# Contents

# 1 Report Introduction

The report consists of two projects; a purposive game production (PGP) consisting of 19 people and a sub-group consisting of six people. The production joined together to produce a game with a purpose, i.e. it has to have a meaning. The report starts out with the production part, which includes various research areas, such as game design and communication. Furthermore, the final game was created and tested several times, throughout this part of the report.

The second part focuses on the interest of the sub-group. The sub-group investigated whether virtual reality has an effect on communicative effort in a collaborative environment or not. Further research of communication has been studied in order to form a working definition to base the project on. The game created by the production was used during the testing phase and the findings are based upon a test with the game.

# Part I

# Purposive Game Production

# 2 Introduction

This report covers the process of a Purposive Game Production. Throughout this production a focus was set on VR in collaborative environments. Several aspects of VR was needed to be investigated, therefore the group investigated areas such as communication, game design, etc. Thus a problem statement was made:

***"Does virtual reality impact team-collaboration during problem-solving"***

The problem statement came from an inspiration to create a tool for teams to practice their collaboration, even when separated. To test this, a game was produced consisting of a series of puzzles where participants had to communicate for them to solve the puzzles and complete the game. To efficiently create such a product, each group member was given a specific role in the production, simulating a game development company. Data was gathered from the test in form of observations, screen recordings and through a questionnaire. In the end the test showed no significant difference between participants using VR, and non-VR.

# 3 Motivation

Virtual reality is currently under heavy development, and major steps are being taken in order to immerse players into virtual worlds. The most prominent tool used to do that are head mounted displays (HMD) which users will wear to be able to look around in a virtual world. It may prove interesting to explore having multiple people interacting in the same virtual world in real time while they are using HMDs.

The beginning of the project was primarily a process of generating ideas and brainstorming. Through this, it was determined that there was an interest within the PGP group regarding teamwork in virtual reality and how a virtual environment could be created that would facilitate cooperation and communication amongst the users. This was a process in which many different ideas and designs were explored and conceptualized, however only to generate a clearer image of what was desired for the project. Once the project's direction had been determined, more specific research areas could be investigated.

# 4 Analysis

Based on the motivation several research areas were to be covered in the analysis chapter. The obtained scientific knowledge allows for a list of design requirements to be formed, used to guide decision making and designs for the implementation phase.

## 4.1 Collaborative Audio-Visual Experiences

### Team Building

Before delving into collaboration in video games and realising how it can be utilised for team-work activities, understanding of basic ideas around team building must be acquired. Team building, according to Dyer (2007), is a term used to address the process of improving productivity and morale in the field of organisation development (OD). In short, it is supposed to increase trust, openness and cohesiveness, and open up channels of communication in a team. This can be, for example, making everyone understand goals and assignments or prevent a leader from dominating too much.

Team building methods originally grew from training groups in the 1940's and 1950s, and featured learning processes with unstructured groups, often strangers, who engaged in activities without a predetermined goal. After the activities, participants would analyse group structure and dynamics, and receive personal feedback on how they could improve their interaction. This process seemed to increase trust, openness and cohesiveness, however, the same could not directly be applicable in already-formed groups, in which roles and tasks were already defined and people familiar with one another (Dyer, 2007). Dyer (2007) further elaborates that simple team-building activities, such as asking open-ended questions, were adopted by organisations in order to make team members reflect upon how they work together.

Another fundamental principle of team building is that it is a process rather than a one-time event (Dyer, 2007). Some research also shows that over time, management interviews in conjunction with team-building activities can sustain positive changes in teams (Dyer, 2007).

If it is desired to create a VR experience, that would enhance teamwork-related skills in participants, extraction of possible design requirements from the presented information about teamwork-related skills is made.

1. The activity (i.e. VR experience) has to be non-predetermined, at least partially. This may refer to roles in the game/scenario, tasks or overall goal.
2. The information suggests that some sort of anonymity can be favourable if the participants are familiar (especially if they are in the same work group), in order to let them experience new roles in a more honest way.
3. Participants should be given tools to properly analyse the group structure and dynamics.
4. Participants should be given tools to properly give and receive feedback on their interaction styles.
5. If participants are in the same work group, they should be given tools to reflect upon and discuss their teamwork skills and roles.
6. Since team-building is often a long process, the experience we create has to be repeatable in some way.
7. Teams differ in sizes (Dyer, 2007); unless there is focus on a particular size, it would be preferable to have an activity which could be scaled.

The VR experience should, ideally, act as a training group activity, which would allow participants to explore group dynamics and structures, as well as get feedback on their interaction styles. It should also act as a sort of facilitator, which would enable participants to discuss and reflect upon themselves, their behaviour and role in the group. It is also of uttermost importance that the process is repeatable, since team-building is a process, rather than a one-time event. This is especially true since we would not be able to test our product throughout a long period of time, as ordinary sessions of team building; meaning, it would be necessary to somehow imitate the long process of team-building, or a few of its aspects.

### Communication in teamwork

Communication is an indispensable part of a group's functioning and dynamics, thus performance (Salas, Sims, & Burke, 2005). As the project aims to concern the field of teamwork, it is necessary to be aware of the different elements of communication, what may hinder it, what may enhance it, and how it can relate to a possible VR video game or experience. In order to reach and complete a task successfully, a group must "coordinate their efforts in a very detailed way" (Fussell et al., 1998). According to Fussell et al. (1998) there are two ways for a group to do so: through team design and communication.

Communication can be defined as "the exchange of information between two or more individuals irrespective of the medium" (Salas et al., 2005). Communication methods are varied and can refer to face-to-face meetings, email correspondences, file exchanges, etc. Communication tends to be more crucial in environments or situations in which the level of complexity increases e.g. emergency rooms in hospitals. This is because communication is not only the tool that distributes the information between group members, but also facilitates the continuous update of the group's shared mental model (Salas et al., 2005). This means that communication is preferable when making a team building exercise in VR. Errors in communication are not uncommon, and appear due to various reasons. Often individuals who receive the same message would interpret it differently due to their own perspectives e.g. cultural differences, characteristics, etc. Communication may also fail due to stressful situations or periods, or when individuals become too focused on their individual tasks, rather than the group. Another important phenomenon that may hinder communication is communication overload, which refers to scenarios in which participants are provided with too much information. This is especially harmful in stressful environments (Salas et al., 2005).

The VR experience we aim to create must consider these issues. If the game consists of a stressful scenario (e.g. defusing a bomb in 5 minutes), it is essential that participants are not expected to interpret many details (i.e. information), as it is most likely bound to cause communication overload. On the other hand, an experience that is not time-based and relies on the users' investigation and problem-solving skills, may allow for a detailed environment and, accordingly, more communication between participants.

Awad et al. (2005) demonstrates the possible significance of pre-task communication in regard to team performance. In their research there was a significant difference in team performance in the operating room of a hospital, when pre-operational briefings were performed (Awad et al., 2005). It may also be the case for a group of participants in a VR-based goal-oriented experience. This also coincides with the idea that face-to-face communication is better to use for highly equivocal tasks, while textual communication is better for less equivocal tasks. It is possible and important to conclude that the more interpretations a task has, the more preferred it is to use face-to-face communication. This also has the effect of establishing relational links faster than text which is important for efficient teamwork (Warkentin & Beranek, 1999). the reason face-to-face communication is preferable, is the information that is conveyed through facial expressions and body language. This is however this is not feasible in VR as current attempts to emulate these virtually have been unsuccessful (Tinwell, Grimshaw, Nabi, & Williams, 2011). However, future research could focus on techniques for adopting facial expressions and body language in VR.

Warkentin and Beranek (1999) had good results with a three-part process called virtual team communication (VTC), in which participants were trained on how to communicate via a computer.

The three parts of the VTC are as follows:

1. Information is given how to communicate via computers, as well as teamwork. Participants are informed that they would get to know each other and form relations, during the early part of the project.
2. Participants are informed of the drawbacks of communication via computer, e.g. information overload.
3. Participants are informed of the rules of conduct and "etiquette". They are informed of how they are best able to convey emotions and avoid confusion, with regards to the meaning of the information exchanged (Warkentin & Beranek, 1999).

Although the study was published in 1999, and thus is considered outdated it could still be of relevance, even if other consideration are to be made in regard to the VR environment. Following a procedure similar to that of VTC could allow for a more efficient starting point for a teamwork-based experience. Naturally, the procedure would have to be tailored to the communication relevant for VR.

#### Developing Efficient Teamwork

In order to know what measures should be considered in order to create good teamwork, it is relevant to first try and understand what would be beneficial to develop in a team. It seems that one of the most important concepts for a healthy functioning team is the idea of psychological safety. A. Edmondson (1999) popularised this term in her article *Psychological Safety and Learning Behaviour in Work Teams*, in which she refers to it as *"a shared belief held by members of a team that the team is safe for interpersonal risk taking"* (A. Edmondson, 1999).

One aspect of interpersonal risk taking is the fear of appearing incompetent. A good example of this, as the article describes, is a study of two different hospital environments - one where mistakes are tolerated and one where mistakes are kept secret, out of fear of the consequences resulting in higher risk for the patients (A. Edmondson, 1999). This example illustrates how, in a hierarchical team structure, there is a need of psychological safety.

Another serious problem that may occur within teams, is the fact that they can occur during

high pressure stress situations, much like we describe in regard to communication. In situations such as these, instead of functioning as a team, each member starts to focus only on themselves, and by that, reject other's suggestions (A. C. Edmondson & Nembhard, 2009). This shows how a specific team structure can function well under normal condition, but badly under extraordinary ones.

Going back to hierarchical team structures, in relation to the idea of psychological safety, it is important to introduce them to periods of high pressure, wherein decisions are required within a short time span. In these scenarios, there could be an increased challenge in encouraging interpersonal risk taking, since the authority might have to be more aggressive (i.e. quickly out ruling alternative ideas that under normal circumstances would have been evaluated). Thus, a team member may fear to appear as confrontational or disagreeable and would so choose not to give beneficial input. Throughout this project it would be prudent to keep the idea of psychological safety in mind since it is an aspect of team-work that can explain many occurring phenomena.

Trust is also a noteworthy aspect. It is defined as *"the expectation that others' future actions will be favorable to one's interests, such that one is willing to be vulnerable to those actions"* (A. Edmondson, 1999). The same article describes the importance of how the actions of a person are being interpreted by the team, when the most beneficial situation is when they are seen as helpful and not critical.

Shared understanding is another key aspect in a well-functioning virtual team (Miles & Hollenbeck, 2013). According to Miles and Hollenbeck (2013), one way to build a shared understanding is through shared experiences and team spirit. The longer a team has been working together, the easier it would be for them to create a shared understanding. Regarding virtual teams, it is especially difficult to develop shared group identity, due to the reduced amounts of the shared information between team members (Miles & Hollenbeck, 2013). Despite that, it could be that VR yields different conclusions, since the amount of shared information is larger and, if done correctly, more tangible than other virtual media.

Another important issue to note is that collaboration is also listed as a key requirement for developing shared understanding. This may mean that the VR experience we create could aim to use collaboration in order to develop shared understanding. If so, we would need to understand how this could possibly be done. The investigations that are described above makes it possible for us to try and formulate basic guidelines for exercises and tools that may be beneficial for a team, as well as an individual.

Firstly, creating situations with different stress levels may force the team to better test their

organisational structure. While this may create chaos, if conducted gradually it may result in useful training for flexibility and adjustment. Other exercises could be designed to train interpersonal risk taking, i.e. creating the feeling that each person's contribution to the team is helpful and not critical. This perhaps could be done by somehow putting emphasis on the contribution of specific roles or traits.

Team members should also be encouraged to express uncertainties that they might feel in regard to their task at hand (feeling of incompetence). Lastly, exercises that develop trust and shared understanding among team members are important, i.e. creating scenarios that could enhance these aspects.

Team exercises, in general, are needed in order to create shared understanding, trust or suitable team structures. Generally, it could be concluded that all of these aspects are directly connected to one another. One could easily assume that a group with meaningful shared understanding is more likely to develop strong trust between members, while a group that does not handle stress situations well may lack interpersonal risk taking.

### Communication methods in VR

It can be assumed that we understand what communication is, but in order to find out what types of communication exists, and how they should or could be used, the mediums will be looked further into, in order to elaborate and conclude on their use in a VR experience.

**Intrapersonal communication** Nonverbal auditory cues tell general information, and this goes for any sound, even ambient sounds. For example, a sound that is alerting might inform that the user should be alert, but not necessarily of what. Auditory cues have an advantage to visual cues since they do not require a specific amount of focus to be noticed, at the same time we do not need to beware of hearing a sound in order to react to it. From a computer usage perspective auditory cues have generally been used for notifications, alarms and such simple informative messages (Brewster, 2007).

While using HMDs for VR, auditory cues have the strength that they will not necessarily interfere with the visual overload that the user is already receiving. The less the user is distracted from the visual experience, the better an HMD experience.The problems that auditory cues might encounter are that they should not break immersion, they must be designed properly for their purpose and thus be diegetic or match the visual theme.

Visual cues are messages that are conveyed through a common recognition and association

with its visual representation. A visual cue can be practically anything, examples of such include signs, traffic lights, notification symbols, etc. This is possible for several reasons. Almost any item is associated with an action that it is used for. Humans have developed simple signs to have a common meaning, e.g. arrows, green is positive, red is negative, etc. It is however important to remember that the user has to learn the meaning of the visual cue first, and that a visual cue can often have a different meaning depending on its context. In this way visual cues work and gives information in the same way that auditory cues do, but on the contrary visual cues require that they are in your view to be noticed and to get the information from them (Brewster, 2007). Visual cues have been implemented in video games in several ways and have the advantage that a sign or logo often can be assigned to an action that it associates with. Therefore, the design possibilities for visual cues are endless.

**Non-interpersonal communication** Non-interpersonalization and recognition of the visual cue must be well thought through. Again considering immersion the visual cues should also be consistent with the theme and be somewhat incorporated in a 'diegetic way'. E.g. the team members could convey roles by wearing specific hats or other items. In comparison, the two non-interpersonal communication methods are quite different but share the fact that their source is not from a language but are simply understood from the human's perceptive abilities.

| | Pros | Cons |
|---|---|---|
| Visual | • Can give any message of a simple recognition or association.<br>• Can be put in context to something<br>• There it is the more detailed option | • Has to be within the user's view to convey its message.<br>• User has to "learn" the visual cues meaning |
| Auditory | • Can convey a message under any condition (as long as it can be heard)<br>• Avoid interference with HMD visual overload. | • Can only convey very simple and undetailed messages<br>• Has to fit the scenery in VR |

From this table it makes sense that the two cue types have different strengths and weakness and that if they are combined in use they strengthen each other and avoid some of the problems that they have alone. It is also stated that *"The combination of visual and auditory*

*feedback at the user interface is a powerful tool for interaction. In everyday life, these primary senses combine to give complementary information about the world"* (Brewster, 2007). In this project there should also be focus on how visual and auditory cues can be used to for communication between people in work groups, e.g. interpersonal communication which will be covered now.

**Interpersonal communication** Just like any other digital game, environments in virtual reality allows for computer-mediated communication with others. However, to make sure that the immersion of being in a virtual environment is not ruined, proper embodiment of not only the user's own avatar, but also others' are extremely important. In addition to the functions that are important in a single user virtual environment, there are multiple other functions that have to work properly in order to keep the user immersed. Capin, Noser, Thalmann, Pandzic, and Thalmann (1997) described these as the following:

- Perception – Allows the user to see if others are around.
- Localization – If there are others around, where are they?
- Identification – Recognizing the other user's avatar
- Visualization of others focus – What is their attention focused at?
- Visualization of others actions – What action is the other user currently performing?
- Social representation – To see what other users task or status is

Ultimately, these six points structure goals to achieve in order to perform proper communication that is not based on neither audio nor visual cues and still keeps the player immersed in the virtual environment they are placed in. Because these six criteria have to hold in order to be able to properly use body language in a video game, it is quite hard to implement. It should be possible with the right program and equipment, but the main issue would be to not break immersion and such a design in combination with the implementation would prove to be quite the task. The key here is to convey these points in an efficient and simple way with auditory and visual cues, also to avoid uncanniness which will certainly break immersion and also impact the desired outcome of more efficient communication.

**Verbal communication** is probably the most commonly used communication type and this is only natural when it is the communication type that involves most analytic channels to take into consideration when communicating. It involves analysing word choice, facial expression, body language, emphasis and expression of words and etc. An analogy written

about interpersonal communication describes it as a dance including skills, moves and counter moves, a set of rules but also a personal touch

Nowadays you can also communicate without sitting face-to-face, this means that you don't get the visual levels of the communication. This leaves out information that could regulate your understanding of the conversation. Communicating effectively requires a mutual understanding of each other and the conversation, and knowing how you should communicate effectively for the purpose of the conversation (Hartley, 2002).

**Textual communication** is basically a communication type similar to verbal communication where the communications form is based on a language. Textual communication does however once again remove another channel of the face-to-face communication being sound, but on the other side adds a new channel, being the your visual style of your text. This factor can once again cause confusion and misinterpretation of the conversation, and a text message should therefore be well thought through when written. Depending on how well you know the person(s) you are messaging with this is more or less important.

Berger (2013) states that a feature/advantage which textual communication has is that the respondent/receiver is not forced or expected to necessarily respond instantly and has the possibility to wait. Therefore textual communication might be better for formal conversations putting facts to the point and allowing the answers to be thought through carefully. For example, emails are often used to give larger amounts of information at the same time, because the mail is able to be written again and the different information can be put into sections to give the reader a better overview.

Therefore, if the design has to evolve around quick and efficient communication, textual communication can be precise but not if it has to be written quickly and the use of textual communication would probably not fit as a solution for the problem. However text can still be used for non-real time communication, e.g. leaving notes for team members with information or questions that is not immediately important.

## 4.2 Game Design

### Why make a game?

Lazzaro (2004) researches why people play games, and elaborates on four key elements that increase the player's emotion, without any deep story. The four key elements are listed as follows:

1. Hard fun
2. Easy fun
3. Altered states
4. The people factor

The first element we will look at is" Hard fun". This element is a reason why a majority of players play the games that they do, meaning that they play to overcome overwhelming obstacles, set new records, see some sort of progression and obtain rewards. In the progress of overcoming these obstacles, emotions such as frustration and personal triumph appears (Lazzaro, 2004).

"Easy fun" is the second key element. This is where there does not have to be a winning condition. Players who plays games with this key element are intrigued by incompleteness and exploration, This key element also allows for mystery and wonder, which can be very intense (Lazzaro, 2004).

The third key element is "altered states", which is where people report how the game makes them feel inside. This is somewhat used to "escape" from reality and change their mental state to a more exciting and relieving one (Lazzaro, 2004).

The last key element is the people factor. People use this mechanism to gain social experiences (Lazzaro, 2004). Players enjoy the emotions of amusement, social experiences of competition, teamwork, personal bonding, and the personal recognition that follows when playing together with other people (Lazzaro, 2004).

With all the knowledge gained above we now have a better understanding of what games can do and why people play them, but it only gives us vague guidelines as to how to design our product. At first glance it makes sense to aim for hard fun in the design of our game since this is what team building is all about; making people better at teamwork so they can overcome more complex challenges than they could on their own. Easy fun brings to mind the feeling of awe and personal wandering and exploration, which can be improved if shared with others, but it does not facilitate problem solving as much as hard fun. However a tint of easy fun may be exactly what the game needs to establish an environment where players feel safe and comfortable and may encourage personal risk taking.

Altered states also makes sense to use in a game for team building, since it makes people and excited and relieved, which also can afford personal risk taking, but also allow players to interact in a neutral environment and get out of their daily roles to explore new ones.

The personal factor is almost an embodiment of what team building is about and is a strong argument for using games for team building. Gamification has been shamed for only being simple operand conditioning, but true gamification of work would be to incorporate the four principle from above.

### Designing for hard fun and collaboration

When a person encounters a problem in both the real world and in video games, they try to solve this problem as easily as possible. This might seem reasonable at first, but when they encounter harder problems, they might have trouble using their acquired knowledge to solve the problem (Gee, 2005). In order to solve a new problem, basic knowledge that can be built upon is needed. It is quite similar to learning how to ride a bike - you often start with some form of assistance, e.g a parent holding the bike or training wheels; and as you get better it is possible to do ride independently of others, and even try new things with the bike.

Similarly, as players progress in games, they may also require help with difficult scenarios. For that, they are often presented with specific tools as a base for developing new knowledge that can assist them.

A good game, difficulty-wise, is a game that sends the player down a path, which forces them to face different scenarios and problems, which they can try or guess how to handle based on knowledge priorly-acquired in earlier stages of the game.

If the progress in the game is very linear with very little increase in difficulty, the game might be perceived as very easy. An example could be a game with a tutorial level, where the user would acquire knowledge, or a set of tools/abilities. They then proceed through the game with very little increase in difficulty, thus utilising the same abilities. This means that the player also uses the same knowledge repeatedly. Eventually, the player would perceive the game as easy and unrewarding, perhaps even boring.

Regarding the tutorial level, Gee (2005) refers to it as a fish tank. It is where the user is exposed to only some of the content of the game, only in order to introduce the player and without overwhelming them at first. It does not necessarily have to be a tutorial it can also be the first few levels. When designing the first levels of the game or when introducing new mechanics to the player, this tutorial approach should be kept in mind.

**Punishment types** While players progress in a game, an enjoyable moment is often experienced when completing a particularly hard task. Yet this is sometimes delayed by failure.

There are several ways to punish a player when they are unsuccessful and each way affects the player differently (Juul, 2009).

1. Game termination punishment: Game over
2. Life punishment: Loss of a life (or "re-try"), bringing the player closer to game termination
3. Energy punishment: Loss of energy, bringing the player closer to life punishment
4. Setback punishment: Having to replay parts of the game

These punishments make the player either play the game from the start again, or give them a minor setback. By using failure as a part of a game, it helps the enjoyment of the game even more. It also helps the players by thinking about new ways to solve the given problem, as they know that the solution they tried did not success. Several punishment types can be incorporated into the game, however they should be balanced so that they do not take away the enjoyment of playing the game.

**Making a problem so difficult that several players are needed** If we want to focus on a collaborative game that relies on team effort, it is important to create an experience with a suitable level of difficulty (whether increasing or otherwise). *"...apparently impossible tasks seem to be one of the strongest factors promoting player collaboration. After all, games are all about facing challenges and succeeding after a series of failures."* (Hämäläinen, Manninen, Järvelä, & Häkkinen, 2006).

If collaboration between players is required, the difficulty level has to be high enough, such that the players will not be able to solve the problem by themselves. So in conclusion there need be a number of challenges in the game that are so complex that all players are needed to figure out a solution. Also these challenges need to be designed such that each player's individual ability has to be utilised, so that is impossible to solve alone.

**Feedback on progression** *"Human beings are driven by feedback. We use feedback from our environment, from one another, and from ourselves to make sense of our world, to interpret social situations, to validate our sense of self, and to learn how to improve ourselves to make us better people."* (Higdon, 2007).

Feedback is a response to something that has been done by someone or something. In terms of human recognition of feedback, it is more relevant to the concept of learning from said feedback in order to make a better judgement of a similar situation later on (Higdon, 2007).

It is also regarded as such in order to create a personal motivation. If the feedback is positive, the person will strive to receive more of the same type of feedback in the future.

Rather than following a universal rule, feedback is received differently from person to person. Through the designed experiment "The Differential Outcomes Effect", this is called differential reinforcement. Differential reinforcement is administered with three things in mind:

*Extinction:* The reinforcement of the feedback is eliminated by the person.

*Non-contingent reinforcement:* the reinforcement is independent of the feedback provided for the person.

*Differential reinforcement of the other:* the reinforcement is provided despite the lack of feedback.

Under the circumstance that these three principles are not going on, it can be believed that the feedback provided will allow for a reinforcement that has an actual effect on the person (Higdon, 2007).

The most common explanation for what feedback will do to a person follows the outcomes theory (otherwise known as expectancies). Within the outcomes theory, there are two associative relationships:

*The stimulus-outcome:* when the response to feedback is associated directly with said feedback.

*The expectancy response relationship:* when the person's expectations to their response to the feedback is compared with the actual result of the response (Higdon, 2007).

So in order to predetermine the results of specific feedback, it will be important to note; will the feedback involve extinction, non-contingent reinforcement, or differential reinforcement of the other, and can the response to the feedback be comparable to the stimulus-outcome, or the expectancy response relationship? Seeing as response to feedback is largely individual, experimentation will become a relevant subject during the design of the game's feedback towards the user.

Also what feedback should be considered from a game design perspective? Negative feedback can be given with the mentioned punishment types and positive can be given also be reversing the punishment, e.g. give boosts in form of lives, time etc. all depending on the mechanics. Of course progression and overcoming the challenges is in itself positive feedback as well. Also audiovisual cues can be used to give feedback, e.g. turning a light green when something is done correctly and red when it is done incorrectly. There are many established norms for

giving feedback like the red/green light example just mentioned.

### Immersive virtual reality

**Visuals in virtual reality** Seibert (2014) hypothesized that VR HMD (Head Mounted Display)s would increase the amount of spatial presence a user feels compared to a user on a normal computer monitor or television. Where spatial presence can be translated directly to spatial immersion or the feeling of being in a space (Weibel & Wissmath, 2011). After testing, Seibert (2014) got results that supported the hypothesis. This shows that using HMDs will in fact grant a user more immersion.

Even though HMDs have potential for creating more immersive experiences , it still has its flaws which might ruin the immersion factor for some people. The most common are (but may not be limited to): nausea and distance underestimation. These topics should be considered when designing the visuals for the game.

**Nausea** is created when the visual input between the user and the users vestibular sense of motion is presented in a mismatch. The main reason to this in VR is hardware lag (Hale & Stanney, 2014). Lag is the time it takes for the hardware to acquire the input from such things as the sensors on the HMDs (Oculus, 2015). What is wanted is a small amount of lag as this will create the most lifelike experience. Lowering the graphical requirements and keeping the details at a minimum might reduce the lag on some hardware (Hale & Stanney, 2014).

**Distance underestimation** is also a common problem within a virtual environment when using HMDs. This is mainly due to the HMDs reduced field of vision, the extra weight on one's face with the HMD on and some visual depth cues (Hale & Stanney, 2014). Adding sound cues to objects might help link a user to its distance within the virtual environment.

So what we can say specifically about visuals in virtual is that they should not require so many resources that they will exceed the technical specifications of the machine which that has to run the game and thereby causing lag. So the graphics should be more or less minimalistic in terms of details and just for safe measure. Actually everything should be tailored to keep technical requirements at a minimum.

Linking objects with sound cues should also be considered since it may also help reduce distance underestimation. We will go more into sound and audio in the following subchapter.

**Audio in virtual reality** Nordahl and Nilsson (2014) use the term presence as a key factor in immersive virtual reality (Nordahl & Nilsson, 2014). They explain the term as the sensation of "being there". This refers to the way a video game can convey realism to a player. They describe auditory feedback in a virtual reality setting to be just as important as visual feedback, to create the sensation of presence.

When trying to simulate a realistic soundscape and thereby creating presence, headphones are the chosen medium, since these can represent binaural hearing (Nordahl & Nilsson, 2014). Binaural hearing is when humans or animals can distinct between sound by hearing differences between each ear (Litovsky, 2008). This allows for the determination of distance and origin of a sound source.

The most important aspect of binaural systems is the possibility of creating spatialised sound, which refers to sound that can be perceived in three dimensions, in comparison to traditional stereo sound which only moves across the x-axis.

Spatialised sounds in a game allows players to perceive the distance and direction of sound originating from sound sources all around them as it would in real life, by filtering sound signals so that they resemble an acoustic signal that has interacted with the torso, head and outer ears of the listener (Nordahl & Nilsson, 2014). This is achieved through *head-related transfer functions* (HRTFs) which is the way an audio signal sounds in our head.

Nordahl and Nilsson (2014) refers to a study created by Larsson, Västjäll, and Kleiner in 2008, in which they conclude that binaural simulation and spatialised sounds are superior compared to stereo sound when it comes to the sensation of presence (Nordahl & Nilsson, 2014).

So in conclusion the sound should be implemented such that they are spatialised, also headphones or earplugs should be used to allow for the sounds to be perceived as spatialised. Also the sound should be coherent with the visuals of the game.

**Music in virtual reality** Since there next to no studies done in the field of music in VR, it is necessary to use studies from normal video games as a point of origin. A 2006 study, where test participants were asked to play 3 different parts of a videogame where differently themed musical tracks were playing, found that the background music in particular helped convey the narrative changes in the story, the emotional status of characters or scenes and the overall theme of the game to the player (Lipscomb & Zehnder, 2004).

A different study which was carried out in 2015, found that immersion was improved signifi-

cantly for test participants who had music set to "on" while playing a game. Reaction times and a feeling of distorted times were reported to be significantly affected too (Zhang & Fu, 2015). The type of music made a difference too, as experiment results from an earlier study proved a stated hypothesis, by indicating a coherence between the players aggression in the game and the aggressiveness of the music that was playing at the time. If what the player was doing matched the background music's level of intensity (e.g. a violent video game with high intensity background music), the player would perform better in the game and behave less aggressively after playing the game than if the music intensity and the game intensity were not properly matched to fit together (Zhang & Gao, 2014).

The studies imply that music can be used to increase immersion, convey emotions of characters, narrative changes and the overall and manipulate the behaviour of the players.

However it is important to mention that these aspects of music in video games not necessarily apply to video games in virtual reality, but it could be interesting to research whether or not this is the case and if music has to be implemented in virtual reality in new ways. It is also interesting to see if non-diegetic music reduces the feeling of presence in VR since the prime role of audio is to convey realism.

## 4.3 State of the Art

This chapter will seek to explore currently used methods to design communication and collaboration within multiplayer games, as well as some of the newest multiplayer productions made for VR. What concepts are current VR and multiplayer productions using and why? How can these concepts be applied to this project?

#### ZombiU

ZombiU is a game originally made for the Nintendo Wii U console, released in 2012. In the multiplayer mode one player works as the Zombie-master and controls spawning of the zombies with the gamepad. The second player's goal is to stay alive and use the traditional controller (see figure 1).

ZombiU uses a special type of gameplay, which Nintendo themselves call asynchronous gameplay. They describe this as being a different experience of a game from each player, within the same game. Hereby referring to the use of two different controllers, the different screen views and the different goals.

Figure 1: Top view for the gamepad player and first person view for the traditional-control player (Nintendo, 2015)

This type of multiplayer game makes each player important for the game, as the abilities and screen visions are different. ZombiU is a competitive game, but the features can be transferred into a collaborating situation since it could force the players to work together by having different views and abilities. This will be relevant in the situation of this project, as collaboration is the main focus. For example, it could be relevant to give each player different views, thereby making communication between players a necessity. In the same way it would be possible to divide features among players. Another instance in utilising the concept of cooperating players presented with different views is a game called "Keep Talking and Nobody Explodes". In this game two players have to work together in order to disarm a bomb however only one of them can see the bomb and are allowed to do so. The other player is only presented with the instructions and has to convey that to the player with the bomb.

**SteamCrew**

Multiplayer in VR is a barely touched subject, however, few projects have been, or are being, made with multiplayer VR in mind. The winning entry of, a game jam named Oculus VR Jam 2015, are titled SteamCrew VR. It aims to create a cooperative VR experience between two players (Kerbeci, 2015).

On the information given at the entry page for the game project, written by one of the developers of the game, it is stated that their goal is to "offer a strong sense of camaraderie between two physically present players, setting a new level in terms of game presence" (Kerbeci, 2015), meaning that their goals are similar to those of this project.

While it is stated that SteamCrew is an experiment on social interactions within VR, it lacks empirical data, to back up that their design decisions are actually valid. Nevertheless, since next to no current research is available within the subject, their considerations and reasoning for design choices will be taken into account in this analysis.

A way to split information between two players, would be to induce communication and cooperation between them.

Another design decision on SteamCrew was to create gameplay sessions that will usually last from 3 to 10 minutes. While it is not stated why, it is most likely for the players not to be forced to play for longer durations, as VR equipment is known to be exhausting to use for longer periods of time.

So in conclusion there are two design requirements which can be derived from the state of the art. Firstly, the game should be designed to be played in relatively short sessions. The amount of time in VR depends a lot on how the game is designed, the equipment, who plays the game and if they are used to HMDs, etc. Secondly, splitting information or abilities seems between players seems like a common practice for developing collaborative experiences in games and therefore it could be used as a safe design choice.

## 4.4 Measuring Success and Collaboration in Video Games

When creating a video game that should enhance collaboration between players, it is crucial to find the factors that could measure whether collaboration is present and to which extent. There are different variables which could measure the success of in a video game Ducheneaut, Yee, Nickell, and Moore (2006).

**Time (how fast?):** How much time do the players use on each task and how much time does it take to complete all?

**Amount (how many?):** What is the total amount of completed tasks?

**Errors (how well?):** What is the amount of errors in each task and total amount of errors before completing the game, if the game is completed?

**Communication (how many words, which kind of words?):** What is the level of teamwork, measured in quantity and quality of the communication (verbal & nonverbal) between the players.

In-game measurements of player collaborations can be made using different variables. Ducheneaut et al. (2006) researched the massive multiplayer online (MMO) game World of Warcraft (WoW), with a focus on three aspects: (1) the amount of time spent on playing, (2) how the players function in groups and (3) which type of social groups players involve themselves in?

WoW is a good example of a collaborative game that utilises difficulty levels in order to encourage cooperation between players, e.g. some quests are too difficult to complete alone (Ducheneaut et al., 2006). Furthermore, WoW gives the players the possibility to have different abilities in the game, which in itself gives a good reason for the players to work together (i.e. different players can be useful for different tasks or strategies).

It is possible to receive data both in-game while the players are playing and outside the game, where you could use, interviews, surveys or observations.

One indicator of a successful video game, is the time a player spends in order to reach a new level (see figure 2). If the game does not consist of a level function, but is task-based, there could be made a measurement of the time the player uses to complete one task.

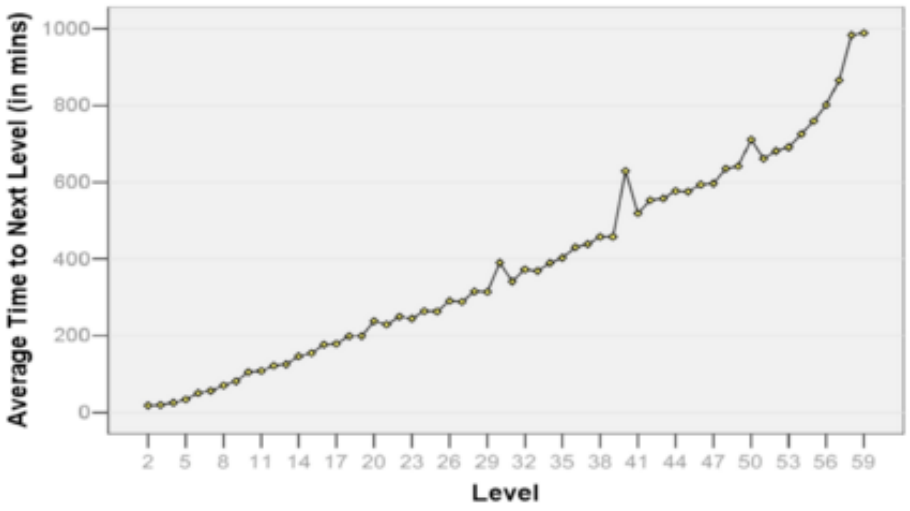



Figure 2: Average time required to reach a level (Ducheneaut et al., 2006)

## 4.5 Communication and Collaboration

When examining collaboration in video games, a focus could be put on interaction; how much interaction exists between the players? Which types of interaction are present?

It is important to define the type of communication, as well as the cultural and social aspects of the players, since different cultures and people communicate and perceive things differently (Hämäläinen et al., 2006).

In a virtual world, digital avatars enable the players to communicate with each other in a game environment (Hämäläinen et al., 2006). The virtual game world supports non-verbal communication, which could create unique interaction and collaboration, e.g. one can use the avatar to guide other players to certain locations or objects instead of having to lead them around by using verbal communication, which could depend on issues such as language barriers or the nature of the task. How a player uses their avatar for interaction and collaboration could be recorded, coded and evaluated both quantitatively and qualitatively. Examples of this could be:

- What do players share with one another e.g goals/tasks, resources, acquired knowledge.
- How does a group of players that works together perform e.g. using strategies, orientation, negotiation, coordination or split roles.
- Communication types and amount of frequencies and content e.g. voice, avatar gestures (Hämäläinen et al., 2006).

After the players are finished playing the game, an interview could be conducted. This would provide knowledge regarding the player's own perception of the gaming experience, which could include collaboration. If needed, the data collected from the game could be compared to the player's experience (note that the players can feel like they are collaborating without actually doing so) (Hämäläinen et al., 2006).

We could also measure how open the players are regarding their mistakes and compare it with their actual ability to solve tasks. Their research also found that the persons most open about their mistakes were the most collaborative ones (Hämäläinen et al., 2006).

You can also test if the players give positive or negative feedback to each other and how it affects the players and overall gameplay.

## 4.6 Design Requirements

In the analysis both the subject of human interaction and cooperation were covered as well as the more technical aspects of virtual reality. Both are equally important as this project is trying to achieve a combination of the two: team building and VR. With regards to team building, one of the best ways to make people better at cooperation is to increase both their need and capability to communicate. The players should have a common goal as well as a need to work together to achieve it. Ideally this goal should scale according to the team size and the tools to accomplish it should be introduced well. The exercises should also be designed so that it is possible for everybody to understand the common goal.

The need to communicate could be increased through adding ways for players to express themselves. This might seem obvious but it is important to remember that by only using, for instance, voice chat, a lot of channels of communication are removed compared to face-to-face communication.

The player has to be presented in a way that insures some degree of anonymity as well as insuring that it is not possible for one person to solve the exercises alone. Furthermore the exercises should be designed so that it is not possible for one person to take a leader-role.

With regards to virtual reality it is important that technical aspects are considered such that the equipment and its features are used properly. Both the auditive and the visual aspects of virtual reality were studied in order to determine what design decisions were going to be made. With regards to visuals, it is important to keep a steady frame rate in order to reduce nausea. With regards to audio it is preferred that the objects has sound cues to help with localisation within the environment they are placed in. The sounds in the game should primarily be diegetic to increase immersion. As for both audio and visual content, care should be taken not to overload the user as every perceptive input in VR is enhanced.

## 4.7 Problem Statement

Based on the research done in this analysis chapter the following problem statement was derived:

***"Does virtual reality impact team-collaboration during problem-solving"***

# 5 Documentation

## 5.1 The concept of Purposive Game Production

This Purposive Game Production is the second generation of an experimental project format focusing on simulating real world game production circumstances while still attaining academic relevance. It does this by enabling larger project group to be formed and divides the team members into specific production roles mimicking those of a real game developing studio.

The format puts more emphasis on the design and implementation phase of the project, amounting in a more ambitious end product and more specialised learning outcomes of the individual team members. The end product was to be presented to a jury of real world industry game developers.

## 5.2 Preliminaries for this format set by the CEOs

The project would be "owned" and supervised by several CEOs who were to overlook the process and development of the game.

As part of the name of the project format states, the game needed to have a purpose. This is the part which really enables the project to be labelled as academia, as the game must be centred around an aspect which needs to be academically tested and contribute to an academic field.

A couple of strict guidelines were set by the CEOs. The game must adhere to the following rules:

- Have no violence
- Have no pornographic content
- Must have purpose
- Must utilize virtual reality (which entails the use of a head-mounted display)

## 5.3 Team structure

A rough team layout of team size, structure and roles was laid out by our to be CEOs at the point of group forming facilitation. An estimate was around 21 people and a handful of different roles with embedded hierarchical structure.

Our own final interpretation of the team layout fitted to our needs can be seen on figure 3 below.

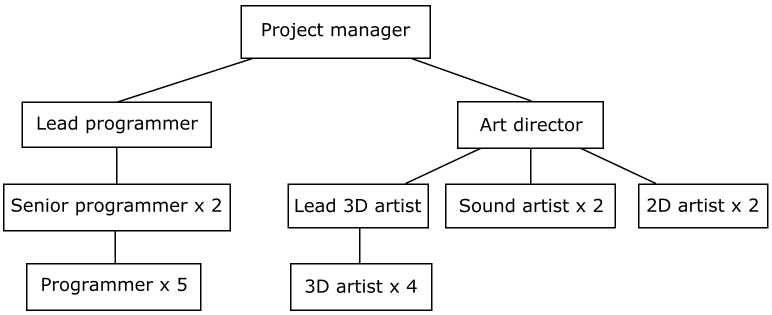



Figure 3: Diagram showing the exact size of the project team and its structure.

## 5.4 Project management documentation

To easily keep a medium sized team (around 20+ in our case) informed and under control takes some kind of system. For our project we chose to utilise a range of communication methods and platforms for this job, such as the agile software development methodology with its sprints and and daily scrum meetings. Platforms for communication were Facebook, Google Drive and Slack. Each platform served its own purpose in regards to the level and type of communication.

### Organisation/sharing of information

Facebook was used for practical information and sudden announcements (Facebook's messenger system for simple 1-to-1 communication and noticing of sudden warnings of absence).

Google Drive were used for file sharing such as assets and work documents for e.g. analysis, management, evaluations etc.

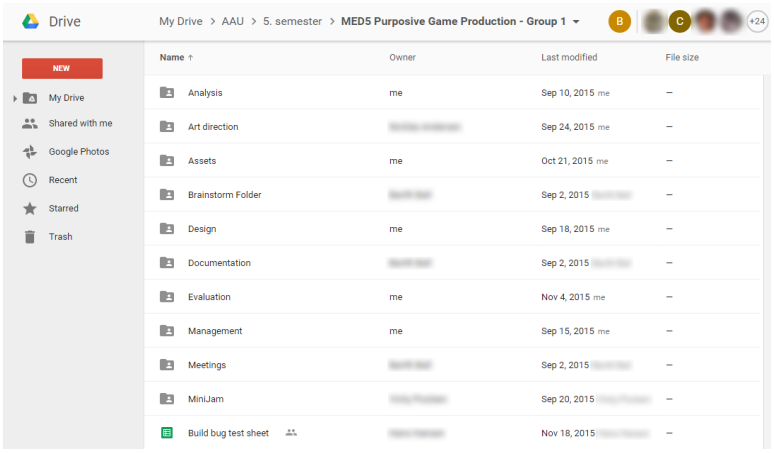



Figure 4: Screenshot of the main Google Drive directory.

Slack was meant to be used as the main communication channel when working, linkage sharing of files (using a Google Drive plugin) e.g. between art department and programming department. The actual usage of Slack was not as prominent as planned as the team members were working in the same physical room, making direct contact much faster and easier, but also more interrupting.

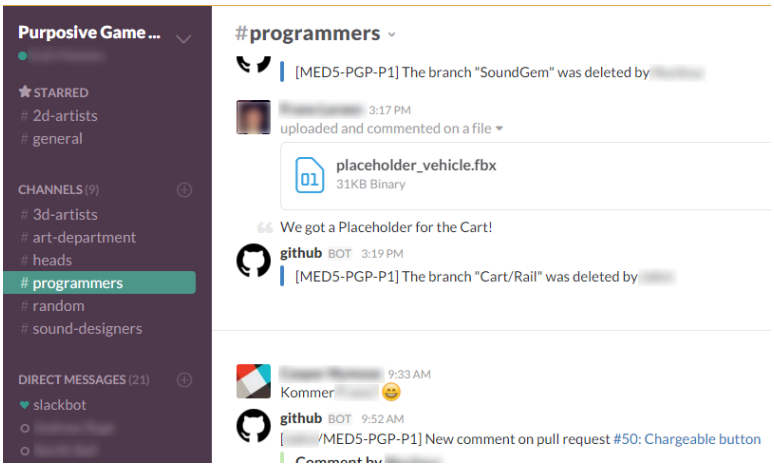



Figure 5: Screenshot from the programmers sub channel of our Slack forum.

### Agile software development

As part of the agile software development model the development process was split up into several so called "sprints". A sprint constitutes an internal process of several steps. A plan is made for the content that needs to be finished for the sprint, this is called a backlog. Each day a short orientational meeting is held, called a scrum.

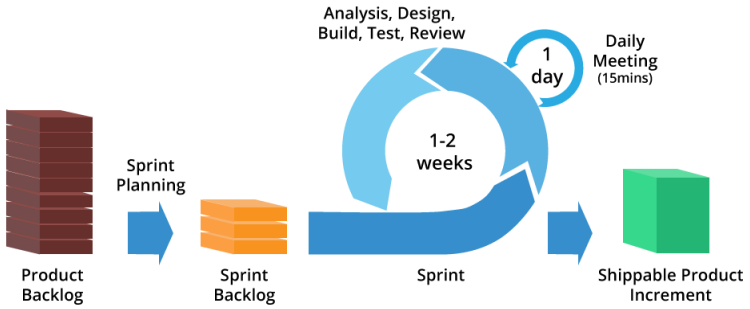



Figure 6: Visualization of the Agile software development process (Mehetrey, 2015).

Our usage of the agile model ended up constituting six sprints (of either one or two weeks length). Daily scrum meetings were held internally in the art department and the programming department.

### Backlog

The backlog is a list of all the tasks which should be completed for the current sprint. Having a backlog of clear tasks which are clearly assigned helps the team organize who is doing what task, and that tasks gets completed in the right order. We chose to organize our backlog with focus on priorities, titles, descriptions, estimated completion time, the assigned person as well as some kind of progress status tied to it.

| | A | B | C | D | E | F | G | H | I | J |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | SPRINT 4 (alpha phase 3rd) | | | | | | | | | |
| 2 | period: 07/10 - 20/10 - 2015 | | | | | | | | | |
| 3 | | | | | | | | | Didn't start | |
| 4 | **Greater goal of the sprint:** | | | | | | | | | |
| 5 | Fundamental game mechanics (interactives) | | | | | | | | Started | |
| 6 | Assets for building a basic level | | | | | | | | | |
| 7 | **2D tasks:** | | | | | | | | Help | |
| 8 | | | | | | | | | | |
| 9 | | **Hours per. person** | **Available hours this sprint:** | | | | | | Done | |
| 10 | | 40 | 120 | | | **Hours total:** | | | | |
| 11 | | | | | | 288 | | | | |
| 12 | | **Task:** | | **Description:** | | | **Work type:** | **Time in hours:** | **Assigned:** | **Status:** |
| 13 | | | | | | | | | | |
| 14 | | Avatar 1 (player) - Thumbnails | | | | | concept | 2 | | |
| 15 | | Avatar 1 (player) - Full body concept | | | | | concept | 4 | | |
| 16 | | Avatar 1 (player) - Shading | | | | | finalize | 4 | | |
| 17 | | Avatar 1 (player) - Color | | | | | finalize | 4 | | |
| 18 | | Tracks - (curved and sloped section) | | | | | concept | 1 | | |
| 19 | | Tracks - (curved and sloped section) | | | | | finalize | 2 | | |
| 20 | | Sound emitting crystal | | | | | Concept | 2 | | |
| 21 | | Sound emitting crystal | | | | | Finalize | 2 | | |
| 22 | | Environment lvl2 - Thumbnails | | https://docs.google.com/document/d/1tYr7qjcyc4Jn | | | concept | 3 | | |

Figure 7: Screenshot of an example of a backlog, this example shows the system for 2D artists.

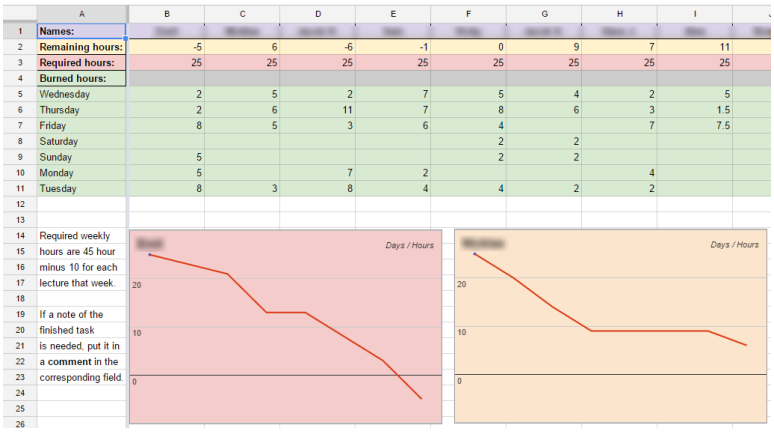
### Burndown charts

To accompany our short term development cycles and scrum meetings we used burndown charts to keep track of the team members' effort. Burndown chart is an organizational method to keep track of individuals effort by dividing tasks into working hours and track

members' completion of given tasks. When a member completes a task, the member gets the amount of hours it was estimated to take to complete subtracted from the total hours that the member is expected to work this sprint.

To keep an overview of bigger tasks they are broken up into smaller parts with a maximum estimated completion time of 4 hours.

The goal is for every member to have hit zero remaining expected working hours at the end of each sprint. This proves to be hard to do on a steady basis as given tasks can take longer than estimated (especially with an inexperienced team).



| | A | B | C | D | E | F | G | H | I |
|---|---|---|---|---|---|---|---|---|---|
| 1 | **Names:** | [redacted] | | | | | | | |
| 2 | **Remaining hours:** | -5 | 6 | -6 | -1 | 0 | 9 | 7 | 11 |
| 3 | **Required hours:** | 25 | 25 | 25 | 25 | 25 | 25 | 25 | 25 |
| 4 | **Burned hours:** | | | | | | | | |
| 5 | Wednesday | 2 | 5 | 2 | 7 | 5 | 4 | 2 | 5 |
| 6 | Thursday | 2 | 6 | 11 | 7 | 8 | 6 | 3 | 1.5 |
| 7 | Friday | 8 | 5 | 3 | 6 | 4 | | 7 | 7.5 |
| 8 | Saturday | | | | | 2 | 2 | | |
| 9 | Sunday | 5 | | | | 2 | 2 | | |
| 10 | Monday | 5 | | 7 | 2 | | | 4 | |
| 11 | Tuesday | 8 | 3 | 8 | 4 | 4 | 2 | 2 | |
| 12 | | | | | | | | | |
| 13 | | | | | | | | | |
| 14 | Required weekly | | | | | | | | |
| 15 | hours are 45 hour | | | | | | | | |
| 16 | minus 10 for each | | | | | | | | |
| 17 | lecture that week. | | | | | | | | |
| 18 | | | | | | | | | |
| 19 | If a note of the | | | | | | | | |
| 20 | finished task | | | | | | | | |
| 21 | is needed, put it in | | | | | | | | |
| 22 | a **comment** in the | | | | | | | | |
| 23 | corresponding field. | | | | | | | | |
| 24 | | | | | | | | | |
| 25 | | | | | | | | | |
| 26 | | | | | | | | | |



Figure 8: Screenshot of parts of a burndown chart showing both slots in which to insert completed working hours as well as the resulting graph for the current sprint.

### Timeline

To steer the overall development process we were given a timeline by our CEOs.The timeline is a top level overview of the entire project process. It divides the project into phases which were further broken into several lesser parts (sprints). The time allocation we set for every sprint varied from what is depicted below.

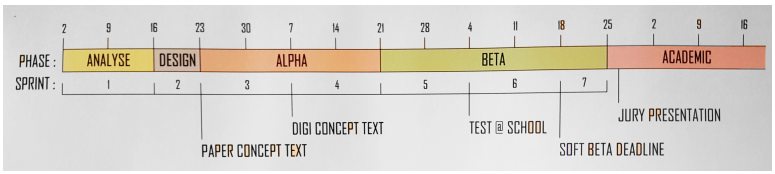




Figure 9: Picture of our top level timeline provided by our CEOs.

### CEO meetings

Clear communication with the project's CEOs was important and orientation meetings were held with them on a weekly basis. On these meetings the project's status would be presented and discussed with the goal of giving the CEOs insight on the progress. It also enabled the different departments to present their work for each other in a more constructive and organized manner (better have one big showcase for everyone than single persons bothering each other about their work all the time).

### Attendance

One of the jobs of the project manager was to keep track on the team members working efforts, and eventual confrontation if work quota was not met. Apart from the burndown charts, team members' attendance was also kept track of to predict possible future problems with the members.

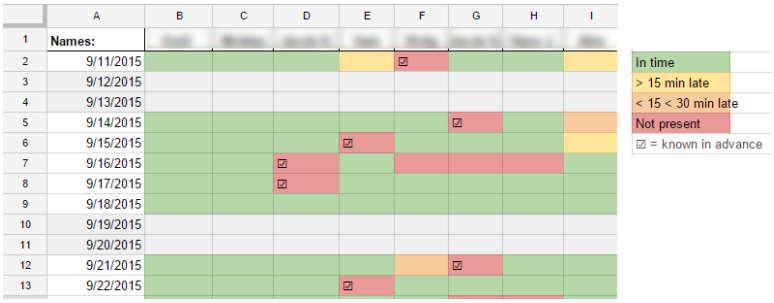



Figure 10: Screenshot of a part of the attendance list showing mixed attendance.

## 5.5 Art department

The art department consisted of 4 different jobs:

- The Art Director
- 2D artists
- 3D artists
- Sound designers

Each job had a part in a creative pipeline, whose purpose was to create visual and auditory content for the game.

### The Art Director

The Art Director was mostly involved in concept development, moodboards, guidance for the members of the art department, quality assurance for the art, insuring collaboration between 2d-, 3d artists & the sound designers, as well as the seniors and lead from the programming department and promotional material. At times the 2D artists were in need of additional manpower which led the art director to do some 2D concept creation as well.

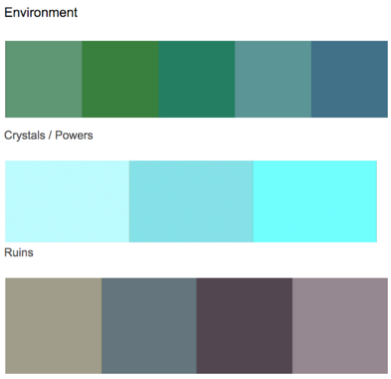



Figure 11: Colour schemes - for keeping consistency

Figure 12: Moodboard. Pictures from upper left to upper right taken from: Firewatch, Disney's Atlantis, No Man's Sky. Pictures from lower left to lower right taken from: God of War, Uncharted.

At the beginning of the project, after the initial design was created by the whole team, concept development began. This meant preparing how the game should look, feel, sound and which objects were needed to accomplish that. To illustrate this, **moodboards** ( 12) and **colour schemes** ( 11) were constructed.

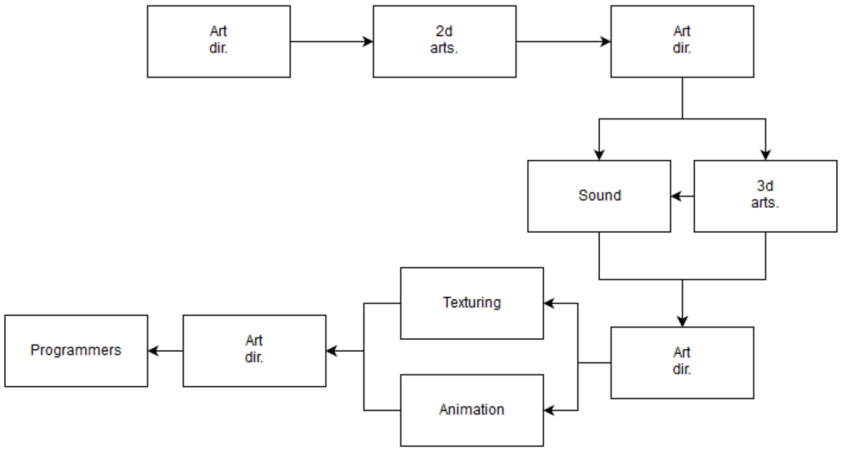



Figure 13: Pipeline for departments

**The position of the art director** In the beginning of each sprint, the art director set up the plan and work load for the 2d artists, as this division is the fundamental for the art department. This was defined as soon as possible, so the 2d artists could begin developing concept for the needed objects in the game. This was handed over to the 3d artists, so they could begin modelling. At the same time the sound designers got the concept, so they could start creating the audio for it. They also got to see the model (and animation if the model had one) so they could match the audio more perfectly. A 3D artist was in charge of the objects that needed the animation, and while he did the animation, textures were drawn for both the objects that did not have animations and the ones who did. At this point the conceptual sound had been created and the sound designers could begin creating the real thing and match it to the frames of an animated object if needed. In between all of these exchanges the art director kept overseeing the development of each piece to keep quality and consistency. The pipeline can be seen in figure 13 above.

Due to complications with the Project Manager leaving the project early and having a 2d artist replacing mostly that position, the art director joined the 2d division at times to create concept and texture, to lighten the heavy workload that this led to. In the late part of the production, the art director was also responsible for how the name, logo and posters should look for the game. This was done in collaboration with the rest of the team, in means of making it look the best it could.

**Forming a bridge** The art directors task was also to be a bridge between the different divisions in the art department, but also what the art department produced and how the programmers should use it. There were some communication problems, as a level designer

was assigned in the middle of project, and all the models that had been made before this change, had not been accounted for before late in the process.

As there were not assigned a Game Director/Designer, the art director in collaboration with the dedicated programmer (later level designer and others) was to make sure that the feedback, the users felt, was designed optimally.

### 2D Artists

The 2D artists have multiple entry points in the process of making an asset. According to our work structure, a 2D artist makes a visual concept for the 3D artists to later recreate in three dimensions and give life by animating. The 2D artist then re-entry the process to apply texture for the surface of the 3D models. Other jobs constitutes of visual effects, user interface and promotional material such as posters etc.

**2D Concepting** The overall theme of the game was decided to be "cave exploration". This is partly due to the main goal of utilising virtual reality, which has the possibility of making a game about exploration feel more real. Furthermore the visual appearance of the game was inspired by the theme of Atlantis as it is presented in the Disney's movie with the same name.

All of the concept drawings were made with respect to the overall visual theme and mood-board. For convenience and faster workflow, 2D content was mainly produced using Adobe Photoshop. The objects to concept were generally listed in a document (backlog), and distributed among the 2D artists.

The general aesthetic in the game were meant to be rough and simple. The main way this was done was by making low detailed props that were mostly devoid of patterns, and if patterns were to be had, it should only consist of very simple shapes. An important aspect of this theme is that the edges of several objects are meant to be rough, and surfaces are meant to be large.

One of the main styles of concept art is the silhouette style. It is meant to focus on creating shape-variety between the different props, and was implemented in the majority of the concept work.

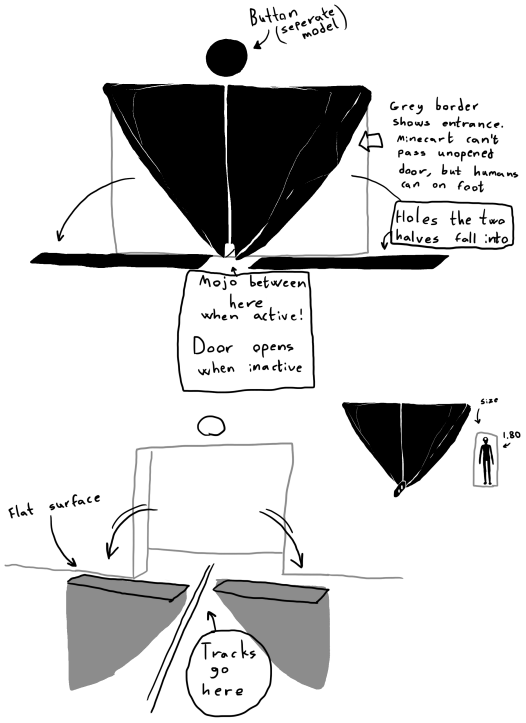



Figure 14: Example of a silhouette-styled concept artwork.

A secondary form of style of the concept art is the rough sketch. Especially rough patterns can be a focus here, as the silhouette requires precise white lines to showcase distinct patterns.

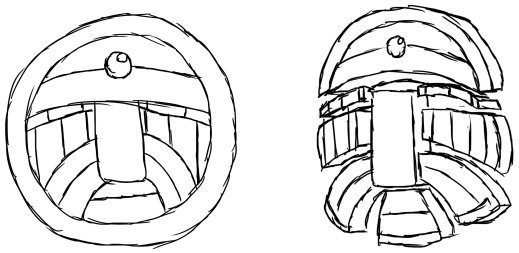


Figure 15: Example of a sketch-styled concept artwork.

The third type of concept art is the finalised/painted concept art. These depict how an object or area should look in-game, and puts in the most specific details that make the subject's matter it's own thing. These details can be anything ranging from stone cracks to plant vines running across the object.

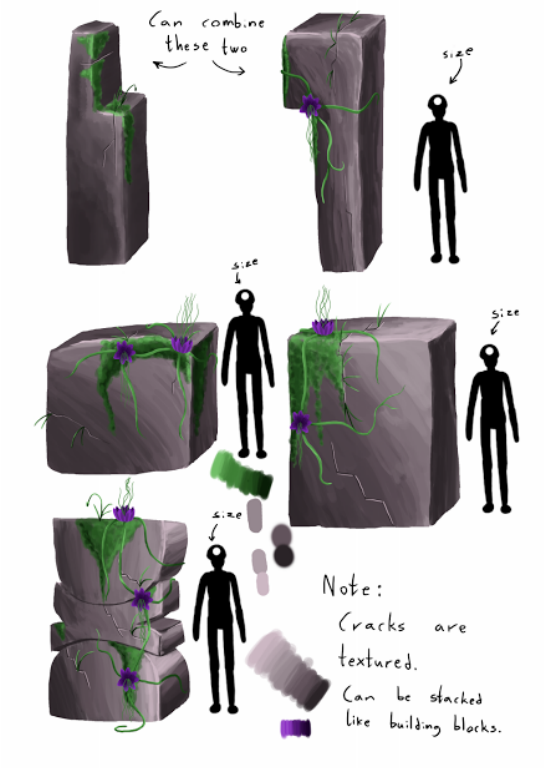



Figure 16: Example of a painted concept artwork.

Most of the world building objects is made of stone as to indicate the place as being an ancient society. The only other material in the world is crystal. All concept drawings was based on this rule. The work flow was to firstly make a lot of different silhouette-like concepts, to choose from. After the most appropriate design was chosen, it was expanded. This process is shown below in the case of the mirror-design, which is the interactive object of the second puzzle.

In the background of the cave, stone-buildings were placed as visual aesthetics. The same thumb-nail iterations are used for this process. The concept shown below is houses chopped out of a cave wall. This type of background architectures is made to fill visual space and for building up the player's expression of the world.

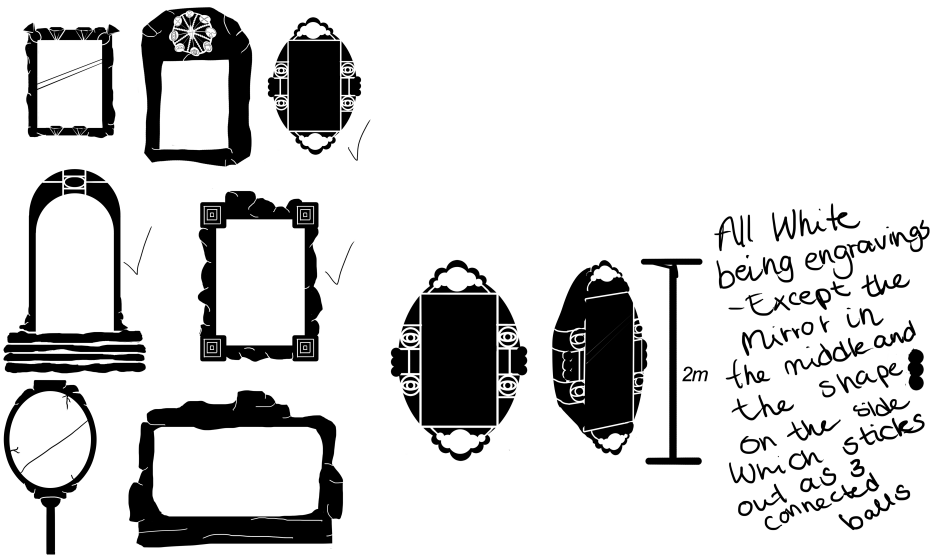



Figure 17: Concept art of mirrors. The silhouette style was used to make several designs fast and simple.

Figure 18: Concept art of houses, for background aesthetics.

Another world aesthetic is the plants. They are supposed to be unnaturally large and their colours are exaggerated. They are made with the same process as the mirror. The notes on the drawings were for the 3D artists to understand what to do with the model. An example is the mushroom below.


Figure 19: Concept art of mushroom. It shows the process from silhouette designs, to final concept.

For the colouring, the plants were usually painted and afterwards exaggerated with photoshop's image adjustments. Lastly small corrections were added. The before and after picture of a flower can be seen below.

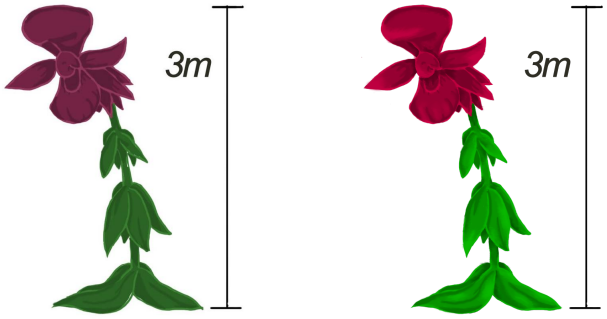



Figure 20: Concept art of flower. It shows before and after of colour exaggeration.

The end goal of the game was decided to be a large crystal heart that keeps the city alive, in connection with the Atlantis reference. It is supposed to be the power that lights all the

crystals in the game world. The concept was made as an actual biological heart, with the rough structure of a crystal. When a concept drawing is done, it is usually saved either as JPEG or PNG file.

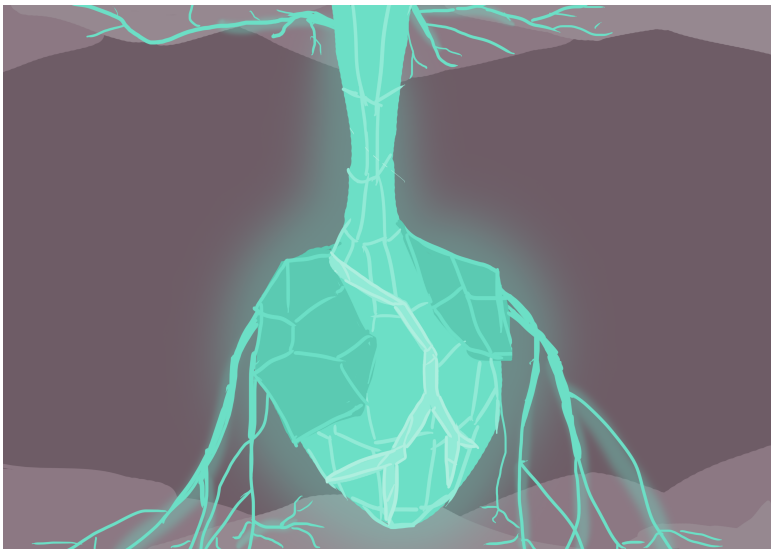


Figure 21: Concept art of crystal heart. The material is crystal and is supposed to be the conductor of light to the crystals in Infra.

**Texturing** For texturing, the software 3D coat was used. The program gives the possibility to paint directly on 3D models, instead of the traditional UV-mapping. This makes for a better work flow and makes it easier to create coherent textures. A FBX file of a 3D model is imported to the program and is unwrapped, meaning that the program automatically makes a UV unwrap from the model. A base colour is laid on an object. Each material has a colour guide, to make sure that objects of the same material upholds colour consistency. Layers are used to separate each step of the texturing process. This makes it easier to remove or change opacity of the separate steps. Layers also make it easier to correct one part of an object without accidentally editing another.

As the figure 22 above indicates, one of the texturing layers can be "highlights". In most materials this means an exaggeration of the edges of the material. Another separate layer could be the cracks in the objects made of a stone material. The texture of a background house can be seen on the picture below. It has the mentioned layers - cracks and edges, and it has darker irregular spots to make it more realistic.

The majority of the texturing that was done for objects was done in relation with pre-specified colour schemes decided by the art director. When an object was textured, it's relation to other objects needed to be taken into consideration. For example, a rock wall segment of

course had to have the same base colour as the rest of the rock wall segments. It was also important that individual rocks had the same base colour as the rock wall segments, because these rocks could be used to cover up areas that were not covered by the rock wall segments alone. Besides that, if a rock was put down as a decoration rather than a cover, it would also stand out unnecessarily if the base colour was different from the rest.

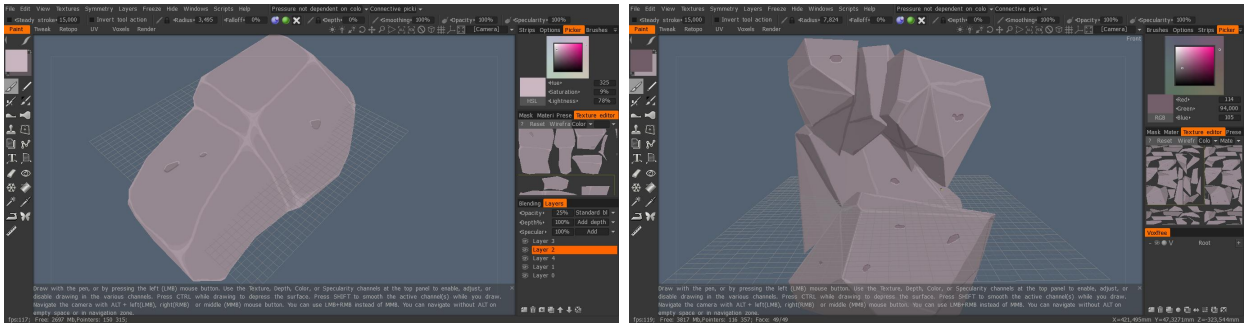


Figure 23: From left to right: A textured rock. A textured rock wall segment.

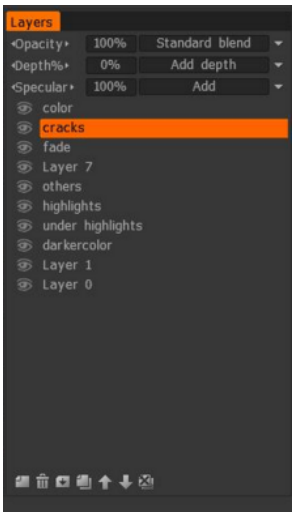



Figure 22: Layers tab in 3D coat.

In order to have the same base colours, a hex code was utilised to determine the exact RGB values of the colours.

Every object had to have a certain kind of highlight and shading applied to it. The main way to do this was to take the base colour and turn down the HSL light value down by 20% for the shadows, and 20% up with the lights.These new values would be up on a new layer on 3DCoat, so that the layer's opacity could be adjusted for the most optimal visual look. After that, more customised values would be applied for more specific highlights or features, mostly lighter or darker than the 20% values.

The other type of world material is the crystal. This material is seen on the statue below. It has a crystal in the forehead and one in each eye. This texture is made with an exaggerated blue colour and darker blue shadows. Lastly transparent blue colour has been added in the area around the crystal, as to indicate the light coming from it

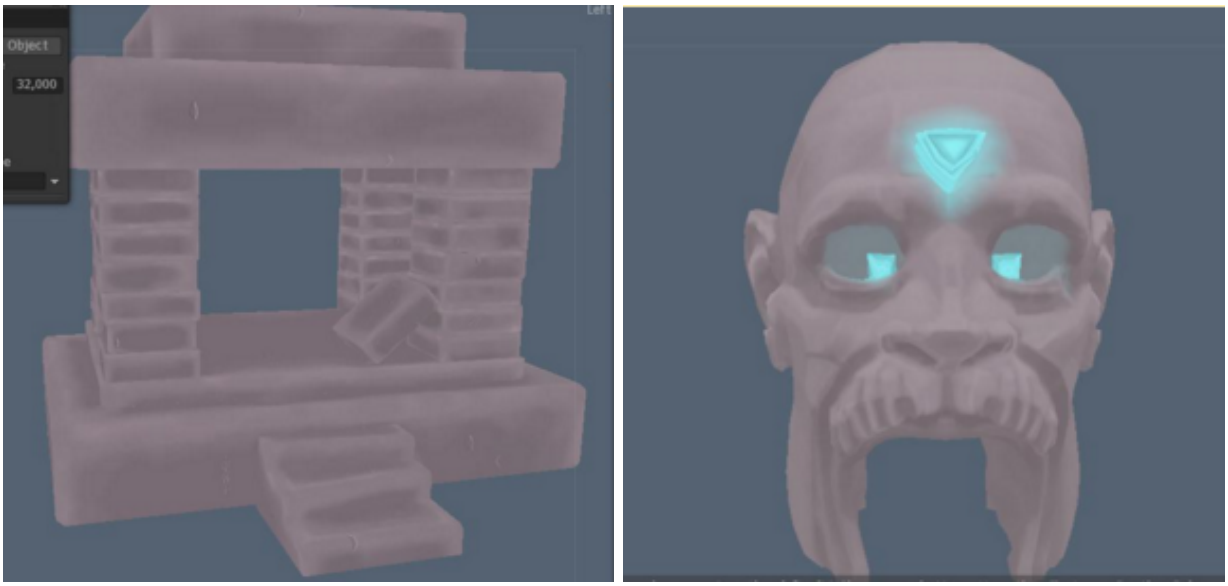



Figure 24: From Left to Right: House in stone. It shows how highlights are laid upon an object to emphasise edges. Statue of head. It shows the two types of material in the environment: Stone and crystal.

After a 3D object is painted, it is exported as a FBX file and the texture is exported as a png file.

### 3D artists

In this section we will go through the work of the 3d artists, their workflow, and which tools and programs they utilised. We will begin with a short introduction to the two 3d modelling programs that was used namely 3DS Max and Maya. Then we will look at the animation and rigging process, and finally the 3d sculpting process done in Zbrush.

**3ds Max and Maya** 3ds Max and Maya are both 3d modelling programs developed by Autodesk. Both programs were used for 3d modelling and have their own advantages and disadvantages - e.g. Maya is better at rigging and animation than 3ds Max. The choice of used software were based on prior experience. The two 3d artists who used 3ds Max only did 3d modelling, and one of them had prior experiences using the software.

For this sub chapter 3ds Max will be used as an example, but most concepts apply for both softwares (although some examples like shortcuts are program specific).

**Workflow** The workspace in 3ds Max gives you an easy overview of a the four windows you can switch between, you can see the workspace in 25.

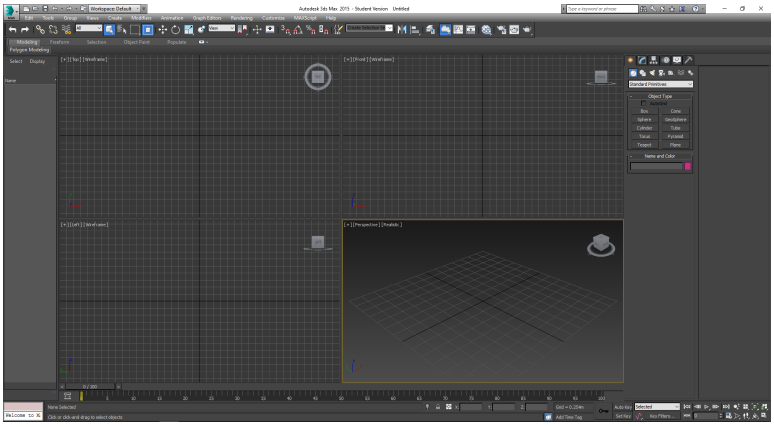


Figure 25: workspace in 3ds Max.

Here you are able to quickly switch between the windows by pressing the alt+w shortcut, maximising the window you cursor is hovering over. In figure 26 you can see the menu where you can create standard primitives and have access to a lot of functionality. This is where most of the tools to 3d model/edit/modify are located, which gives an neatly packed overview of the tools so you are able quickly switch between the tools you need.

After you have created the standard primitive you need with the right segments and the different sides, you have to right click the object and convert it to an "editable poly", which gives you the ability to form the object the way you need. You are able to select different selectors, such as: vertices, edges, faces and the whole object, likewise you are also able to perform different edits depending on what you have selected, this can be seen in figure 27.

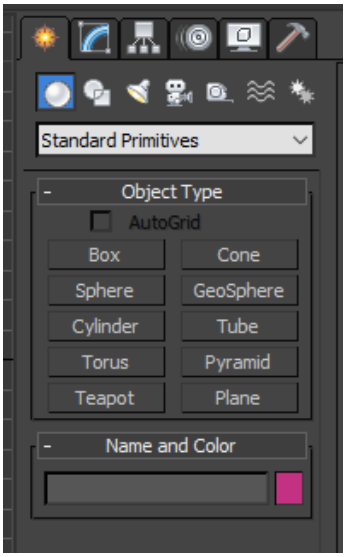



Figure 26: 3ds Max menu.

**Example: The process behind epic gate** In this example we will go through the process behind the creation of the epic gate, starting of with the pipeline and the workflow, then the actual 3d process of the 2d concept art and the exportation of the final 3d model.

**Pipeline and Workflow** Before the 3d artists can start working, they need finalised concept art provided from the 2d artists which has been confirmed and accepted by the art director. The concept art for the epic gate can be seen in figure 28.

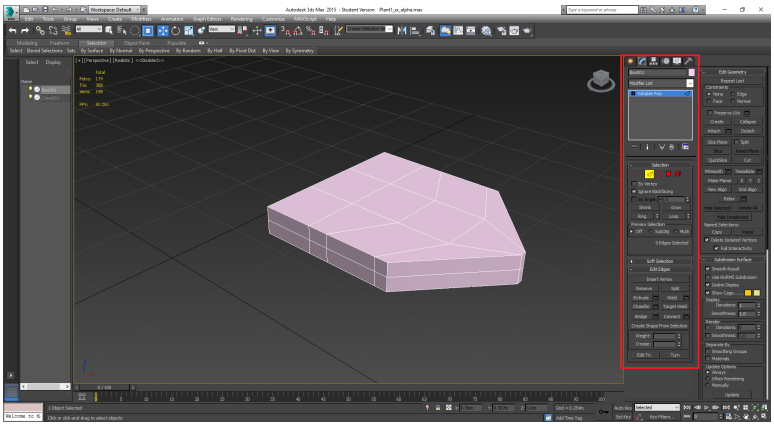


Figure 27: Red rectangle indicates selector and edit tools.

Figure 28: Concept art for epic gate.

When the concept has been finalized the 3d artists can start working on the first iterations and show them to the art director.

**3D process** The epic gate was created by a lot of different sized boxes stacked on each other. After the model has reached the specified scale given by the 2d artists and art director, all the boxes was converted to editable polys, so the 3d artist was able to edit the objects. In order to make the objects more organic, all the objects were bevelled and there were added dents to the objects using the "proboolean" tool. You can see the final model in figure 29.

After the model is done, meeting the expectations of the concept artist, the model would be triangulated/optimized ensuring it would have a low as possible poly count(for data reasons within the game, the more polys the higher performance needed), ensuring the metric systems match and the size is as it's supposed to be. Then items in the model would be attached

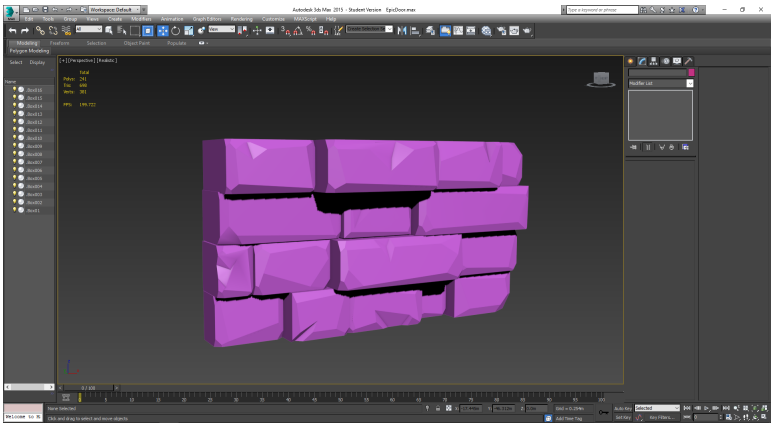


Figure 29: Final model in 3ds Max.

together and saved, first as a regular 3D MAX project. Then it would be exported to an OBJ and a FBX file, so it could be worked on by others(using other 3D programs) and so it can be imported into unity, or imported into 3d Coating enabling the model for texturing.

**Animation** Animations an important part of many video game productions and it is the job of the animator to bring life and movement into the world.

**Keyframe animation** All the animations are done via keyframes in the Autodesk Maya software A keyframe is basically a specific moment in the animation timeline which stores the location of certain vertices or objects, defined by the animator. For instance, if you select a certain set of vertices (or a whole object) and place a keyframe, the coordinates of the selection are stored in the keyframe. So when two adjacent keyframes in the same vertices or objects have a difference in coordinates there will be a transition in the timeline between the two.

**Graph editor** Another way to further control the animation is the graph editor The graph editor is a way to control the translation, rotation and scale of each axis separately by manipulating the tangents of each respective curve. This way we could fine-tune the movement of each axis separately and control the speed of the animation in different points of time and make the animation nice and fluid.

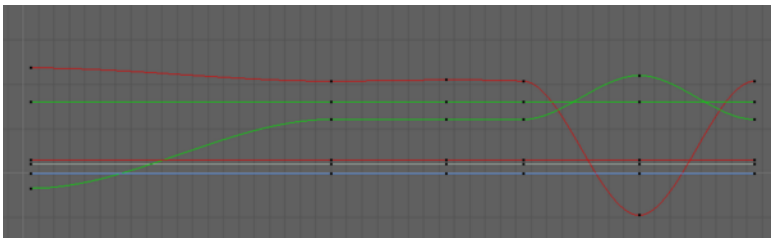


Figure 30: The graph editor shows the time on the x axis and the value of the translation, rotation and scale on the y axis. Keyframes are marked with black dots.

**Rigging** Rigging is the process of connecting vertices to a set of joints which can be said to form a skeleton of the model. The vertices can be connected with various degrees of influence depending on the animators' needs.

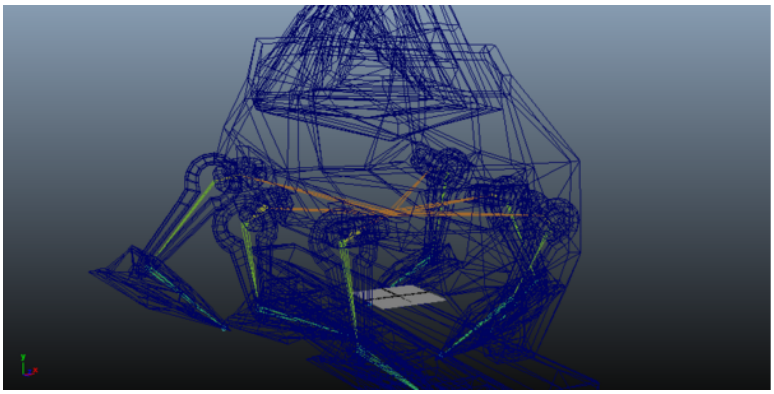


Figure 31: The rig can be seen in colour when the geometry is viewed in wireframe mode.

For the minecart (figure 31), the model had to be rigged in such a way that it resembled a mechanical being rather than a biological one. Therefore each part of the minecart were bound individually by selecting one part and the corresponding joints and clicking the Skin -> Smooth bind -> Selected objects. Once the binding is done the minecart were animated using normal keyframe animation. Ideally we would have used inverse kinematics but that did not seem to function correctly.

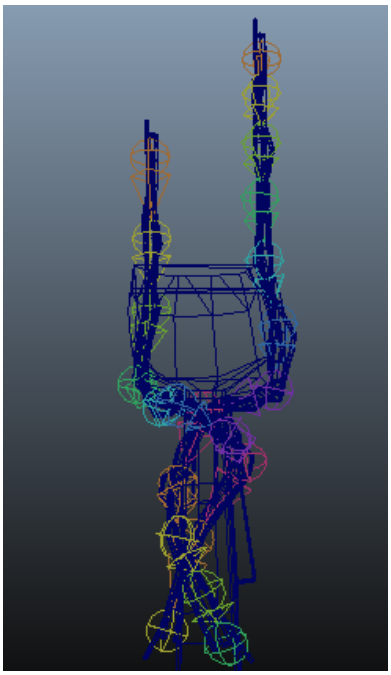


Figure 32: the vines have been wrapped around the crystal via rigging

Rigging can also be used in other areas than animation. An example of this (figure 32), being done in the project, is the vines that holds the sound crystal. These had to wrap around the crystal which was made easier by binding them with joints running in a straight line.

**Pipeline and working process** It is important that animations are done after the model has been UV-unwrapped. in the beginning we wasted a lot of hours because we animated models before they were UV unwrapped which made it pretty much impossible to texture. The animator should be in close communication with the 3d artists to ensure that the topology is optimal for the required animations. This includes making sure that there are enough vertices in the areas that has to move. In the case of this project that was less important as many of the animations were of mechanical nature rather than organical.

We used animal videos as reference material on several of the animations. The most prominent example of this is the minecart animation where a slow motion video of an ant was used. Another example is the button in which the movement is inspired by that of a jellyfish.

Many of the animations had to loop so in these cases it was important that the last keyframe was identical to the first.

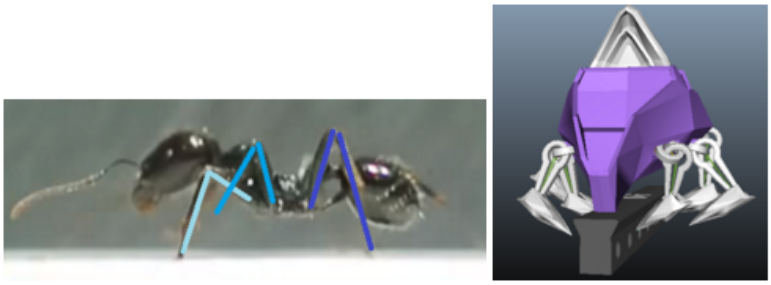


Figure 33: The ant used as inspiration for the animation and the 3D-cart with the legs in the same position.

The minecart (as well as many other animations) had to loop. In order to ensure that the loop went smoothly we simply had to ensure that the first keyframe were identical to the last.

**3D Sculpting** The 3d application Zbrush from Pixologic was chosen to do the complex organic sculpting. It is used in many AAA games, such as "The Last of Us", "Gears of War", "God of War" and "Assasins creed" (Pixologic, 2015). It differs from other 3d applications in that it makes use of brushes, (similar to photoshop) and that you work with a high polygon count. When done sculpting it becomes necessary to do a retopology, which means making a low poly version of the high poly sculpt, either manually or using algorithms.

**Workflow** The workflow in Zbrush is very similar to digital painting in Photoshop, as well as having a strong similarity with traditional clay sculpting. 3d sculpting could very well be referred to as digital sculpting in the same way that Photoshop is digital painting. The similarity with Photoshop is in the way that brushes and alphas are used, and the similarity with traditional clay sculpting is in the way that the artist has to think in terms of big forms at first before moving into details. This will further be elaborated in the following chapters but first the technical foundation for 3d sculpting in Zbrush has to be explained.

**Dynamesh** When sculpting it is natural that the mesh (polygons) stretches, because of this throughout sculpting the artist uses a function called "dynamesh" which is an algorithm that calculates a new mesh. In this way Zbrush sculpting becomes more of a fluid process, much like real clay sculpting. The following three pictures are close-up's of the lower beard of one of the ingame models demonstrating dynamesh in action.

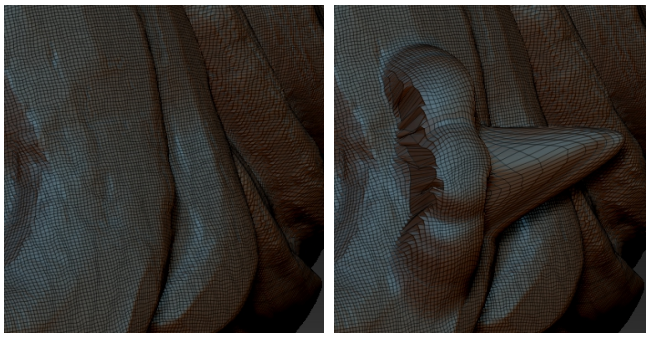


Figure 34: Here one can see a uniform mesh. And the same mesh deformed.

In figure 34 one can see how "heavy" sculpting can damage the mesh making it impossible to sculpt details on it (deformed into too big polygons). In figure 35 dynamesh has evened out the mesh density making it possible to continue sculpting.

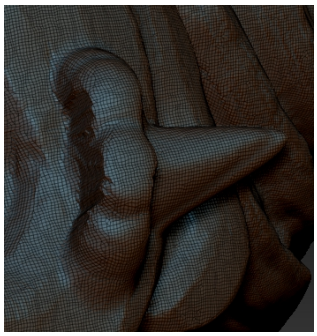


Figure 35: The same mesh as before, but made uniform through using the dynamesh algorithm.

**Organising into subtools** In Zbrush it is possible to separate a sculpture into "subtools" and in this way control visibility and only work on one "part" at a time. This workflow was also used for practically all sculpts . Later one can merge all subtools into one mesh (with many islands) and then if necessary use "dynamesh" so that they become one mesh (no islands, not separated). In this way it becomes possible to export a zbrush model as an .obj file. A .obj file is a standard 3d model format that can be imported in any other standard 3d application such as Maya, 3ds Max or Blender. Then from these applications one can export as .fbx in order to import this in Unity, or paint in 3d Coat.

**Building up form** The main work flow is not different from working with real life clay in the sense that the artist has to think in terms of building up big forms before moving into details. The big advantage however, is that Zbrush allows the use of symmetry, automatically

projecting changes done on one side of the x-axis onto the other side. As can be seen in figure 36, the moustache is first formed as one big simple form. Then this big form can be shaped a bit less simple and then broken down into smaller parts using Zbrush's standard brushes marking out where the form has to change.

Figure 36: From left to right: Big form of moustache. Less simplified version. Marked form changes. Final detailed.

The other much used brushes are the clay brushes since their algorithm is similar to the "feel" of real clay - small bumps and holes are evened out. Sharp edges are created with the standard brush set to a negative focal shift (very sharp edge). This can be seen in the final version of the moustache. The standard brush is then used in combination with the clay brush (to even out the form). Finally the trim-dynamic brush is used since it works similar to a piece of sandpaper, in order to make a completely flat surface. Also the move brushes are used, they work like a hand pulling a piece of clay. They are especially useful in the beginning when shaping the big mass of the form, but also when moving details.

**Usage of concept art and moodboard** The 2d artists made concept art that was meant to serve as a basis for the sculptures, but since these concepts was not sufficiently detailed, the art director and the high poly modeller (3d sculptor) designed detailed moodboards. E.g. figure 37 shows the concept art for the King and figure 38 shows the moodboard.

Figure 37: The main concept for the king sculpture

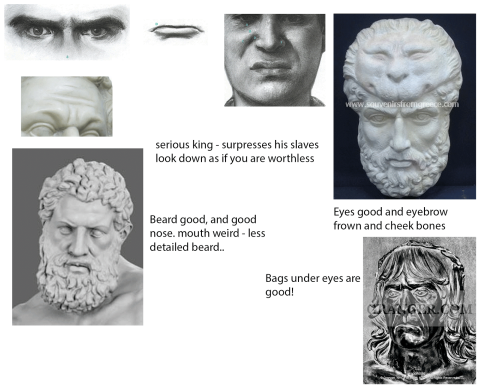



Figure 38: A detailed moodboard with comments

**Automatic and manual retopology** The decision on whether to use manual or automatic retopology methods is tied to the requirements of how dense the final low poly mesh should be, as well as what kind of "flow" would be optimal. In the case of the avatar (figure 39) the decision was to allow a triangle count of about 20.000, and also have a good "flow" around areas that might be rigged and animated (elbow joint etc.). Since automatic retopology is immediate (press of a button) it makes sense to always try this first and then judge if the result is usable in the game.

For the avatar the automatic retopology algorithm "Zremesher" was used since this algorithm often give good results in terms of mesh flow. Furthermore, Zremesher allows the artist to manually choose (paint) what areas of the mesh that should contain more dense (red colour) polygons and what areas should contain less dense (blue colour) polygons. As can be seen in figure 40 the result around the elbow joint has a very good flow usable for rigging and animation, however the face has certain problem areas around the nose and the mouth. In this case it was decided to stick with the result at hand, but fix the problematic areas of the face, using manual retopology methods in the 3d application 3d Coat (3d-Coat, 2015), thus effectively a combination of automatic and manual retopology.

Figure 39: The Player Avatar

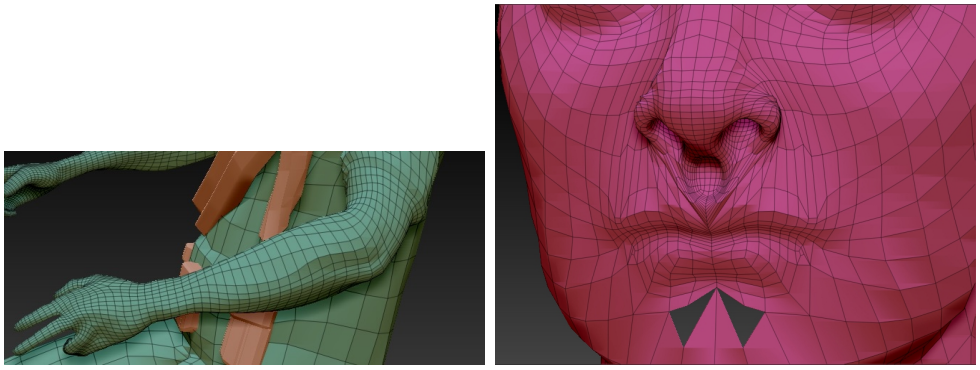


Figure 40: The topology flow using the Zremesher algorithm and problems using the Zremesher algorithm.

With regards to the sound sculpture it was decided to do a completely manual retopology in 3d Coat, essentially building a new mesh on top of the high poly sculpt (figure 41). The main reason was that the triangle count should be kept as low as possible, since this was a requirement for the scene inside Unity.

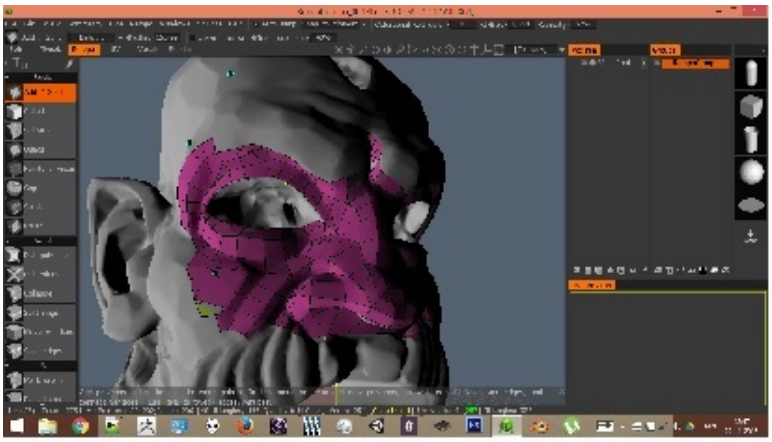


Figure 41: Manual retopology inside 3d Coat.

For the King statue (fig 42) it was decided to use the "decimation master" algorithm, since the requirement for the triangle count was a maximum of 20.000. The final result gave about 8000 tries (fig 42). Alternatively the Zremesher algorithm usually gives a much higher polygon count.

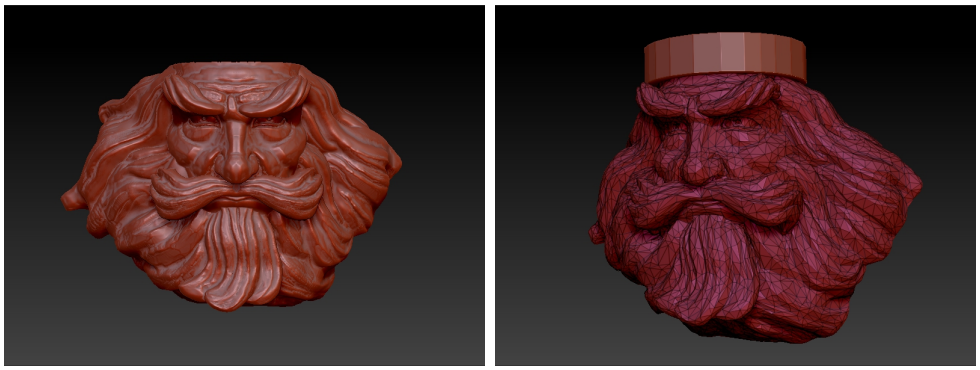


Figure 42: The high poly King sculpture and the sculpture after using decimation master.

### Sound designer

The sound designer has the role of creating sounds for the game. This includes sounds such as effects, ambience and interface sounds although In the case of this game there were no interface sounds. The sound designer is in close communication with the rest of the creative department with particular interest to the Art Director and the Animator.

**Recording** The sounds for the game were gathered from creative commons websites, synthesising programs and through Foley recording. Some of the gathered sounds that were not

recorded by the sound team, were downloaded from the website `http://freesound.org`. Other sounds were synthesised in FL Studio, using the Morphine special synthesiser plugin. Most of the downloaded sounds were sounds that were difficult to replicate in real life, while the synthesised sounds were used for the musical components of the game. The sounds that were recorded through Foley recording, were primarily natural sounds.

**Equipment** The recording sessions were recorded in various locations with a setup consisting of a Samson Meteor Microphone, which was hooked up with a laptop through a USB connection. The Samson Meteor Microphone is a small condenser microphone with a cardioid pickup pattern. and a 16-bit, 44.1/48kHz resolution. It is very lightweight and easy to use, but is primarily designed for studio recordings and desktop setups, meaning it isn't well made for outside use. This recording setup is not very mobile, as the microphone has to be connected to a computer through a USB connection.

**Procedure** After designing a given set of sounds on paper, the sound designers would either find a location where the desired sounds would be occurring naturally, or gather the items that would be able to produce the desired sound(s), then setting up a controlled recording environment to record them. Several takes would have to be recorded of each sound for two reasons: first of all, knowing whether or not a sound was useful for a given event in the game would be discovered later in the editing process, so having several versions of the same sound ready would help finding the best one. Secondly, a diverse soundscape would help sell the "reality" part of Virtual Reality, so having more than one sound effect for something like a water drop, would help with immersion.

**Sounds generated and recorded with Foley** The natural sounds that were required for the game were split up into categories specified by what kind of origin the sounds would have in real life. The core types of sounds that were needed for the soundscape of the game were primarily "Water", "Wind" and "Rock" based, since the game was determined from an early point to be set in a large underground cave, which would be heavily inspired by the ancient legends of the sunken civilisation of Atlantis.

Depending on the type of sound, a controlled environment could be set up so that the sounds could be recorded in a optimal way. A soundproof room was available to the sound team on the university campus, but it was not always possible logistically to carry all the needed equipment and materials to that room. Therefore, several controlled environments were made

at locations that were as quiet as possible while still being in close proximity to the materials that were needed for the foley recording session.

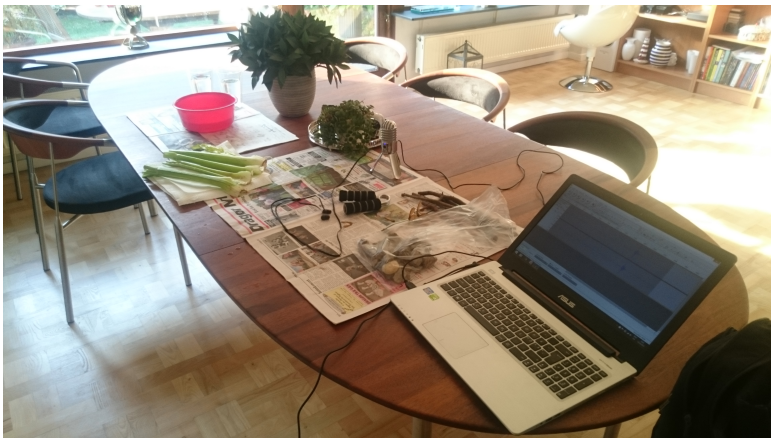


Figure 43: A controlled recording environment for small-scale sound effects

The water based sounds were recorded through different means. One of the flowing water sounds was recorded from an indoor water fountain. Several different water splash sounds and water droplet sounds were recorded by interacting with a bowls of water in several different ways.

Figure 44: Water fountain and bowl

Some of the wind sounds that were used for the main ambiance soundscape, were recorded by covering the microphone in a T-shirt and holding it outside of a window while a light breeze was blowing. The T-shirt stopped the wind from hitting the microphone directly, which would have resulted in too much white noise. Some normal household machinery was recorded, like the humming from a fridge and the rumbling of a washing machine, but neither were used in the game.

A lot of the rock sounds in the game were recorded at the university campus. Several rocks, concrete blocks and large crates were found and recorded by pushing them around on different surfaces or kicking them. These were used later for large stone doors and background mechanisms in the game. Ambient sounds were also gathered from small pebbles and nuts to make the sounds of crumbling rocks. Of these two, the nuts made much more believable rock sounds than actual rocks.

Figure 45: Concrete block being pushed along the ground and nuts for pebbles.

**Sound Editing** As mentioned before, our sound library consists of recorded foley sounds and creative common sound effects found online. The latter refers to sound effects uploaded by people around the world, which can be used however you like. We chose to use the free sound editing software Audacity for editing and manipulating the sound effects before importing them into Wwise. The general effects we used were:

**Noise removal** Audacity has a great noise removal effect, which we used on every sound file to remove all background noise and to make sure that only the frequencies that we are interested in remained. Noise removal was especially effective on the recordings we did ourselves, such as outdoor recordings of rocks sliding on concrete. These recordings contained background noises like wind and traffic which we removed.

**Pitch altering** Since our game takes place in a very large area, and the objects in the scenes are generally huge, we used pitch altering to lower the pitch of most sounds to fit with the environment. As described in the analysis, deep pitched sounds are often used to simulate large objects. Take the sliding rocks as an example. These were used for different sounds of large stone doors opening. Two different rock slides were pitched down and mixed together for each door slide. A deep rumbling bass was also added for the extra feeling of something extremely large being moved.

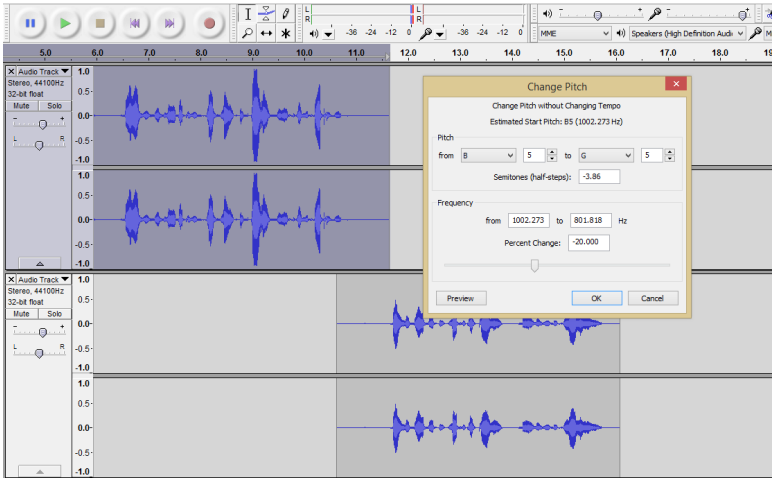



Figure 46: Figure showing pitch altering in Audacity

Pitch altering was also used for other sound effects, such as waterfalls and ambiance. Again these sound effects required a deeper pitch than the original sound. The waterfalls needed to be huge, and the ambiance had to resemble the deep rumble of a large cave. Some sounds were also pitched higher, such as the sound of a switch being activated. This was done using a time track curve in Audacity, continuously increasing the pitch during the short sound. This is a feedback to the user, telling them that the switch is now activated. The pitch of the cart motor sound also increases as the cart moves faster. This was done using an RTCP pitch curve in Wwise (See further below).

**Reverb** The reverb effect was used in both Audacity and Wwise, as well as 3Dception's room modeling in Unity (see further below). Reverb was used on every sound, as it is a main factor in creating the feeling of a huge space. Each sound required a different level of reverb depending on its size.

**Delay** Like reverb, delay was used in both Audacity and Wwise. We actually did not use delay that much for an echo effect, but more as a way to create "more sound". An example of this is some of the running water sounds, where three or more different water recordings are noise filtered and then put together. A very short delay, only repeating once is added to create the effect of there being even more water. The actual echo effect is added in Wwise instead.

**Low pass filtering** Both low and high pass filtering was used on different sounds where only the high or low frequencies were of interest. Low pass filtering was amongst other things

used on the sliding doors to remove most of the treble from the rocks. It was generally used on many of the sounds to create the illusion of a large object, or to simulate the mechanics that was going on behind a wall. In Unity, low pass filtering is used automatically to resemble distance as an object becomes further away from the player. Low pass filtering was also used as an RTPC curve in Wwise, increasing the cutoff frequency as the railway barricade descends under water.

**Looping** In a video game, it can be difficult to determine the length of audio files as you can never be sure how long a player will stay in the same area. This is not a big problem with interaction feedback sound, but can be particularly complicated when it comes to ambient sounds, since it will most of the time have to be played constantly as long as the player is in the area. This can be done through looping of sounds. A simple looping procedure is made by crossfading the start of a sound into the end of it. This will create a perfect loop as long as the sound is almost constant. With sounds that are not constant, we used more than one version of that sound, slightly altering the pitch whenever it is played. This makes the sound less repetitive.

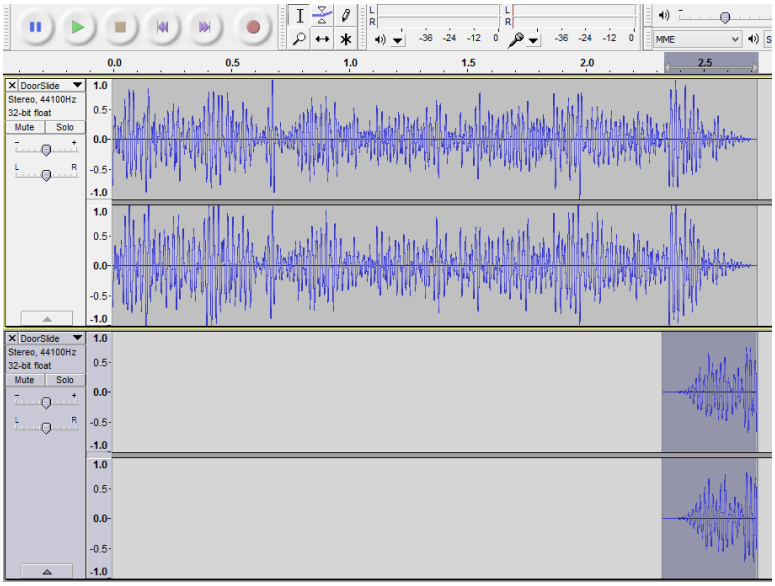



Figure 47: Figure showing looping in Audacity

Other tools and effects such as equalisation and bass enhancement were also used. The former was the primary effect for the vocal introduction in the game.

**Wwise** For integrating sound into Unity 5, we chose to use Audiokinetic's sound engine Wwise, because of its wide selection of sound editing tools that are specially made for games, and because of the existing plugin tools for integrating 3Dception into a Wwise project.

When working with Wwise, imported sound files are edited and put into different containers in an *Actor-Mixer* Hierarchy as seen in figure 48. An Actor-Mixer can contain a wide array of sound containers, and it controls all of the attributes of the sounds. Depending on the type of container, different behaviours can be attributed to sounds. Random Containers can contain several different sound files and will, depending on its individual settings, play a randomly selected sound when selected. A blend container can contain any number of sounds, and will play all of them at the same time once activated. A Sequence container can contain a list of sounds that it will play in a determined order once selected. It can even play these sounds in an endless loop.

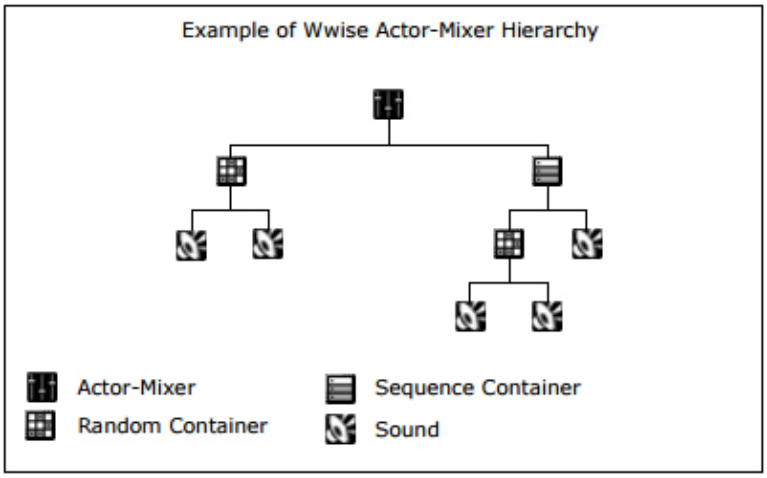



Figure 48: A schematic example of an Actor-Mixer Hierarchy in Wwise

An example from our final Wwise project that utilises all of these mentioned types of containers to create something that would have been very complicated to create otherwise, is our Main Ambiance sound (figure 49). This sound needed to be a very subtle whooshing sound paired together with a low bass rumble to give the illusion of being in an underground cave, and it was of the utmost importance that the players never noticed it.

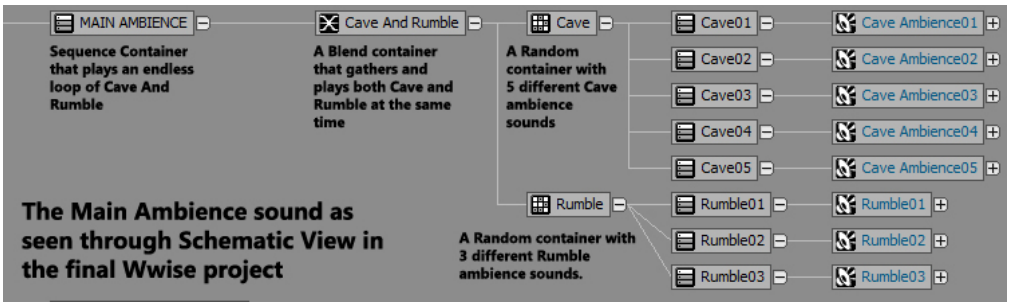



Figure 49: Our own Main Ambiance Hierarchy in Schematic view format (with step-by-step descriptions)

Therefore, we imported 5 similar, but slightly different wind sounds from our Foley sound library, put them into sequence containers and put those into a random container. The same was done with the rumble sound. Both of these were then put into a Blend container which was finally put into a sequence container that would continue playing that contents of that Blend container in an endless loop for as long as it was activated. The end result was an ambient sound that could loop infinitely in a completely random pattern, making it so that the player should never notice the same sound playing, thereby not breaking their illusion of really being in a cave.

Wwise integration to other programs functions primarily through the use of events. Events are placed into one or more soundbanks, which is then integrated into the game engine. Each audio file or container in Wwise can be placed in an event. Events will then execute an action with this file or container once it is called in the code. The actions could simply be playing a sound and/or stopping a sound. An example of how we used events in our game could be the sound of the sliding doors. Two events were created for this operation: one event, DoorSlideStart, starts the looping blend container which includes the sound files that makes up the huge sliding door. This sound will then loop until the other event, DoorSlideStop, is called and quickly fades out the looping slide sound, followed by the the playing of the sound of the door stopping. We have used these type of events for every sound in the game. The Wwise integration into Unity will be explained further in the programming departments subchapter.

### Environmental/Visual Design

Next to the game/level/puzzle design an environmental design was done. The purpose of this design was to shape the visual feeling of the game and to frame the correct experience of the game. At the same time it also had the purpose to guide the players through the game with

the help of visuals. This meant that the environmental design covered quite a large purpose and included a design focus to strengthen the following areas:

- Making the experience a whole and design the environments for continuity in order to convey a story.
- Lighting design to put visual emphasis on the games interactables, and thus guide the player's gaze to the interesting and important parts of the game.
- Designing visual areas and landscapes that would be interesting to experience in VR and thus putting the potential of HMD to use.

Hand in hand with a 2D environmental-artist and the art director, a 3D artist experienced with working in Unity was tasked to put the non-interactable parts of the game together and shape the artistic game atmosphere that was aimed for. figure 50 depicts the concept art that was created to first try and visualise how the game should look.

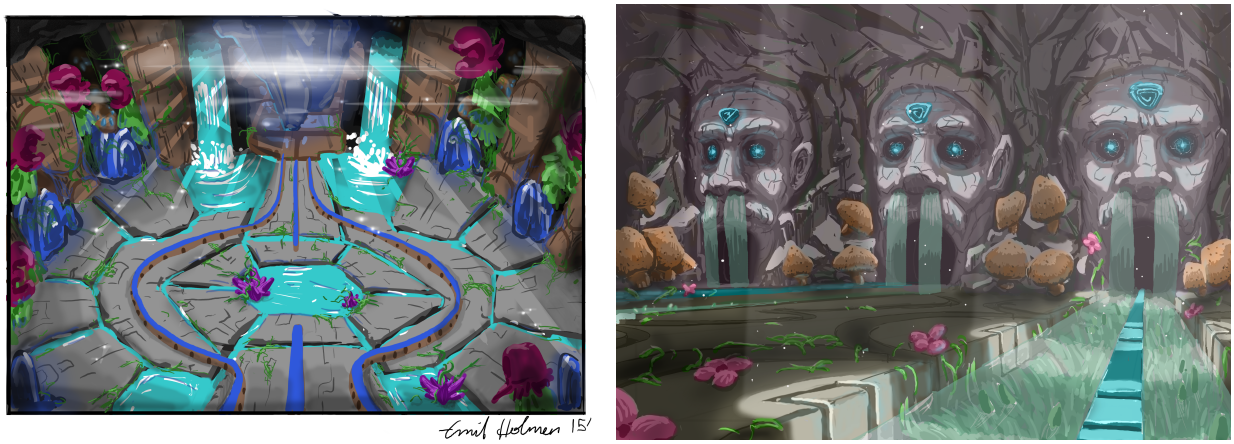


Figure 50: From Left to Right: First environmental concept design. Second environmental concept design.

To begin with these two exact environments were created to have a sort of blueprint of how the visual style of the game should be. The rest of the environments were designed with the purpose that they should fit the game but still differ from each other, and should also create new and exciting visual experiences when seen through a HMD. With these purposes we looked at the strengths of the HMD and decided on some experiences that would work well, such as; large spaces, narrow-turning spaces, and the ability to really perceive depth on a virtual medium. With these things in mind and knowing what 3D assets were at disposal to work with, simple ideas were created and accepted or denied by the art director. These were the environments chosen:

1. Small entrance rooms (intro puzzles).
2. A narrow down-spiraling tunnel full of crystals.
3. A large room showcasing an overgrown terminal used by the "civilisation". (mirror puzzle).
4. A tunnel with lots of lava.
5. A small room looking up at a higher plateau. (elevation puzzle).
6. A large room showcasing a city in the middle of a lake.
7. A medium size room with emphasis on 3 large statues. (sound puzzle).
8. A gigantic room where you can't see the floor, roof or walls.
9. A very small room with a crystal heart in it.

The design of the rooms were also chosen from their purpose. If the room contained a puzzle, the visual experience would not be as emphasised, in order not to distract the players from their task.

To convey a story without telling any to the players, the designed environments were placed in continually order. The order was chosen to make the tour as logical as possible when travelling through caves under ground. This had the purpose to try and make the tour seem as real or natural as possible to the player, and make the tour a showcase of the cave and it's "civilisation".

The above listed environments were also placed in the same order as listed above. This order was chosen on background of a discussion that included the following topics:

- There should maximum be 1 "in-between-level", between the puzzles.
- Lava should appear after going further down into the caves.
- The visual experience should grow gradually as the game progressed, by increasing environment scale and intensifying smart use of the HMD strengths.

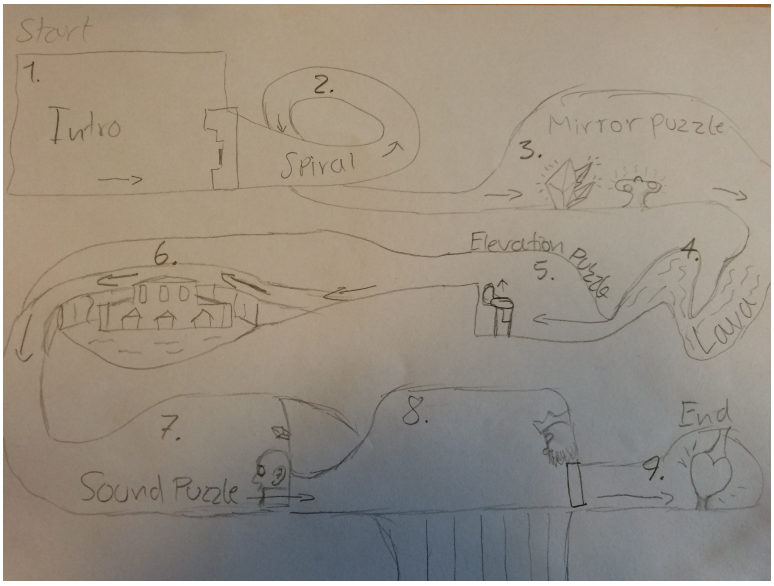



Figure 51: Shows how the environments were placed in continuity of each other.

Once the wanted environments were in place, some choices on how to keep them similar were made.

### Visual Continuity

Since a lot of visual 3D assets (such as crystals and plants) were created merely to define and shape the visual style of the game, they had to be used in a specific and continually way. This had the purpose to keep the visual style of the game continually and fit the art direction. Some things that was chosen was:

- Crystals grew in groups (like sea corals).
  - Square crystals grew out of cave walls.
  - Triangle crystals grew close to the water.

  - Layered crystals grew in the water.
- Mushrooms grew everywhere.
- Water Lilies grew relatively far from each other, but were surrounded by lilypad.

After designing the environments and the visual experience in its whole, a lighting design iteration was done on the environments.

### Lighting Design

The lighting design was done with some key points in mind:

- The lighting should be natural.
- Lighting should be more intense around interactable objects to guide players visually and easily give them the feeling that they found the solution.
- The environments should look deep and dark.
- Keep the amount of used lights to a minimum for optimisation.
- Try to keep lights and focal points within the golden circle.

Keeping lighting natural, keeping the environments deep and dark and keeping the amount of lights to a minimum all tied well together. These implementations were done by choosing objects that would seem natural as light sources and which were quite manyfold in the environments (crystals and plants).

Point lights were chosen as the lighting type for the games natural/ambient lighting source. This was because point light works as a light bulb which is a light source that emits light equally in all directions but fades gradually, which was what we wanted. In few cases a spotlight was placed without an actual light source. It was placed in a way that it was as subtle as possible (to not seem unnatural) but had the best effect of giving the object in the spotlight a good focus.

Last it was attempted to place focal points and their lights, so that it would meet the rule of thirds to the perceived of the game. This was however not something that could be ensured since the players view direction was not locked. It also made it harder, that the field of view in the Oculus Rift is different from the Unity standard.

## 5.6 Level design

A description of the design of all the levels in the game was needed in order to create an overview over the order of the appearing levels and content of the levels so the game would progress in difficulty the further the players got in the game. There were designed three introductory levels which should introduce the players to the game mechanics. After the introductory levels the game content consisted of three levels each with its own puzzle to solve. The whole game was based on the main game mechanic of a headlight which the players could focus and thereby interact with game objects. Each puzzle was created with the requirement of the player to use the headlight in order to solve the puzzle. All the levels were designed so the players had a common goal throughout the whole game. The players could not solve the puzzles and complete the game without all players actively participated.

A total number of three players were chosen since three people is assumed to be the minimum of people for teambuilding. Another reason for having a total of three players was that the game was using colour combination which works best with three players because the number of the basic colours are three. Each of the three players had a colour in the RGB additive system (Red, Green and Blue). The RGB system was chosen based on the result of the combined three colour was white, where the CMYK subtractive system gives black when combining all the three colours. A black light would be a issue since it would not be possible to see in a dark room.

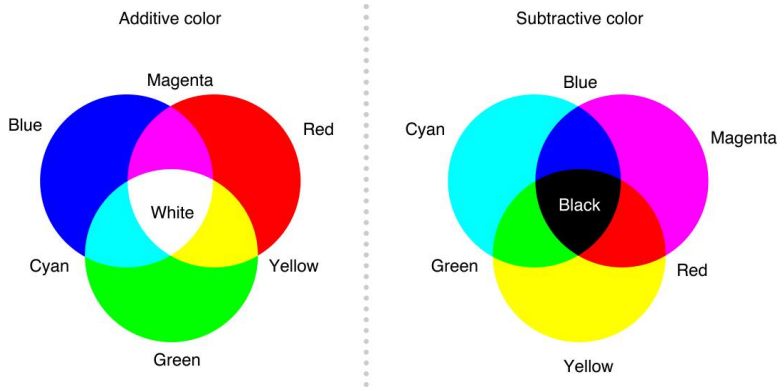



Figure 52: Shows the two different colour systems RGB and CMYK (Apple, 2015).

**Intro puzzle 1** The first introductory level was designed in order to introduce the players to the mechanics of the game, this being the player movement, their headlights, how to activate the buttons and how barricades and doors function. The players were introduced to activating the buttons by focusing their headlight and they were introduced to barricades and

the giant door, which enabled the players to move forward until they had solved a specific puzzle.

Each player had their own button which corresponded with the same colour as their headlight. This should teach the players that a button of the same colour as their headlight means that they must focus their head light on it. The location of the button was placed on the barricades (figure 53) so the players could see that the button was activating the barricades and thus moving them down. A giant door would open when all the barricades were activated.

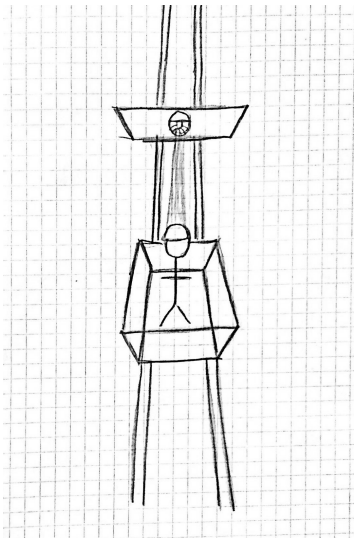


Figure 53: shows the player interacting with the button placed on the barricade

Since this puzzle was mainly created to introduce the players to the game mechanics the main communication expected was regarding sharing information on how to move, how to focus headlight and how to get through the obstacle being the barricades and the giant door.

**Intro puzzle 2** The second introductory level was created in order to introduce the players to combining two headlights colours and thereby creating a new colour. The players got introduced to the colour combinations magenta, cyan and yellow. Magenta is the output of a combination of red and blue. Cyan is a combination of blue and green and yellow a combination of red and green.

The location of the three colour combination buttons were changed based on several usability tests. Usability testing of the game showed that the players did not understand the possibility of combining the colours because they thought that they each had their own button. This was the case since the three buttons were placed in front of the three players (figure 54).

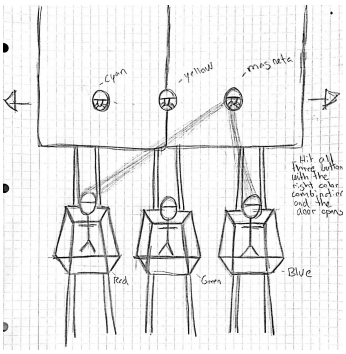



Figure 54: Shows the players combining their headlight colour. Player red and player blue are shining on the same button in order to create magenta.

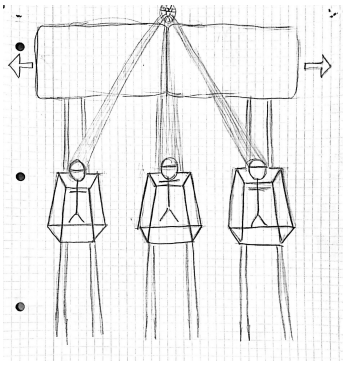


Figure 55: Shows all the three players shining their headlight on the same button in order to create the colour white.

Another reason was that the colours of the buttons were too similar to the players being closest to the buttons. The blue player had the cyan button in front of him/her, the red player had the magenta button and the green player had the yellow button in front of him/her which were the issue.

This puzzle was designed in order to create a discussion about solving the problem. Optimally the players should discuss the possibilities for activating all the three buttons through combining the different colours. The players were able to see the colour combinations throughout the whole game if they shined their light on the same spot, e.g when red and blue shined their light on the same spot the combined colour becomes magenta. If one or more of the players knew colour theory they could share this information to the other players and thus

succeed in solving the puzzle.

**Intro puzzle 3** The second introductory level was created in order to teach the players that all players can combine their lights which gives the colour white (figure 55).

The communication level expected were not extended since this level should be fairly simple since it is not very different from the first two introductory levels.

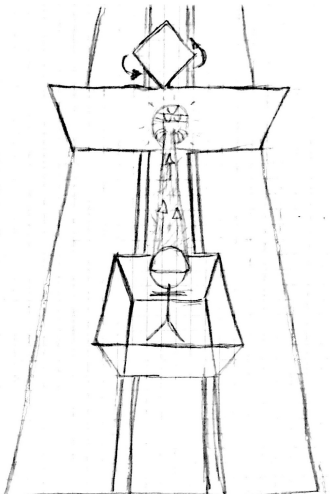


Figure 56: Shows the player interacting with the button placed on the barricade and thus rotating the mirror.

**Mirror puzzle** The fourth level was designed with a common goal. The players each had their own barricade and mirror placed in front of them. The location of the mirror had to be near the barricade in order for the players to see the players move their mirror while they rotate it. Usability testing of the game showed that the mirror must be in front of the player and barricade (figure 56) since some players moved all the way up to the barricades and thus was able to miss the mirror object.

This puzzle did not only require communication, but it also required the players to actively share information. The level was created with different height level in order to limit the vision of the players. This puzzle required the green player to share the visual information of seeing a red button, ergo give away part of the solution to the puzzle, that red light must reach this button through the use of all the players' mirrors. If the green player was holding back this information or was not able to see the red button the whole team could not succeed in understanding the puzzle. In short, all the players must do their part in turning their mirror so they could reach the common goal and the light could reach the final button which would resolve in the barricades going down.

**Elevator puzzle** This puzzle was designed to use the same colour combination of magenta, cyan and yellow which the players learned from the second introductory level. This level contained an elevator for each of the players. The players had to help each other in order to activate the elevators and reach the tracks on the higher ground (figure 57). The red player had to activate both the blue and red players' elevators and thereby help them while still being at the lower ground. The red player had to create magenta with the blue player in

order for blue's elevator to go down and up. The red player had to create yellow with the green player in order for the green's elevator to go down and up.

Finally, the two players who reached the higher ground had to bring down the last elevator and transport the last player to the higher ground, through combining their colours hitting a cyan button. The green and blue player (on the higher ground) could choose to continue without the red player (on the lower ground), requiring the red player to communicate that his elevator was not down.

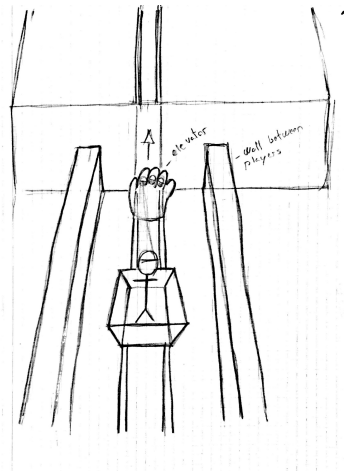



Figure 57: Shows the setup with a players elevator which goes up to a higher level when activated.

Because of this specific design, the puzzle requires a high level of communication to reach the solution. In general, all the players had to communicate when they were on their elevators ready to be lifted, especially the red player needed to communicate where he/she was since the blue and green could not see the location of the red player.

**Sound puzzle** This level was designed in order to create verbal communication through making the players recreate a melody through a set of tones. The players each had a button with a specific sound which the other players could not hear (figure 58). The melody could be replayed by focusing the headlight on a crystal hanging from the ceiling. It was discovered through usability testing that the players did not see the sound crystal as an interactable object and was therefore not able to replay the melody. A button was placed on the sound crystal as a cue for the players so they would see the sound crystal as interactable.

The communication expected throughout the puzzle was, beside discussion of how to solve the puzzle, singing, humming or describing the type of sound e.g by its pitch like low, middle or high pitched.

## 5.7 Programming Department

The programming department handled the technical implementation of the game, including level editing, sound implementation, and some graphics implementation. As a programmer, the main tasks were to write code, and making sure assets were imported, and behaved as intended.

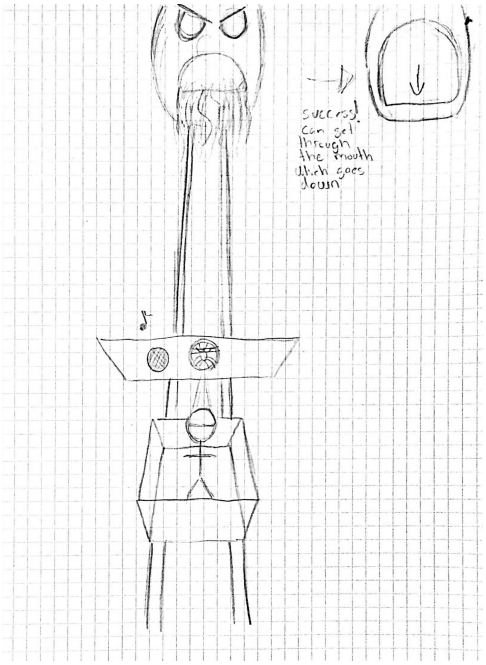



Figure 58: Shows the player interacting with the button placed on the barricade and thus playing a tone which only he/she can hear.

### Work structure

The programming team consists of eight members; one lead programmer, two senior programmers, and five programmers. The lead programmer would keep an overview of the tasks to be completed, hours spent on each task, and would merge new features into the game as they were implemented. The lead programmer and the senior programmers would help the remaining programmers when problems occurred.

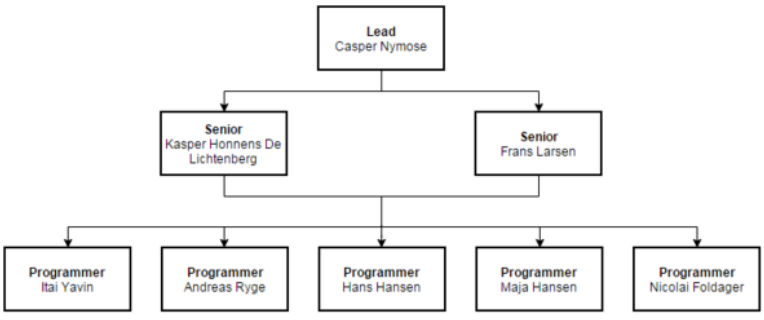



Figure 59: Structure of how the programming team was set up.

### Programming structure

Almost all code is written in C#. Generally, the core functionality of the game is programmed in a component based manner, which means that, ideally, when fully functional, we can set up game levels with any number of our interactable objects, of any type, in any order wanted. This is a huge benefit both performance wise, and allows for as many level designs as possible to be tested for usability, so that the design of the final levels be decided upon.

### Software

**Unity3D** Unity3D was chosen as the game engine to work with due to its flexibility, reliability, and support for the Oculus Rift. It also allows for easy debugging, and real time profiling, so that the most performance heavy elements can be optimized, which is important when creating a game, and even more so, when creating a game for virtual reality.

**Git** To allow the programming department to work at the project at once it was decided to utilize version control. Based on prior experiences and what the team felt comfortable in, Git was chosen. Since Git is a distributed version control system, where every developer has their own local copy of the entire project. This allows entire teams to work on the same project at once as each development environment is isolated from the others, while still preserving filenames and directory structure consistent for all members, making changes to the source code without fearing it will break functionality, as scripts can be reverted to prior versions. Furthermore, as Git keeps track of who change what it allowed the lead and senior programmers to backtrack prior commits to understand who did what and when it happened.

The method of workflow was decided to be feature branch development. Git allows each member of the programming team to work on their own branch of the project. This means that they can add and remove stuff from the project and not having to worry about how it conflicts with others. When they are at a point where their work is finished they push their branch to the Git servers and create a pull request. This tells the lead and seniors that the programmer would like to have their work integrated into the main branch.

**TeamSpeak** Instead of creating our own in-game VoIP functionality we decided to use a popular application called Teamspeak. This does not require a lot of effort to get up and running, and it served the purpose of providing voice communication, but at the cost of

abandoning the idea of having spatial sound. Since all our computers were connected to a local area network, we just had to host start a server on one of the computers and then all the others could connect to that server.

**Wwise** The games' sound was implemented using a Wwise plugin for Unity. Wwise is an audio engine made by Audio Kinetic, designed for sound implementation in games. Wwise was chosen due to its' flexibility in mixing sounds in realtime, and easy configuration of when and how each sound should play. For more information about the use and functionality of Wwise, we refer back to the subchapter on the sound artist's use if Wwise.

**3DCeption** 3DCeption is an audio engine with binaural functionality. 3DCeption was implemented in the project through Wwise. Meaning that 3DCeption was implemented into Wwise, which was then implemented into Unity as a plugin. 3DCeption was used for sound spatialization, which was meant to give the users an idea of the space in which they travel as well as the position of specific objects, such as the other player models, and the origins of different sounds.

### Player mechanics

This section introduces how player mechanics such as controls, movement, and interactions have been implemented.

**Controls and movement** The players are seated in each their own cart, which they can control by moving forward and backwards along the rail tracks in each level. This movement is simply bound to a controller's left analog stick on the vertical axis. For the players to look around, the camera is rotated with the controller's right analog stick in the non-VR version of the game. In the VR version, the player view is fixed to the rotation of the Oculus Rift, so that the players control what they see in the game with their heads. The trigger button on the controller is used to focus the lamp mounted on the player's head.

To easily be able to generate levels of different shapes and sizes, a system was created to allow us to place varying rail track segments, and connect them.

**Rails and Cart** As mentioned in the controls and movement section, rail tracks are added to the levels through a system that allows us to easily add tracks, and each single rail track

contains a railpoint script that looks for the previous rail and the next rail, which means that these have to be connected before a player can move from one rail to the other. Listing 60 shows how the rails script is seen in the Unity Inspector, where we are able to drag and drop previous and next rails into the selected rail.

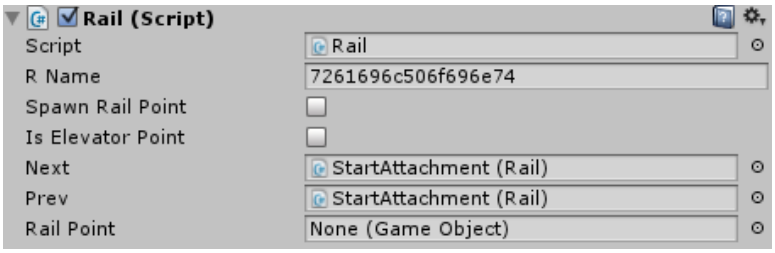



Figure 60: Rail script that is attached to each rail track in order to connect them and allow a player to move.

The rails script makes it possible for players to move back and forth on the rails that are connected to each other, but the players have to solve different tasks in order to be able to continue from one point to another, and this means that we need to be able to control when some rail tracks get connected to each other. The RailConnection script was created for this purpose. This script contains methods that can be used to connect or disconnect rails. An example for when this is needed is when there are barricades in front of the players, where the players only should be able to proceed when they have solved a puzzle and lowered the barricade, then this is where we call the ConnectToNext method and connect the rail that is right before the barricade to the rail that comes after the barricade. The ConnectToNext method is seen in listing 1.

Listing 1: The method ConnectToNext taken from the RailConnection script.

```
public void ConnectToNext(){
        if(self.next == null){
                self.next = nextRail;
                nextRail.prev = self;
        }
}
```

The cart script functioned in tandem with the rail script. The rail script would hold all the positions of where the cart script could move, and the cart script would simply interpolate between positions. Listing 2 shows a portion of the Move method in the cart script. This method simply interpolates between the two rail points that the cart is currently between. When one rail is reached it then updates its next and previous rails, using the one it is

currently on (each Rail component knows what rail is next and previous from it, so the cart can simply take that information from there).

Listing 2: A cutout portion of the Move method from the Cart script.

```
if(railMoveTowards != null) {
        float length = Vector3.Distance(currentRail.transform.position,
            railMoveTowards.transform.position);

        currentStep +=  (1 / length) * verticalAxis * movementSpeed * 0.0075f *
            (Time.deltaTime * 1000) * (Input.GetKey(KeyCode.R) ? 10f:1f);

        transform.position = Vector3.Lerp(currentRail.transform.position,
            railMoveTowards.transform.position, Mathf.Abs(currentStep)) +
            (currentRail.transform.up / 4)*2;
        transform.rotation = Quaternion.Lerp(currentRail.transform.rotation,
            railMoveTowards.transform.rotation, Mathf.Abs(currentStep));

        if (currentStep >= 1){
            currentRail = currentRail.next;
            currentStep = 0;
        }  else if(currentStep <= -1){
            currentRail = currentRail.prev;
            currentStep = 0;
        }  else if(currentStep <= 0 && currentRail.Equals(startingRail)){
            currentStep = 0;
        }
}
```

**Interaction** An object is interacted with by a player pointing the light from the lamp on his head on the object, and is focusing the light with one of the triggers. This is implemented through a "Raycast". A ray is created from the player's position, which has a set length and a direction that is forward from the player. The first object that the ray collides with is told that it is "interacted with", if allowed by the trigger system, which takes into account player information such as colour and if the interactable is allowed to be interacted with.

In order for the players to communicate with each other visually they were given a light that was placed on top of their helmet. The direction their individual lights shines are fixed to the direction the player looking. The light object was a simple spot light set as a hierarchical child to the camera object in Unity. This makes any translations performed on the parent, i.e. the camera, to be performed on the children objects as well. If a player were to rotate

their head in order to change the direction they are looking, the spot light object would follow along.

The helmet light was used as an active part of the interaction process, as it gave a visual feedback when the users were operating it. When the triggers on the back of the controllers were pressed the cone of the spot light would narrow down. This was performed by a simple line in the 'HelmetLightScript.cs', as seen below.

Listing 3: The content of the method LightUpdate(float t) that updates the light cone angle.

```
helmetLight.spotAngle = Mathf.Lerp(angleNormal, angleFocus,t);
```

helmetLight refers to the light source, angleNormal and angleFocus are the angles of the cone when it's not focused and focused and t refers to the time. The mathf.Lerp method allows for a linear interpolation between two values, in this case from the normal angle to the narrowed down angle. This means that when the triggers are pressed the cone will smoothly narrow down over the value specified in t.

Along with the cone of light angle being narrowed down, the raycast that was described earlier is also sent from the object in the forward direction.

### Feedback

**Sound** To implement sound into the game, a singleton class was made called 'SoundManager'. This script utilizes the inherent methods from the Wwise plugin (see 'Software' section about Wwise for more information). This script is supposed to have a single purpose, it held all of the necessary methods for playing and setting of events, it is also the only component in the game that handles and plays sound at different given positions. This made it possible for any other script to call the methods in the SoundManager and it would then handle the execution. The creation of the SoundManager simplified the implementation of sound throughout the rest of the project.

Listing 4: SoundManager script example.

```
public void StopAllEventsOnObject(GameObject g){
        AkSoundEngine.StopAll(g);
}

public void PlayEvent(string eventName, GameObject g){
        AkSoundEngine.PostEvent(eventName, g);

        for(int i = 0; i < objectList.Count; i++){
```

```
                if(objectList[i] == g)
                        return;
        }
        AddToList(g);
}

public void StopEvent(string eventName, GameObject g){
        AkSoundEngine.ExecuteActionOnEvent(eventName,
AkActionOnEventType.AkActionOnEventType_Stop, g, 0, 0);
        RemoveFromList(g);
}

public void StopAll(GameObject g){
          AkSoundEngine.StopAll(g);
}

public void ToggleSwitch(string switchName, string switchMode, GameObject g){
        AkSoundEngine.SetSwitch(switchName, switchMode, g);
}
```

Above is shown a portion of the SoundManager script. As seen the SoundManager mostly consists of a series of Wwise functions, some of which have been simplified either in name or in use, to make them easier to implement.

**Animation** The animation of objects in the game was a mixture of using unity object movement through transform or using mecanim to play animations connected to the 3D model.

The button were created with an animation feedback. When a player interacted with the correct button the button would start pulsating in and out in order to indicate it being activated.

For running animations on the game objects there were used Unity's inbuilt animator Mecanim. This system controlled the animation of the giant door opening, the button pulsating in and out as feedback, the player cart walking(legs moving) and the sound crystal moving back and forth. The sound crystal was looping its animation all the time while the other game objects animations were activated by a boolean which were turned true in different scripts and thus turning a specific animation on. The giant door's animation could only be played once while the button type in the sound puzzle could be activated several times and should therefore play its animation each time.

When a boolean was changed from false to true like when for instance a player pressed the movement button on the controller, the state of the cart's animator would change from the Entry state to the CartAnimation state (figure 61). The CartAnimation state contained a animation where the carts legs were moving around. This state does also contain the speed of animation which was accessed in the Cart code and changed accordingly to how long the player pressed the movement button. When the player stopped pressing the movement button the carts animator statement would change and thus stopping the animation.

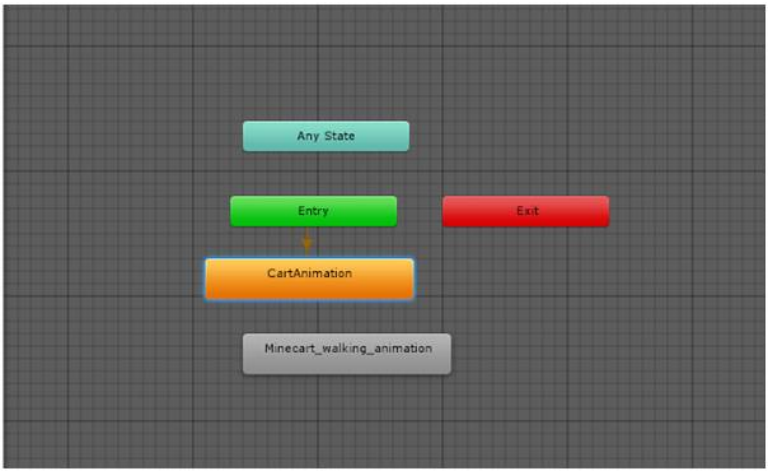



Figure 61: Unity's user interface for mecanin with the current state being CartAnimation.

**Effects (particles & lights)** There were created several effects as player feedback in the game in order to insure a high level of usability and understanding for the game mechanics. The players should be able to see what objects are interactable and when the objects are activated.

In order to insure that each player knew which button were their button a colour feedback was created. This was made by giving the button the colour of the individual players colour or the colour of combining more colours. The button object had a constant particle effect which were running all the time, while another particle effect started a short burst when a player activated a button. When a button was activated a button would start shining a dim light in the same colour as the button colour indicating that the players no longer should shine on the button. These button feedback effects were turned on and off by the InteractableButton script. A part of the script is showed in listing 5. The bool in listing 5 is setting the player index to the correct colour in the RGB system done with a boolean checking for r being player one, g being player 2 and b being player 0. Section 2 is checking which player is needed to activate a specific button. This is done for each of the three players, the three colour combinations and the button combining all the three players. The code in

the if statement is setting all the light feedback to correspond with the colour of the player needed to activate the button, this being the light, the particle system and the burst system. A for loop was used to go through all the objects in the button's 3D model and setting the colour of the whole object to the correct colour.

Listing 5: Code used for setting the buttons colour based on the player that is required to press it.

```
public void setButtonColor(bool[] a){
        bool    r = a[1],
                g = a[2],
                b = a[0];

    if(r && !g && !b){
        buttonLight.color = Color.red;
        particleSystem.startColor = Color.red;
        puffSystem.startColor = Color.red;
        for(int i = 0; i < rend.Length; i++){
            rend[i].material.color = Color.red;
        }
    }
}
```

The mirror object in the game was created with a ray cast ability which meant that the mirror could reflect a player's light shaft in the corresponding colour of the player who were shining their headlight on the mirror. The reflected light shaft on the mirror had the direction in which the mirror was facing. This light feedback system would insure that the player could see the current turn position of the mirror and which way the mirror should turn. This light feedback effect was created in the mirror script. The script contained the same method content as in listing 5 which create the same colour output as the player who was shining on the mirror.

The soundcrystal was created with a light feedback which was created with a blue/green colour while the sequence was being played and pink when the players had made an error. The original colour of the error was red, since red was a colour assumed to give the feeling of error. The issue with using red as an error colour was that one of the player colour was red and the players would might assume that the red player was related to the error. The light feedback while the sequence was playing was indicated by the blue/green light source going up and down in intensity between a minimum and maximum value. See listing 6. The light feedback was supposed to be visible in the whole room, the maximum value were therefore set to a high value in the inspector. The light feedback was created in the LightFade script.

Listing 6: Code used for pulsating a colour if there is a sequence playing

```
    if(Sc.sequenceIsPlaying == true){
        myLight.color = new Color(0.2F, 0.7F, 0.5F, 1F); //Green/Blue
        currentIntensity = Mathf.MoveTowards(myLight.intensity,targetIntensity,
```

```
            Time.deltaTime*pulseSpeed);
        if(currentIntensity >= maxIntensity){
            currentIntensity = maxIntensity;
            targetIntensity = minIntensity;
        }else if(currentIntensity <= minIntensity){
            currentIntensity = minIntensity;
            targetIntensity = maxIntensity;
        }
        myLight.intensity = currentIntensity;
    }
```

**Player feedback (from other players)** A light shaft script was implemented from a GitHub user (robertcupisz, 2015). This light shaft script was attached on a spotlight on the player prefab. The light shaft was only visible from a camera which was added into an array in the script. The player was not able to see its own light shaft since this would distort the visuals, so other players' camera were added into the array, except the players own camera. The colour of the each of the players' light shaft was set from another script similar to how the players colour was set.

**Networking** A networking solution called DarkRift was utilised to implement syncing of clients. We decided to use an Embedded Server within Unity, to allow for monitoring during tests and easier debugging.

Initially the networking system was implemented while striving for having it to be the least amount of intrusive to the existing codebase. Additionally we wanted to ensure that as few classes as possible had 'network access'. For the server, the class ServerManager controls all traffic and locally pipes necessary information to other classes like the TriggerHandler. The server among other things responsible for changing levels, which can be seen in figure 62.

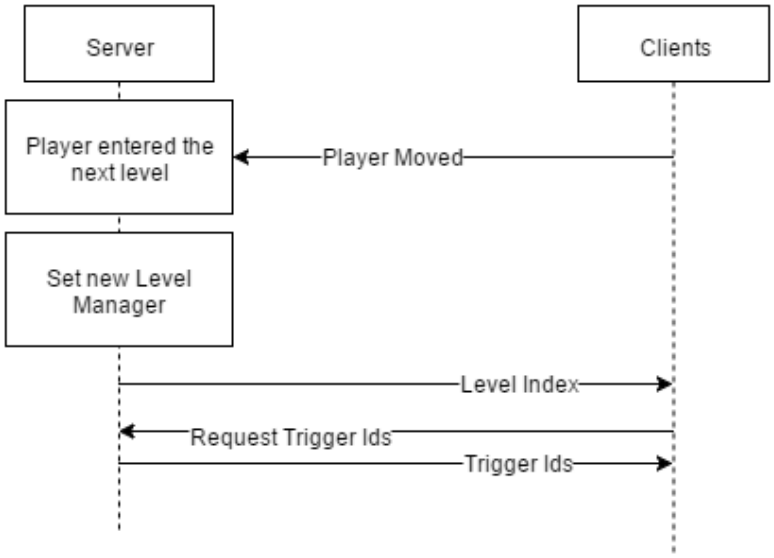



Figure 62: Activity Diagram that details how the server know when to change level and what information it give the clients.

For the client, the ClientManager controls only part of the traffic. The ClientManager connects to the server, a diagram showing the process of connecting to the server can be seen in figure 63. The ClientManager handles spawning of the player, spawning a version of the other clients player. When it spawns the players it gives them a network ID, and syncs it between other players and the server. It also manages the current level, and starts the game by fading the players in when all are present. The TriggerHandler handles trigger syncing, The Player class syncs the player to the other clients.

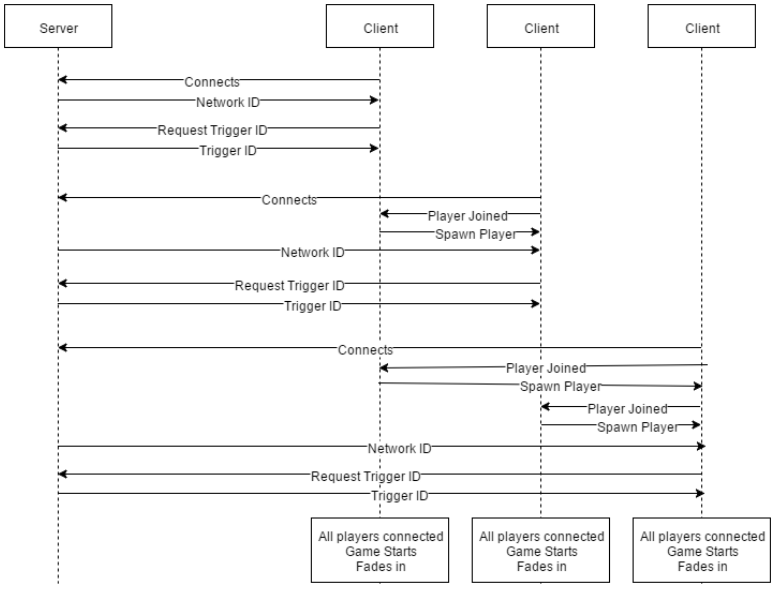



Figure 63: Activity Diagram showing what happens when clients connect to the server.

The player class called "NetPlayerSync" contains both the code for a client's player and the client's shadow players. When the ClientManager spawns a player, it initializes it as either a sender(client's own player) or receiver(a version of other client's player), which enables or disables several scripts. A sender will send information when it moves or looks around, in order for the receivers to receive the information and interpolate to the new position and or rotation, which makes the receivers movements more smooth.

A system, internally named the "Trigger system", was created for the purpose of handling the elements that the players interact with. When the first iteration of the system was made, the implementation included a GameManager and a LevelManager. Each of these were needed to be present in each scene. The LevelManager would contain all the objects to be triggered. The GameManager would check the objects in the LevelManager all the time. The early class diagram and sequence diagram of the LevelManager and GameManager can be found in Appendix A.

The current LevelManager keeps track of all interactable objects in a scene, the order that they are supposed to be triggered in and if they are the correct objects players need to trigger. Whether the objects are triggered, is defined by the trigger script attached to each of the objects. The LevelManager also contains a boolean array which holds a state, which is set to true if the trigger component have been set to true at any point. The trigger component serves the purpose of whether they are allowed to be interacted with by a player or not, and also stores information about whether a player has interacted with the object it is attached to. The trigger component gets information about its status from other objects in the game, as it's single purpose is to control the current status.

The GameManager detects what objects have been triggered, how many there are of them, and which sequence of triggers is currently in play. It sets the first objects in the level ready to be interacted with. It checks for objects in a specific sequence and if they have a desired order. It checks if players trigger the objects in the current sequence, and whether they fail to trigger the correct objects in the sequence. If they press the wrong objects the GameManager resets the triggers. It can be seen what happens when a player interacts with an object on figure 64. The state of the trigger is sent to the server, and it runs the GameManager, and sends trigger information back to the clients.

It has been divided into several different parts in order to optimize it. The parts it has been divided into are:

- Start
- SetNewLevelManager

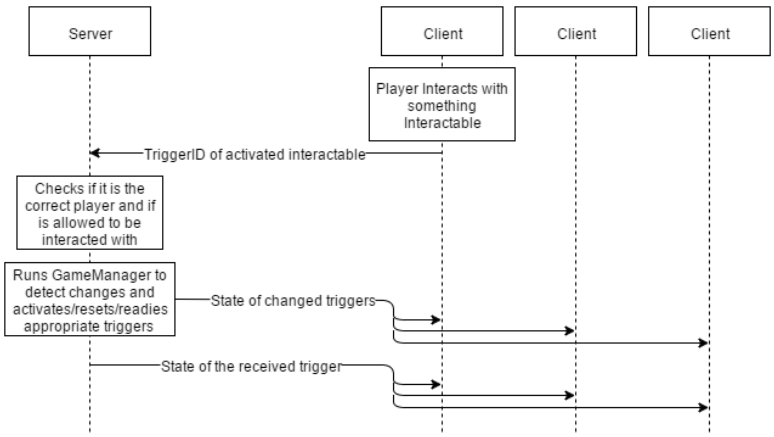



Figure 64: Activity diagram of what is sent when a player interacts with an object.

- DetectTriggerChanges
- ResetTriggers

In the start method the GameManager goes through and fetches serverManager, LevelManager and a LevelHandler. which is needed to check what level is currently being used and getting the objects for the level. SetNewLevelManager is used every time a new level is started. It sets a new LevelManager and goes through the first sequence in the level and sets the object ready to be triggered. This is needed in order to start each level as the objects otherwise could not be activated. The first thing DetectTriggerChanges does is to check if there is a desired order in the current sequence. If there is no order in which the objects needs to be triggered, it simply goes through the objects in the sequence and detects if they all are triggered. If objects in the sequence are required to be reset, then it goes through the current objects and makes them ready to be triggered again, and sets the trigger component's trigger to false. If there is a desired order in the sequence, the GameManager goes through several for loops to detect the previous objects, the current and next in the sequence. The GameManager first goes through the objects which should be triggered based on the order and sequence, and if they are triggered it goes to the next order of objects in the sequence. If the previous or next objects are triggered, the GameManager resets the triggers, which will be explained how, last in the Trigger system. It then goes through and checks all the objects in the sequence and if they all are triggered it proceeds to the next sequence. After a given delay, the ResetTriggers goes through all the objects in the current sequence and resets the trigger component to the original state. It also goes through the boolean array in the LevelManager which holds the trigger states, and sets the corresponding states to false. After each trigger change has been detected, the GameManager tells the ServerManager what happened.

The TriggerHandler is specifically for receiving and setting triggers for both the client and server. An issue occurred when syncing the mirrors and elevators, because of how the Trigger System was set up and synced, when some packets were lost the objects would not be placed and rotated in sync. To hotfix this up to the test, an additional system was appended to the triggers that would send the state of triggers(for example the position of the elevators) from the server to the clients continuously every two seconds, this was only enabled when needed.

In order to send information over the network through DarkRift the data needs to be serialized, meaning converting to bytes. For serializing two convenient extensions for Vector3 and Quaternion to serialize them into byte arrays. Then a separate Deserializer class for deserializing them back again.

### Voice Communication

It was decided that the players should be able to communicate in-game by using microphones. This was mainly necessary because of the actual physical distance between each player and the headsets that the players needed to wear were somewhat noise cancelling, meaning that they would probably have a hard time hearing each other speak without a Voice over Internet Protocol (VoIP) application

An attempt was made to create in-game voice communication, using VoiceChat [Reference] a Unity plugin originally made by Fredrik Holmström but forked and updated to newer version of Unity by Lucas Vasconcelos. VoiceChat can either use Unity's own Networking System (UNet) or it can be used with own Networking Implementation. As the current implementation of the game uses DarkRift, an integration for VoiceChat with DarkRift was planned.

In order for VoiceChat to work a network id have to be given to the VoiceChatRecorder, as it creates VoiceChatPackage(s), which is a struct with the voice data, Compression type, network id and packetid. Such that when a package have to be played, you can know which client it came from and have an individual VoiceChatPlayer for each client. This was one of the advantages of choosing a built-in VoIP, as it would be possible to spatialize the audio between the players.

Using the networking subjects a new voice package ushort is defined. Whenever the NetPlayerSync is receiving a VoiceChatPacket it then sends the data to a reference of VoiceChatPlayer's function called OnNewSample. This will make sure to queue the different voice clips

and play them in correct order.

Although this was implemented there were a lot of troubles with DarkRift not being able to throughput all the data from all the clients and there were a lot of trouble Serializing the VoiceChatPacket's - In which DarkRift would discard some of the packages. This both created a lot of performance issues and when multiple client were connected it would create a lot of artifacts.

Therefor Teamspeak were used instead as an alternative solution - But for future works, VoiceChat would be fixed.

## 5.8 Usability Tests

**First usability test**

A few simple levels of the game were created that introduced some of the mechanics that were to be implemented. These simple levels were merely created for usability testing, since it was imperative to figure out if people could understand the mechanics that were created for the game. The primary focus in the first usability test was to test basic functionality, such as movement, focusing the headlight, and activating buttons etc.

The networking part of the game was not ready in the first usability tests, which resulted in only testing one person at the time. The levels could of course not be designed exactly like we imagined the final levels would look like, since there was only one person playing at the time, and these final levels would require three persons to be able to progress in the game. Though, the usability test levels were constructed in a way that was as similar to the final designs, as this could maybe give some ideas on what could be improved in the puzzle designs.

The usability test was conducted at Aalborg University Copenhagen (AAU-CPH), and the test participants were mostly Medialogy students. Each test participant was given a short introduction on how to put on the Oculus Rift, and how to move back and forth in the game. They had to figure out the rest of the game's workings themselves, as the test's purpose was to see if the game was intuitive enough. They also had to answer a questionnaire after the test. There were only six participants, but they were very helpful as we managed to get answers for most of our questions. The whole questionnaire can be seen in appendix B. figure 65 and 66 are two examples of some of the results we acquired.

Some of the features in the game seem simple enough to its creators, but figure 65 shows that

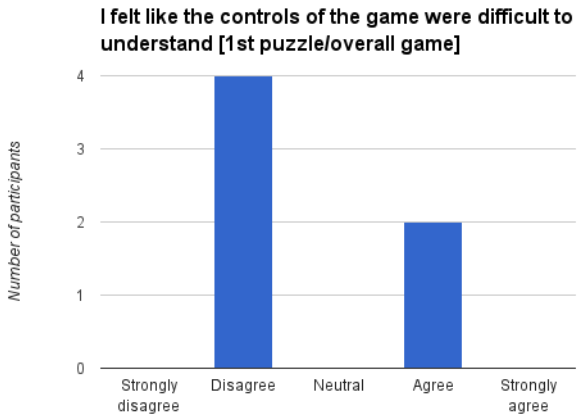



Figure 65: A question about an interactable sound crystal in the game.

none of the participants understood that they could interact with a so called "sound crystal", which was present in the usability test. The sound crystal is essential to the game's puzzle, as it is very hard to solve one of the presented puzzles without its usage.

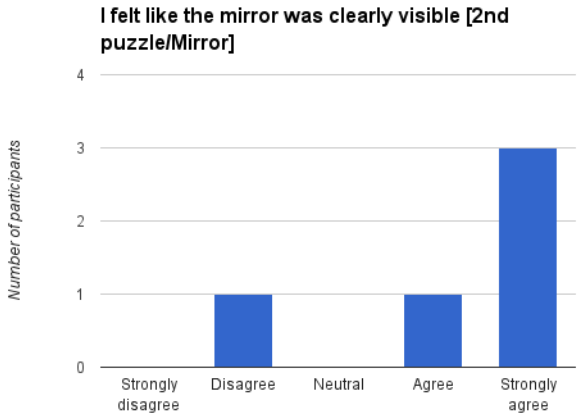



Figure 66: A question about whether or not the controls of the game were difficult to understand.

Figure 66 represents the results of a question regarding the controls in the game. The participants did not have free forward and backwards movement, in the game at this point, which meant that they moved from one fixed point to another fixed point each time they wanted to move either direction. It seemed frustrating for some of the test participants. figure 66 also shows how two participants agreed that the controls were difficult to understand. It could be caused by the novel experience of trying VR and using an Oculus Rift as a controller for the first time. Observations were used during the test, see appendix C for observation

notes. The observations showed that some of the people did not completely understand the concept of rotating their head in order to look around in the game. This may be caused by the lack of prior experience with VR and HMDs.

One of the levels had a mirror, which the participants had to interact with. Observations showed that almost all of the participants had problems noticing the mirror. Most noticed it as soon as they accidentally focused their headlight on the mirror, giving off a visible reflection.

This first usability test made the realization that some of the interactable objects had no feedback designed for them, like the sound crystal. It was not clear enough for the test participants how they were supposed to rotate a mirror and reflect their own lightsource in the mirror and onto a button. The movement of the characters had to be less frustrating. The focus was to improve these problems before the next usability test, so that future participants would hopefully provide us feedback on other possible problematic aspects of the game for the next test.

#### Second usability test

The second usability test was very similar to the first, since the only difference was that more final features had been implemented in the game, as for example sound and a new sound puzzle level along with other improvements. This usability test was also conducted at AAU-CPH and consisted of five test participants. Each participant was observed and had to answer a questionnaire. The results from the questionnaires can been seen in appendix D.

Some mechanics had been improved since the previous usability test and were tested again. An example being the mirror. In the previous test, the users barely noticed it, while in the implementation used for this usability test, it had been improved.

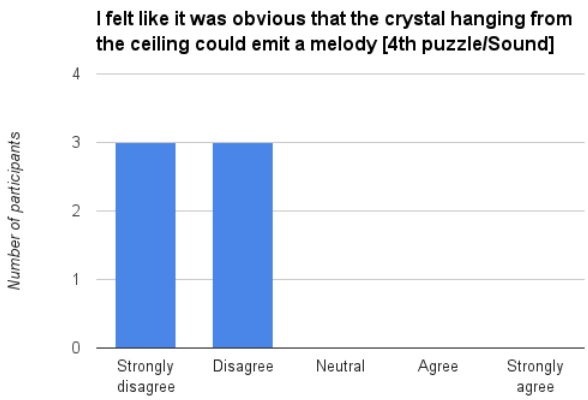



Figure 67: Question regarding the visibility of the mirror.

Figure 67 shows that only one user said the mirror was not clearly visible, and the observations from this test also approved that the participants noticed the mirror almost right away, which was a great improvement compared to the first usability test.

Sound had been implemented for this test, but there were not much feedback on it. The participants explained that the volume of two of the sounds used were high volume, while another of the used sound was too low in volume. The lack of feedback could also mean that the participants had an overall good experience and had no further comment on it.

# 6 Test method

The test was set up in two groups; VR and Non-VR. Each included several groups consisting of 3 participants. All participants were provided with a computer, controller, headset, microphone, and the VR group participants also had an Oculus Rift Development Kit 2 each. Each group also had a Teamspeak setup that the participants communicated through verbally. During testing, one test participant's computer screen was recorded, together with all the test group's communication. Test observants were set to observe each individual's screen and take notes of their behaviour on a checklist.

# 7 Results

## 7.1 Prepared data

The following section will present the data gathered. Each question in the observation sheet has been weighed from -2 to 2 according to our own judgement of how important they are for teamwork in a group.

Table 1: Weighing of the questions

| **Teamwork** | **Weight** |
|---|---|
| Did they communicate about the puzzle | 2 |
| Did they smalltalk(not about the puzzle) | 1 |
| Did they share information(what they observe/do/understand) | 2 |
| Did they discuss their approach(discuss misunderstandings/their POV) | 2 |
| Did anyone take the role of leader | 0 |
| Did he dictate? | -2 |
| Did he listen to feedback? | 1 |
| Did they negotiate / solve as a group | 2 |
| Did they refer to objects in the scene while explaining | 1 |

The recorded data was observed by 4 different people in each group and afterwards their average was found to avoid subjective opinions. Then the average for each group, depending on the weight of each question, was conducted:

Table 2: Average from each group. Higher points equals better teamwork rating.

| **Group** | **Group with Virtual Reality (G_VR)** | **Group without Virtual Reality (G_NVR)** |
|---|---|---|
| **1** | 3.44 | 3.89 |
| **2** | 4.19 | 3.75 |
| **3** | 3.86 | 2.92 |
| **4** | 2.61 | 3.69 |
| **5** | 3.42 | 3.83 |
| **6** | 4.03 | 3.72 |

The histogram below (figure 68) depicts the data in bins, with the blue columns being the experimental group and the red ones are the control group.

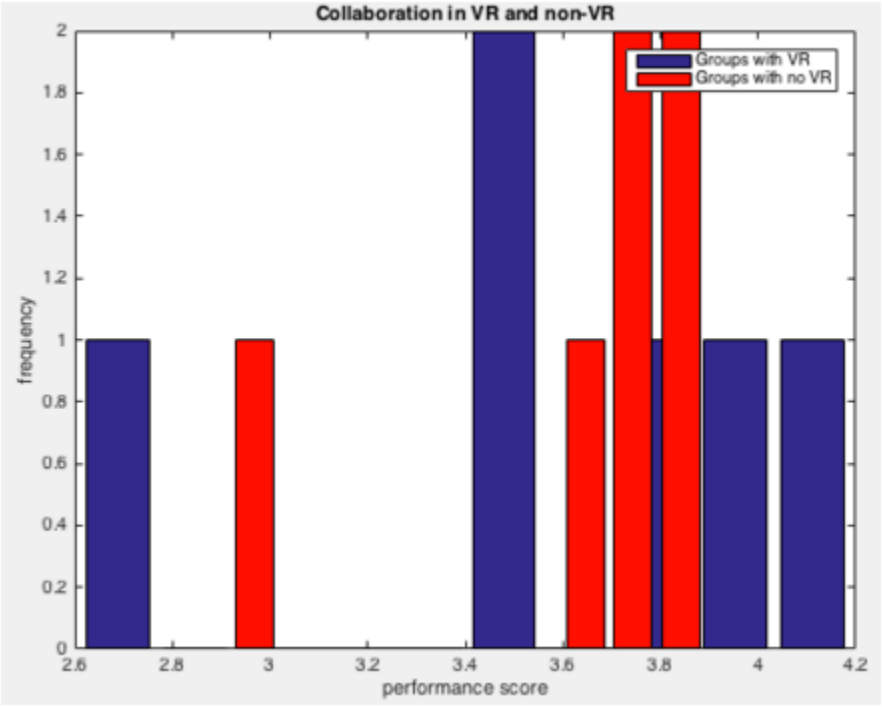



Figure 68: Histogram of the data distribution

## 7.2 Analysis of data

### Measurement level

Due to the used point system, it is judged that the data set roughly fulfills the requirements for interval level data since the points can be ordered, and they can be said to have a set interval between points.

### Normality

Descriptive analysis of the data was performed to get general information.

Table 3: Descriptive data for the experimental- & control group

| Test Type | Mean | Standard Deviation | Anderson Darling Test | Standard Error |
|---|---|---|---|---|
| **VR** | 3.591 | 0.572 | 0 | 0.233 |
| **Non-VR** | 3.633 | 0.357 | 0 | 0.145 |

Both groups fail to reject the null hypothesis in the Anderson Darling Test, meaning that according to the Anderson Darling Test, the samples are normally distributed.

### Variance

```
Levene's statistic (absolute)    1.45601
Degrees of freedom                 1, 10
p-value                          0.25534
```

Figure 69: Output from Levene's test of homogeneity of variance

The p-value of 0.255 provided by Levene's test is higher than the significance level of 0.05, so it can be assumed that the samples are drawn from the same variance.

On the basis of these results the given data fulfills all of the prerequisites for parametric data.

### Test results

The Wilcoxon Rank-Sum Test is a non-parametric test, which will check if the mean of a group varies from the other, and how much. The following results were given from the rank-sum test.

Table 4: Test results from Wilcoxon Rank-Sum Test

| **p-value** | **h** | **ranksum** |
|---|---|---|
| 0.9372 | 0 | 38 |

A Paired-Sample T-Test, which is a parametric equivalent to the Wilcoxon Test, gives the following results.

Table 5: Results from Paired-Sample T-Test

| **h** | **p-value** | **ci** | **tstat** | **df** | **sd** |
|---|---|---|---|---|---|
| 0 | 0.8950 | -0.7295<br>0.8128 | 0.1389 | 5 | 0.7348 |

## 7.3 Conclusion

We can see two similar outcomes from both the non-parametric test and the parametric test:

- From the Wilcoxon Rank-Sum Test there was found no significant difference between the experimental (M = 3.591, SE = 0.233) and control (M = 3.633, SE = 0.145) groups collaborative abilities (W(5) = 38, p = 0.9372, h = 0).
- From the Paired-Sample T-Test there was found no significant difference between the experimental (M = 3.591, SD = 0.572) and control (M = 3.633, SD = 0.357) groups collaborative abilities (t(5) = 0.1389, p-value = 0.8950).

# Part II

# Individual Group

# 8 Introduction

What has previously been presented so far was made as the PGP group that consisted of 19 people, however, from this point the report is presenting the study of a six person group. The focus of the project ahead will still be related to what has been written in the previous chapters, since the game that was created will still be used to investigate a more specific subject in VR.

After the PGP group tested for collaboration in VR, the sub-group became interested in delving more into the subject of communication within virtual reality in a collaborative environment. The joint analysis chapter already mentioned different ways of communicating, for example face-to-face communication and textual communication. Some of them have advantages in certain situations, and these are needed for a group to manage their efforts as a collaborative team. The report is not going to investigate which method is better for communicating in VR, but is going to investigate whether the communication effort in a collaborative environment can be affected by VR.

Therefore, the following problem statement was created:

*"Does virtual reality have an effect on group communication effort in a collaborative environment?"*

# 9 Analysis

This section is built on top of the previously presented knowledge from the joint analysis chapter. Special attention should be brought to section 4.1 Communication in Teamwork, as well as section 4.4 Measuring Success and Collaboration in Video Games, and 4.5 Communication and Collaboration. The following section will explain several aspects of collaboration from the viewpoint of Dillenbourg (1999). This chapter is used to help define what collaboration and collaborative interactions are.

## 9.1 Collaboration

Collaboration can be explained as a situation in which several people are working towards a common goal (Dillenbourg, 1999). Dillenbourg tries to answer the question of what collaboration entails and in which situations collaboration is likely to occur, in order to figure out what is meant by collaborative learning. During this investigation Dillenbourg proposes several aspects of collaboration. Dillenbourg explains that for a situation to be collaborative, four criteria must be fulfilled. The first two of these that the collaborators are equal and are capable of the same actions. To these he explains three levels of symmetry.

- Symmetry of action, the notion that all collaborators are capable of performing the same actions.
- Symmetry of knowledge, that collaborators have the same level of knowledge.
- Symmetry of status, that no hierarchical order is present between the collaborators.

The last two criteria are that the collaborators share a common goal, and that they work together to achieve this. These criteria mean that it is not any situation that can be defined as collaborative.

### Collaborative interactions

Dillenbourg also present the approach of viewing collaboration as interactions between collaborators. For this Dillenbourg propose another set of criteria; interaction, synchronization and negotiation (Dillenbourg, 1999).

**Interaction** Since a collaborative situation requires the common work of multiple people, it also requires that the sense of interaction is present. For this Dillenbourg (1999) explains

that it is not the frequency of interactions that counts, but rather the nature of the interaction itself. More precisely, the interactions of the collaborators must affect each other's approach and actions (Dillenbourg, 1999). Say for example two students are tasked with writing a report together. In one example, each students simply writes their part of the report, and from time to time interact to compare their work. In this case, the students can be said to share a common goal, i.e. to finish the report. Though the work division they have set has left little to no possibility for interaction with each other, and as thus won't necessarily be a collaborative interaction. On the other hand, if the students together device a disposition for each chapter, and together write the work in a manner based on these dispositions. Each student now has a direct influence on the others performed actions and work.

**Synchronization** Synchronization as a criteria refers to the collaborators communicating synchronously. The boundary to when something refers to a synchronous- or an asynchronous communication is hard to define. Dillenbourg explains synchronicity as less of a technical aspect and more of a social one. A collaborator may expect his co-collaborator to listen and answer to what is said to him. For this to be a possibility it requires that the communication medium affords such an interaction. The conversational medium must inherently allow for direct conversation. For example, a chat system (e.g. instant messaging) or a phone call (Dillenbourg, 1999).

**Negotiation** Another criteria of the collaborative interaction, is the fact that the interaction must be negotiable. This is one of the reasons why collaborators have to be of symmetrical status, as little to no negotiation would be present in a hierarchical situation. This means that a collaborative situation would have to afford negotiation, meaning that there needs to be something to negotiate about, and the collaborators would need to have a symmetry of actions.

The point is that a collaborator should have the ability to argue for his viewpoint as well as actions, this way the collaborators mutually gain more influence in each other's actions and their approach to the task at hand (Dillenbourg, 1999).

Throughout this section, collaboration was presented through the viewpoint of Dillenbourg (1999). A set of criteria were presented and explained. This was done so to create a basis of understanding for the problem statement at hand.

# 10 Evaluation

The following chapter will present how the tests were conducted and a brief rundown of them, afterwards the test results will be presented along with graphs visualising the data. Finally, statistical analyses will be performed, on the results and the data will be showed.

## 10.1 Test Rundown

The first half of the results was from a test conducted at Aalborg University Copenhagen on November 23rd. The second half was conducted as the same location on December 9th and 10th. Both of the tests were conducted in a room secluded from other humans, so the test participants and conductors were isolated. The two tests were conducted in a similar manner to each other in order for them to produce the same kinds of data.

For a description of the first half of the test, see section 6. The second test was made on the same base as the first though it only had one facilitator and one technician present. Because of this only one group was tested at a time in the second test.

In both tests the data was collected using the screen recording software 'Open Broadcaster Software', this recorded the point of view of a single participant along with the verbal communication from the group as a whole, all recordings can be found on the attached disk. To complement the screen recordings the participants were asked to complete a questionnaire afterwards (Appendix E & F).

In the first test, there were seven groups, each consisting of three participants in each of the two categories, namely non-VR and VR, making a total of 14 groups and 42 participants. However, because of complications and possible bias, the last group from each side was discarded. In the second test there were four groups in each of the two categories, making it a total of eight groups and 24 participants. In all, ten groups in each category were tested, making it a total of 60 participants.

## 10.2 Communicative Effort

During the tests a working definition called 'Communicative Effort' was used. This definition was based on knowledge gained from the analysis chapter. The definition of communicative effort, in the context of a collaborative task, encompass either of the following points:

1. The sharing of information between individuals (Salas et al., 2005).

2. Negotiation or discussion (Dillenbourg, 1999).
3. Synchronicity (Dillenbourg, 1999).

This definition helped limit what communication that was measured, to only communication relevant for the participants collaborative tasks at hand.

## 10.3 Data Gathering Methods

The recordings from the tests were analysed to gather the data that was needed. Every recording was analysed, and a time stamp was made every time a communicative effort was made. The time spoken is calculated to percentages based on the full play time. When the timings of all the groups were noted, they were double checked and looked through, a few were checked for a third time when a timing deviation was found too large. Afterwards, the mean of the two closest was calculated.

## 10.4 Test Results

The hypotheses that were used during the testing can be found below.

- $H_0$ - VR has no effect on a group's communication effort in a collaborative environment
- $H_1$ - VR has an effect on a group's communication effort in a collaborative environment

### Percentage Spoken

Based on the data gathered from the screen recordings during the tests the percentage of play time used speaking, was calculated.

Figure 70 shows the VR results for the calculated percentages, while 71 shows the results from the non-VR groups. For the VR group a mean of 43.37%, a standard deviation of 7.66% and a standard error of 2.42 were calculated. While for the non-VR group has a mean of 48.90%, a standard deviation of 9.95% and a standard error of 3.15 (Appendix G).

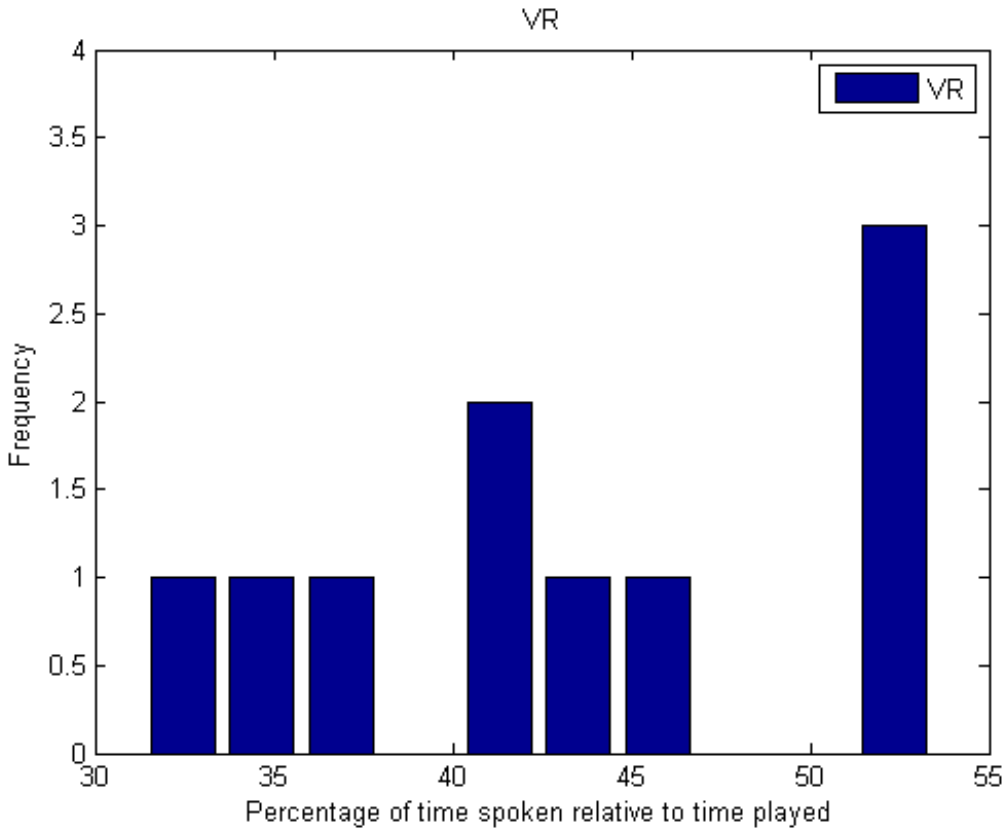



Figure 70: Percentage time used speaking for VR groups.

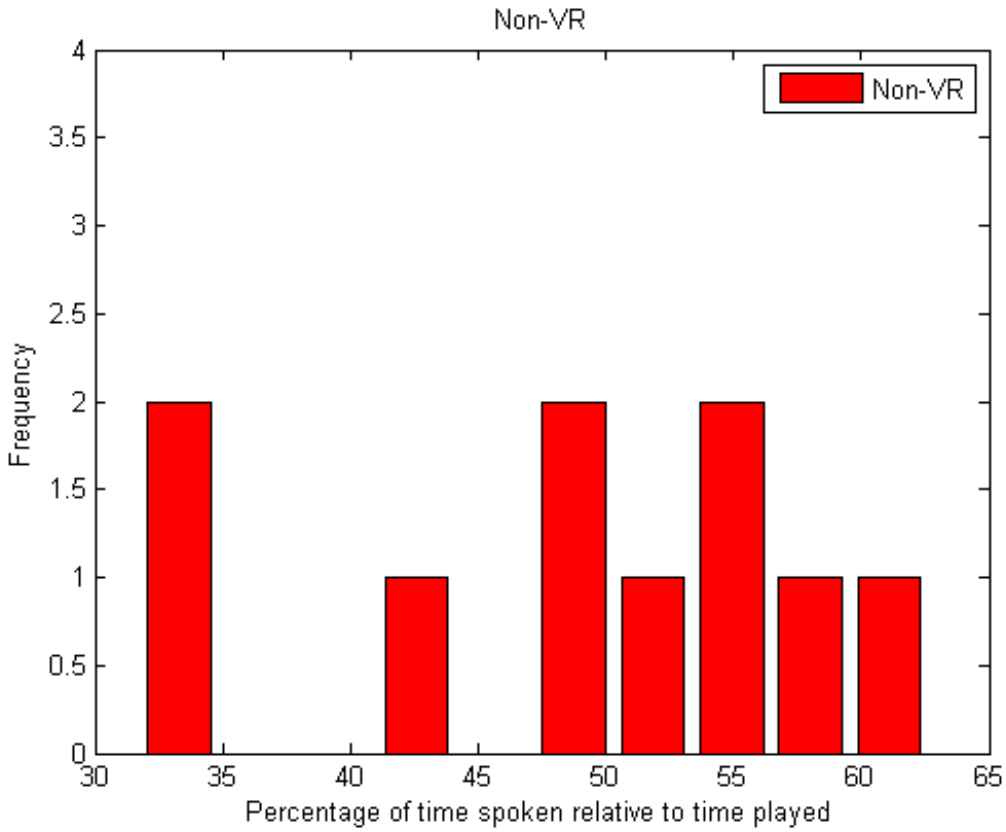



Figure 71: Percentage time used speaking for non-VR Groups.

### Time Played

Figure 72 shows the measured time used for each group playing and solving the game's puzzles. For the VR groups a mean of 18.46, a standard deviation of 6.12 and a standard error of 1.96, were calculated. For the non-VR groups a mean of 14.89, a standard deviation of 4.30 and a standard error of 1.36 was calculated (Appendix G).

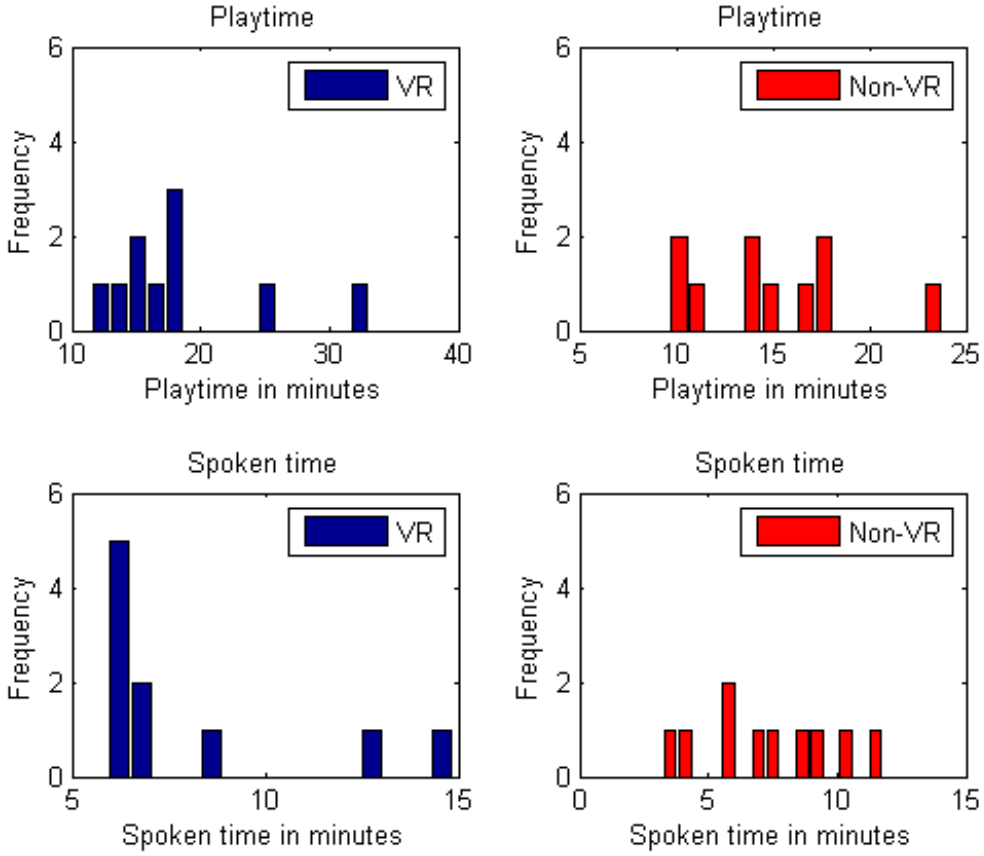



Figure 72: Spoken and played time data

### Time Spoken

Figure 72 also shows the measured time spoken in minutes within the VR groups and non-VR groups. For the VR groups a mean of 8.03, a standard deviation of 3.23 and a standard error of 1.02 was calculated. For the non-VR groups a mean of 7.33 and a standard deviation of 2.64 and a standard error of 0.83 was calculated (Appendix G).

## 10.5 Questionnaire Results

The following graphs are based on the data gathered from the questionnaires given to each participant at the end of the tests (Appendix H & I).

In figure 73 and 74 the participants answers based on the questions, “How often do you

participate in a collaborative task?" and "How would you evaluate your own collaborative level?", are shown, these answers are based on their own opinion about how often they partake in collaborative tasks and their collaborative level.

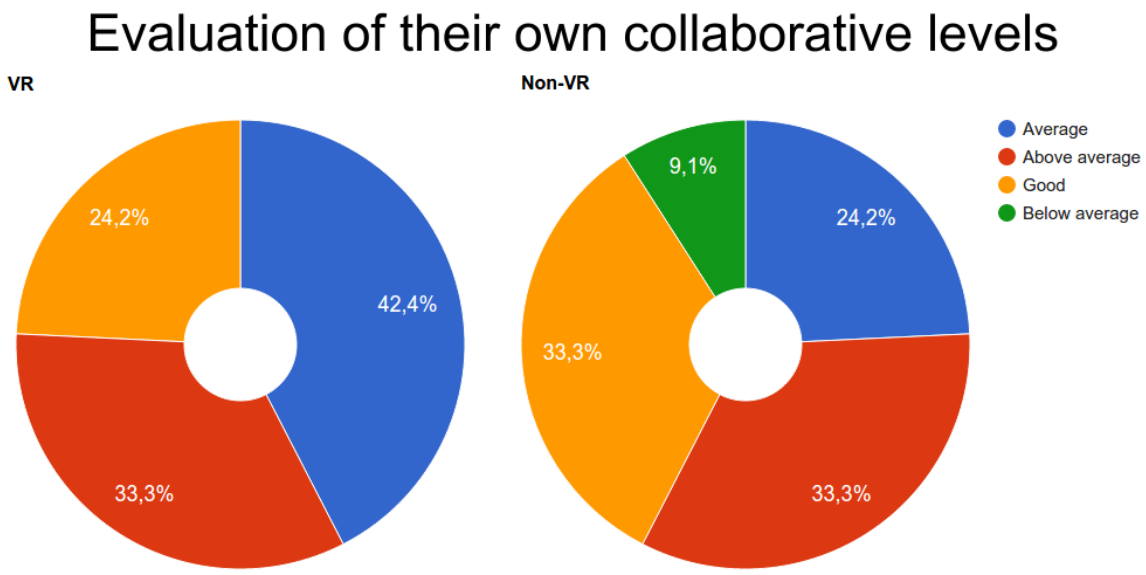



Figure 73: How each participant within the VR and non-VR groups evaluated how often they participate in collaborative tasks

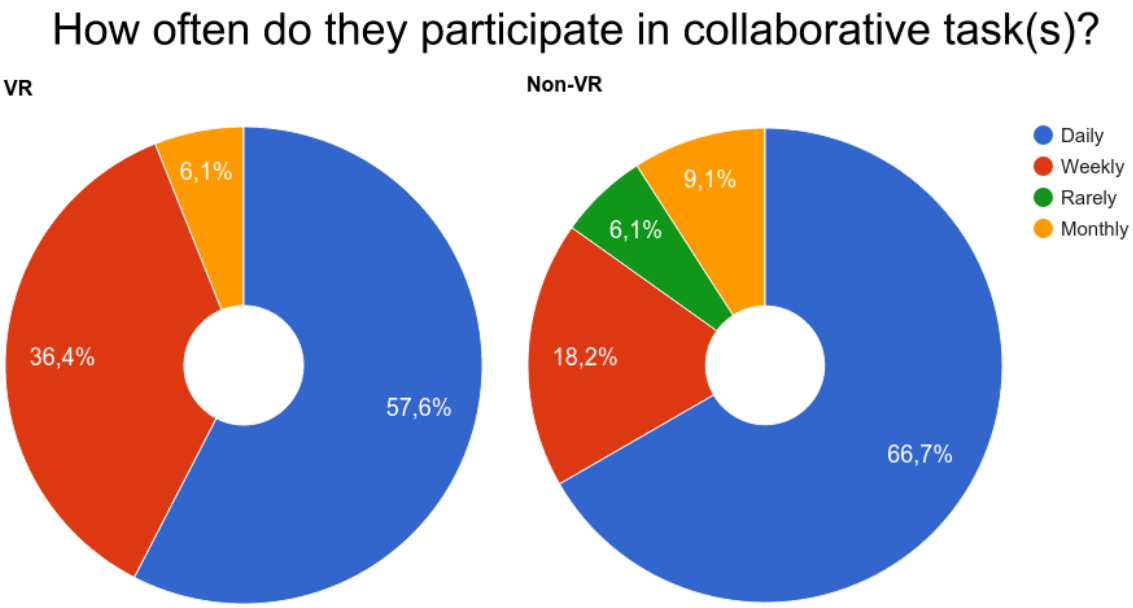



Figure 74: How each participant within the VR and non-VR groups evaluated their own collaborative abilities

### Age

In figure 75 and 76 the different test participants age span is shown, which varies from 19 to 47 (Appendix H & I).

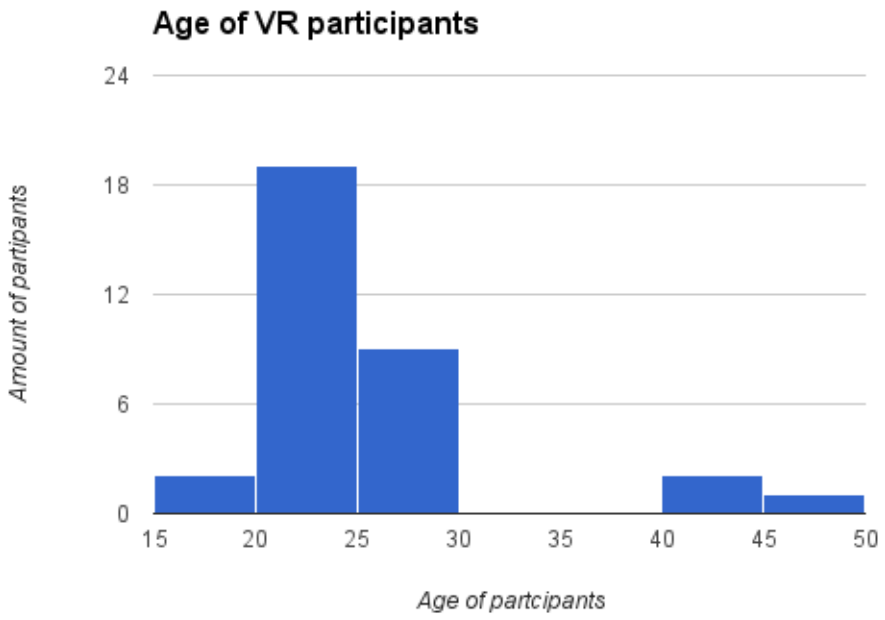



Figure 75: Distribution of participant ages within the VR groups.

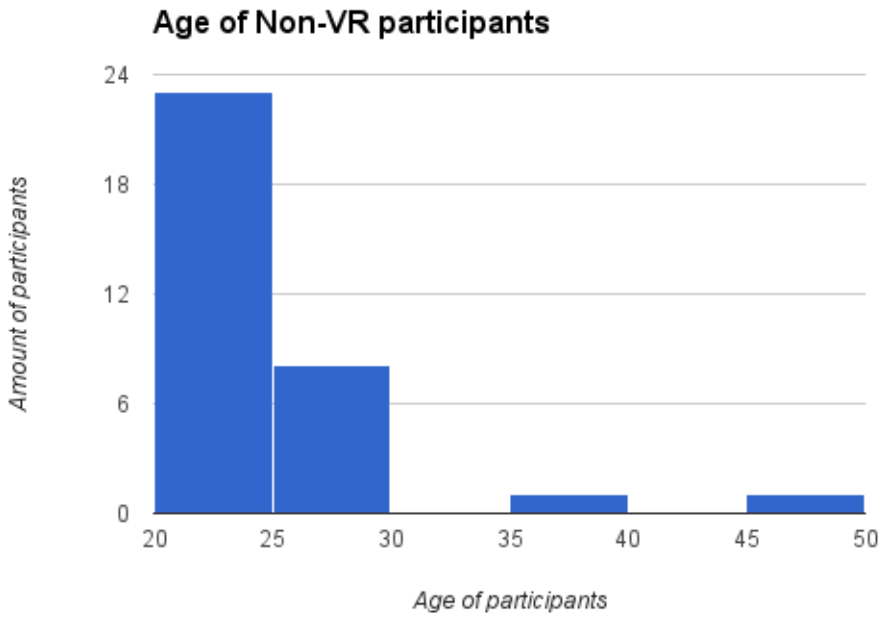



Figure 76: Distribution of participant ages within the VR groups.

### Previous Experience with VR

In total 14 out of 30 participants had no prior experience with VR.

### Outlier

The average play time of all the VR groups were 18.68 minutes, where the non-VR groups had an average time of 18.11 minutes. However, the VR category had an outlier with a play time of 24.55 minutes. If this group is exluded then the average play time of the VR category falls to 15.82 minutes.

### Nausea

Out of the 30 participants that had tested the VR version, eight noted that they felt sick or nauseous at some point during the test. The groups that had at least one nauseous participants spoke 42.91% of the time, where groups without any nauseous participants spoke 44.46% of the time.

## 10.6 Student's t-Test

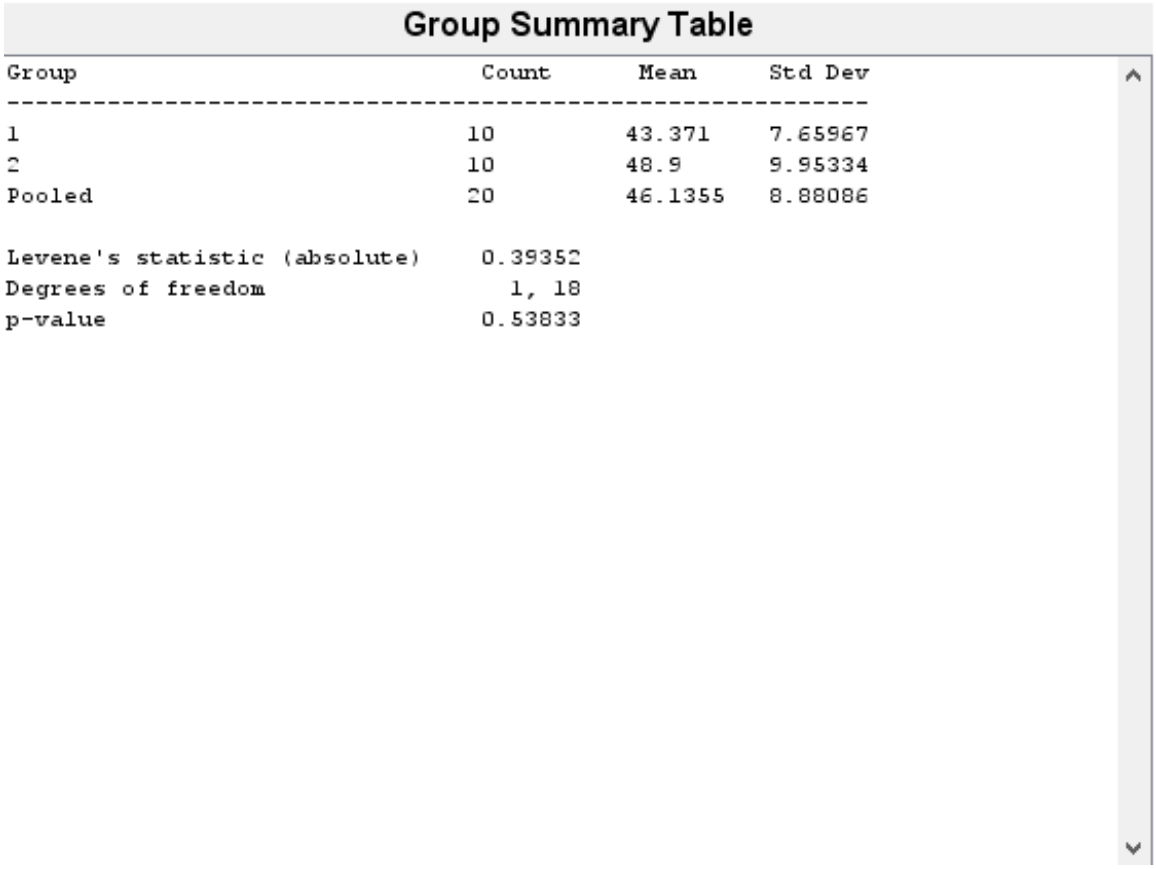
Since the data consists of results from only two groups a student's t-test has been decided to use to process the data. Since the hypothesis is simply looking for a difference without any assumptions, this t-test is going to be two tailed. Before using the t-test, the data must be proven to be parametric, which is tested by looking at three aspects of the data. First, it is necessary to figure out what the level of measurement is. Then whether the data fulfills the assumption of homogeneity of variance (Field & Hole, 2003). Lastly, both sets will have to be normally distributed.

### Level of Measurement

In order to utilize a t-test, it is necessary for the data to have a high level of measurement, for example, interval or ratio. As the data gathered is time, there is a zero value and measurements are divisible, the data fulfills all the prerequisites for interval level measurements, and more, so it is fair to claim that the data gathered from these tests would be of ratio level.

### Variance

The variance of the two data sets needs to be similar in order for them to be parametric. The variance tells how much variation there is between each test participant's score, i.e. percentage of time spoken relative to playtime. If the variance is small, it can be assumed that the variation is the same in both sample groups. The Levene's test is used to test homogeneity of variance. The test results from the Levene's test can be seen in figure 77, it shows a p-value of 0.53833, this means that the Levene's null hypothesis is unable to be rejected and it can therefore be assumed that there is no significant difference in the variance.



**Group Summary Table**

| Group | Count | Mean | Std Dev |
|---|---|---|---|
| 1 | 10 | 43.371 | 7.65967 |
| 2 | 10 | 48.9 | 9.95334 |
| Pooled | 20 | 46.1355 | 8.88086 |

| | |
|---|---|
| Levene's statistic (absolute) | 0.39352 |
| Degrees of freedom | 1, 18 |
| p-value | 0.53833 |

Figure 77: Results from the Levene's test.

### Normal Distribution

To test whether the data is normally distributed, Q-Q plots for each dataset was made. The first Q-Q plot (figure 78) is from the non-VR data set, while the second Q-Q plot (figure 79) is from the VR data set. While both plots do not perfectly follow a normal distribution they both seem to roughly do so. In addition to the Q-Q plots Anderson Darling test was utilized. The Anderson Darling test challenges the null hypothesis that the data set in question is following a normal distribution.

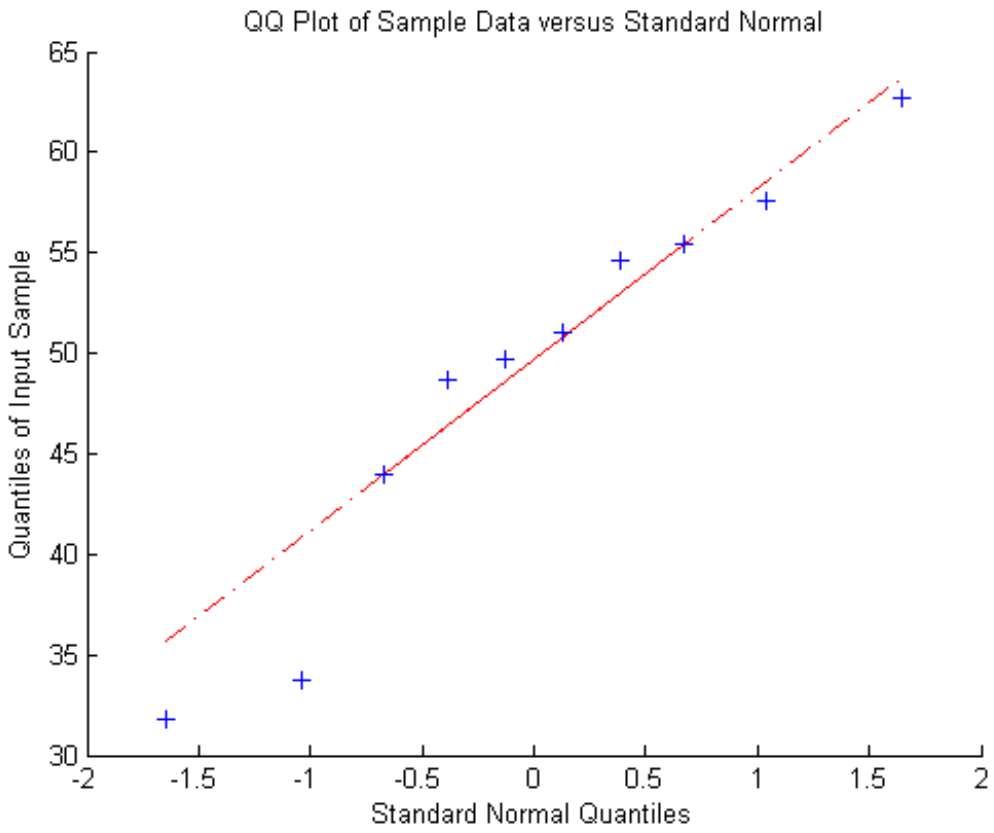



Figure 78: Q-Q plot for non-VR data set.

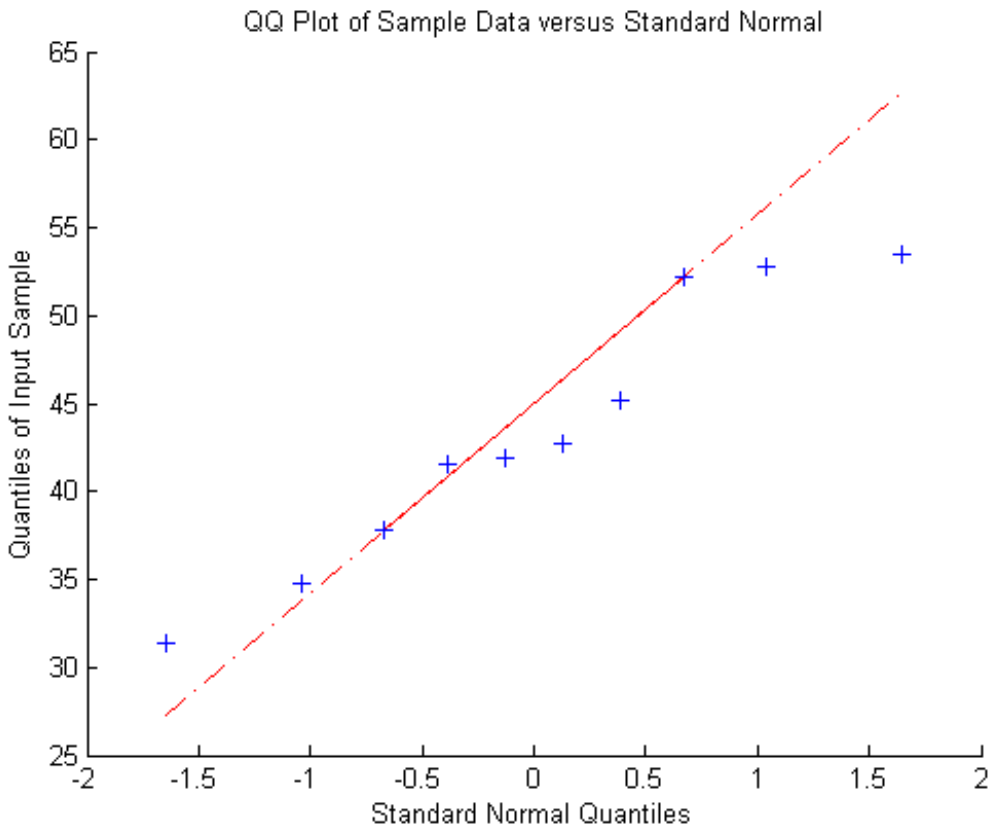



Figure 79: Q-Q plot for VR data set.

### Anderson Darling

The results from the Anderson Darling test for the VR groups returns a p-value of 0.53. This value is well above the significance value of 0.05. For the non-VR group the p-value is 0.42 which is also well above the significance value. This means that the null hypothesis cannot be rejected for either data set, therefore it is assumed that both of the data sets follow a normal distribution.

## 10.7 Results

The result from the two tailed t-test is a standard deviation of 8.89, a degrees of freedom of 18 and a p-value of 0.18. The p-value is well above the significance level so the null hypothesis, that VR has an effect on a group's communication effort in a collaborative environment, cannot be rejected ($t(18) = -1.3921, p > 0.05, r = 0.31$).

# 11 Discussion

## 11.1 Student's T-Test

Based on the results from the student's t-test in the evaluation chapter, it was not possible to reject the null hypothesis since the p-value ($p = 0.18$) is way above the significance level of 0.05. The reason for this might be due to the size of the sample size, a larger sample size could have shown no correlation with a higher certainty. This can be seen with the calculated effect size which is 0.31, which should indicate a medium sized effect. The sample size used for the test would be too small for a medium effect. Furthermore, there are two possible bias' with the data gathered from the two tests. First, the data is collected from the two tests might have come to existence in different ways. The facilitator might have been more elaborate in one of the tests, but there has been taken measure to ensure that they have performed as similar as possible, however it is possible that there might have been a small divergence between the two tests. One of the bias' could be that one of the tests were more isolated physically than the other. In the first test the only thing separating the two groups (VR and non-VR) was a partition, and while listening to the recordings you are sometimes able to hear the other test group. In the second test they were completely isolated, meaning that they were in a small room where only 1 group could be at a time, so it was impossible to be disturbed by external noise.

## 11.2 Direct Effect

The following section will discuss some of the factors that could have had a direct effect on the result of the statistical operations performed on the data.

### Previous VR Experience

Even though the statistical calculations performed on the data sets shows that there is no significant difference on group communication effort in a collaborative environment in VR compared to non-VR; the raw numbers from the data set might present another result, or help explain the result that was reached.

The groups that tested VR spent 43.37% of the time talking compared to the non-VR groups who spent 48.90% of the time talking. While a difference in 5% does not seem like much, when looking at the raw numbers the VR groups spoke on average 8.03 minutes over 18.46 minutes

of play time, where the non-VR groups spoke on average 7.33 minutes over 14.86 minutes of play time. This is a difference of 3.57 minutes more playtime for the VR category.

This could be explained by the fact that 14 out of 30 of the VR participants had no previous experience with VR, which could have made them more focused on the new experience instead of focusing on the task of solving the puzzles. In total there were six groups that had at least one participant who had prior experience with VR, while there were four groups that had no participants who had experience with VR before the test. The groups with prior experience spoke on average 7.99 minutes over 18.69 minutes, where the non-VR groups spoke 8.09 minutes over 18.11 minutes of play time. This points towards a contradiction of the previous statement, that if participants had prior experience with VR then they would complete the game faster, i.e. have a lower playtime. However, the VR category had an outlier that differed significantly from the average of the category. Excluding this group shows a different result; the average spoken time went down to 5.51 minutes over 15.83 minutes of play time. If this group was excluded, then the difference between play times could be because of VR being a new experience to 14 of 30 participants. This would also skew the results of the t-Test in favour of the non-VR group, since this would lower the spoken percentages of the VR group.

### Nausea

In total eight participants spread among seven groups felt that the VR experience gave them nausea, which is 26% of the entire VR category (30 participants). It was hypothesized that there might be a correlation between how much a group spoke and how many in the group felt nausea.The seven groups spent on average 42.91% of their play time speaking with each other, which is only a slight decrease of 0.47% compared to the overall mean. The groups that did not have a nauseous participant had a mean of 44.46%, which is an increase of 1.09% compared to the overall mean. Given how few groups actually makes up these means, more groups would have to be tested in order to make a definite conclusion on whether this had an affect on the result of the test or not.

### Measurement Method

The used measurement method shows the percentage of time used speaking, over the full time played for each group. The way that the data has been measured assumes that the groups spoke evenly over their playtime. This inherently has a problem as the game is split

into several levels and it is entirely possible that each level may have different communicative needs. For example if it turns out that the non-VR groups had used significantly more time in the last puzzle than the VR groups, then it would heavily affect the gathered results if that same puzzle also required more communication to effectively complete. This problem could have been avoided if for each group, each level was measured independently. This would have resulted in four means each for the VR and non-VR groups (if the three introductory levels are counted as one). These four means could then have resulted in a more precise conclusion. This would only be necessary if each level did require differing amounts of communicative effort, which should have been tested beforehand to see if this is the case.

The used way of measuring the communication between the test participants was to record their play session along with the verbal communication between them. The videos were then analysed and the rate of communication between participants was done by timing verbal activity. In order to acquire the most precision of these results, each video was played through at least twice by different people, so that each analysis was individual and an overall difference between the observations would be determined. If the difference between the observations were under 10%, it would be reevaluated as an applicable result for further evaluation. There was the alternative method of taking the videos and counting the total words said, but there were a few problems with using that method. While trying this method on a recording, it was discovered that the results were inaccurate, and it was difficult to distinguish important words (words in relation to solving a puzzle or the game in general) from the unimportant ones (other general communication such as teasing, personal compliments, etc.). The main reason for this was that the audio recorded put all of the participants voice channels into one recording, and it was often the case that multiple people would be talking at the same time, making measurements difficult. This problem could be avoided by separating each test group participant's microphone audio feed into their own separate channels, and record those individually. Essentially, the method of analysing time spoken was used because it proved to be the easier alternative.

**Indirect Effect**

The questionnaire from the tests provided information on how each participant rated their own collaborative abilities, where they also were asked how often they were involved with collaborative tasks. It would be interesting to investigate whether their answers relate to how well they performed in the game. Furthermore, It would be interesting to see if the participants who claimed to have good collaborative skill completed the puzzles faster and whether those who stated to collaborate daily communicated more throughout the test session, than

the other participants.

The VR groups communicated on average for 43.37% of the time, while 57.6% of those participants claimed to participate in collaborative tasks daily and 36.4% did it weekly. By looking at the non-VR groups, who on average communicated for 48.90% of the time, then 66.7% of them were part of collaborative tasks on a daily basis, while 18.2% did it weekly. This could be a reason to think that the non-VR groups communicated more during the test because of their 9.1% difference in performing collaborative tasks daily. Though it is difficult to conclude, the difference could point towards a possible correlation between daily collaborative interactions and the amount of percentage spoken time in our game. Though it is, with these numbers, impossible to conclude, and even then it would be hard to conclude a causality. A future test could be beneficial for testing this claim, since this could have had an effect on the results gathered for this report.

Some of the participants were a bit older than others, and it is interesting to see whether the age of the participants could have had an effect on the results. Two groups will be taken from the VR participants; the one with highest age in average (see VR_2.mp4 on the disk), and the one with the lowest (see VR_10.mp4 on the disk). The group with the highest age has participants that are 32.67 years old in average, while the lowest are exactly 20 years old in average. The older group had a lower overall playtime with 15.5 minutes, and they talked 8.36 minutes of that time, meaning that they communicated 53.47% of the time. The younger group had a playtime of 17.43 minutes and spent 6.58 minutes communicating, which means they communicated 37.76% of the time. The participants in the older group spent 15.71% more time communicating with each other, and this might also be the reason for why their playtime is shorter, because they might have been more effective in solving the puzzles due to their effort to communicate. Though this is just an example taken from two of the groups, which is not enough to conclude whether age has an effect on the communicative effort. If age was to be investigated further, then there would be a need to compare many more groups, or preferably conduct a test to check whether an effect is existent or not.

## 11.3 Summary

Throughout this chapter the results from the evaluation chapter has been discussed. It was found that the t-test did not find any significant difference, and thus we could not reject the null hypothesis. In light of this, several points were discussed that could have had an effect on the results. These points were divided into factors that were judged to possibly have had a direct effect and points that could have had an indirect effect. For this it was

found that the lack of previous experience with VR, as well as nausea could possibly have had an effect, though with the given sample size it would be impossible to judge. Among indirect effects it was discussed that participants with daily collaborative interactions may have had a correlation to their communicative effort percentages, though with the sample size this would again be impossible to say, and future tests would thus be beneficial. Last, age was also discussed as a possible factor, since some of the groups with older participants seem to have used more time communicating.

# 12 Conclusion

The purpose of the second part of the report was to investigate whether VR has an affect on group communicative effort in a collaborative environment or not. Therefore, the problem statement was defined:

*"Does virtual reality have an effect on group communication effort in a collaborative environment?"*

The problem statement was formulated using a working definition called 'Communicative effort'. Using this definition it was possible to define a specific form of communication concerning collaborative interaction. The problem was tested by utilising the puzzle based game that was created for a similar purpose. The test had two categories; one consisting of groups playing the game in VR and one consisting of groups playing without VR. During the test the working definition was used to outline the parameters of what was measured. The gathered data resulted in an insignificant difference, thus it was impossible to reject the null hypothesis. In light of this result, several possible explanations were discussed. While nothing conclusive was found, multiple possible explanations were found, where one of the most prominent was concerning the measuring method. If any effect was present, there would probably be a need for a larger sample size, or a more precise measurement method. But since neither of those were present in the test, it was impossible to detect any significant difference.

# 13 Future Works

In this chapter, possible future adjustments and directions will be proposed and argued for. Most of these are also discussed in the discussion chapter, though in less detail.

## 13.1 Test Adjustments

After conducting the tests, several shortcomings were found, some of which could be explained in part due to the lack of a decent sample size. The effect size found from the test suggested a medium effect, however the sample size was too small to detect an effect of this size. In future tests a larger sample size would be necessary, though this would often be the case. The time spoken was used as a measurement for the test. For future tests it may be better to utilize different methods, such as counting or defining categories of words and/or sentences and utterances. Though at this time it is hard to say exactly whether these would be significantly more accurate or not. One of the major problems with the test was that it was assumed that communication would be evenly distributed throughout the different sections of the game. Although, if a test group were to spend more time in levels that required higher amounts of communication than the other test groups, it could potentially skew the data in one or another direction. For future tests a solution would be to measure the levels individually.

## 13.2 Other Directions

Throughout the discussion chapter it was speculated that age, VR experience and nausea potentially could have had effects on the data. An interesting approach could be to test whether these have an actual effect on communication in collaborative environments when utilizing VR. If any differences were to be found, it could help reach a conclusion on the problem statement. Making adjustments to the test around these could also be a solution. For example with nausea it could help to give participants breaks between levels, as longer exposure to VR could have a negative effect, since it would mean further exposure to elements that could induce nausea in VR (see analysis section 4.2). Adjusting the game with this in mind could be helpful to prevent the aforementioned complications.

# 14 References

Apple. (2015). *Color spaces.* Retrieved from `https://developer.apple.com/library/mac/documentation/GraphicsImaging/Conceptual/csintro/csintro_colorspace/csintro_colorspace.html`

Awad, S. S., Fagan, S. P., Bellows, C., Albo, D., Green-Rashad, B., De La Garza, M., & Berger, D. H. (2005). Bridging the communication gap in the operating room with medical team training. *The American Journal of Surgery*, *190*(5), 770–774.

Berger, J. (2013). Beyond viral: Interpersonal communication in the internet age. *Psychological Inquiry*, *24*(4), 293–296.

Brewster, S. (2007). Nonspeech auditory output. In *The human computer interaction handbook: Fundamentals, evolving technologies and emerging applications* (pp. 248–263). CRC press.

Capin, T. K., Noser, H., Thalmann, D., Pandzic, I. S., & Thalmann, N. M. (1997). Virtual human representation and communication in vlnet. *IEEE Computer Graphics and Applications*(2), 42–53.

Dillenbourg, P. (1999). What do you mean by collaborative learning? *Collaborative-learning: Cognitive and Computational Approaches.*, 1–19.

Ducheneaut, N., Yee, N., Nickell, E., & Moore, R. J. (2006). Alone together?: exploring the social dynamics of massively multiplayer online games. In *Proceedings of the sigchi conference on human factors in computing systems* (pp. 407–416).

Dyer, W. G. (2007). *Team building.* Wiley Online Library.

Edmondson, A. (1999). Psychological safety and learning behavior in work teams. *Administrative science quarterly*, *44*(2), 350–383.

Edmondson, A. C., & Nembhard, I. M. (2009). Product development and learning in project teams: the challenges are the benefits*. *Journal of Product Innovation Management*, *26*(2), 123–138.

Field, A., & Hole, G. (2003). *How to design and report experiments.* Sage.

Fussell, S. R., Kraut, R. E., Lerch, F. J., Scherlis, W. L., McNally, M. M., & Cadiz, J. J. (1998). Coordination, overload and team performance: effects of team communication strategies. In *Proceedings of the 1998 acm conference on computer supported cooperative work* (pp. 275–284).

Gee, J. P. (2005). Good video games and good learning. In *Phi kappa phi forum* (Vol. 85, p. 33).

Hale, K. S., & Stanney, K. M. (2014). *Handbook of virtual environments: Design, implementation, and applications.* CRC Press.

Hämäläinen, R., Manninen, T., Järvelä, S., & Häkkinen, P. (2006). Learning to collaborate: Designing collaboration in a 3-d game environment. *The Internet and Higher Education*, *9*(1), 47–61.

Hartley, P. (2002). *Interpersonal communication.* Routledge.

Higdon, J. (2007). *Effects of goal orientation directed feedback on learning and motivational outcomes as moderated by trait goal orientation.* ProQuest.

Juul, J. (2009). Fear of failing? the many meanings of difficulty in video games. *The video game theory reader*, *2*, 237–252.

Kerbeci, A. (2015, April). *Steamcrew vr: Collaborate to steer and maintain your steam submarine! sink horrid monsters in an abyssal quest for magic gems!* Retrieved from http://vrjam.devpost.com/submissions/36153-steamcrew-vr

Lazzaro, N. (2004). *Why we play games: Four keys to more emotion without story* (Tech. Rep.). XEODesign.

Lipscomb, S. D., & Zehnder, S. M. (2004). Immersion in the virtual environment: The effect of a musical score on the video gaming experience. *Journal of Physiological Anthropology and Applied Human Science*, *23*(6), 337–343.

Litovsky, R. (2008). *Cochlear. binaural hearing* (Tech. Rep.). University of Wisconsin-Madison.

Mehetrey, P. (2015, April). *Let me teach you agile.* Retrieved from http://poonammehetrey.blogspot.dk/

Miles, J., & Hollenbeck, J. R. (2013). Teams and technology. *The Psychology of Workplace Technology*, 99.

Nintendo. (2015). *Zombieu.* Retrieved from http://www.nintendo.com/games/detail/vjLSrMs9WUOA215UXth_bVcjOkIPQico

Nordahl, R., & Nilsson, N. (2014). The sound of being there: Presence and interactive audio in immersive virtual reality. *The Oxford handbook of interactive audio. Oxford University Press, New York*, 213–233.

Oculus. (2015). *Oculus rift development kit 2 (dk2) - oculus - oculus vr.* Retrieved from https://www.oculus.com/en-us/dk2/

Pixologic. (2015). *Pixologic :: Pixologic :: Industry :: Scientific visualization.* Retrieved from http://pixologic.com/zbrush/industry/video-games/

robertcupisz. (2015, Juli). *Light shaft.* Retrieved from https://github.com/robertcupisz/LightShafts

Salas, E., Sims, D. E., & Burke, C. S. (2005). Is there a "big five" in teamwork? *Small group research*, *36*(5), 555–599.

Seibert, J. (2014). *An exploratory study on virtual reality head mounted displays and their*

*impact on player presence.* (Unpublished doctoral dissertation).

Tinwell, A., Grimshaw, M., Nabi, D. A., & Williams, A. (2011). Facial expression of emotion and perception of the uncanny valley in virtual characters. *Computers in Human Behavior*, *27*(2), 741–749.

Warkentin, M., & Beranek, P. M. (1999). Training to improve virtual team communication. *Information Systems Journal*, *9*(4), 271–289.

Weibel, D., & Wissmath, B. (2011). Immersion in computer games: The role of spatial presence and flow. *International Journal of Computer Games Technology*, *2011*, 6.

Zhang, J., & Fu, X. (2015). The influence of background music of video games on immersion. *Journal of Psychology & Psychotherapy*, *2015*.

Zhang, J., & Gao, X. (2014). Background music matters: Why video games lead to increased aggressive behavior? *Entertainment Computing*, *5*(2), 91–100.

# 15 Appendix

## Appendix A

### GameManager Script

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

public class GameManager : MonoBehaviour {

        public static GameManager _instance;

        protected int numberOfTriggeredEvents = 0; //Is used to determine where we
            should continue from each time a puzzle sequence is finished. is only
            updated after each sequence is finished
        protected int currentNumberOfEventsTriggered = 0; //counts up when a single
            event is triggered, is reset when all events in the sequence is triggered
        protected int index = 0; // used to count which sequence is currently in play

        public LevelManager LM; //Used in order to access the LevelManagerObject in
            the scene
    [HideInInspector]
    public LevelHandler levelHandler;

    private bool readyForNextSequence = false;

    public ServerManager server;

        // Use this for initialization
        void Start () {
                _instance = this;
        server = GetComponent<ServerManager>();
                GameObject g = GameObject.Find("LevelManagerObject"); //accessing the
                    LevelManager script on the LevelManagerObject
        levelHandler = GetComponent<LevelHandler>();


                if(g != null)
                        LM = g.GetComponent<LevelManager>();
        }
```

```
        // Update is called once per frame

    public void DetectTriggerChanges(){
        try{
            //Checks if the current object is triggered, and if they are ready to be
                triggered
                                                if(LM == null || LM.events.Length ==
                                                    0){
                                                        return;
                                                }

            if(LM.eventOrder[index] == 0){ //Checks if there is a desired order in the
                sequence, runs if there isn't
                for(int j = 0; j < LM.eventsInSequence[index]; j++){ //Goes through
                    the events in the sequence
                    //Used in order to not get double values
                    if(LM.events[j + numberOfTriggeredEvents].isTriggered == true &&
                        LM.events[j + numberOfTriggeredEvents].isReadyToBeTriggered ==
                        true){ //Checks if they are triggered
                        LM.events[j + numberOfTriggeredEvents].isReadyToBeTriggered =
                            false; //sets the event untriggerable
                        LM.triggeredEvents[j + numberOfTriggeredEvents] = true;

                        currentNumberOfEventsTriggered++; //counts up the events in
                            sequence by 1
                    }
                    resetTriggers(j);
                }
            }
            else { //Checks if there is a desired order in the sequence. Only checks
                the objects that should be interacted with in the sequence
                for(int i = 0; i < LM.eventOrder[index]; i++){
                    if(LM.events[i + numberOfTriggeredEvents +
                        currentNumberOfEventsTriggered].isTriggered == true &&
                        LM.events[i + numberOfTriggeredEvents +
                        currentNumberOfEventsTriggered].isReadyToBeTriggered == true){
                        //Checks if they are triggered
                    LM.events[i + numberOfTriggeredEvents +
                        currentNumberOfEventsTriggered].isReadyToBeTriggered = false;
                        //sets the event untriggerable
                    LM.triggeredEvents[i + numberOfTriggeredEvents +
                        currentNumberOfEventsTriggered] = true;
                    server.TriggerChanged(LM.events[i + numberOfTriggeredEvents +
                        currentNumberOfEventsTriggered]);
```

```
            currentNumberOfEventsTriggered++; //counts up the events in
                sequence by 1
            }
        }
        //This if statement is used in order to reset interactables if they
            require it
        for(int i = 0; i < LM.eventsInSequence[index]; i++){
            if(LM.events[i + numberOfTriggeredEvents].canReset == true &&
                LM.events[i + numberOfTriggeredEvents].isReadyToBeTriggered ==
                false){
                StartCoroutine(TimedReset(i));
            }
        }
        //This loop goes through the objects, which is not part of the current
            order, but is still in the sequence
        for(int i = LM.eventOrder[index] + currentNumberOfEventsTriggered; i <
            LM.eventsInSequence[index]; i++){
            //It then checks if they are triggered
            if(LM.events[i + numberOfTriggeredEvents].isTriggered == true &&
                LM.events[i + numberOfTriggeredEvents].isReadyToBeTriggered ==
                true){
                LM.events[i + numberOfTriggeredEvents].isReadyToBeTriggered =
                    false;
                //If they are triggered which they shouldn't be, then we reset
                    the sequence
                currentNumberOfEventsTriggered = 0;


                for(int j = 0; j < LM.eventsInSequence[index]; j++){ //goes
                    through all the objects in the sequence and untrigger them
                    StartCoroutine(FailedReset(j));
                    LM.events[j + numberOfTriggeredEvents].isTriggered = false;
                    LM.triggeredEvents[j + numberOfTriggeredEvents] = false;

                }
            }
        }

        //This if statement goes through all objects earlier then the current
            in the event order and detects if they trigger them, which they
            shouldn't
        if(currentNumberOfEventsTriggered > 0){
            for(int i = 0; i < currentNumberOfEventsTriggered; i++){
                //Checking events earlier in the sequence
```

```
                if(LM.events[i + numberOfTriggeredEvents].isTriggered == true
                    && LM.events[i +
                    numberOfTriggeredEvents].isReadyToBeTriggered == true){
                    LM.events[i +
                        numberOfTriggeredEvents].isReadyToBeTriggered = false;
                    currentNumberOfEventsTriggered = 0;


                    for(int j = 0; j < LM.eventsInSequence[index]; j++){
                        //goes through all the objects in the sequence and
                        untrigger them
                        StartCoroutine(FailedReset(j));
                        LM.events[j + numberOfTriggeredEvents].isTriggered =
                            false;
                        LM.triggeredEvents[j + numberOfTriggeredEvents] =
                            false;

                    }
                }
            }
        }
    }

    for(int i = 0; i < LM.eventsInSequence[index]; i++){
        if(LM.triggeredEvents[i + numberOfTriggeredEvents] == false){
            readyForNextSequence = false;
            break; //breaks if one of the objects is not triggered
        } else {
            readyForNextSequence = true;
        }
    }

    if(readyForNextSequence == true){
        if(LM.triggerEvents[index] != null){
            LM.triggerEvents[index].isTriggered = true; //Triggers an object
                with should trigger when a sequence is finished. could for
                example be a door
            server.TriggerChanged(LM.triggerEvents[index]);
        }
        if(index < LM.eventsInSequence.Length - 1){ //Checks if it is the last
            sequence of events - if it is: skip this
            numberOfTriggeredEvents += LM.eventsInSequence[index]; //Increase
                the total number of events by the amount of events that was in
                the current sequence
```

```
                index++;
                for(int i = numberOfTriggeredEvents; i < numberOfTriggeredEvents +
                    LM.eventsInSequence[index]; ++i){ //Goes through the next
                    sequence of events
                    LM.events[i].isReadyToBeTriggered = true; //Makes the next
                        sequence ready to be triggered
                    server.TriggerChanged(LM.events[i]);
                }
            }
            currentNumberOfEventsTriggered = 0; //Resets the amount of objects
                that was triggered in the current sequence
            readyForNextSequence = false;
        }
    }
    catch(System.IndexOutOfRangeException e){
        Debug.LogWarning("This was an error -- needs to implement Andreas
            GameManager Fix");
                                            Debug.LogWarning(e);
    }

}

    public void setNewLevelManager(LevelManager levelManager){
            LM = levelManager;

 if(LM == null)return;

 index = 0;
 numberOfTriggeredEvents = 0;
 currentNumberOfEventsTriggered = 0;
            if(LM.eventsInSequence.Length != 0){
                    for(int k = 0; k < LM.eventsInSequence[0]; k++){ //makes the
                        first events in the scene triggerable
                            LM.events[k].isReadyToBeTriggered = true;
                    }
            }
    }

 private void resetTriggers(int triggerIndex){
    //This if statement is used in order to reset interactables if they require it
    try{
        if(LM.events[triggerIndex + numberOfTriggeredEvents].canReset == true &&
            LM.events[triggerIndex + numberOfTriggeredEvents].isReadyToBeTriggered
            == false){
```

```
                LM.events[triggerIndex + numberOfTriggeredEvents].isReadyToBeTriggered
                    = true;
                LM.events[triggerIndex  + numberOfTriggeredEvents].canReset = false;
            }
        }catch(System.IndexOutOfRangeException e){
            Debug.LogWarning("This was an error -- needs to implement Andreas
                GameManager Fix");
                    Debug.LogWarning(e);
        }
    }

    IEnumerator TimedReset(int resetIndex){ //Coroutine used for resetting objects
        which needs to be resetted
        yield return new WaitForSeconds(1.5f); //Resets after x seconds
        LM.events[resetIndex + numberOfTriggeredEvents].isTriggered = false;
        LM.events[resetIndex + numberOfTriggeredEvents].isReadyToBeTriggered = true;
        server.TriggerChanged(LM.events[resetIndex + numberOfTriggeredEvents]);
    }

    IEnumerator FailedReset(int resetIndex){ //MOVE SOME OF THIS STUFF UP!
        yield return new WaitForSeconds(1.5f); //Resets after x seconds
        LM.events[resetIndex + numberOfTriggeredEvents].isReadyToBeTriggered = true;
        server.TriggerChanged(LM.events[resetIndex + numberOfTriggeredEvents]);

    }
}
```

### LevelManager Script

```
using UnityEngine;
using System.Collections;

public class LevelManager : MonoBehaviour {

        [ContextMenuItem("Force Update GameManager", "ForceUpdateGM")] //add extra
            functionality on the right click context menu
        [Tooltip("Interactable gameobjects in the game, should be dragged on in the
            right order for the eventOrder to work. Because it will be used in
            eventOrder!!")]
        public Trigger[] events;                                    // Events which are
            dragged into the levelmanager       - Should be specified in the unity
            editor!!
```

```
        [ContextMenuItem("Force Update GameManager", "ForceUpdateGM")] //add extra
            functionality on the right click context menu
        [Tooltip("Bool used to detect if se is finished. Size must be the same size as
            'events'")]
        public bool[] triggeredEvents;

        [ContextMenuItem("Force Update GameManager", "ForceUpdateGM")] //add extra
            functionality on the right click context menu
        [Tooltip("This contains how many of the gameobjects there is in a specific
            sequence. You must specifiy the corresponding event size to the number of
            gameobjects. e.g if you have 6 elements, you must make sure that the total
            number is also 6 in this array. could be written as the element0 is 2 and
            element1 is 4. Element0 is the first sequence. Array size is the amount of
            sequences")]
        public int[] eventsInSequence;

        [ContextMenuItem("Force Update GameManager", "ForceUpdateGM")] //add extra
            functionality on the right click context menu
        [Tooltip("This contains the event order in which you want gameobjects
            triggered in a sequence. The number specifies the amount of gameobjects
            that should be triggered, 0 = no order, 1 = first gameobject in the
            sequence needs to be triggered before the rest, then second, and then
            third, 2 = first and second can be triggered, then third and fourth. Array
            size must be the same as amount of sequences")]
        public int[] eventOrder;
        [Header("Defining Level start and end")]

        [Tooltip("The 3 rail points where the level starts, Organized Blue, Red,
            Green")]
        public Rail[] levelStartRail = new Rail[3];

        [Tooltip("The 3 rail points where the level end, Organized Blue, Red, Green")]
        public Rail[] levelEndRail   = new Rail[3];

        private void ForceUpdateGM(){
                GameManager._instance.DetectTriggerChanges();
        }
}
```

# Appendix B



## Questionnaire

You will now be given a series of questions based on the game you just played.

* Required

### Objects

1. **1st puzzle/overall game** *
   *Mark only one oval per row.*

| | Strong disagree | Disagree | Neutral | Agree | Strongly agree |
|---|---|---|---|---|---|
| I felt like it was obvious that I could interact with the red button | | | | | |
| I felt like it was obvious that the feedback from the button indicated that it was activated | | | | | |
| I felt like the controls of the game were difficult to understand | | | | | |
| I felt like the animation of the door was too slow | | | | | |
| I felt like the door model was visually pleasing (ignore missing textures.) | | | | | |

2. **2nd puzzle/Elevator** *
   *Mark only one oval per row.*

| | Strong disagree | Disagree | Neutral | Agree | Strongly agree |
|---|---|---|---|---|---|
| The elevator ride made me uncomfortable | | | | | |

3. **3rd puzzle/Mirror** *
   *Mark only one oval per row.*

| | Strong disagree | Disagree | Neutral | Agree | Strongly agree |
|---|---|---|---|---|---|
| I felt like it was obvious how the mirror reflected my light source | | | | | |
| I felt like it was easy to understand how the rotation of the mirror worked | | | | | |
| It felt like it was clear that i had to reflect my light source in the mirror to activate the button on the left side | | | | | |

## Colors

4. **4th puzzle/Sound** *

*Mark only one oval per row.*

| | Strong disagree | Disagree | Neutral | Agree | Strongly agree |
|---|---|---|---|---|---|
| I felt like it was obvious that the crystal hanging from the ceiling could emit a melody | | | | | |
| It was clear to me that focusing my light source on the crystal hanging from the ceiling would play the melody again | | | | | |
| I understood that each button had a tone which I had to activate in the correct order | | | | | |

## The image below shows colors used for the following questions


## Choose the right color as a result from each color combination

5. **Red mixed with blue** *

*Mark only one oval.*

- Magenta
- Cyan
- Yellow
- Green

6. **Green mixed with red** *
*Mark only one oval.*
- Magenta
- Blue
- Cyan
- Yellow

7. **Blue mixed with green** *
*Mark only one oval.*
- Yellow
- Red
- Cyan
- Magenta

8. **Yellow mixed with magenta** *
*Mark only one oval.*
- Blue
- Green
- Red
- Cyan

9. **Yellow mixed with blue** *
*Mark only one oval.*
- Red
- Cyan
- Green
- Magenta

10. **Cyan mixed with magenta** *
*Mark only one oval.*
- Green
- Blue
- Red
- Yellow

# Appendix C

### Notes from Observer 1

Participant 1 Lv1: Easy. Lv2: Focuses on the button ASAP = has to wait for elevator to come down again. Lv3: Figured out how to use the mirror right away. Lv4: Did NOT notice the sound crystal until he asked for help. After that, the first sequence is easy for him. "WOW WTF?!=!..... Too hard" Didn't even try to solve the 2nd sequence because it was so hard.

Paricipant 2 Lv1: Easy Lv2: Focuses on the button ASAP = has to wait for elevator to come down again. Lv3: Does not notice that the mirror is rotating. Keeps lighting on the two buttons. Figured out the mirror eventually. Lv4: Did NOT notice the sound crystal until he asked for help. He understood what to do after focusing on the crystal. "WTF?!" was the reaction when he heard the 2nd sequence. He thought he had solved it, but he couldn't.

Participant 3 Lv1: Did not understand that he had to use his head to control. He was trying to look around with the controller. It was easy after that. Lv2: Again, a lot of waiting because they send the elevator up. He understood it easily. Lv3: Saw the mirror because he accidentally focused on it, after that it was easy. Lv4: Kept focusing on all 6 different buttons. Did NOT notice the sound crystal until he asked for help. We had to tell him that he only was suppose to use the first three buttons for the 1st sequence. Then it was easy. He kept trying to solve the 2nd sequence, but could not solve it. He said that there was a big delay

Participant 4 Lv1: Didn't know how to focus. Lv2: He focused on the button asap, but he thought that he had to constantly focus on the button for the elevator to move. We had to tell him that he could just do it once to trigger the elevator. Lv3: Noticed that the mirror rotated when he focused on the light. Was easy for him to understand it all. Lv4: I think he saw the crystal, but he had no idea that he had to focus on it. He just kept focusing on all the buttons. We had to tell him about the crystal, but he could not really solve the 1st sequence. He said that the sounds were too much alike...

Participant 5 Lv1: I was talking to another test participant. No notes from me :( Lv2: He did focus on the button so the elevator when up, but then he was kind of lost because he was positioned right in front of the elevator while it was elevated. He was then instructed to back up a bit, then he figured it out. Lv3: Does not really notice the mirror. He kept rotating it, but he did not try to focus on it. He also tried to focus on the button behind the crystal many times. Lv4: He keeps focusing on the same button. Then tries other buttons.

Did NOT notice the crystal, we had to tell him about it. He was trying all the 6 buttons, and could not really figure out the 1st sequence. We told him to try to focus on the first 3 buttons, but I do not think he know what is going on, has no idea that he has to replicate the melody. Did eventually solve it.

**Notes from Observer 2**

They quickly understand that lighting on button activates things.

All the test participants except the last (6th) tried to activate the button before being on the elevator

Several player did not find it clear that you had to light on the big crystal to replay the sound. Solve: Add constant light on it.

3. Found it difficult to understand that you could control light with the head, Keeps lighting on the button even when it is activated.

The light reflected from the mirror seemed to guide the player to rotating the mirror in the correct position. Tries to light the button behind the crystal(not important factor) The sound crystal is too slow to play the melody

4. keeps holding the focus button down. Tried to light button behind crystal Mirror was not noticed. Could not find the sound crystal Several tried to light on the buttons on the second row. Wanted to have a higher pitch difference.

5. Only lights short to activate the button. It was not an issue to know to light the button while being on the elevator. Tries to light the button behind the crystal. Did not notice mirror Lights on buttons on the back row. Tried several times only lighting on the button infront of him. It seemed like he did not understand the puzzle.

6. Several of the players tried to move forward while the door was opening. Did go on the elevator before activating the button. Tried to light the button behind the crystal Had difficulty hitting the right position on the mirror, when it keeps turning. Lights on the second row buttons

# Appendix D



## Questionnaire

You will now be given a series of questions based on the game you just played.

* Required

### Game

1. **Overall game** *
   *Mark only one oval per row.*

| | Strong disagree | Disagree | Neutral | Agree | Strongly agree |
|---|---|---|---|---|---|
| I felt like the player movement was comfortable | | | | | |
| I felt like the speed of the player movement was too fast | | | | | |
| I felt like the headlight could reach the places I wanted | | | | | |

2. **2nd puzzle/Mirror** *
   *Mark only one oval per row.*

| | Strong disagree | Disagree | Neutral | Agree | Strongly agree |
|---|---|---|---|---|---|
| I felt like the mirror was clearly visible | | | | | |
| I felt like I was in control of the mirrors rotation | | | | | |

3. **3rd puzzle/Sound** *
   *Mark only one oval per row.*

| | Strong disagree | Disagree | Neutral | Agree | Strongly agree |
|---|---|---|---|---|---|
| I felt like it was obvious when the puzzle restarted through the light feedback | | | | | |
| It was clear when the buttons were active and not active | | | | | |

### Sound

4. **Were the volume of the sounds too loud? If yes which one/ones?**
*Check all that apply.*

- [ ] Player movemet(cart moving forward and backwards)
- [ ] Door opens
- [ ] Focuslight
- [ ] Activating button
- [ ] Mirror turning
- [ ] Sound from button (sound puzzle)
- [ ] Melody from crystal (sound puzzle)
- [ ] Background sounds(ambience)
- [ ] Error sound (sound puzzle)

5. **Were the volume of the sounds too low? If yes which one/ones?**
*Check all that apply.*

- [ ] Player movemet(cart moving forward and backwards)
- [ ] Door opens
- [ ] Focuslight
- [ ] Activating button
- [ ] Mirror turning
- [ ] Sound from button (sound puzzle)
- [ ] Melody from crystal (sound puzzle)
- [ ] Background sounds(ambience)
- [ ] Error sound (sound puzzle)

6. **Any further comments about the sound**

.....

Powered by
Google Forms

# Appendix E



## Post-test questionnaire (VR)

*Required

**ID**
Group Number && Participant Number (Ex. 1_2)

**Gender** *
- Male
- Female

**Age** *

**Have you played computer games before?** *
- Yes
- No

**How often do you participate in a collaborative task?** *
- Daily
- Weekly
- Monthly
- Rarely

**How would you rate your own collaborative skills?** *
- Poor
- Below average
- Average
- Above average
- Good

**Do you have any previous experience with Virtual Reality?** *
- Yes
- No

**Did you feel any nausea during the test?** *
If yes, please elaborate where and when

# Appendix F



## Post-test questionnaire (Non-VR)

*Required

**ID**
Group Number && Participant Number (Ex. 1_2)

**Gender** *
- Male
- Female

**Age** *

**Have you played computer games before?** *
- Yes
- No

**How often do you participate in a collaborative task?** *
- Daily
- Weekly
- Monthly
- Rarely

**How would you rate your own collaborative skills?** *
- Poor
- Below average
- Average
- Above average
- Good

**Comments**

Submit

*Never submit passwords through Google Forms.*

# Appendix G

| VR | 1st Play time | 2nd PlayTime | 3rd Playtime | 1st Talk time | 2nd Talk Time | 3rd Talk Time | 1st % | 2nd % | 3rd % |
|---|---|---|---|---|---|---|---|---|---|
| Group 1 | 14 | 15,05 | 16,3 | 5,58 | 9,52 | 6,41 | 42,62% | 65,41% | 40,51% |
| Group 2 | 17 | 14 | | 9,53 | 6,5 | | 58,14% | 48,81% | #DIV/0! |
| Group 3 | 11,31 | 11,39 | | 6,03 | 6,11 | | 52,53% | 53,08% | #DIV/0! |
| Group 4 | 18,11 | 19 | | 6,05 | 5,5 | | 32,02% | 30,70% | #DIV/0! |
| Group 5 | 16,3 | 18,54 | 17,3 | 7,3 | 5,57 | 6,4 | 45,45% | 31,48% | 38,10% |
| Group 6 | 33,5 | 31,7 | | 14,19 | 15,25 | | 42,32% | 47,93% | #DIV/0! |
| Group 7 | 24,55 | 24,57 | 24,56 | 22,29 | 13,26 | 12,38 | 90,23% | 53,84% | 50,67% |
| Group 8 | 14 | 14,35 | 13,45 | 10,13 | 5,57 | 5,55 | 72,98% | 40,80% | 43,03% |
| Group 9 | 16,1 | 15,39 | | 7,34 | 6,02 | | 46,80% | 38,55% | #DIV/0! |
| Group 10 | 18,25 | 17,21 | 17,3 | 10,24 | 5,54 | 7,16 | 56,47% | 34,01% | 41,52% |
| | | | | | | | | | |
| | | | | | | | | | |
| Non-VR | Play time | 2nd PlayTime | 3rd Playtime | Talk time | 2nd Talk Time | 3rd Talk Time | % | 2nd % | 3rd % |
| Group 1 | 9,02 | 11,02 | 10,3 | 3,22 | 5,31 | 6,13 | 37,27% | 50,00% | 59,21% |
| Group 2 | 18,3 | 13 | 17,33 | 5,43 | 6,56 | 6,25 | 30,90% | 53,33% | 36,56% |
| Group 3 | 9,26 | 11 | | 3,16 | 3,11 | | 34,63% | 28,94% | #DIV/0! |
| Group 4 | 18,1 | 17,2 | 17,03 | 10,14 | 7,08 | 10,01 | 56,33% | 41,15% | 58,75% |
| Group 5 | 9,6 | 9,35 | 9,23 | 4,05 | 6,1 | 4,25 | 40,83% | 64,35% | 47,07% |
| Group 6 | 23,2 | 24,1 | | 12,37 | 10,56 | | 54,07% | 45,24% | #DIV/0! |
| Group 7 | 13,3 | 13,55 | 13,2 | 8,47 | 8,23 | 5,33 | 65,06% | 60,24% | 41,63% |
| Group 8 | 16,01 | 16,56 | | 9,21 | 8,53 | | 58,38% | 52,46% | #DIV/0! |
| Group 9 | 14,12 | 14,06 | | 7,05 | 6,41 | | 49,88% | 47,40% | #DIV/0! |
| Group 10 | 14,5 | 14,05 | | 7,48 | 6,58 | | 52,58% | 49,47% | #DIV/0! |

# Appendix H

| Timestamp | Gender | Age | Have you played computer games before | How often do you participate in a collaborative task | How would you rate your own collaborative skills | Do you have any previous experience with Virtual Reality | Did you feel any nausea during the test? | ID |
|---|---|---|---|---|---|---|---|---|
| 11/23/2015 11:48:0 | Male | 22 | Yes | Daily | Above average | Yes | no | 1_2 |
| 11/23/2015 11:49:0 | Male | 26 | Yes | Daily | Above average | No | Only slightly, but I think its something you have to get use to. | 1_1 |
| 11/23/2015 11:49:0 | Male | 26 | Yes | Daily | Above average | No | Yes. Every time the vehicle turned it felt weird in the stomach. | 1_3 |
| 11/23/2015 12:22:2 | Male | 47 | Yes | Daily | Above average | Yes | No i did not! | 2_2 |
| 11/23/2015 12:22:5 | Female | 25 | Yes | Daily | Average | Yes | NOOO NO NOOOO | 2_1 |
| 11/23/2015 12:22:5 | Male | 22 | Yes | Daily | Good | Yes | nope | 2_3 |
| 11/23/2015 13:06:3 | Male | 26 | Yes | Daily | Above average | Yes | A little bit when the lanes curved, and when the light of other players burned in front of me. | 3_1 |
| 11/23/2015 13:06:4 | Male | 27 | Yes | Weekly | Average | Yes | Yeah there where a bit in the beginning, but i think it was a question of me getting used to OK2 compared to OK1 | 3_3 |
| 11/23/2015 13:07:3 | Male | 23 | Yes | Daily | Good | No | No, I was good. It was scary to "peek" out of the wagon though, because I was afraid of hitting something with my head. | 3_1 |
| 11/23/2015 13:38:0 | Female | 27 | No | Daily | Average | Yes | No | 4_3 |
| 11/23/2015 13:38:3 | Male | 25 | Yes | Weekly | Average | Yes | Only in the end, I think i just got confused about the last puzzle | 4_1 |
| 11/23/2015 13:38:4 | Male | 23 | Yes | Daily | Above average | Yes | Some, but I've tried worse<br>When the framerate drops at some points it starts lagging slightly<br>And when we take quick turns at corners | 4_2 |
| 11/23/2015 14:06:0 | Male | 24 | Yes | Daily | Above average | Yes | right side of the screen flickered resulting in a eye irritation. | 5_1 |
| 11/23/2015 14:06:0 | Male | 22 | Yes | Daily | Good | Yes | I did not. | 5_2 |
| 11/23/2015 14:06:1 | Male | 22 | Yes | Weekly | Above average | No | No | 5_3 |
| 11/23/2015 14:59:5 | Female | 22 | Yes | Weekly | Good | Yes | no | 6_3 |
| 11/23/2015 14:59:5 | Male | 23 | Yes | Weekly | Good | Yes | no | 6_2 |
| 11/23/2015 15:00:0 | Female | 41 | Yes | Daily | Good | Yes | no it was ok | 6_1 |
| 11/23/2015 16:14:3 | Male | 24 | Yes | Daily | Good | Yes | No | 7_3 |
| 11/23/2015 16:14:3 | Male | 44 | Yes | Weekly | Average | No | yes | 7_2 |
| 11/23/2015 16:15:3 | Male | 22 | Yes | Monthly | Average | No | Yes, after a while I began feeling slightly nauseas and it progressively got worse, but by removing (taking the headset off) the nausea never progressed to unbearable levels. | 7_1 |
| 12/9/2015 12:40:3 | Female | 23 | Yes | Weekly | Average | No | No | 1_3 |
| 12/9/2015 12:40:4 | Female | 23 | Yes | Weekly | Average | No | Når man bevægede sig og det drejede | 1_1 |
| 12/9/2015 12:41:0 | Female | 27 | No | Weekly | Average | No | yes, in the middel of the game game i started feeling a bit nausiated | 1_2 |
| 12/9/2015 13:01:2 | Female | 22 | Yes | Daily | Above average | No | NO | 2_3 |
| 12/9/2015 13:01:3 | Male | 20 | Yes | Daily | Above average | No | yeah, from he start | 2_1 |
| 12/9/2015 13:01:4 | Male | 20 | Yes | Daily | Good | No | No i did not | 2_2 |
| 12/9/2015 13:29:1 | Male | 21 | Yes | Weekly | Above average | No | No | 3_2 |
| 12/9/2015 13:30:0 | Female | 21 | No | Daily | Average | No | YES, WHEN IT'S GOING OP AND DOWN | 3_3 |
| 12/9/2015 13:30:2 | Male | 25 | Yes | Weekly | Average | No | No | 3_1 |
| 12/9/2015 13:52:5 | Female | 19 | Yes | Weekly | Average | No | no | 4_1 |
| 12/9/2015 13:53:0 | Male | 22 | Yes | Daily | Average | No | I felt no nausea | 4_3 |
| 12/9/2015 13:53:2 | Female | 19 | Yes | Monthly | Average | No | Yes, when the wagons went down. I am probably affected by getting sick enough | 4_2 |

# Appendix I

| Timestamp | Gender | Age | Have you played computer games before | How often do you participate in a collaborative task | How would you rate your own collaborative skills | ID | Comments |
|---|---|---|---|---|---|---|---|
| 23/11/2015 11.50.20 | Male | 23 | Yes | Daily | Average | 1_1 | |
| 23/11/2015 11.50.21 | Female | 24 | Yes | Weekly | Above average | 1_2 | |
| 23/11/2015 11.50.21 | Male | 35 | Yes | Daily | Good | 1_3 | |
| 23/11/2015 12.28.53 | Female | 22 | Yes | Daily | Above average | 2_2 | |
| 23/11/2015 12.28.55 | Male | 27 | Yes | Daily | Good | 2_1 | |
| 23/11/2015 12.29.05 | Male | 23 | Yes | Weekly | Above average | 2_3 | |
| 23/11/2015 12.57.54 | Male | 26 | Yes | Weekly | Average | 3_3 | |
| 23/11/2015 12.57.57 | Male | 25 | Yes | Daily | Above average | 3_2 | |
| 23/11/2015 12.57.57 | Male | 25 | Yes | Daily | Average | 3_1 | |
| 23/11/2015 13.36.52 | Male | 22 | Yes | Rarely | Above average | 4_1 | |
| 23/11/2015 13.36.57 | Female | 28 | Yes | Monthly | Below average | 4_3 | |
| 23/11/2015 13.36.57 | Male | 26 | Yes | Daily | Below average | 4_2 | |
| 23/11/2015 14.00.02 | Male | 23 | Yes | Daily | Good | 5_3 | |
| 23/11/2015 14.00.14 | Male | 21 | Yes | Daily | Good | 5_1 | |
| 23/11/2015 14.00.34 | Male | 25 | Yes | Daily | Good | 5_2 | |
| 23/11/2015 14.54.44 | Male | 22 | Yes | Daily | Good | 6_3 | |
| 23/11/2015 14.55.00 | Male | 23 | Yes | Monthly | Average | 6_1 | |
| 23/11/2015 14.55.14 | Male | 45 | No | Weekly | Above average | 6_2 | |
| 23/11/2015 16.00.41 | Male | 21 | Yes | Daily | Above average | 7_1 | |
| 23/11/2015 16.00.43 | Male | 26 | Yes | Daily | Average | 7_3 | |
| 23/11/2015 16.00.43 | Male | 23 | Yes | Monthly | Good | 7_2 | |
| 10/12/2015 12.08.25 | Male | 23 | Yes | Daily | Good | 1_3 | |
| 10/12/2015 12.08.30 | Male | 23 | Yes | Daily | Below average | 1_1 | |
| 10/12/2015 12.08.37 | Male | 23 | Yes | Rarely | Above average | 1_2 | |
| 10/12/2015 12.32.06 | Male | 23 | Yes | Daily | Average | 2_1 | |
| 10/12/2015 12.32.16 | Male | 20 | Yes | Weekly | Good | 2_2 | Fede puzzles :D |
| 10/12/2015 12.32.16 | Male | 23 | Yes | Daily | Above average | 2_3 | |
| 10/12/2015 13.23.32 | Male | 22 | Yes | Daily | Average | 3_2 | No comment |
| 10/12/2015 13.23.34 | Female | 21 | Yes | Daily | Above average | 3_3 | Super nice game ! nice details |
| 10/12/2015 13.23.35 | Female | 20 | Yes | Daily | Good | 3_1 | nice game |
| 10/12/2015 13.53.53 | Male | 21 | Yes | Daily | Above average | 4_3 | |
| 10/12/2015 13.54.31 | Female | 22 | Yes | Daily | Good | 4_1 | |
| 10/12/2015 13.54.49 | Male | 20 | Yes | Weekly | Average | 4_2 | game crashes, but fun |