Growth - A virtual reality game and controller system

Björgvin Brynjarsson
Björn Ingi Baldvinsson
Daníel Jóhannsson
Kristján Ingi Einarsson
Bachelor of Science
2016
School of Computer Science
Reykjavík University

T-404-LOKA
Supervisor:
Hlynur Sigurþórsson
Examiner:
Hannes Pétursson

B.Sc. Final Report

Contents

List of Figures

Chapter 1

Introduction

In August of 2012, a project dubbed "Rift" showed up on Kickstarter, a crowd-funding site. It was presented by a company called Oculus and promised to bring consumers the world of high-definition virtual reality technology at an affordable price. The Kickstarter campaign managed to raise a total of 2.437.429 USD in donations from enthusiasts that wanted to own such a device, and with that resounding success came a renewed interest in virtual reality headsets from consumers and companies alike.[1] In 2016, the first generation of a new wave of consumer virtual reality devices arrived on the market. Devices spearheaded by the giants of the industry dominating the scene, Oculus along with the Rift was obtained by Facebook, HTC co-produced the Vive alongside Valve, Samsung created the Gear VR for their latest generation of Galaxy smartphones and Sony developed Playstation VR for their already popular gaming console.[2]–[6]

In addition to the selection of dedicated devices ranging in prices from 100 USD to 800 USD, there were a number of smaller names selling headsets that fit a smartphone for less, similar to the way Samsung entered the market. Google also has a product called Cardboard, a minimalist software focused approach where the first headsets were literally made of cardboard and they housed the user's personal smartphone. Google made the software available to everyone for free and with that may have ensured that most smartphone owners will eventually have tried virtual reality. This also made sure that most people will experience virtual reality for the first time with a headset that houses a smartphone. For this reason the headset called Freefly VR was chosen by the team for this project. The Google Cardboard SDK is also the team's choice for its widespread adoption and the large network of developers using it.[7]

A fairly large flaw with the smartphone VR method was the lack of well-designed controls. Limiting the experience to just watching with minimal input options puts severe constraints on any aspiring developers for smartphone VR. Kári Halldórsson, an adjunct professor at Reykjavík University, proposed an experiment which he offered as a final project for undergraduate students. Some smartphone VR headsets include rudimentary controllers for some basic user interface purposes but why not make use of something most people already have available? So he made a request for a virtual reality game where smartphones would become meaningful controllers. The inspiration came from his own small game where, using the gyroscopic sensors of two smartphones, he managed to simulate winged flight where the player had full control over the wings.

Kári had two non-negotiable requirements for the project:

1. The application must support at least one smartphone that connects wirelessly to a virtual reality headset, the connection process should be simple and convenient for players.

2. The wireless smartphone controller(s) should be essential to the game's design and gameplay.

The team got to work on the controller system immediately, an early simplistic but functional version was ready only a week after the project started. By making good use of the popular game engine Unity, the team could write the code for the controller application in a way that blends it with the game seamlessly. The team opted to focus on programming for the Android platform for convenience. Every team member used an Android smartphone at the time.[8]

Next came the design phase for the game development. The team, in conjunction with Kári, brainstormed a game design using a modified version of a framework called Mechanics-Dynamics-Aesthetics. During that creative process the idea of a feature where the player could change in size was well received by everyone. That feature would become the central mechanic of the game.

The team devised a method to increase the size of the player that was rather flawed to begin with. The two in-game "cameras" would drift apart and cause the view of the player to become unfocused, like the player had suddenly become severely cross-eyed. This was later fixed when the team's understanding of the necessary positioning and angle adjustments improved.

After the effect became sufficiently convincing for team members the decision was made to set up a preliminary research project. The purpose of this research was to determine if the effect felt convincing to randomly selected volunteers. Preparing for and conducting the experiment helped improve the size effect considerably. The results were generally that participants did not experience the effect in the way the team had hoped. After analyzing the data, the team theorized that this was due to other factors than the size effect simply not working.

Managing the development of a game with Scrum was also an experiment for the team. Early on, during the first presentation of the project to the judges, one of the judges expressed an interest in knowing if Scrum could work well in the design and development of a stand-alone game project. The team was also eager to find out since, according to Mountain Goat Software, Agile methodologies tend to work well during long-term game development, where estimates can carry over from one project to the next. It is unclear however if Scrum works well in small game development projects. As far as the team is concerned, Scrum worked out very well in the end. For more details on the team's success with Scrum and everything else please refer to this road map:[9]

1.1 Road map

Section 2 goes into details about how the controller system works and the concept behind it.

Section 3 is focused on how the team came up with the game idea, what that idea was and how the game plays.

Section 4 explains the method used to achieve the growth/shrink effect.

Section 5 is about the research, how it was conducted, the results of the experiment, any conclusions reached and suggestions the team had for further research.

Section 6 describes how the project was managed, what methodology was used and how effective it was when managing the development of a game.

Section 7 discusses the project conclusion and what the team learned from it.

Chapter 2

Controllers

As was mentioned earlier, Kári Halldórsson had made a prototype of a VR flying experience using two smartphones as controllers where each phone's orientation mapped to the orientation of the player's "wings" in-game. This project's goal was to expand on this idea and make the smartphones fully functional controllers that would be meaningful inputs for a full-blown game. Originally the idea was to design the controller system with the possibility to use two mobile devices as controllers at the same time but also to make it possible to play with a single controller. After the idea for the game had been finalized and development started, it soon became apparent that implementing the option to play with a variable amount of controllers was problematic to the design. The challenge was not technical in nature, it was usability that suffered if the two controller design persisted. When the controller was used for more than just its orientation e.g. gestures and button presses, the interface became more complicated and less intuitive. With this in mind, the team changed the design so the implementation focused on a single controller.

There were other design challenges the team encountered. Since the player could not see the screen of the controller while wearing the VR headset, the ways in which the touch surface could be used was limited to some degree. A decision was made to design the controller UI in a way that would split the screen into two halves, each acting as a single button.

However, there are other options to interact with the controller than touch. The mobile devices that the controller application is designed for have built-in motion sensors to detect the orientation and movement of the device in three-dimensional space. The controller application utilizes both the mobile device's touch surface and motion sensors as input for the game which is sent over in the form of different messages. The controller's interaction with the game could be split into three categories:

Gyroscopic The controller's orientation is used to control the movement direction of the player.

Accelerometric The controller's movement is used to detect gestures.

Touch The controller's two buttons are contextually used to perform certain actions in the game.

The possibility of touch gestures was also considered but deemed unnecessary to achieve every critical design goal for the game. The team decided to forego touch gestures. The added complexity of the controls could have been detrimental to the user experience.

2.1 Messages

The controller application checks the status of the device's motion sensors at regular intervals, it also listens for any events regarding touches on the device's screen. After this data has been

interpreted the controller sends the relevant information to the game. The types of messages which are sent from the controller can be categorized into two types:

Events Something that "happened", e.g. a button was clicked.

Status The current reading of the motion sensors in the device.

There is a fundamental difference in how these types are handled. Events have no tolerance for loss of data. If the player taps the screen you want to be certain that the game application actually receives that information, so it can respond with the appropriate changes to the game state. Status messages are sent at regular intervals. They are more tolerant of loss since it doesn't affect the game as dramatically if a message with motion sensor data is lost every once in a while. Status messages are sent once per frame, assuming a frame-rate of 60 frames per second the difference between 59 status messages per second and 60 is negligible. Event messages, however, are specific actions requested by the player and must therefore always be resolved correctly.

2.2 Communications

As can be seen in Figure 2.1, the game is running on one device attached to the VR headset. Wirelessly connected to that device is another device running the controller application.

Figure 2.1: Single-player communications diagram.

The code for both the controller and the game is written in C#. The communications between the game and the controller are implemented with the Unity Transport Layer API which is a "low-level" alternative to Unity's other networking API called The High-Level API (HLAPI). The Transport Layer API is built on top of User Datagram Protocol (UDP), it introduces the concepts of connections and channels. A single connection, like the one between the controller and the game, can have several channels. Those channels can have different properties, mainly related to Quality of Service (QoS). What this means for the different types of messages the controller

needs to send is that we can have different channels, with different reliability guarantees, over the same connection for each type of message.[10], [11]

Figure 2.2 depicts how data flows between different components of the controller system after a connection has been established between the game and controller application. The `ControllerNetwork` and `GameNetwork` on each side take care of sending and receiving data with the Transport Layer API. On the game side, the `GamePad` component is the actual interface with which the various game scripts get the current status of the controller. On the controller side, the `Touch input & Motion tracking` component takes care of sending the controller messages through the network using `ControllerNetwork`. The `GameInfo` component reads the current status of game specific states for debugging purposes.

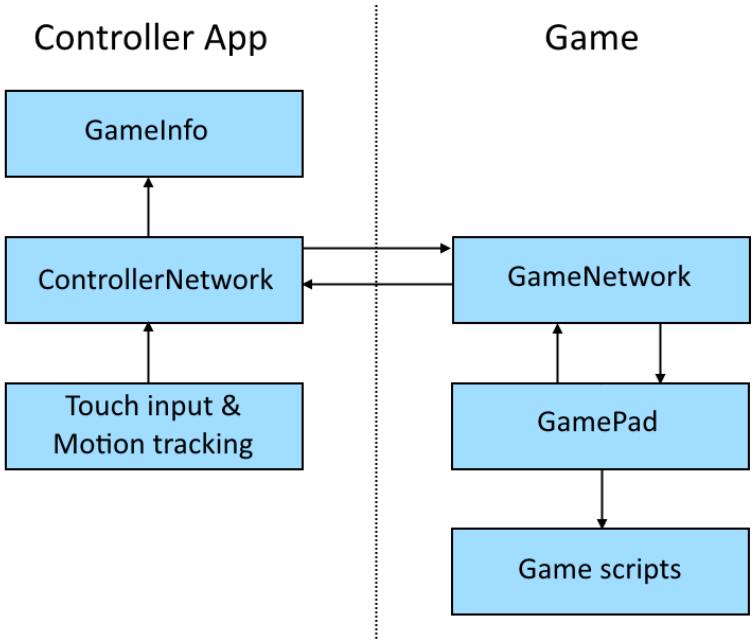

Figure 2.2: Data flow within applications.

2.2.1 Multiplayer communications

Although networked multiplayer was not within the scope of this project, there were discussions about how that could be implemented in the future. Figure 2.3 shows how each player's game is running a local server to handle the communications with it's controller but the game is also a client of the multiplayer server. To the multiplayer server the player is a single entity and no data flows from the player's controller to it.

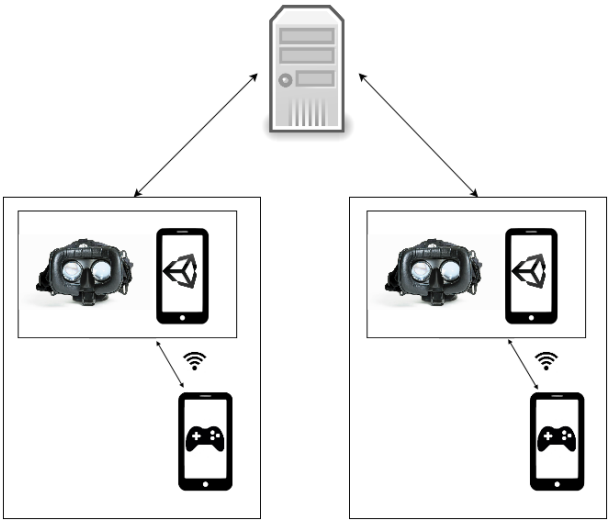

Figure 2.3: Multiplayer communications diagram.

2.3 Controller interface

Since the player can not see the screen of the controller while wearing the VR headset, the ways in which the touch surface can be used is limited to some degree. A decision was made to design the controller UI in a way that would split the screen into two halves, each acting as a single button (See Figure 2.4). What helped this decision was the fact that most touchscreens are flat and slick surfaces. If the user can not use haptic feedback to navigate the controller with touch, the buttons must be very large and easy to locate.

Figure 2.4: The gamepad interface of the controller application.

Before the player can start using the controller they need to connect it to a game. As can be seen in Figure 2.5, a dropdown menu is presented with all the games that are currently broadcasting a need for a controller. The games are identified by a combination of 2 random

words, in this case the selected game ID is “surprise fabulous”. After the player has selected the game they wish to connect to they press the “Connect” button and the game can start.

What the different controller actions actually do in the game will be covered in the next chapter.

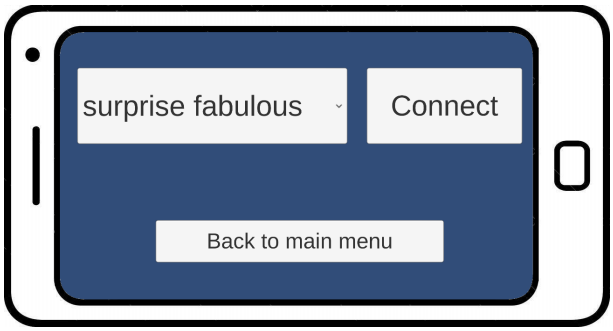

Figure 2.5: Establishing a connection to a game.

Chapter 3

The Game

Rewinding back to the beginning of the project. The basic implementation of the controller system was finished but there was still no real design for the game. At this point, the team decided to make use of a game design framework in an attempt to make the design robust, with a unified vision that all team members could agree on.

3.1 Idea generation

The idea generation framework the team used is a modified version of the Mechanics-Dynamics-Aesthetics framework, abbreviated MDA. MDA works by brainstorming ideas for a specific aspect of the game at a time, aspects such as setting, game mechanics etc. Ideas are pitched in anonymously from everyone and after a set time the suggestions are revealed and discussed. A small number of suggestions are chosen after the discussion to become definite parts of the design. The suggestions for that category then build up ideas for the next category, mechanics within a game can build up an idea for the setting and vice versa. This process continues until every aspect has ideas allocated to it, then the game's overall design is discussed. Then the process can start over again on a new game idea. Once the team considers the number of game ideas to be sufficient or that a good enough idea has emerged, the game ideas are compared and one is chosen.[12]

Simplicity was an important aspect of the design, 15 weeks was a very small amount of time to develop a game, there were many pitfalls that the team could have lost time to. The suggestions for the game's design took special note of this so overly ambitious game designs were simplified or discarded. The core design the team came up with is best explained by briefly describing all the elements that were generated with MDA. Some of these elements have been elaborated on for clarity. Reference Figure 3.1) for a visual representation of the flow of the MDA model.

Mechanics The player's possible interactions with the game.

- Player Gets stuck in areas and needs to find a way out.
- Player can spit projectiles.
- Player can collect power ups that have different effects on gameplay.

Dynamics The experiences a player can have in the game.

- Player can eat almost everything in the game when they are big enough and when they eat they grow in size.

- Player can spit out a projectile to break objects or open up new areas but doing so will cause the player to shrink in size.

Design Goals What effects the designers want to have on the player, this could be teaching something or having the player feel some emotion.

- Player should at first feel trapped and enclosed in a small area and as the game progresses they will gradually feel freedom as they open up new areas and spaces to explore and grow bigger.

Context The theme or setting of the game.

- The player is a mutated fly monster in an irradiated city.

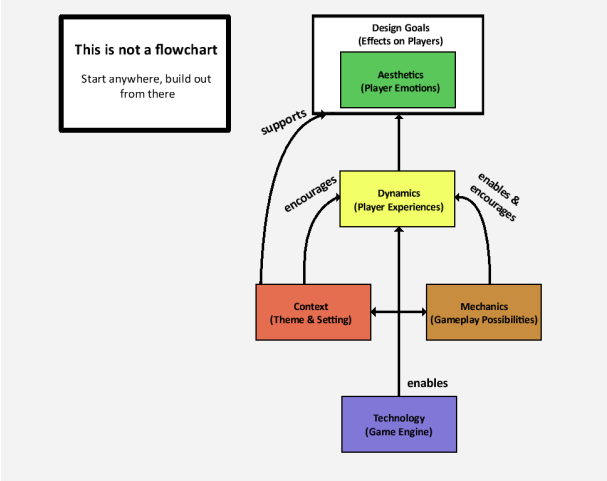

Figure 3.1: The MDA framework.

3.2 Game summary

The player takes on the role of a mutated fly in an irradiated city. The fly has gained the ability to grow rapidly after it eats objects smaller than itself. The fly also has the ability to fly around freely in a three-dimensional space, the player can directly control the fly's direction with gyroscopic sensors in their smartphone controller.

The player must help the fly overcome various obstacles in the form of puzzles and enemies that approach the fly whenever it lands on the ground. While the fly has the ability to grow by eating, it also has the ability to shrink by spitting projectiles. Some puzzles will require the fly to become small or to spit projectiles at specific targets.

The player should be feeling trapped in their limited space for a while until they finally escape and find freedom. The player should also feel relaxed when flying to contrast the tension they feel when the fly is landed. Lastly, the game allows the fly to eat almost any visible object in the game.

Chapter 4

The growth effect

Like we mentioned in the introduction, the team wanted to come up with an effect that could convince people that they were growing or shrinking. In this chapter, there will be an in-depth explanation of the method. Let us first define the term "camera" and then describe the implementation of the effect. "Camera" has the same meaning as "eye" or origin point of the viewing frustum in 3D graphics. It is essentially the eye of the player inside the virtual world.

The team speculated that by moving the cameras away from each other we could simulate the effect of growing. By increasing the horizontal distance between cameras as well as the vertical distance between the cameras and the ground surface, the player should feel as if they are growing.

Virtual reality headsets use two cameras to create the stereoscopic 3D effect to make the player feel as if they are immersed in a virtual world. This caused a problem with how cameras in virtual reality are set up by default. Their positioning is very close together and the cameras' forward vectors are parallel to each other. This is done to maintain clear focus on objects close to the player. If the cameras drift apart but maintain their forward vectors then the result is extremely blurry visuals that induce kinetosis, also known as motion sickness, in players. To counter this loss of focus the team calculated a correction, a new vector for each camera that converges in a new focal point. Adjusting the forward vectors by 2.8 degrees towards each other(see Figure 4.1) causes the player's focal point to move farther away from them in direct proportion to their virtual size. The distance between cameras could then be adjusted without causing players major eye strain. Refer to Figure 4.2 for a visual representation.

While the horizontal distance between the two cameras grows, the vertical height of the cameras also increases proportionally.

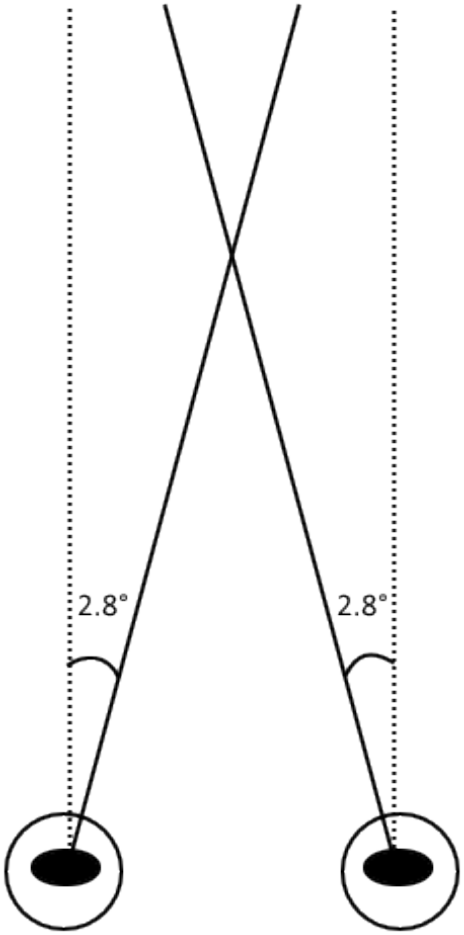

Figure 4.1: Top-down view of the cameras, dotted lines are parallel.

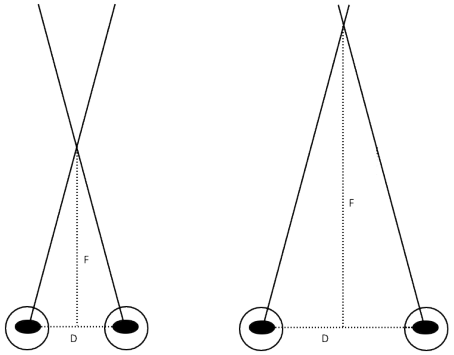

Figure 4.2: Focal point moves further away when distance between eyes is increased

Chapter 5

Research

Team members were enthusiastic about implementing a method to convincingly simulate growing rapidly. To confirm that the method was effective, a prelimary research study was conducted. This research took time away from developing the game to instead contribute to the academic study of the effects of virtual reality on people.

5.1 Experiment method

The experiment relied on separating participants into two groups, one group that was aware of what was being researched and another group that was unaware. The experiment spanned two days, with each group of 10 participants conducting the experiment each day. The experiment was conducted at the facilities of Reykjavík University.

First participants were asked a few background questions. Next, they were handed a virtual reality headset unit and the experiment began. The virtual avatar of the participants would increase in size at set intervals, the experiment conductors had control over the avatar's size. During the experiment, participants were asked a few questions regarding what was happening in the virtual world, how they were feeling and if they could spot certain objects. Once the experiment was over they were asked to fill out a form containing follow-up questions about their experience.

5.2 Results

The results of the experiment were not what the team had anticipated. Most participants felt like they were going up in an elevator rather than becoming bigger. The difference between the two test groups was minimal, although the group that was told about the experiment described their experience as "growing" more frequently. This could be attributed to the experiment conductors telling one group of participants what feelings to expect from the experiment.

The background questions form provided some necessary context about the participants. The team could conclude that, while using this method, there was no correlation between a participant's height and how they sensed their scale within a virtual environment, which the team had theorized would have a correlation. Most participants did not feel nauseous after taking part in the experiment, the background information had no bearing on this result. Three participants expressed that they felt anxiety during the experiment due to their fear of heights. That could be an indicator that they felt as if they had been lifted up rather than feeling like they had grown.

5.3 Conclusion

Did the growth effect work as intended? The data obtained from performing the preliminary experiment seemed to indicate that the method was flawed. A lot of participants expressed feeling as if they were being lifted, as if they were gliding or that they were being pulled upwards. The team proposed that there were two probable causes for these descriptions. Either the growth effect in the experiment was happening too quickly, or the participant had no avatar as a point of reference to the rest of the virtual world. Despite the results of the experiment, the team continued working on the method. Too little time remained of the project to conduct further experiments or fundamentally change the effect.

Chapter 6

Project Management

At the start of the project, there was very little certainty regarding most aspects of it. It would take a few weeks to establish project goals and the general direction of the project. Once the plan was formed it became significantly easier to organize the team and stick to what was important. The use of Scrum would help the team immensely with this organization although members doubted it's effectiveness at the beginning.

6.1 Why Scrum?

The team spent some time investigating what Agile frameworks would fit a project with very few determined aspects. Scrum, Kanban, and Scrumban were the first clear choices. The team quickly decided that Scrum offered a better option for observing and presenting work progression, every member was also familiar with the basics of Scrum and it was therefore chosen to start with. If Scrum would not work out then the team planned to switch to Kanban within the first few weeks.

The team still worried that game development with an unclear direction would be difficult to work on with Scrum. The backlog was originally filled with epic stories and vague requirements. After the game's design phase was completed and a long term list of requirements was prepared, those worries were put to rest. After the long term stories were decided the team held a planning poker session to estimate their scope. Stories added later would then be given story point values based on other stories and the votes of team members.

6.2 Time logging

The overall time plan was to spend 1200 - 1440 hours on the project, splitting work hours evenly between team members. The team decided to use an application called toggl to log work hours. With toggl the team could accurately monitor every team member's activity and compare real progress with the planned progress derived from Scrum. For the most part, this combination was very effective at motivating team members as well as estimate how much work could be done each sprint.[13]

6.3 Sprints

Sprints were organized to last two weeks at a time, starting on Mondays and ending on Sundays. Each sprint would begin with a meeting with the product owner, followed by a sprint planning

meeting. This helped focus the team on the most important goals for each sprint. Meetings with the product owner occurred twice each sprint. One of those meetings was used to update the product owner on the status of the project. This arrangement proved very effective at synchronizing the team's understanding of the goals with the product owner's expectations. Meetings with the project instructor were also twice each sprint. At the end of each sprint the team would discuss the status of any remaining tasks, if there were any, review the sprint and hold a retrospective meeting.

6.4 Retrospectives

Retrospective meetings were originally set up with a "Stop doing, keep doing, start doing" format. Team members would write down anonymously on notes what they felt the team should stop doing, keep doing or start doing based on how that sprint went. These notes would then guide the team through the next sprint. Later in the project the team found this method insufficient and unproductive, it was then decided to change retrospective meetings. After researching retrospectives online the team found a promising method. The format was "Sad face, neutral face, glad face". Writing on post-it notes continued but this time, the idea was different. Team members could write positive encouragements, compliments and "kudos" to each other for work well done in the passing sprint, this particular aspect made the team more enthusiastic about retrospective meetings. Neutral and negative things that occurred during the sprint were written down and put on the neutral face or sad face part of the board. These notes would then be grouped together by relevance and classified as "actionable" or "non-actionable" problems. Actionable problems would then be discussed and given proposed solutions. These solutions would then be especially considered in the next sprint. This method proved far more effective, boosting team morale as well as identifying the most important problems the team was facing and what could be done to solve them before the next sprint began.

6.5 Conclusion about Scrum in game development

Using Scrum to develop a game which had very few requirements at the start of the project proved to be very successful. Maintaining a good perspective of work done versus work to be done was trivial after the initial work cost of setting up long term requirements. Similarly setting up a fast way to retrieve progress reports on the project was time consuming prior to status meeting 1. However, it became immensely time saving when preparing for status meeting 2 and 3. Spending time early to save time later was very helpful near the project deadline. The team's overall plan was successfully completed, partly thanks to the flexibility of Scrum. It was effective at motivating the team to complete the right tasks at the right time. The team recommends that other development teams try using Scrum for their game projects.

Chapter 7

Conclusion

In conclusion, the team believes every major goal was met along with most minor goals. The smartphone controller application was feature complete according to the parameters set by this project.

The game itself was nearly feature complete. What it required most was graphical optimization, original artwork and more levels to make it a full-fledged game ready for release.

The research report was a tremendous success. The results were interesting and gave the team a good idea of what could be done to improve the growth effect.

Project management proved to be easy and productive for the entire team. Scrum was successfully used to manage the development of a small game with many unknown variables, something the team did not expect at the beginning. The backlog was completed at the estimated time, team work capacity lined up with original predictions and work progressed well ahead of schedule. The team also accomplished these goals in less time than was originally anticipated, additional time of the planned 1200-1440 hours was devoted to improving the game.

From working on this project, team members have learned a great deal about cooperation on a large-scale project over a long duration. The project also gave everyone valuable experience with developing games and conducting academic research. The game engine Unity was something all team members were familiar with at the start, but after the project was over everyone had become more adept with it. Finally the team learned how to apply Scrum more effectively to manage a project, focusing more on eliminating waste, properly estimating the scope of work and using that to stick to the long-term plan.

Bibliography

[1] (2016). Oculus rift: Step into the game. Kickstarter funding page, [Online]. Available: https://www.kickstarter.com/projects/1523379957/oculus-rift-step-into-the-game.

[2] (2014). Facebook oculus rift acquisition, [Online]. Available: https://www.facebook.com/zuck/posts/10101319050523971.

[3] (2016). Oculus, [Online]. Available: https://www.oculus.com/en-us/.

[4] (2016). Samsung gear vr, [Online]. Available: http://www.samsung.com/global/galaxy/wearables/gear-vr/.

[5] (2016). Playstation vr - virtual reality headset for ps4, [Online]. Available: https://www.playstation.com/en-us/explore/playstation-vr/.

[6] (2016). Htc vive, [Online]. Available: https://www.htcvive.com/us/.

[7] (2016). Google cardboard, [Online]. Available: https://www.google.com/get/cardboard/.

[8] (2016). Unity - game engine, tools and multiplatform, [Online]. Available: http://unity3d.com/unity.

[9] (2008). Scrum in game development - mountaingoat software, [Online]. Available: https://www.mountaingoatsoftware.com/presentations/agile-and-scrum-for-video-game-development.

[10] (2016). High level api - unity manual, [Online]. Available: https://docs.unity3d.com/Manual/UNetUsingHLAPI.html.

[11] (2016). Transport layer api - unity manual, [Online]. Available: https://docs.unity3d.com/Manual/UNetUsingTransport.html.

[12] (2016). Mechanics-dynamics-aesthetics framework, [Online]. Available: https://en.wikipedia.org/wiki/MDA_framework.

[13] (2016). Toggl - time logging service, [Online]. Available: https://toggl.com/.

School of Computer Science
Reykjavík University
Menntavegur 1
101 Reykjavík, Iceland
Tel. +354 599 6200
Fax +354 599 6201
www.ru.is