Debug.Log(whichPart + "Decreasing");
      }
      else if (posCount < negCount)
      {
            isExhaling = false;
            isInhaling = true;
            isConstant = false;
            //Debug.Log(whichPart + "Increasing");
      }
      else
      {
            isExhaling = false;
            isInhaling = false;
            isConstant = true;
            //Debug.Log(whichPart + "Constant-ish");
      }

      }

      public float theVarianceDiff = 0;
      public bool quick = false;
      public bool slow = false;
      public bool hold = false;
      public void ImmediateAccelerationCheck(List<float> theWindow, float
mean)
      {
      float diff = 0;
      float max = 0;
      float min = 0;
      //float variance = 0;
```

```
        max = theWindow.Max();
        min = theWindow.Min();
        diff = max - min;

        theVarianceDiff = diff;

        if (theVarianceDiff <= 7)
        {
              hold = true;
              slow = false;
              quick = false;
        }
        if((7 < theVarianceDiff) && (theVarianceDiff <= 30))
        {
              hold = false;
              slow = true;
              quick = false;
        }
        if (theVarianceDiff > 30)
        {
              hold = false;
              slow = false;
              quick = true;
        }
        }

}
```

Own code

Background

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class background : MonoBehaviour
{
      public float speedBackground;

      public bool tutorialMessage;
```

```
	// Update is called once per frame
	void Update()
	{
	if (tutorialMessage)
	{
		float t = Time.deltaTime - speedBackground;
		transform.position = new Vector3(transform.position.x + t,
transform.position.y, transform.position.z);
	}
	}

	public void Reset(Vector2 startPos)
	{
	transform.position = new Vector2(startPos.x+37.4f,
transform.position.y);
	}
}
```

Breath Control

```
using UnityEngine;
using UnityEngine.UI;
public class BreathControl : MonoBehaviour
{
	public Callibration calScript;
	public Jump jumpScript;

	float directionGravity = 1f;
	public float inhaleTimer;
	public float exhaleTimer;

	float recordTime = 20f;
	float personalBestTime;

	public bool levitateZone;
	public bool gravityZone;
	public bool gravitySwitched;

	public string textZone;
	int pantTimes;
	float inhaleText;
```

```
    float exhaleText;

    public bool boostZone;
    public bool boost;

    int checkpointsReached;

    public Text record;
    public Text ownTime;

    public Sprite balloon;
    public Sprite pump;
    public Sprite dog;
    public Sprite player;
    public SpriteRenderer render;

    public AudioSource inhaleAudio;
    public AudioSource exhaleAudio;

    public PantingSound pantingAudio;

    [SerializeField] bool hitTrap;

    public GameObject breathInStim;
    public GameObject breathOutStim;
    public GameObject pantStim;
    public Animator animIn;
    public Animator animOut;


    public float gameTotalAbd;
    public float gameTotalCh;
    public int gameNumber;
    private void Start()
    {
    }

    // Update is called once per frame
    void Update()
    {
    if (Input.GetKeyDown(KeyCode.Mouse1))
    {
          Physics2D.gravity = new Vector2(0, 9.8f * directionGravity);
```

```
            directionGravity *= -1;
      }
      //print(textZone);
      MoveCar();
      }

      void MoveCar()
      {
      gameTotalAbd += calScript.normedAbd;
      gameTotalCh += calScript.normedCh;
      gameNumber++;
      //if inhaling, move car left until -xlim
      if ((calScript.isInhaling))
      {
            if (exhaleTimer < 1.25f && exhaleTimer > 0.2f && gravityZone &&
!gravitySwitched && calScript.quick == true)
            {
                  pantTimes++;
                  Physics2D.gravity = new Vector2(0, 9.8f *
directionGravity);
                  directionGravity *= -1;
                  gravitySwitched = true;
            }
            if (inhaleTimer > 0.2f)
            {
                  if (boostZone)
                  {
                  boost = false;
                  //textZone += " " + exhaleTimer + ", ";
                  }

                  if (levitateZone)
                  {
                  if (exhaleText <= exhaleTimer)
                  {
                        exhaleText += exhaleTimer;
                        textZone += exhaleText + ", ";
                  }
                  }
                  exhaleTimer = 0;
            }
            //Debug.Log("Inhale");
            inhaleTimer += Time.deltaTime;
```

```
            if (inhaleTimer > 0.1f && levitateZone)
            {

                  transform.position = new Vector3(transform.position.x,
0.55f, transform.position.z);
                  this.GetComponent<Rigidbody2D>().gravityScale = 0;
            }
            else
            {
                  this.GetComponent<Rigidbody2D>().gravityScale = 3;
            }

      }


      // If exhaling, move car right until +xlim
      if ((calScript.isExhaling))
      {
            this.GetComponent<Rigidbody2D>().gravityScale = 3;
            exhaleTimer += Time.deltaTime;
            if (exhaleTimer > 0.2f)
            {
                  gravitySwitched = false;

                  if (levitateZone)
                  {
                  if (inhaleText <= inhaleTimer)
                  {
                        inhaleText = inhaleTimer;
                        textZone += inhaleText + ", ";
                  }
                  }

                  inhaleTimer = 0;
                  if (boostZone)
                  {
                  if (ownTime != null && record != null)
                  {
                        ownTime.text = "Current Exhale Time: " +
Mathf.Round(exhaleTimer) + "s";
                        if (personalBestTime < exhaleTimer)
                        {
                              personalBestTime = exhaleTimer;
```

```
                        }
                        if (personalBestTime > recordTime)
                        {
                              record.text = "Your Record: " +
Mathf.Round(exhaleTimer) + "s";
                              recordTime = exhaleTimer;
                        }
                  }
                  boost = true;
                  }
            }

            //Debug.Log("Exhale");
      }
      }
      private void OnTriggerEnter2D(Collider2D collision)
      {


      if (collision.gameObject.tag == "Breakable wall")
      {
            if (boost)
            {
                  //Destroy(collision.gameObject);
            }
            else
            {
                  hitTrap = true;
                  jumpScript.Respawn();
            }
      }

      if (collision.gameObject.tag == "LevitateZone")
      {
            textZone += checkpointsReached + "___levitateZone: ";
            if (pump != null)
            {
                  animIn.Play("CircleBreathIn", -1, 0);
                  breathInStim.SetActive(true);
                  inhaleAudio.Play();
                  //render.sprite = pump;

            }
```

```
            levitateZone = true;
      }

      if (collision.gameObject.tag == "GravityZone")
      {
            textZone += checkpointsReached + "___gravityZone: ";
            if (pantStim != null)
            {
                  //animPant.Play("CircleBreathIn", -1, 0);
                  pantStim.SetActive(true);
                  pantingAudio.playPanting = true;

                  render.sprite = dog;
            }
            gravityZone = true;
      }

      if (collision.gameObject.tag == "BoostZone")
      {
            textZone += checkpointsReached + "___boostZone: ";

            if (balloon != null)
            {
                  animOut.Play("CircleBreathOut", -1, 0);
                  breathOutStim.SetActive(true);
                  exhaleAudio.Play();
                  render.sprite = balloon;
            }
            boostZone = true;
      }

      if (collision.gameObject.tag == "Trap")
      {
            hitTrap = true;

            jumpScript.Respawn();

      }
      if (collision.gameObject.tag == "checkpoint")
      {
            checkpointsReached++;
            jumpScript.StartPos = new
Vector2(collision.gameObject.transform.position.x + 3f,
```

```
collision.gameObject.transform.position.y);

      }
      }

      private void OnTriggerExit2D(Collider2D collision)
      {
      if (collision.gameObject.tag == "LevitateZone")
      {
            levitateZone = false;
            render.sprite = player;
            if (inhaleAudio != null)
            {
                  inhaleAudio.Stop();
            }

            if (breathInStim != null)
            {
                  breathInStim.SetActive(false);
            }

            if (inhaleText <= inhaleTimer)
            {

                  inhaleText = inhaleTimer;
                  textZone += inhaleText + ", ";
            }

            if (!hitTrap)
            {
                  textZone += inhaleText + " cleared levitate zone--- \n";
            }
            else
            {
                  textZone += "failed levitateZone --- \n";
            }
            hitTrap = false;
      }

      if (collision.gameObject.tag == "GravityZone")
      {
            if (pantStim != null)
            {
```

```
                pantStim.SetActive(false);
            }
            gravityZone = false;
            render.sprite = player;
            if (pantingAudio != null)
            {
                pantingAudio.playPanting = false;
            }

            if (!hitTrap)
            {
                textZone += pantTimes + " cleared gravity zone--- \n";
            }
            else
            {
                textZone += "failed gravityZone --- \n";
            }
            hitTrap = false;
            pantTimes = 0;
        }

        if (collision.gameObject.tag == "BoostZone")
        {
            if (breathOutStim != null && exhaleAudio != null)
            {
                breathOutStim.SetActive(false);
                exhaleAudio.Stop();
            }
            textZone += personalBestTime + ", ";


            if (!hitTrap)
            {
                textZone += "cleared boost zone--- \n";
            }
            else
            {
                textZone += "failed boostZone ---\n";
            }
            hitTrap = false;
            boostZone = false;
            render.sprite = player;
            boost = false;
```

```
      }
   }
}
```

Calibration Screen

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class CalibrationScreen : MonoBehaviour
{
      bool startCalibrationTimer;
      public float CalibrationTimer;
      public float maxCalibrationTime;
      public Text CalibrationTimerText;
      public Button startButton;

      ButtonsManager callibrationScript;

      public GameObject WelcomeScreen;
      public GameObject CalibrationInfoScreen;

      public Text nameTester;
      // Start is called before the first frame update
      void Start()
      {
      callibrationScript = GameObject.FindObjectOfType<ButtonsManager>();

      }

      // Update is called once per frame
      void Update()
      {
      if (startCalibrationTimer)
      {
            CalibrationTimer += Time.deltaTime;
            int time = Mathf.RoundToInt(maxCalibrationTime -
CalibrationTimer);
            CalibrationTimerText.text = " " + time;
```

```
            if (CalibrationTimer >= maxCalibrationTime)
            {
                  CalibrationFinished();
            }
      }
      }

      public void GoCaliScreen()
      {
      PlayerPrefs.SetString("nameTester", nameTester.text);
      WelcomeScreen.SetActive(false);
      CalibrationInfoScreen.SetActive(true);
      }

      public void StartCalibration()
      {
      startCalibrationTimer = true;
      }

      public void CalibrationFinished()
      {
      CalibrationTimer = 0f;
      startCalibrationTimer = false;
      startButton.gameObject.SetActive(true);
      callibrationScript.DoTheCallibrationButtonFunctionality();
      }
}
```

Camera Move

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class CameraMove : MonoBehaviour
{
      public GameObject player1;

      // Start is called before the first frame update
      void Start()
      {
```

```
	}

	// Update is called once per frame
	void Update()
	{
		this.transform.position = new
Vector3(player1.transform.position.x + 4.5f, 0, -10);
	}
}
```

Email

```
using System.Net;
using System.Net.Mail;
using System.Net.Security;
using System.Security.Cryptography.X509Certificates;
using UnityEngine;

public class Email : MonoBehaviour
{
	public string fromEmail = "thomasrebel1997@gmail.com";
	public string toEmail = "thomasrebel1997@gmail.com";
	public string subject = "SubjectName";
	public string body = "Body of the email";
	public string password = "nuclearPinguin01";
	// Start is called before the first frame update
	void Start()
	{
	//EmailSending();
	}

	// Update is called once per frame
	public void EmailSending(string bodyTemp)
	{

	MailMessage mail = new MailMessage();
	mail.From = new MailAddress(fromEmail);
	mail.To.Add(toEmail);
	subject = PlayerPrefs.GetString("nameTester");
	mail.Subject = subject;

	body = PlayerPrefs.GetString("firstGame") + "\n" + bodyTemp;
```

```
      mail.Body = body;
      // you can use others too.
      SmtpClient smtpServer = new SmtpClient("smtp.gmail.com", 587);
      //smtpServer.Port = 587;
      smtpServer.Credentials = new System.Net.NetworkCredential(fromEmail,
password) as ICredentialsByHost;
      smtpServer.EnableSsl = true;
      ServicePointManager.ServerCertificateValidationCallback =
      delegate (object s, X509Certificate certificate, X509Chain chain,
SslPolicyErrors sslPolicyErrors)
      { return true; };
      smtpServer.Send(mail);

      }

}
```

Game Finish Stimulating

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class GameFinishStimulating : MonoBehaviour
{
      public ScreenControl script;
      public BreathControl breathScript;
      public Email emailScript;
      public GameObject secondGame;
      private void OnTriggerEnter2D(Collider2D collision)
      {
      if (collision.gameObject.tag == "Player")
      {

            emailScript.EmailSending(PlayerPrefs.GetString("firstGame") +
"\n" + secondGame.name + ": " + breathScript.textZone +
                                    "\n Average Abdomen Value: " +
(breathScript.gameTotalAbd / breathScript.gameNumber) +
                                          "\n Average Chest Value: " +
(breathScript.gameTotalCh / breathScript.gameNumber));
            script.ActivateEndScreen();
      }
```

```
      }

      public void Skip()
      {
      emailScript.EmailSending(PlayerPrefs.GetString("firstGame") + "\n" +
secondGame.name + ": " + breathScript.textZone +
                                    "\n Average Abdomen Value: " +
(breathScript.gameTotalAbd / breathScript.gameNumber) +
                                          "\n Average Chest Value: " +
(breathScript.gameTotalCh / breathScript.gameNumber));
      script.ActivateEndScreen();
      }
}
```

Go To Middle Screen

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class GoToMiddleScreen : MonoBehaviour
{
      public ScreenControl script;
      public BreathControl breathScript;
      public GameObject firstGame;
      private void OnTriggerEnter2D(Collider2D collision)
      {
      if (collision.gameObject.tag == "Player")
      {
            PlayerPrefs.SetString("firstGame", firstGame.name + ": " +
breathScript.textZone +
                                    "\n Average Abdomen Value: " +
(breathScript.gameTotalAbd/breathScript.gameNumber) +
                                          "\n Average Chest Value: " +
(breathScript.gameTotalCh / breathScript.gameNumber));
            script.ActivateScreenBetweenGamesScreen();
      }
      }

      public void Skip()
      {
      PlayerPrefs.SetString("firstGame", firstGame.name + ": " +
```

```
breathScript.textZone +
                                    "\n Average Abdomen Value: " +
(breathScript.gameTotalAbd / breathScript.gameNumber) +
                                              "\n Average Chest Value: " +
(breathScript.gameTotalCh / breathScript.gameNumber));
      Time.timeScale = 1;
      script.ActivateScreenBetweenGamesScreen();
      }
}
```

Jump

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Jump : MonoBehaviour
{
      public float playerSpeed = 5f;
      public float jumpHeight = 10f;
      public Rigidbody2D playerRB;
      public bool isTouchingGround;
      public Vector2 StartPos;
      public background backgroundScript;

      public bool tutorialActive = false;

      // Start is called before the first frame update
      void Start()
      {
      StartPos = new Vector2(-6f, -1.2f);
      }

      // Update is called once per frame
      void Update()
      {
      GoJump();
      }

      void GoJump()
      {
      transform.Translate(Vector2.right * (Time.deltaTime * playerSpeed));
```

```
	if (Input.GetKeyDown(KeyCode.Space) && tutorialActive == false)
	{
		if (isTouchingGround)
		{
			playerRB.velocity = new Vector2(0, 0);
			playerRB.AddForce(new Vector2(2f, jumpHeight),
ForceMode2D.Impulse);
			isTouchingGround = false;
		}

	}
	}

	private void OnTriggerEnter2D(Collider2D collision)
	{
	if (collision.gameObject.tag == "Ground")
	{
		isTouchingGround = true;
	}
	}

	public void Respawn()
	{
	Physics2D.gravity = new Vector2(0, -9.8f);
	backgroundScript.Reset(StartPos);
	playerRB.velocity = Vector3.zero;
	transform.position = StartPos;

	//PlayerPrefs.SetString("mailMessage", mailMessage);
	}
}
```

Menu

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Menu : MonoBehaviour
{
	public GameObject menuObject;
	// Start is called before the first frame update
```

```
	void Start()
	{

	}

	// Update is called once per frame
	void Update()
	{

	}

	public void MenuStart()
	{
	Time.timeScale = 0;
	menuObject.SetActive(true);
	}

	public void MenuClose()
	{

	menuObject.SetActive(false);
	Time.timeScale = 1;
	}
}
```

Panting Sound

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class PantingSound : MonoBehaviour
{
	float timer = 0.5f;
	public bool playPanting;
	public Animator animPant;

	public AudioSource pantingAudio;

	bool first;
	// Start is called before the first frame update
	void Start()
```

```
	{

	}

	// Update is called once per frame
	void Update()
	{
	if (playPanting)
	{
		if (!first)
		{
			animPant.enabled = false;
		}
		timer += Time.deltaTime;
		if (timer > 1.5f)
		{

			first = true;
			animPant.enabled = true;
			animPant.Play("CirclePanting", -1, 0);
			pantingAudio.Play();
			timer = 0;
		}
	} else
	{
		first = false;
	}
   }
}
```

Screen Control

```
using UnityEngine;

public class ScreenControl : MonoBehaviour
{
	public GameObject BeginScreen;
	public GameObject TutorialScreen;
	public GameObject GameScreen;
	public GameObject ScreenBetweenGames;
	public GameObject StimulateGameScreen;
	public GameObject EndScreen;
```

```
    // Start is called before the first frame update
    void Start()
    {
    ActivateBeginScreen();
    }

    // Update is called once per frame
    void Update()
    {

    }

    public void ActivateBeginScreen()
    {
    BeginScreen.SetActive(true);
    TutorialScreen.SetActive(false);
    GameScreen.SetActive(false);
    EndScreen.SetActive(false);
    }

    public void ActivateTutorialScreen()
    {
    BeginScreen.SetActive(false);
    TutorialScreen.SetActive(true);
    GameScreen.SetActive(false);
    EndScreen.SetActive(false);
    }

    public void ActivateGameScreen()
    {
    BeginScreen.SetActive(false);
    TutorialScreen.SetActive(false);
    GameScreen.SetActive(true);
    EndScreen.SetActive(false);
    ScreenBetweenGames.SetActive(false);
    }

    public void ActivateScreenBetweenGamesScreen()
    {
    GameScreen.SetActive(false);
    StimulateGameScreen.SetActive(false);
    ScreenBetweenGames.SetActive(true);
    }
```

```
	public void ActivateStimulateGameScreen()
	{
	BeginScreen.SetActive(false);
	StimulateGameScreen.SetActive(true);
	ScreenBetweenGames.SetActive(false);
	GameScreen.SetActive(false);
	TutorialScreen.SetActive(false);
	}
	public void ActivateEndScreen()
	{
	BeginScreen.SetActive(false);
	TutorialScreen.SetActive(false);
	StimulateGameScreen.SetActive(false);
	GameScreen.SetActive(false);
	EndScreen.SetActive(true);
	}

	public void EndGame()
	{
	Application.Quit();
	}
}
```

Start Normal Game

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class StartNormalGame : MonoBehaviour
{
	public ScreenControl script;
	private void OnTriggerEnter2D(Collider2D collision)
	{
	if (collision.gameObject.tag == "Player")
	{
		script.ActivateGameScreen();
	}
	}
}
```

Tutorial Hit Trigger

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class TurorialHitTrigger : MonoBehaviour
{

      public TutorialScreenController script;
      private void OnTriggerEnter2D(Collider2D collision)
      {
      script.hitTrigger();
      Destroy(gameObject);
      }
}
```

Tutorial Screen Controller

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class TutorialScreenController : MonoBehaviour
{
      public Jump playerScript;
      public background backgroundScript;

      public GameObject[] popupMessages;
      public GameObject blueBackground;
      public GameObject whiteBorder;
      int currentMessage = 0;

      public void hitTrigger()
      {
      playerScript.playerSpeed = 0;
      playerScript.tutorialActive = true;
      backgroundScript.tutorialMessage = false;

      blueBackground.SetActive(true);
      whiteBorder.SetActive(true);
      popupMessages[currentMessage].SetActive(true);
```

```
	}

	public void Next()
	{
	blueBackground.SetActive(false);
	whiteBorder.SetActive(false);
	popupMessages[currentMessage].SetActive(false);
	currentMessage++;

	playerScript.playerSpeed = 4;
	playerScript.tutorialActive = false;
	backgroundScript.tutorialMessage = true;
	}
}
```

Video Script

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class VideoScript : MonoBehaviour
{
	public Transform player;
	public GameObject[] vid;
	// Start is called before the first frame update
	void Start()
	{

	}

	// Update is called once per frame
	void Update()
	{
	print("hello");

	for (int i = 0; i < vid.Length; i++)
	{
		float dist = Vector2.Distance(player.position,
vid[i].transform.position);
		if (dist < 20f)
		{
```

```
                vid[i].SetActive(true);
            }
            else
            {
                vid[i].SetActive(false);
            }
        }
        }
}
```

Appendix B

Consent form participants

Information sheet and consent form lo-fi evaluation
"Breathing exercises, how to stimulate execution"

YOU WILL BE GIVEN A COPY OF THIS INFORMATION SHEET

Introduction
You have been asked to participate in a research about breathing exercises. This form provides information about the research and the possibility to give your consent for participating. Read the form carefully, before deciding to participate. If there are any questions, please feel free to ask them to the researcher.

Description of the study
The main question for this research is "How do you implement stimulating elements and breathing inputs in a serious game on breathing techniques for multiple domains to increase engagement and the correctness of the execution of breathing exercises?" This specific research will be testing whether these elements and inputs have a positive effect of doing breathing exercises. You will be asked to do an interview and to play a game that will include breathing inputs to control the game. There is a risk that the breathing inputs will cause shortness of breath. Therefore, you may always choose to stop doing the experiment or are asked to stop if the researcher sees this effect arise.
You will first do an interview, followed up by playing the game. The first interview will contain questions about current experiences with breathing exercises, motivation and will contain an explanation about the game. The game will be a basic, traditional game from one of the most popular game genres. Part of the game will be

controlled by doing breathing exercises. After playing the game, another interview will take place. This interview will be about your experiences with the game. The total duration will be around 20 to 30 minutes. The experiment will be free of charge. Before the experiment, the participants will be asked to download the game on their phone or computer. Furthermore, there is potentially also an external sensor needed to play the game. This will either just be a microphone, or a soft strap around your belly and chest. These straps will slightly pressure you when breathing, but not enough to make you feel unpleasant. The interviews and prototype testing will help have a better understanding of the influence of the stimulating elements and if the theory about the use of different kinds of game input is correct. This research is part of a graduation project in the Creative Technology bachelor at University of Twente.

Covid procedures

The external sensor will be delivered by the researcher beforehand. To minimize risks, the sensor will be cleaned before and after each use, and the researcher will always be more than 1.5 meters away from the participant. He will also follow every present corona-guideline when delivering. The rest of the experiment is online, so there is no risk for covid transmission.

Procedures for withdrawal from the study

You are free to leave the experiment at any moment, upon which all collected data from you from our research will be removed, unless you give consent for using this data. You also have the option to ask to remove all collected data after the experiment, but maximum of 24 hours after finishing the experiment. There will be no repercussions for leaving the experiment early or asking to delete your data.

Use of personal information

For this research, personal information will be gathered. Personal data is an audio recording of the interview. This will be done to get more insight into breathing exercises. For one, the interview will be audio recorded if permission is given. The recordings are used to use the information of the interview without losing part of it through incomplete notes or memory loss. Information and possibly quotes will be used from the audio recording, but you will stay anonymous. The audio recordings will not be shared with anyone except the researcher, and you will be completely anonymous in the paper. You have the right to ask for access to your personal information which will be used for this research and the right to ask to erase it.

Usage of data during the research

The audio recordings from this research will be kept in an online data cloud to be used by other researchers involved in this research. Every piece of data concerning your personal information that could identify you, such as this consent form and audio recordings, will be kept offline and safe. Audio recordings will be used only by other researchers involved in this research.

The audio recordings can also be used in the graduation report about this research. This

will be made public, but the data that is used from the audio recordings, quotes, in the report will be anonymous, thus will not trace back to you. After the research is complete, all the audio recordings will be removed, only the final report will be kept online.

Furthermore, logs of your gameplay will be saved for research purposes. This will contain in game actions, scores and sensor data. This data will be anonymously processed and used in the final report. After the research, the data will be written down into the report. These are still anonymous. All other files will be deleted.

For any further information/questions, you can contact the researcher:
Researcher: t.rebel@student.utwente.nl
Supervisor: D.Reidsma@utwente.nl

Contact Information for Questions about Your Rights as a Research Participant

If you have questions about your rights as a research participant, or wish to obtain information, ask questions, or discuss any concerns about this study with someone other than the researcher(s), please contact the Secretary of the Ethics Committee for Computer and Information Science at the University of Twente, by ethicscommittee-cis@utwente.nl

UNIVERSITY OF TWENTE.

Please tick the appropriate boxes

Required boxes to be "Yes"

Taking part in the study	**Yes**	**No**
I have read and understood the study information dated [.../.../....], or it has been read to me.	◯	◯
I have been able to ask questions about the study and my questions have been answered to my satisfaction.	◯	◯

I consent voluntarily to be a participant in this study and understand that I can refuse to answer questions and I can withdraw from the study at any time, without having to give a reason. ◯ ◯

I understand that taking part in the study involves an audio recorded interview and a survey questionnaire completed by the participant. After the research is complete, all personal information and audio recordings will be destroyed, only anonymous data and the final report will be kept online. ◯ ◯

Use of the information in the study

I understand that information I provide will be used for publication of the report for this research. This report will be made public, available for everyone. However, the data used in the report will be anonymous. ◯ ◯

I understand that personal information collected about me that can identify me, such as gender and age, will not be shared beyond the researcher. ◯ ◯

Required Boxes to be "Yes" or "No"

I agree with being audio recorded in the interview. I agree quotes of mine from the audio recording can be used in the final research paper. ◯ ◯

Signatures

_____ _____ _____

Name of participant Signature Date

I have accurately read out the information sheet to the potential participant and, to the best of my ability, ensured that the participant understands what they are freely consenting.

_____ _____ _____

Researcher name Signature Date

Study contact details for further information:
Thomas Rebel, t.rebel@student.utwente.nl

Results gameplay data

Start game without stimulating elements/ start game with stimulating elements	User 1	User 2	User 3	User 4
Amount levitate failures	1	4	4	2
Amount boost failures	8	7	7	9
How far panting average	0.7	1.1	0.7	1.7
Best single exhalation time (seconds)	4.4	7.3	8.6	10.3
Average abdomen	0.49	0.63	0.68	0.51
Average chest	0.79	0.67	0.56	0.83
2nd game without stimulating elements/ 2nd game with stimulating elements				
Amount levitate failures	3	2	1	1
Amount boost failures	4	3	3	6
How far panting	0.9	1.5	0.8	2.2
Best single exhalation time (seconds)	4	7.8	9.35	8.9
Average abdomen	0.43	0.58	0.59	0.48
Average chest	0.76	0.65	0.55	0.79

Results questionnaire

Full name	User 1	User 2	User 3	User 4
Why were you chosen for this experiment		other		
If you are pregnant, for how long are you pregnant? If you are a singer, for how long are singing with lessons?		-	Zero	
Did you ever do breathing exercises?	yes	yes	yes	yes
If yes, how much experience do you have?	I did it for a short time a while back	I did it for a short time a while back	I do it weekly	I did it for a short time a while back

Do you had problems doing breathing exercises?	No/I didn't do any breathing exercises in my life	No/I didn't do any breathing exercises in my life	No/I didn't do any breathing exercises in my life	I had trouble building a routine around them
If you have done breathing exercises, did you think you did them correctly?	Yes	I have no idea	Yes	Slightly yes
If you did breathing exercises, did you do it with a teacher or alone?	Alone	with a teacher	Alone	Both
Did you enjoy playing this phase of the game	Yes	Yes	No	Yes
Did you finish the level?	No	No	No	No
How difficult was it to finish the game? (1 is most difficult)	2	3	2	1
What made it difficult?	The panting was impossible, timing the breath was difficult	it felt like the sensors wheren't propperly responding to my breathing, the panting was unclear	There was a lag between the breathing and the game. Sometime I felt I would exhale by the game wouldn't register it	There was a lot to focus on, i struggled both with just the actual gameplay (im a bad gamer) as well as balancing it with breating
Did the breathing sensor worked? Was it responsive and reliable?	It did work but not always (I think)	unsure, it responded to inhaling and exhaling, but to panting it felt lacklustered	Somewhat, the inhaling was reliable and exhaling. But when changing very quicky it felt it wasn't reliable	Mostly
Is this a more enjoyable way to do breathing exercises compared to your alternative?	Yes	No	No	Yes
Do you feel the game helps with execution of the breathing exercise? (So if you do all the actions correctly, like breathing through your stomach)	Maybe	Yes	Yes	Maybe
Did you feel the game helps with the performance of the breathing exercise? (IE	Maybe	Yes	Yes	Maybe

breathing out longer than normal)				
Do you want to use this game instead of what you are doing now for breathing exercises	Yes	No	No	No
Any other comment?	I would use this in addition to what I already do. Also other breathing exercises are relaxed, this is not.	it could be a fun gimmick but for me to use it regularly would be a hassle to set it up instead of just starting with the exercise without any other tools	At least for me, my lungs would start to hurt from going to max. It doesn't feel that great when you can't last that amount of time. So, maybe some kind of improvement bar for long term progress could be a cool idea	The combination of breating and gameplay did not help, i breathe better when sitting upright but then needed to bend over to use the keyboard etc. at times ihad to breathe out so long i did not have any breath yet, but i dont know if that is a bad thing or not in regards to breathing exercises
Did you enjoy playing this phase of the game	Yes	Yes	No	Yes
Did you finish the level?	No	No	No	No
How difficult was it to finish the game? (1 is most difficult)	3	4	3	1
What made it difficult?	The timing and panting	the callibration wasn't working properly and the game desyncronised sometimes where I would keep on floating at start of when I was exhaling	Very difficult to control your breathing in a game setting. I couldn't last the 4 seconds because of the way I do my breathing exerises	Needing to use the space bar to jump was tricky, especially in combination with needing to suddenly focus on the breathing
Did the breathing sensor worked? Was it responsive and reliable?	yes but sometimes not	No	Somewhat, same as phase 1	No
Is this a more enjoyable way to do breathing exercises compared to your alternative?	Yes	Yes	No	Yes
Do you feel the game helps with execution of the breathing exercise? (So if	Maybe	Yes	Yes	No

you do all the actions correctly, like breathing through your stomach)				
Did you feel the game helps with the performance of the breathing exercise? (IE breathing out longer than normal)	Maybe	Yes	Yes	Maybe
Do you want to use this game instead of what you are doing now for breathing exercises	Maybe	Maybe	No	No
Which phase of the game helped better with the execution of the breathing exercises?	Phase 2	Phase 2	Phase 1	Both feel the same
Which phase of the game helped better with the performance of the breathing exercises?	Phase 2	Phase 2	Phase 1	Phase 1
Did some visual and/or audio game elements help with doing the breathing exersizes? If yes, which ones and how did they help you?	The green one for inhaling. The red one did not really help.	the lowering of the tone during the exhalation part helped, together with the bubble around the cube filling up and emptying helped	The circles that show how close you are to your max breath	Having an indicator if the game was recognizing inhale or exhale is nice, but i'm not sure if in phase one the circles were supposed to show how I was breathing, or how I was supposed to breathe, and it ended up distracting me more than not.
Any other comments?	Would use it next to my breathing exercise not instead.	it was buggy and sensor wasn't the most comfortable on the belly		I don't really know what i'm supposed to compare the breathing exercises too, like what it'd be without the game. Starting over everytime, it is difficult to get back into the rhythm of breathing exercizes and in my experience, finding a rhythm helps me most and usuallytakesa little bit. the interruptions do not help with that