Norwegian University of
Science and Technology

Conductor Hero - Project Report

Authors

Rikhart Vigdal Bekkevold
Sabina Niewiadomska
Per-Morten Straume
Ida Ellinor Syverinsen
Andreas Wang
Yijie Zhou

IMT4310 - Experts in Teamwork
Norwegian University of Science and Technology

04.05.2018

Summary of Project

Title:	**Conductor Hero - Project Report**
Date:	04.05.2018
Authors:	Rikhart Vigdal Bekkevold Sabina Niewiadomska Per-Morten Straume Ida Ellinor Syverinsen Andreas Wang Yijie Zhou
Contact Person:	Andreas Wang, andrwan@stud.ntnu.no, 48048162
Keywords:	Project Report, Virtual Reality, SteamVR, Unity Engine, Interaction Design, Game Design, 3D Modelling
Pages:	22
Attachments:	1

Abstract: Experts in Teamwork is a course where students from various disciplines are put together in a team to work on a project throughout the semester. For the course, our group worked on a VR game called Conductor Hero where the player takes the role of a Conductor in a fantasy environment. The development of this project has taught us the importance of working with tight schedules as well as provided experience with cross disciplinary teams. The project report for Conductor Hero contains the technical details and methods for how the game was developed as well as discussions around the decisions that led to the final product.

Contents

List of Figures

1 Introduction

1.1 Background

Experts in Teamwork (EiT) is a compulsory course for master degree students at NTNU Gjøvik. The main goal is to help the students to develop interdisciplinary teamwork skills [1], such as responsibility for project development, understanding the possibilities which appear from interdisciplinary teams and sharing feedback in a constructive way [2]. Our village, "Virtual and Augmented Reality for Games, Health and Education", focuses on how new technology like virtual reality (VR) and augmented reality (AR) can be used not only for entertainment but also in resource demanding sectors like healthcare and education. We were free to come up with our own ideas for this project, as long as it was related to virtual or augmented reality.

1.2 Conductor Hero: An introduction to ensemble conducting

Conductor hero is a VR rhythm game where the player takes on the role of a conductor. Conducting an ensemble throughout a song, the player ensures that the instruments follow the beat of the song and play at the appropriate times.

The goal of our project was to make an enjoyable game which introduces players to the world of ensemble conducting, an area that can seem quite mysterious to the layman. Through this game, we wanted to expose the player to the domain of ensemble music and encourage them to expand their horizons, build interest and understanding, develop musicality and sensitize the player to musical experiences. Additionally, we are hoping that the game could facilitate the promotion of conducting as a hobby or potentially future choice of education.

The game aims to improve the rhythmic skills of the user through conducting, requiring rhythmic hand movements corresponding to the beats of the music. Additionally cueing helps the user understand how musical tensions are built through different instruments entering and exiting the song in different sections. VR is a new technology presenting new opportunities and challenges. The gesture tracking that exists in modern VR systems allows for an authentic conductor experience. The immersive nature of VR also allows us to place the player within a fantasy world which strengthens the mood of the music being present. Current room scale solutions are sufficient for this type of game, as we do not require the user to move around.

1.3 Market exploration

Predictably, the number of VR conductor games available at the current time is low. We were not able to find any products directly related to our ideas, however, we looked into similar games in order to see what worked well and what did not.

1.3.1 Audioshield

Audioshield is one of the more well known VR games on the market. It is a rhythm game where the player blocks incoming spheres in beat with the music. The rhythm of the song determines the patterns and frequencies of the spheres. A sphere is blocked when it hits the player's shield of the corresponding color. Audioshield served as an inspiration for our placement of the player, and their point of view.

1.3.2 Guitar Hero

While not a VR game, Guitar Hero is a well-known series that covers many of the same aspects as those we want to achieve in our conductor game. In this game the player take on the role as a guitar player in a rock band, and must match the on-screen incoming colored notes. This is done using a plastic guitar controller, and by hitting the notes you score points and engage the audience. While our game focused on conducting rather than guitars, we wanted to create the same feeling of immersion and state of flow as players feel when playing a guitar hero game.

1.3.3 Wii music

Wii music is a console game that allows the player to take different roles in bands and ensembles, including the role of a conductor. Theme-wise, this was the game that came closest to our idea, with similar use of a musical track and beat indication. The game allows the player to mimic a conductor's movements, but the musical expressiveness is limited. The game has received mixed feedback, and we have found it especially interesting to also look into the aspects that had been poorly rated by players; this way trying to avoid the same pitfalls.

2 Methods

2.1 Idea Generation

The idea generation followed a ”divergent - emergent - convergent” process [3], as shown in Figure 1. At the first stage, we conducted a brainstorming session, where each group member freely expressed their ideas related to VR and AR in the field of games, health or education. A range of ideas was generated, including a rehearsal room for musicians or dancing classes, a public speaking practice place, collaborative painting or sand painting, a virtual memory palace, a 3D-data visualisation room and simulators for activities like gardening, parachuting or car racing.

After discussing all ideas we decided to apply dot-voting [3] which helped us to prioritize our favourite ideas. All of us received three votes and we were asked to distribute the votes (dots) as we saw fit. This meant that we could put all three dots on one of the ideas or we could put one dot on three different ideas instead. When everyone had placed all their dots, we narrowed our focus to the three ideas with the most votes.

One of the ideas was to create a service which lets the musicians or DJs play their own music and artists to display visualisations, while an audience could meet in virtual space to experience the audio-visual performance and dance together. Another was an educational chemistry game, aimed at school children. The final idea was to create an orchestral conducting game.

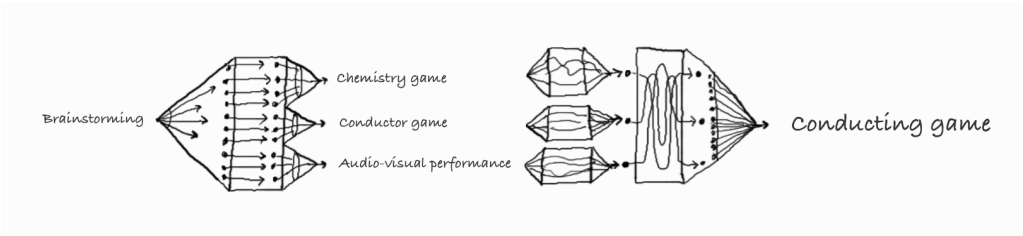

Figure 1: The process of idea generation [3]

In the following discussion of these ideas, we sketched the scenarios of the games, as well as the implementation details. We now had a better feel for, and understanding of, each idea. Together we decided that we would like to try to develop the orchestra conductor game. Among the reasons for this choice was the strong musical background of four group members, as we believed our musical experience would help us to create a more interesting game scenario. The lack of market competitors also played a part in our decision, additionally, the VR technology seemed to suit conducting well. Sketching of ideas was further used in the project to illustrate our ideas to each other as shown in Figure 2 which contains a sketch of how we envisioned the various elements of the Conductor Table.

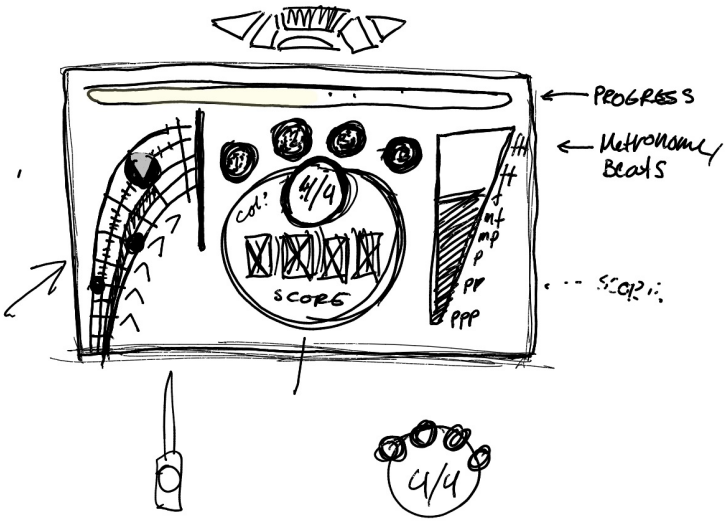

Figure 2: An initial concept sketch from the development of the Conductor Table.

2.2 Interview

After choosing to develop a conducting game, we interviewed a professional conductor over Skype to learn more about his experience of conducting and conductor training. The purpose of the interview was to better understand the different aspects of conducting. We also wanted to explore possibilities of involving VR games to promote the positive sides of conducting and to make any negative sides of the training process seem more engaging. During the interview, we found one motivation for the role of conducting, which is the feeling of control and leadership, as the orchestra follows your directions. It is relatively hard to get access to a whole orchestra for conducting training in real life, especially for conducting fanciers or novices. However, in a VR environment, it is possible to have access to a virtual orchestra with a customized composition at any time as long as the player has access to a VR device. With the interactions and feedback from the virtual orchestra, players can achieve greater motivation to keep on training. From the interview, we also got the idea that using unique virtual environments and game elements may ease the feeling of dullness and stress during repeated training. The feedback from the conducting expert confirmed pre-existing assumptions for the features in the conductor game and helped us to have a clearer vision about potential users.

2.3 Tools

The game was created using the Unity engine (version 2017.3.0.3f) with C# as a language for the scripting components. Microsoft's Visual Studio 2017 Community was used as the Integrated Development Environment. Version control was done through Git using Github as our hosting service. Google drive was used for storing sound files, as these came close exceeding Github's restrictions on file size, and to a large extent slowed the system down. Valve's Unity SteamVR plugin was used for communicating with the HTC Vive. Blender was used for modeling 3D assets, like the instruments and game world. The UI for the conducting table was created with Adobe Illustrator and Adobe Photoshop. The

soundtrack used for the game was composed in Guitar Pro 5. Cubase 5 was used as the Digital Audio Workstation when generating audio from the midi tracks exported from Guitar Pro 5, using Virtual Studio Technology instruments from EastWest/Quantum Leap symphonic orchestra to produce the desired individual instrument tracks. Discord was used for intragroup communication.

2.4 Development Methodologies

The development followed a mixed methodology model with a focus on aspects that would allow us to work in parallel and quickly prototype features. Code hygiene factors was a low priority due to the limited time and we did not expect that performance would be an issue with such a limited game. We, therefore, adopted a ”critical path coding” [4] approach to get as much done with as little code as possible.

Developers usually worked individually after short discussions on architecture, however, pair programming techniques were employed for more challenging or unfamiliar problems. We originally intended to have a focus on the process, making proper use of issue tracking, branching strategies, and code reviews. However, early in the project, we came to the conclusion that this added unnecessary overhead to a time-limited project. We decided it would be more beneficial for the project that we spent more time on implementation rather than documentation and process.

In terms of meetings, we employed daily three minute meetings during the morning of every village day in a similar vein to those used in agile software methodologies. These meetings consisted of every group member mentioning any additional progress that had happened since the last day, what they planned to work on during the village day and discussing any potential problems that may arise. We kept these meetings to a maximum of three minutes per group member as a means to avoid situations where certain group members could end up dominating the discussions.

3 Results

3.1 Game Design Document

We generated an initial design document for the game, giving a brief overview of the envisioned game, and a plan for the game mechanics. This document can be found in the Appendix A.

3.2 Game Implementation

After creating the game design document we implemented a functional prototype of the conductor hero game. The source code and binaries are located on github[1]. In order to play the game, a functioning HTC Vive headset is required. Instructions on how to install and play the game can be found on the main GitHub repository page. A youtube video[2] was recorded to demonstrate the features of the game.

3.2.1 Asset Attribution

The majority of assets used in Conductor Hero were created by us with a few exceptions:

- Particle Fog textures were acquired from Unity's Standard Assets.
- The skybox used in the game was acquired from the Unity Asset Store[3]
- The models for the HTC Vive Controllers were provided with the SteamVR plugin.
- The font used in the Conductor Table was acquired from dafont[4].

3.2.2 Core Game Components

The four core components that make up the game are: GameManager, Metronome, AudioManager, and MotionTracker.

The GameManager component manages musical cues as well as scoring. The Metronome component handles the core beat logic. It allows outside components to ask how close they are to being on beat. Other components can also add callbacks to the metronome which will be called every beat. The default sound for the metronome was removed, partly due to uncertainties regarding licensing. The AudioManager manages simultaneous playback of the multiple instrument tracks, metadata about song length and supports muting and unmuting individual tracks.

Finally, the MotionTracker component handles game logic related to the right hand conducting. This includes managing the visuals of the pattern and any logic for tracking the player's movement within the pattern, using the tracking spheres in the scene.

[1] https://github.com/Per-Morten/imt4310_conductor_hero
[2] https://www.youtube.com/watch?v=YQQTDyfQb-Q
[3] https://assetstore.unity.com/packages/2d/textures-materials/sky/10-skyboxes-pack-day-night-32236
[4] https://www.dafont.com/white-rabbit.font

3.2.3 Standard game flow

The following section describes the standard game flow in the prototype level of Conductor Hero:

- The player starts the game and the stage will go to an unlit state.
- The two controllers are separated by having one with a laser pointer and one without. The "laser pointer" on one of the controllers indicates that it should be held in the left hand.
- In order to start the game, the player needs to hold their right controller in an upwards position where they would feel comfortable having the top of the conducting pattern and press the menu button to start. The game starts a countdown to the song which ends with the first cue.
- While playing, the player's left hand is used to hit cues on beat while their right hand is used to keep the conducting pattern going.
  - A cue appears at designated times during the song. The cue has a circular progress bar which implies when the player is supposed to hit it.
- The game ends when the song is finished.

3.2.4 Shortcomings of the Prototype

In the prototype, the game will start the countdown to the song and show the pattern to the player, but currently, it will also start counting score for movement even if the song has not properly started yet. This is unintended as we would like to start measuring score after the first cue, but we did not have the time to fix this by the end of the project. The game is also supposed to end once the song duration has expired, but we do not have any additional end states to signify this due to time constraints.

3.2.5 Playtesting

As part of the development, we performed a lot of internal testing within the development team. We also had an initial playtesting session with the entire group and also asked some members of other teams to try out the game and give feedback. This playtesting session is further discussed in the Discussion Chapter.

3.3 Presentation

As part of the course, we also had a presentation towards the end where we showed off the project to the class, the supervisor and industry externals. The slides from the presentation can be found on Google Slides [5]

[5] https://docs.google.com/presentation/d/1unr5goORWJpajkHhiNFVHJUfyByFeLB_AzXME653Kd0/edit?usp=sharing

4 Implementation

4.1 Implementation Details and Encountered Challenges

4.1.1 Beat Logic

Since the game relies heavily on fixed beats that correspond to the music, we decided to create a central metronome object. The metronome ended up supporting two ways of identifying the current beat, a beatID property returning the current beat, and a callback called on each beat update. All game objects were originally intended to register as listeners to the ”OnBeatTickCallback”. However, as we were not in total control over the order of callback executions, we ran into logical inconsistencies in the cueing system. We worked around this issue by only registering callbacks in order independent systems. During these callbacks, each system would be responsible for updating their order dependent subsystems through direct function calls. This introduces some more coupling into the codebase, but not to an unmanageable degree.

4.1.2 Audio Playback

Requiring the player to cue in different instruments throughout a song, un-muting individual instruments on success or muting them on failure, presented an interesting challenge. Controlling the different tracks was done through a simple enum, however, we ran into issues that the tracks were not necessarily synchronized, which was devastating to the game. Luckily Unity’s AudioSource supports scheduling tracks, fixing the synchronization problem [5].

4.1.3 Cues

The musical cues were implemented as a finite state machine as seen in Figure 4. The cue transitions between different states are based on the number of beats left before the instrument should enter and whether or not it has been hit by the player. Implementing the cues as a finite state machine allowed us to structure the code in a way that was consistent with the state transitions of the cue, making it easy to follow the logical execution of the cue behavior. An alternative solution could be to go with a full state pattern[6], with separate classes for each state. However, we did not deem the cue logic to be complex enough to justify such a solution. An example of a cue can be seen in Figure 3.

4.1.4 Creating an illusion of glowing light for objects without light sources

In terms of the in-game environment, the original plan was to include bright glowing mushrooms scattered around the area. A problem arose when the number of illuminated mushrooms in the scene exceeded a manageable amount, severely reducing the performance of the application. We wanted to keep the visual integrity of the scene as close as possible to its original Blender model, so we came up a workaround for the problem.

Instead of illuminating the mushrooms, we worked around the issue by changing their material to a bright colour. We then added a bloom post-processing effect, creating an illusion of glowing light around the mushrooms. This did not light up the scene in any

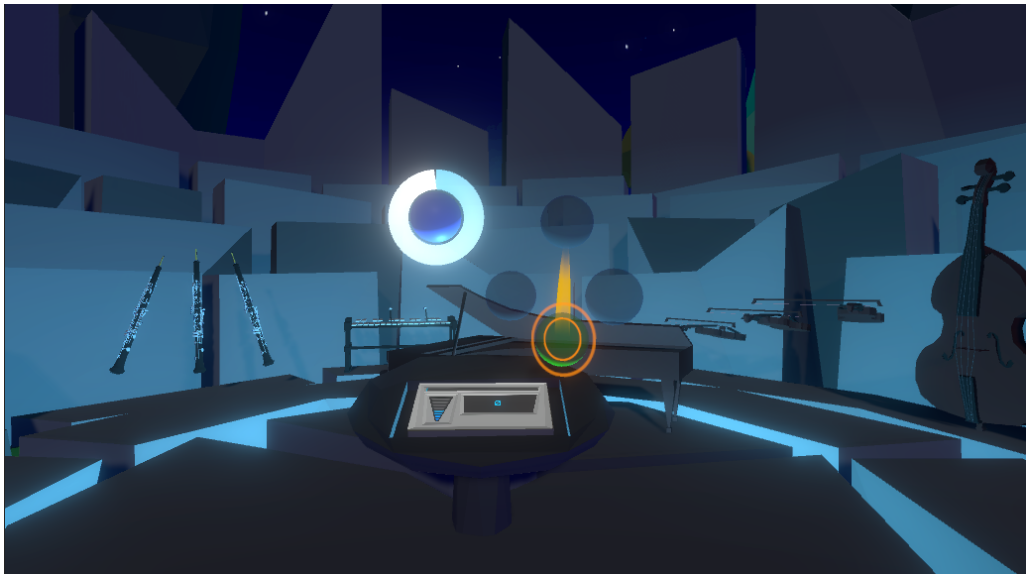

Figure 3: This screenshot shows a Cue that is ready to be hit by the player to signify that an instrument track should start playing.

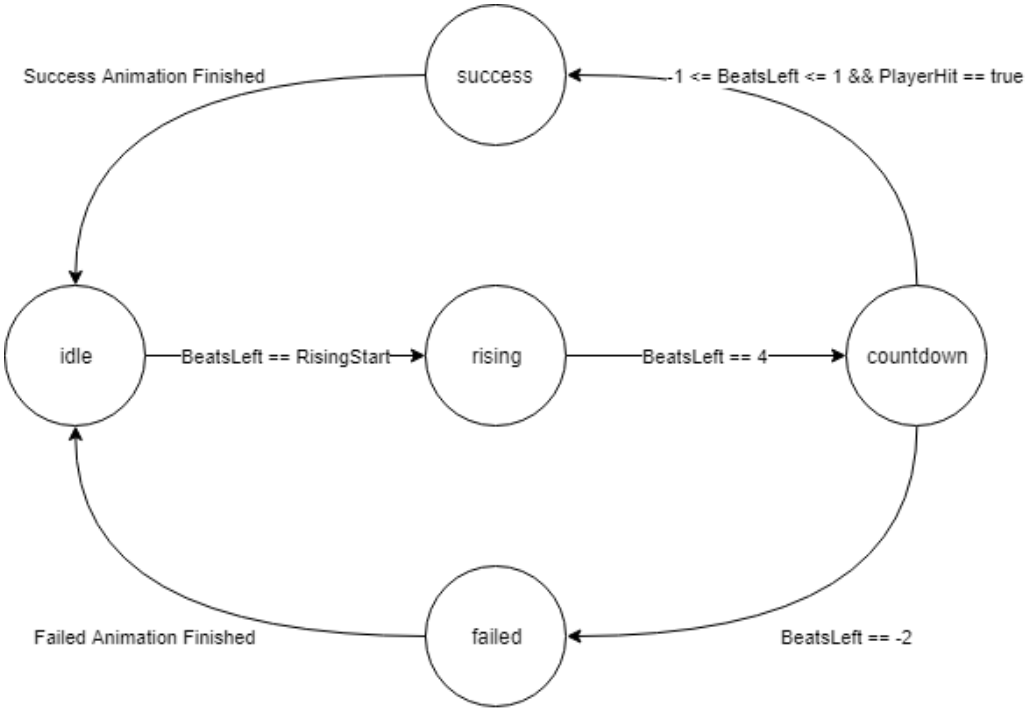

Figure 4: This model shows the finite state machine used for the cueing logic

way, but the small size of the mushrooms and the sheer amount of them made this the most logical solution to keep the visuals intact without sacrificing performance.

4.1.5 Creating simple fog effects using Unity's particle system

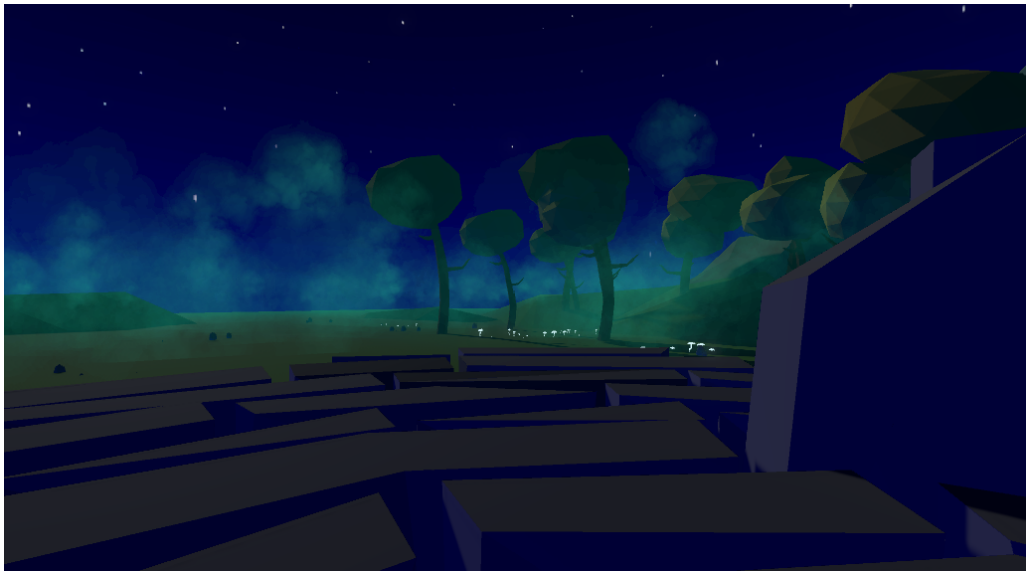

Figure 5: This screenshot showcases how the ingame fog looks like.

The original concept idea for the in-game environment also included some green fog that was spread around in the scene. Ideally, this could be handled through the use of Volumetric Fog or similar implementations, but due to limited time to work on the project, we had to find a quicker and simpler solution to implement while still providing a relatively good approximation of the fog effect.

We implemented fog using Unity's Particle System component, utilizing a single fog texture from the Unity particle effects standard assets pack. We then layered several particle systems on top of each other with small variations in settings like particle size, rotation, lifetime, and, emission rate to create the illusion of fog. We combined this with Unity's distance fog functionality which makes geometry fade towards a gradient colour based on distance from the camera. This enhances the illusion of fog. The end result does not look equivalent to Volumetric Fog, but we believe it serves its purpose, and saved us a good amount of development resources. A screenshot illustrating the visuals of the fog can be seen in Figure 5.

4.1.6 Demonstrating the conducting pattern visually

The initial implementation for demonstrating the conducting pattern included four transparent spheres. These spheres would change their material from a gray to a green colour, indicating that the player was supposed to touch it at that beat. This method made it possible to see the conducting pattern as only one sphere would be green at any given time, and the colour changed on beat which reinforced the rhythmic movement to the player. While the initial implementation was fine for internal testing, later playtesting with externals provided some insight into the fact that more visual feedback and information

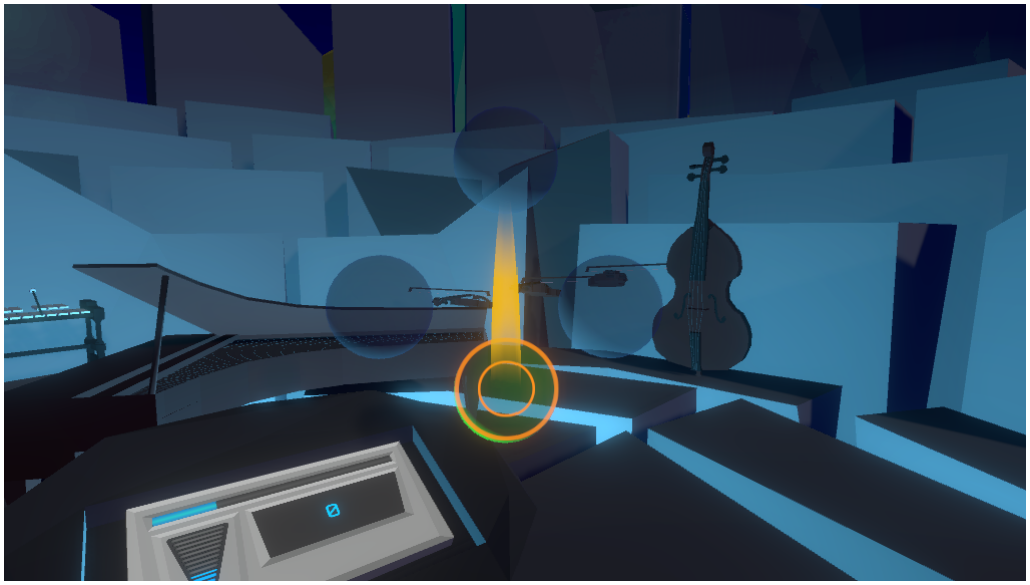

Figure 6: This screenshot illustrates the four motion tracking spheres used for conducting. The currently active sphere is coloured green with expanding circles emmitting at the start of the beat. There is also a trail that further illustrates how the player should move their hand between beats.

would be preferred. This was particularly apparent for testers with no prior experience in conducting as they did not understand how they were supposed to move their hands.

Our solution to this was to create trails between the four different spheres which demonstrated the pattern in a slightly more visual manner. This was handled by taking an invisible ”GameObject” with a ”TrailRenderer” and interpolate it between the previous beat’s sphere and the next beat’s sphere.

To emphasize the rhythm we added expanding circles instantiated on the beat. These also indicated the next step in the conducting pattern. Additionally, the first beat was emphasized with a different colour to more easily allow the player to get back into the rhythm. This can be seen in Figure 6.

If more time was available, we would probably have used Bezier curves to demonstrate the conducting pattern in a more correct manner rather than having linear trails between the spheres. The added complexity of this approach is negligible, but it would take a fair amount of time to create proper looking curves without any visual aids in the Unity editor for bezier curve control.

4.1.7 Binding animation states to the beat

To emphasize the rhythm of the song in Conductor Hero we ended up tying the majority of animation states to the ”OnBeatTickCallback” in the Metronome component. While this was an interesting way to work with animations it also posed a fair share of problems. An issue with changing animation states on beat was the cases where it would be preferable to change the animation at different times than the callback. An example of this issue is detailed below:

The player is supposed to start moving their hand towards the next beat before it

starts, but the interpolated trail only starts moving towards the next sphere whenever the next beat triggers the beat callback. With a linear interpolation, this means that the trail lags behind where the player should have moved their hand between two beats, potentially confusing the player and making it hard to read the rhythm from the movement of the trail. To work around this we slightly modified the linear interpolation by parameterizing the input time through the use of an ”AnimationCurve”. This makes it possible to control the velocity of the animation throughout its lifetime. Using this, we made the velocity of the trail very fast early on, to create a more natural motion towards its destination to where the player needs to move their hand.

4.1.8 Providing Visual Player Feedback

Figure 7: This screenshot shows the dashboard on the Conductor Table where the top bar is current song progress, the left bar is the volume bar and the display on the right shows current score compared to maximum score.

It is important to provide the player with enough visual feedback for them to understand how their actions are being reflected in the game. In Conductor Hero, visual feedback is provided using several methods.

Particle effects in the tracking spheres indicate whether or not the player is conducting correctly. Red particles indicate that the player is off beat, while green signifies being on beat.

We are also providing some visual feedback on the conductor table whenever the player gains score for either conducting on beat or hitting cues.

A small popup appears in the table’s canvas whenever the score is increased, telling the player how much score their actions have resulted in. A score of ”+10” is given for hitting cues correctly while a score of ”+1” is given for every conducting motion that is on beat. The score values obtained by hitting cues could have been bigger, and more visually distinct, to further emphasize the larger amount of points gained, but we did not implement this due to time constraints. The visual feedback used in the conductor table

can be seen in Figure 7.

4.1.9 Design of the in-game environment

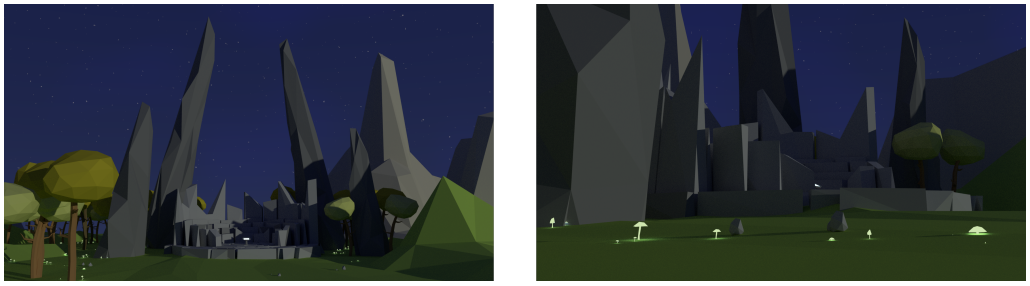

Figure 8: These screenshots showcase the 3D environment made in Blender.

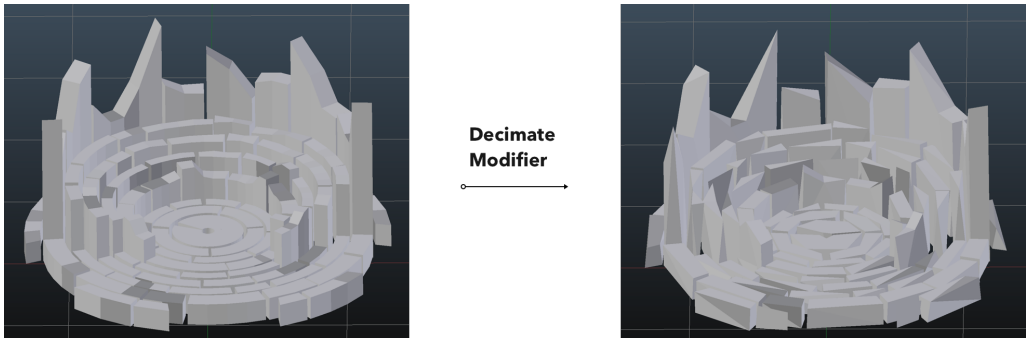

Figure 9: This figure illustrates how the Decimate Modifier in Blender can create a low poly version of a regular 3D model.

A goal of Conductor Hero was to immerse the players into a unique VR environment while conducting the orchestra. We discussed various environments like a seabed, or the outer space, but settled on the forest scene for the prototype due to time constraints. We envisioned that the musicians of the orchestra could change to fit the environment, turning into beasts, fish, or aliens. The game has a low poly style, as the concept did not require high-fidelity graphics. We believe that the visuals look appealing to the audiences while being stylish, they are easy to create and iterate upon, which has the added benefit of fitting within the time scope of our project. To achieve this style we used the Decimate modifier in Blender to decrease the number of polygons in each object. An example of how the modifier was used can be seen in Figure 9 while the final 3D model of the environment can be seen in Figure 8. We reviewed existing games using low poly styles and imagery of forests to create a harmonious colour palette for our scene. The colour palette we ended up using is shown in Figure 10. During this review, we found that many existing rhythm games use lighting to emphasize the rhythmic feel of the song. To achieve the same effect we introduced lighting into the floor that the player is standing on, which pulsates with the rhythm. The cyan light below the stage and the ”glowing” mushrooms around the environment further enhance the fantasy theme of the scene.

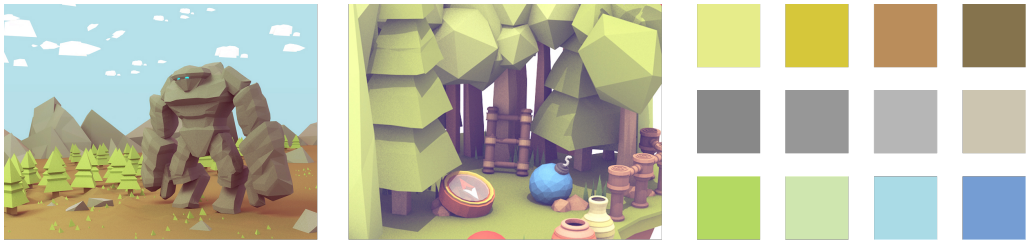

Figure 10: The colour palette we used in the project was influenced by styles similarly to what is illustrated here. The final colour palette is seen on the right.

5 Discussion

5.1 Changes between the initial game design and the final product prototype

The final prototype ended up retaining a lot of the core ideas from the initial design. The changes that were made are discussed in this Section.

Originally, it was intended to let the player use their dominant hand for conducting the time signature. This would be the case in a real-life situation, as conductors can freely choose to use whichever hand they desire. However, due to time constraints, we only allow for right-handed time signature conducting. This functionality had a low priority for implementation as our left-handed testers were able to play ”right-handed” without any issues.

We also wanted to score the player on the accuracy of their conducting movements as well as their rhythmic precision. In the end, only rhythmic scoring was implemented. Our implementation for motion tracking lacks the sophistication needed to compare the player's movement to a predefined motion. Instead, the player is now required to follow a simple pattern that corresponds to where their hand should be at certain beats to gain points.

Volume control was not implemented in the final prototype due to time constraints, although this was a difficult decision to make. We realized that it would be challenging to find a suitable way to integrate this with the other mechanics.

The Conductor Table/Dashboard was kept as originally designed, except for two changes. The volume bar was left on the table with logic for increasing/decreasing volume attached but ended up being unused as the volume control, in general, was dismissed during development. The other change relates to cueing, which was originally planned to be indicated on the table. The cues are now indicated by a floating sphere that the player has to hit at the right time. The rationale behind this change was a desire to avoid the player constantly looking down at the table for essential information, but rather supply it to the player as they look at the orchestra.

We originally envisioned two game modes, a performance mode, and a freestyle mode. In the performance mode, the player is expected to follow a given pattern as close as possible. The freestyle mode would allow the players more artistic freedom by controlling both the volume and cueing at will. We decided to only implement the performance mode, based on the scope of the project, as the chosen mode would fit better with our envisioned implementation of the cueing system. The freestyle mode where the player could cue in instruments at will would also require a more robust fading system, and potentially another song that better fit with the removal and additions of the different instruments.

5.2 Playtesting discussion

After finishing the integration of all the game components and having a working prototype we had a small and informal playtest with all group members. Some members of other groups were also invited. This Section describes what we learned from playtesting, what we changed due to feedback and what we would have liked to change given more time.

5.2.1 What did we learn?

We learned that our choice of song was too fast and difficult for the average playtester. Our song of choice would probably end up at the higher end of the difficulty scale in a finished game. In hindsight, it could have been beneficial to have a simpler track for playtesting as it could have given us more information about the interactions within the game. Although most testers struggled with the fast rhythm, those with previous musical experience fared much better. After a few attempts players with musical experience generally performed at a high level in terms of score acquisition.

Several testers mentioned that it was hard for them to visually understand the conducting pattern they were supposed to follow. The test build only indicated the pattern by having the four tracking spheres light up with a green colour per beat. Testers mentioned that they would like more visual feedback on their actions, reporting difficulty in understanding whether or not their actions were correct. Some confusion was also connected to when the song started, which hints at the requirement for a clearer indication of this. Unintentionally the game starts tracking the score before the song has started.

Another thing we noted from playtesting was player engagement throughout the gameplay. Testers were generally at their peak of engagement when needing to handle both cues and conducting at the same time as it really reinforces the rhythm of the song with both hands. When asked what parts of gameplay they enjoyed the most, the testers also confirmed this as they felt just conducting with one hand could become boring for longer tracks.

We discovered during playtesting that our world space placement of tracking spheres was too high for shorter testers. They struggled with hitting the tracking spheres in a natural and comfortable manner.

Most testers also seemed a bit too focused on the tracking spheres, sometimes distracting them from noticing cues, or looking at the environment. We believe this is due to the testers being too worried about trying to hit the visual spheres. While in reality hitting this would not have been necessary for the gameplay as the game is quite forgiving in terms of hit detection. The rationale behind the hit detection being forgiving was to make it easier for players to look around the scene.

5.2.2 What did we change based on feedback?

One of the first changes we implemented after the initial playtesting was adding a trail going between the tracking spheres to visually demonstrate the conducting pattern to the player. The animation for the trail is tied to the beat of the music so it should also help the player stay in rhythm. We also added more visual feedback to scoring on both the table and the size/visibility of the red/green particles around tracking spheres, used for demonstrating whether the player is on or off beat. Whenever the player gains score

a ”+1” or ”+10” will pop up in front of the score on the table. A ”+1” is given for each beat the player manages to conduct correctly, while a ”+10” is given when the player cues an instrument.

When it comes to the player height and tracking sphere position disparity we added a simple means of calibrating the y position of the spheres. The player is able to change the height of the spheres before starting the song which should let them place them at a more comfortable height.

5.2.3 What would we like to change based on feedback?

One thing in particular that we would have liked to change given more time is to make it more obvious when the song starts. While starting the countdown after calibration is fine for now, it would be nice to better indicate with the tracking spheres that a song has started, replacing the current system. The score should not be tracked during the countdown period, but showing beat countdowns towards the start of the song through additional means could potentially help the player realize when the song is supposed to start.

5.3 Game design decisions

5.3.1 The choice of method for tracking player conducting movement

Our solution to motion tracking is quite simple compared to proper gesture tracking, which could have been acquired from third-party libraries or assets. We went with this solution because it was the quickest and easiest way we could track motion for the prototype. We looked at other gesture tracking solutions, but none of these allowed the developer to see where in a gesture the player was, which is essential for checking where the player’s hand is in relation to the rhythm.

5.3.2 Whether to place motion tracking components in world space or camera space

We discussed within the group whether to place the tracking spheres in world space or camera space. Initially, the spheres were in camera space and moved relative to where the player was looking. Having the spheres locked to camera space circumvented the issue of calibrating the player’s height. However, it became a problem when cueing was added, as the players struggled to move their hands according to the spheres when looking for cues. To fix this we moved the spheres into the world space and added a simple calibration mechanism.

5.3.3 Critical Path Coding

Following a critical path coding [4] approach helped us focus on writing the minimum amount of code needed to solve our specific problems. We believe this approach enhanced the productivity of the programmers in the group and helped us avoid the trap of trying to create too many generalized or overly complicated solutions. However, we might have done a bit too many ”hacks” to get the game to function, especially towards the end of the development cycle. While we might be able to reuse some parts of the code if we were to create a full game from this concept, others are too coupled with the current design or contain too much technical depth.

5.3.4 Dropping Process

As previously mentioned we originally planned to have a greater focus on documentation and work process, however, we abandoned this early in the project because we felt it added unnecessary overhead. None of us had ever programmed a rhythm game before and was therefore busy enough with exploring the problem space. Our inexperience also made planning and engineering solutions ahead of time more difficult. Because of these factors, it felt contrived and unnecessarily time-consuming to try and follow best practice guidelines. While this led to less documentation of our workflow and process we believe it was beneficial for our project, allowing us greater flexibility and the option to dedicate more time to implement the product.

6 Conclusion

In conclusion, we are very happy with the project which has led to the Conductor Hero game prototype. We believe the game can serve its intended purpose of being an introduction to the world of conducting ensemble orchestras, but that further work will be required for a more complete and immersive experience.

The interdisciplinary nature of the course has also taught us many valuable lessons. We have learned the importance of working on a tight schedule towards a prototype using a critical path coding method to ensure that we finish on time. This experience will help us be more pragmatic when we meet similar projects in the future. We have gained valuable insights into some of the challenges and gains of working in an interdisciplinary project in terms of communication, methodology, and differences in skill. With this knowledge it should be easier for us to identify and solve areas of friction in future projects, allowing for more effective teamwork.

For some of us, this project has been a view into the area of leadership and management, sparking an interest in possibilities of pursuing this field as a career. For others, it has been an introduction to the area of game programming, with the unique and varied challenges present within the field. The more experienced members have been presented with the issue of asset and resource pipeline management, ensuring the communication between different applications to realize the groups' artistic vision. As a group, we have experienced some of the challenges of working in the young field of designing interfaces for VR applications, and how many traditional approaches to design falls short in the VR space. After a successful prototype, we all have an increased interest and confidence in the field of VR technology, which will be invaluable in future VR endeavours.

7 Future Work

While we managed to get further than we had expected, there is still a lot of work left unfinished. Time constraints was a challenge throughout the project, and we believe that the finished product could be significantly improved with further testing. With more time available, user testing and user experience (UX) design would have been a priority for the group. User feedback can be an invaluable resource during the development of a product, and we believe that it would be beneficial to make this a priority in the possible further development of Conductor Hero.

Turning the prototype into a fully fledged game would require a complete rewrite of most of the game systems, to create more generalized and reusable solutions. However, this rewrite would be necessary to achieve other desirable features, like more levels, songs, and the potential for user-generated content. A fully fledged game would also require us to design a complete user interface for the user to interact with, which would present us with more challenges in the VR space.

During the initial design phase, we came up with several ideas for extending the mechanics of the game, for example, a level where you don't just conduct the instruments but also the virtual environment, creating a visual experience that reinforced the feeling of the music. To keep the scope and mechanics at a manageable level these ideas were not pursued, however, we would like to explore these ideas in an extended project.

Bibliography

[1] Experts in teamwork. ntnu.edu. (accessed: 11.04.2018). URL: https://www.ntnu.edu/eit.

[2] What is experts in teamwork - questions and answers. ntnu.edu. (accessed: 11.04.2018). URL: https://www.ntnu.edu/web/eit/what-is-eit.

[3] Gray, D., B. S. M. J. 2010. *Gamestorming: A playbook for innovators, rulebreakers, and changemakers.* O'Reilly Media, Inc.

[4] Blackshaw, C. 2017. Critical path coding. Konsoll 2017. (accessed: 04.05.2018). URL: https://www.youtube.com/watch?v=3XKUp8z7hjQ.

[5] Unity api - audiosource.playscheduled. docs.unity3d.com. (accessed: 03.05.2018). URL: https://docs.unity3d.com/ScriptReference/AudioSource.PlayScheduled.html.

[6] Nystrom, R. 2014. *Game Programming Patterns*. Genever Benning.

A Appendix

A.1 Game Design Document

A.1.1 What is "Conductor Hero"?

Conductor Hero is a VR conducting game that allows the player to take the role of a conductor as the name implies and conduct a band/orchestra of instruments to given music tracks. We have taken inspiration from games like Guitar Hero™ where the focus of the game is not necessarily to be authentic to real life experience, but rather provide an approximation and highlight the experience of conducting instead.

A.1.2 Planned Game Functionality

Our initial plan for the game includes several different functionalities. First off, we plan to have two different types of game modes. Freestyle mode is where the player can freely cue instruments in and out as they wish where the cueing itself is not scored. Performance mode on the other hand requires the player to cue in instruments at specific points in the music track. In Performance mode we would also like to score the cueing according to how close to correctness it is in relation to timing.

The band/orchestra will include several instruments. This should include violins, violas, contrabass, harpsichord, glockenspiel and oboes.

In terms of conducting the player can use their dominant hand to conduct the music with a time signature of 4/4 for the minimum viable product. The other hand can then be used to cue in groups of instruments. The player is scored depending on how close their conducting movements are to being correct and how rhythmically correct it is. We want to be very lenient on the movement itself as it is hard to expect perfect conducting motions every time and rather focus on scoring the rhythmic aspect. At the same time it should not be possible to cheat the conducting pattern. The nondominant hand could also potentially be used to control the intensity of instruments. By doing so we can for example have three different instrument intensities and allow the player to switch between them with the nondominant hand.

Within the game we also want to have a Conductor Table/Dashboard that contains all relevant UI elements. The list of what we want to display on this screen is as follows: Score, communication on where to cue instruments in, duration left on song and current song volume.

A.1.3 Project Scope

For this project, we plan to scope for a minimum viable product containing only one level as we have a limited amount of development time. If the MVP milestone is met early then we can also potentially add the ability to share the player's performance from freestyle mode as a performance mode challenge.