Bachelor End Project - Pingball - Final Report

Berge, Drew Bettencourt, Danilo Lageweg, Stanley
Overman, Willie Zaidi, Amir

June 2020

Coach
Dr. ir. Rafael Bidarra

Client
Lars Lommers MuZIEum

Bachelor Project Coordinator
Otto Visser

Foreword

This project gave us a new perspective on the importance of sound design. The game would not have been as playable as it is without the help and feedback from a couple of people. We would like to especially thank our coach dr. ir. Rafael Bidarra and the client Lars Lommers for all their help during the project. The feedback from them on the game design and the sound design proved to be very helpful for the final product. We would also like to thank TU Delft and the MuZIEum for making this project possible.

Table of Contents

Chapter 1

Summary

The team was tasked with making an audio-only pinball game. The main requirement of this game is that it is fully playable with only audio. Making an audio-only game presents many issues not typical to game development. All design decisions had to be made explicitly with the thought that nothing would be visually apparent. This is reflected in our final design, as all features serve in some way to further the main goal of making a game that is played without any visuals. While the design process was unique, the development process itself is largely the same as for a normal software development project. However, there were multiple important differences. The relative unexplored nature of the project lead to the development process requiring much more prototyping and testing of ideas to find out what worked. The implementation itself was not particularly unique. While being an audio-only game, at its core it was still a game, so once the technologies to use were decided, there was nothing unusual about the implementation of the game. The final user evaluation was enlightening. While users in general enjoyed the game, it was clear that there were both things that were immediately understandable, as well as things that were not obvious to new players. This project gives more insight about the capabilities and limitations of audio-only content. It also shows the importance of user testing in this type of project, as in hindsight more testing with end-users would likely have been beneficial. At the end of a project, the team believes that a valuable product has been created that brings a new experience to both those with visual impairments, as well as those who wish to experience an audio-only game as such.

Chapter 2

Introduction

Over the last number of weeks, the team has worked on an audio-only version of the classic game pinball. We were asked by the muZIEum, a museum about vision and vision impairments, to make this audio-only pinball game so they can showcase it in their museum in Nijmegen. Almost all video games that are available use visuals as a central way of conveying information. This, however, leaves people with significantly impaired vision little choice for games to play, and those with complete loss of sight have even less choice. While many games try to accommodate people with various disabilities, the central role that visuals play in games means that for most games, playing with exclusively sound is not feasible. There are games that are playable with only their audio, but there are far fewer games that have audio-only design as a central principle. This is the niche that Pingball hopes to take a role in.

Designing a game to be played without any visuals is a task with relatively unique troubles, as the team has come to learn over the course of its development. Both finding areas to take advantage of the audio space as well as the inherent limitations of not having visuals come with their own problems to overcome. During development of the game, many ideas were thought up, prototyped and scrapped. As audio-only games are sparse, there was not much previous work to look to for inspiration or ideas.

In this report the problem that was presented and the solution that was devised are discussed. Both overall design as well as specific implementations are laid out in detail. The process, including our development methodology and a number of scrapped ideas are discussed as well. The user testing that was performed and the results thereof is also discussed later on in the report. The hope is that by reading this report, the reader has a good idea of what has been created, and why the decisions that were made, were made.

Chapter 3

Research

3.1 Problem definition

Pinball has many variants and has gone through large changes since its conception. The main problem is adopting one of these pinball variants to the audio game format. To make the game work as an audio game, some changes will always have to be made to the gameplay. However, the goal is to adapt Pinball, and not to create a game vaguely resembling it. In the many existing variants of Pinball there could be one that works well as an audio game without major changes. If there is one, that variant should be the base of the audio game. If not, a new variant has to be developed. Testing would then be required to determine if players of the game still consider it a Pinball game.

3.2 Problem analysis

There are various elements to Pinball that make adapting it into an audio game a difficult task.

First, giving constant feedback, such as with a score counter on the screen, is only possible with a constant sound. However, that might quickly become distracting. It is then important to decide what aspects of the game should provide feedback and when, as not all information needs to be constantly available to the player. Some information which the player is used to being able to view at any point, such as the score, requires more attention in an audio-only game than when there are visuals, as the ability for someone to change where they are looking does not trivially analogue to sound.

Second, object localization with only audio is hard, and becomes even harder when too many objects are added to the surroundings. Tracking the precise location of a moving object can probably only be done by those who have practiced that skill their entire life. Because of this, methods of tracking the ball and the game state in general should not rely entirely on spatialization. It

becomes important to consider what information can be sonified, in ways that the player can learn without hindering them in other ways.

Third, the difficulty of the game has to be adjusted to account for the loss of visuals. Since localization with audio is much harder than with visuals, the game should be made easier than regular pinball. If the difficulty is too high, players might get frustrated by it and quickly stop playing the game. There are a number of ways to adjust the difficulty of pinball, such as the speed of the ball and the layout of the play field, and attention needs to be put to adjust these to a desirable level.

3.3 Related work

Blindfold Pinball [1] is a game on the iOS App Store™ [2] that claims it can be played blindfolded. The game is designed to be able to be played using only audio or visuals. When using visuals it is really easy to track the ball on the playing field, but when playing blindfolded the game fails in this aspect. The speed of the ball is too high and the amount of bumpers on the map produces too many sounds that overpower the sound of the ball. You can also not really tell when the ball is close to the flippers. This is due to the speed of the ball and because the sound of the ball does not get amplified enough when it's approaching the drain. A rectangular playing field also feels limiting for an audio-based game, as only a small range of spatial directions is used. It's difficult to differentiate a sound which is slightly to your left, from a sound that is straight in front of you.

Another related initiative to make a set of audio games is the Audio Game Hub [3], which is an assortment of audio games created to be playable both by sighted and non-sighted people alike. Here, both spatialization and sonification techniques are used to convey the game state. For example, in one of the games, Hunt, a target moves around the screen that the user must shoot. Here, the position of the target is conveyed in a few ways: as the user aims closer to the target, a targeting sound plays with increasing frequency; a stereo panorama is used to locate the object left and right; and the pitch of the targeting sound varies with the height of the target on-screen. While the target being visible makes this game trivial for a sighted person, it is entirely playable and more engaging when the player relies entirely on the audio.

Outside of the individual games in the hub, the creators also gave much attention to the menus. All important information is conveyed through voiceover, giving detailed tutorials of the games and their settings. The menu controls are also usable without seeing the screen. Changing selections is done through swiping the screen and other buttons, such as the back button or information button, are placed relative to the corners of the screen, making them easy to locate.

Even compared to normal pinball, the games are relatively simple. They generally feature few individual game elements, letting the player use multiple

audio cues for similar information, aiding in making the state of the game be extremely clear. The techniques used however are still applicable to Pingball.

Legend of Iris [4] and Loud and Clear [5] are two audio games developed at Delft, University of Technology. These games were made to gamify a regular task, while Pinball is a game with no additional purpose. However, the research done by their development studios is still useful for the development of Pingball by looking at their findings. For example, during their testing the Loud and Clear team discovered that "For many players, it was rather difficult to estimate how far a sounding item was. Indeed, detecting extreme relative distances is quite easy, e.g., when an item is really far or really close to the player; but distinguishing between two intermediate distances turns out to be rather difficult." [5, p. 9] Since determining the distance of the ball to the drain is an important aspect of pinball, this is important information to keep in mind.

3.4 Requirements

3.4.1 Requirements analysis

From the project description that can be found in Appendix A, multiple sub-requirements can be extracted.

> *There are very few digital games suitable for blind or partially sighted people, let alone designed on purpose for them.*

The game should be designed from the ground up for blind and partially-sighted people, and the requirements are made in the context of designing a game for them.

> *Design and implement a digital game that conveys the core game mechanics of the classic pinball game.*

The game should include the core mechanics of the classic pinball game. This refers to a moving ball falling down to a drain due to gravity, two player-controlled flippers to keep the ball out of the drain, and obstacles in the map that give the player points when the ball hits those obstacles.

> *Purely based on spatial audio features (and possibly some haptics), without any visuals.*

It should be playable without any visuals, and by default not have any visuals for that reason.

> *The game should require no complicated setup.*

When the player starts the game, he should be guided towards the basic gameplay loop within seconds. Any required setup could be hidden in a tutorial, but this should not become complicated. If play-testers have an issue with any aspect of this tutorial it must be removed.

Be playable by both blind and sighted people on a mobile phone.
Target platform: Android smartphones

This game should be compiled for Android smartphones, so there are limitations to which engines and frameworks can be used, as not all support the Android platform. User interface elements introduced in the game should be compatible with and navigable by Android's accessibility services.

Which would then drive the headphones and serve as gamepad as well.

No external controller should be required, and the Android phone must be used as the input device. The sound mixing should be optimized for headphones, as that is the set-up the client wishes to use.

This game is meant to be tested and actually deployed at the Muzieum (https://muzieum.nl/), in Nijmegen, The Netherlands.

As the game will be placed in a noisy environment, the sound mixing should not rely on any sounds with low loudness.

3.4.2 MoSCoW

Using the requirements analysis, the following MoSCoW requirements list was created. Must have items are requirements that cannot be missed in the final product, and have to be prioritized. Should have items are similar, but might be dismissed as infeasible. Could have items are interesting options that must be looked in to but have a low chance of appearing in the final product. Won't have items are ideas that were discussed previously and already determined to be excluded from the final product, but could be reconsidered at a later point in the project, when the other requirements have been met.

Must Have

- (Semi)circular playfield
- User input
- Flippers
- Sound when ball is rolling
- Spatialized audio
- Score points when hitting bumpers

Should Have

- Identifiable but not obnoxious sound when ball approaches flippers
- Sound when ball touches flippers
- High Score system

Could Have

- Ramps
- Slow down field near the flippers
- Force field type of flipper, throws a force towards a certain direction and pushes the ball back like that.

Won't Have

- Head tracking
- Custom HRTF audio engine (Unity already has this)

3.4.3 Requirements elaboration

The previous subheader describes the requirements of the game in the MoSCoW system.

Some important parameters that need extra attention are the speed of the ball, the amount of sounds and map design. The speed of the ball is an important parameter that will need to be extensively tested and tweaked during the development phase. The ball cannot be too slow or it will make the game boring and not too fast or the player will have no idea what is happening.

The sound of the ball should be the most audible one, other components should have (slightly) lower volume. This will ensure that the player does not lose track of the ball.

3.4.4 Success criteria

The project is considered successful if the following requirements hold. First, the must-haves mentioned in 5.1 are implemented. Second, players are able to effectively track the ball only using sound. Third, players find enjoyment in playing this game, as determined by a questionnaire.

Chapter 4

Design

Many concepts that were created before and during the development process have been scrapped or reworked for the final product design. This section describes the design of the final product and justifications for decisions made for this design.

4.1 Setup

Games commonly employ a menu screen to give the player the possibility to change the settings of the game or launch alternative modes. However, since Pingball does not have any alternative modes, this is not worth the added complexity for the visually impaired players of the game. Therefore the decision was made to not include any menu. This reduced the design space as it was now impossible to add any settings to the game as well, but it was deemed a necessary trade-off for accessibility.

When the player opens the game by clicking the application shortcut, he or she is asked to press the left side of the screen for an introduction, or the right side to skip the tutorial and start. The introduction explains the game to the player. It is given in the form of voicelines that explain the controls of the game, and the goal of the game. The explanation that the player receives goes as follows:

> *Welcome to Pingball. In this game you have two flippers that are activated by pressing the left and right side of the screen. Listen for warnings as the ball approaches the drain. Use the flippers to prevent the ball from going into the drain. As the ball rolls over a flipper, a tone will play. This tells you where the ball is on the flipper and in which direction it will be shot. Score points by hitting objects. There are three sections. Clearing them in the order indicated while playing, opens the minigames. Good luck!*

When the player is done listening to the explanation he or she can choose to listen to the tutorial again or start the game.

4.2 Gameplay

The player hears the audio as if he or she is positioned at the pinball drain, the hole where balls disappear, and has to keep a ball out of the hole. When objects on the playing field are hit by the ball, the player earns points. The player gets three balls, and each ball is similar to a life. The score of the three playthroughs from the three balls is added up for one final score. As all feedback comes purely from spatialized audio, everything is kept as simple and understandable as possible.

A vital part of game design is keeping the player interested and engaged in the game. This is relatively easy to achieve in a video game format, but requires a bit more reflection when only using audio, since standard game mechanics often don't apply here. Pinball as a game is generally very visual, and a lot of game-play engagement comes from this aspect, therefore it requires a focus on alternative methods to keep players engaged.

4.2.1 Scores

An important part of pinball is keeping track of the player score and communicating this with the player. On top of keeping the player engaged during game-play, it's also important to give the player an incentive to play the game again. This is done by keeping track of player high-scores during different games. The score system consists of the following elements:

1. Current game score being continuously updated in the back-end, when bumpers are hit by the ball
2. Score announcements for milestones on big numbers terminating with multiple zeroes
3. Final score announcement when the ball rolls into the drain
4. Persistently tracking and informing the player about the high score
5. Saving the high score if the final score has beaten the high score

On top of this, when the currently obtained score is higher than the highest stored score, the game announces a special 'New High Score!' achievement.

4.3 Map

The map is a semi-circular field of about 150 degrees. The drain is placed in the center of the semi-circle and the flippers are placed close to the drain. The playing field has an incline to give the ball acceleration towards the drain due to gravity. It consists of five separate sections, three of which are accessible when the game is started. (see figure 4.1)

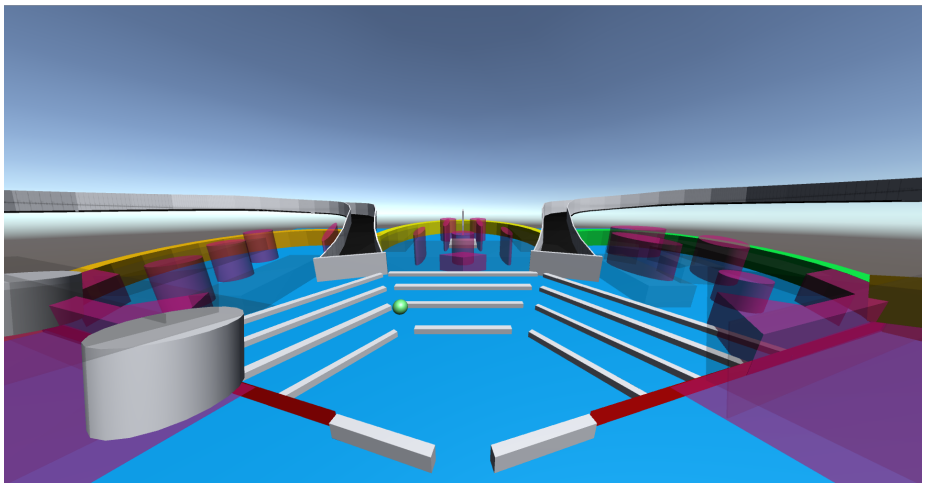


Figure 4.1: Perspective view of the map design.

4.3.1 Player

In this game, the player is positioned slightly in front of the drain, to make him or her feel as immersed in the game as possible. A side effect of the rotation of the field is that the player, with their location at the edge of the map in front of the drain, can hear the height of the ball increasing when it rolls far away. There is no option to rotate or move in the game. This ensures that the player never becomes confused about his or her own orientation and always knows that sounds positioned in exactly the middle of the sound field are also in the middle of the map. The volume roll-off curve of sounds is adjusted so the strongest volume reduction caused by the distance from the player to the edge of the map does not make sounds inaudible, yet the distance to sounds can be guessed from volume differences.

4.3.2 Sections

The game has three separate sections, each of which has its own instrument sounds as a theme. (see figure 4.2)

The first section is themed to sound like a guitar. It was designed by taking inspiration from an existing pinball machine (see figure 4.3), which used this bumper placement on its wall. When the ball rolls into the section, it is usually launched towards the ramp, where a flat bumper shoots it back into the middle of section. There it usually hits one of the circular bumpers placed in a vertical line, bouncing it back to the flippers or further into the section to repeat the cycle.

The second section is themed to sound like a drum kit. This section is also inspired from an existing pinball machine, although it is a different design. The

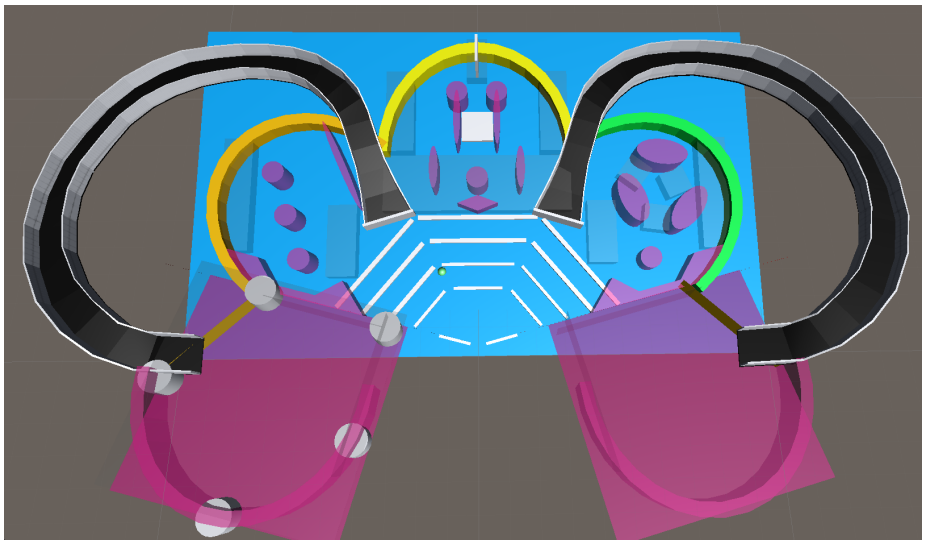


Figure 4.2: Map design with the guitar section left, drum kit section in the middle, and piano section right. The pink squares are mini-games.

goal of this section is to trap the ball between two large horizontal bumpers, making it bounce from side to side rapidly as it builds speed. When it enters this bouncy section, the player also receives a large amount of points, as this is hard to aim for. The outside of the section has bumpers located in such a way that the ball usually hits around three of them before bouncing back to the flippers.

The third section is themed to sound like a piano. This section features three bumpers in a triangular configuration, which is common in many pinball machines. The bumpers are slightly oval shaped, to make it more likely for a satisfying series of hits between them to occur. The bottom wall features the same bumper setup as the first section, to nicely fill out the space.

In Pingball, the player can play the game like standard pinball by following the basic rules. To add more diversity to the gameplay however, Pingball offers an extra challenge for more points and alternative gameplay. Since the map is split into 3 sections each with corresponding instrument sounds, the game plays special, instrument-specific sounds in a random order to guide the player into shooting the ball into the right section. For example, the game will play a piano tune to signify the player to shoot the ball into the piano section first. When the player has reached a certain amount of points in the piano section, the game will play the guitar tune to tell the player the piano section has been cleared and that the guitar section is the next target. When all sections have been cleared, the ramps towards the mini games are opened which is announced with a voiceline.

Figure 4.3: Pinball machine used for inspiration.

4.3.3 Ramps

Once all three sections have been visited in the correct order, two ramps open. When the ball enters one of these ramps, it takes the ball to one of the two mini-game areas on the map. These areas are located slightly higher and directly to the left and right of the player. The location of these areas makes the movement of the ball, as it rolls through the ramp, sound interesting, as it moves around the player until it stops very close to the ear of the player.

4.4 Components

The game uses most of the core components of classic pinball such as balls, flippers, bumpers, ramps, rollovers and a drain. But some special components are added as well, namely force areas and sound plates.

4.4.1 Ball

The ball is the most important component of the game. The player can control the ball with the flippers, and send it to any part of the map. It is an object that is perfectly round in all dimensions, rolling throughout the map. The physics of this ball are tuned in such a way to make it feel like a physical ball rolling inside of an arcade pinball machine. However, the speed and friction are slightly modified to make it easier to control the ball, as the difficulty to do so is already much higher because of the lack of visuals. The reason the ball is the most important component is that colliding with the ball is the only form of interaction most objects on the map have. If the feeling of the ball is not perfect, the entire game will feel worse as a result. The bounciness, mass, friction and gravity are therefore fine-tuned extensively with weeks of playtesting feedback. To get the best possible feeling of friction and bounciness, different values per material have been used. Walls have no friction and almost no bounciness, while the floor has no bounciness and almost no friction. The other components all have their own values as well. The mass of the ball is very low, to reduce the impact the ball has on the flipper when it hits them, and make the flippers more consistent when shooting the ball. The gravity is set to $9.81 m/s^2$, with a playing field angle of 7 degrees. This translates to an acceleration towards the player of $9.81 m/s^2 * \sin(7^\circ) = 1.2 m/s^2$.

The ball has multiple interaction sounds with objects that all need to be audible when playing. First, there is the rolling sound that increases in volume depending on the speed of the ball. It changes its pitch following an exaggerated Doppler effect model: when the ball rolls away from the player the pitch drops, and when the ball approaches the player the pitch increases. Players might know this effect from cars passing them on the road. The ball also plays a sound that repeats based on its distance from the flippers. This aids the player by making it more explicit when the ball is very close, instead of relying exclusively on spatialization. This sound plays very slowly when the ball is far away from the flippers, which gives it very little presence when at a distance. As the ball then

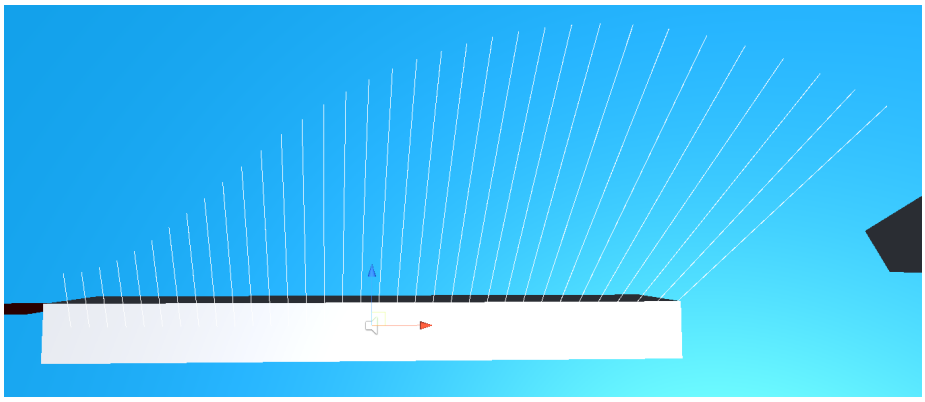


Figure 4.4: Static flipper with manually tweaked force field.

gets closer and closer, the sound then repeats faster and faster, enhancing the sense of distance regardless of factors such as speed or direction.

4.4.2 Flipper

The flippers in Pingball are similar to flippers from classic pinball games, but do not physically move in the physics simulation. This gives more fine-tuned control over forces applied by these flippers on the ball. When the ball is shot from the base of a flipper, it is slowly shot towards the other flipper so the player can shoot into the opposite direction. When the ball is shot from the middle, it is shot upwards into the second section. When the ball is shot from the tip of a flipper, it is shot into one of the side sections, depending on which side that flipper was on. (see figure 4.4)

There are a few sounds associated with the flippers. Firstly, when a flipper is activated or deactivated, it plays a corresponding sound. This provides the player with feedback as to when and which flipper has been pressed, which is particularly useful for touch screen where there is otherwise no direct feedback. The flippers also have sounds when coming into contact with the ball. There is a unique impact sound when the ball hits a flipper at a high enough speed, to differentiate between the relatively low risk state of hitting a wall, and the relatively high risk state when the ball impacts near the flippers (and thus near the drain). Aside from that, a tone also plays as long as the ball is in contact with the flipper. This sound plays a few important roles. Firstly, it serves as a clear indication when the player can perform a consequential action in the game, as the ball being in contact with a flipper means that activating that flipper will launch the ball. Secondly, the tone itself changes pitch based on where on the flipper it is contacting. This gives an indication of how the ball will be hit, if the player activates the flipper. As normal flippers rotate around a hinge, hitting the ball near the tip of the flipper will send it in a different angle and speed

than near the hinge. This makes it so that, even though the player cannot see the field, they still have the same sense of control over the ball as they do when they can see the ball, as in normal pinball.

4.4.3 Bumper

Bumpers are objects that launch the ball away on contact. They do so regardless of how hard the ball hits them, with roughly the same force. Their main purpose is to make the game more dynamic. The ball loses momentum as it rolls due to forces like friction and air resistance. This is counteracted by active components like bumpers, which impart additional momentum into the ball, keeping it moving and in play. Bumpers also give score and make a flashy sound when hit. Due to this, placing multiple bumpers in close proximity such that they cause bounces between each other gives the player a way to get many points quickly, and makes it exciting when this happens.

There are a few important considerations when making and placing bumpers. One such consideration is the position. A badly placed bumper can lead to situations where the ball frequently gets knocked directly back to the flippers, which can lead to repetitive situations of hitting the ball, and having it come straight back repeatedly. Another possibility of a carelessly placed bumper is to enter a stable loop of bounces between one or more objects, outside of the player's control. This then causes a situation where the player slowly acquires more and more points, but cannot interact with the ball anymore. This then soft locks the game, preventing it from being played further.

Another consideration when placing bumpers is the shape of the bumper. The angle the ball gets launched in depends on the shape of the bumper. This can be used to direct the ball in certain directions, or to add larger variations depending on where you hit the bumper. In either case, it is important for the level designer to consider how these interactions will happen, to avoid frustrating or otherwise unenjoyable situations.

4.4.4 Ramp

Ramps are tubes on the field that absorb the ball and move it to another part of the map. They serve as the way to access otherwise inaccessible parts of the map. Because of this they are elevated above the map so that the ball can move above certain sections of the map. This also prevents having the ball go too far away from the player, making it harder for the player to hear and track the ball. Due to the steep slope of the ramp, the ball might require a boost to reach the top part of the ramp. Force areas are used to provide this boost to the ball when it enters the ramp. The position of the ramp is important. The ramp needs to be positioned somewhere in front of the flippers so that the player can hit the ball in the direction of the ramp.

4.4.5 Rollover

Rollovers are targets which award points when the ball rolls over them. These can be used to award players for getting the ball in a hard to reach area of the map, and as such a rollover is used in a hard to reach area in the middle section of the map. They also play a special sound to let players know that they hit the rollover.

4.4.6 Force area

There are areas throughout the map where the ball gets a constant force acceleration to improve the feeling of the game. For example, the wall on the left side of the left flipper, and the one on the right side of the right flipper have a force pointing inwards, towards the drain. This prevents situations where the ball is rolling very slowly against those walls and the player has nothing to interact with for multiple seconds. There are also areas where the ball is held inside so there is a higher chance of it bouncing against more components. Some force areas only apply when the ball is already rolling in the direction of the force to give the ball a boost, and others always apply to ensure the ball never rolls into the opposite direction. The force areas break the laws of conventional physics, but since the player cannot visually see the ball being pushed, and pinpointing the exact location through audio is virtually impossible, this is not a major problem.

4.4.7 Sound plate

The ball passes multiple weak force areas that make a sound on collision. These areas are used to give a better indication of what the ball is doing, as just localizing the ball by its moving sound through spatialization is difficult. The areas do not move, so the player knows the location of the ball when it passes through them.

4.4.8 Drain

The drain is not a physical object, but rather an abstract concept of the ball rolling in between the flippers and consequently falling off the playing field. Once it has fallen for over a meter, it triggers a sound effect to let the player know what happened, and then resets back to the starting position. The sound effect when the ball rolls into the drain is a sinking sound to indicate the ball is falling. This gives player the feeling of their stomach dropping because of something “bad” happening. The goal of this effect is to have players put more meaning into keeping the ball in the field, as they do not want to have this happen again. Raising the stakes this way improves player immersion and dedication.

4.5 Mini-games

The game includes 2 mini-game sections to make the game more interesting and fun, and to add more replay value. Each mini-game section can be reached through the use of a ramp and not by any other means. Every mini-game starts with a short introduction clip explaining to the player how the mini-game is played. Afterwards the player has a limited amount of time to play the mini-game before the ball is ejected to the main part of the map.

4.5.1 Smash

This mini-game requires the player to tap the screen as fast as possible. The section where the mini-game is played contains multiple bumpers/targets and the user can earn points by hitting these bumpers/targets. The ball receives a force to a random direction every time the user taps the screen. The faster the user taps the screen, the more targets the ball hits in the short time, and the more points the user gets. There are two reasons the force is used instead of simply giving points directly. The first is that the sound effects of the ball rolling around and colliding into bumpers gives an enjoyable experience as the player feels like he or she is influencing the game more than he or she would otherwise. The second reason is that the ball carries momentum, and still receives some points when the player stops tapping. This adds a deceleration to the collection of points that feels more natural than abruptly stopping.

4.5.2 Direction

This mini-game plays a sound to the left or right of the player. The player needs to then touch the side of the screen that corresponds with the direction of the sound. The player will hear different sounds depending on the correctness of the player's input. The player receives points for each correct touch he or she makes.

4.6 Tried concepts

This section describes concepts that were experimented with but ultimately removed from the game, and gives reasons for the removal of each of them.

4.6.1 Resonance audio

Resonance Audio is a spatialization plugin for Unity made by Google in 2018. It is similar to Oculus's Audio SDK, but also supports compilation for iOS devices. However, there were two reasons the spatializer was switched to Oculus during development. The first reason was that Resonance created a bug where the position of the listener was only updated when an Audio Source was enabled and disabled, making the initial testing phases tricky as it was never clear if the position was properly updated or not while playing. The second and more

important reason was that the quality of HRTF convolution seemed to be subpar on Android builds compared to Windows builds, while Oculus's builds were of similar quality. Therefore the decision was made to replace the Resonance audio plugin with Oculus's Audio SDK.

4.6.2 Maps

During the early stages of the development phase the team experimented with different types of maps. These included the classic pinball (see figure 4.5) map and a cone shaped map (see figure 4.6). The main reason the classic pinball map is not used, is because of the angle of the map. The angle of the map is too small to take full advantage of spatialization. The player isn't able to tell left from right easily if the ball was constantly in front of the player. Another reason is that the classic map is also longer and distance is not easy to determine with HRTF [6].

The cone map is shaped like a cone where the convergence point is at the lowest point of the map and the drain is positioned at center of the convergence point. The problem with this map is that the ball tends to roll towards the middle of the drain and this makes it impossible to push back using the flippers when there is a gap between the flippers.

4.6.3 Force Flipper

The force flipper (see figure 4.7) was one of the first ideas that was planned to be tested during the development phase. It was supposed to act like a force push that pushed the ball away and could be pointed to any direction around the center of the drain. The force flipper could be imagined like a plane that moves around the drain within a given distance (like a circle). When the user touches on the screen the (x, y) coordinate of that touch will then convert into a direction that the flipper should move in. The plane flipper will then move away from the the drain in the generated direction and when colliding with the ball add a force to the ball in the generated direction. There were 2 configurations developed for the force flipper that added different constraints for the direction generation part of the flipper: sectioned and half circular direction.

The half circular direction generator takes the x coordinate of the screen touch and converts it into a direction from 0 to 180 degrees around the drain position. The sectioned direction generator also adds a half circular constraint to the direction but also breaks that half circle into parts, for example a 3 section generator would split the half circle into 3 parts so touching the screen in the middle area would create a direction pointing north, the left area of the screen would make a direction pointing west and right area would point east.

The reason this flipper design is not in the final product is because it was too complex to figure out without visuals. It would need a big introduction to explain and would not fit our idea of a quick and easy casual game. We also found out that determining if a source of a sound is to the left or right is really

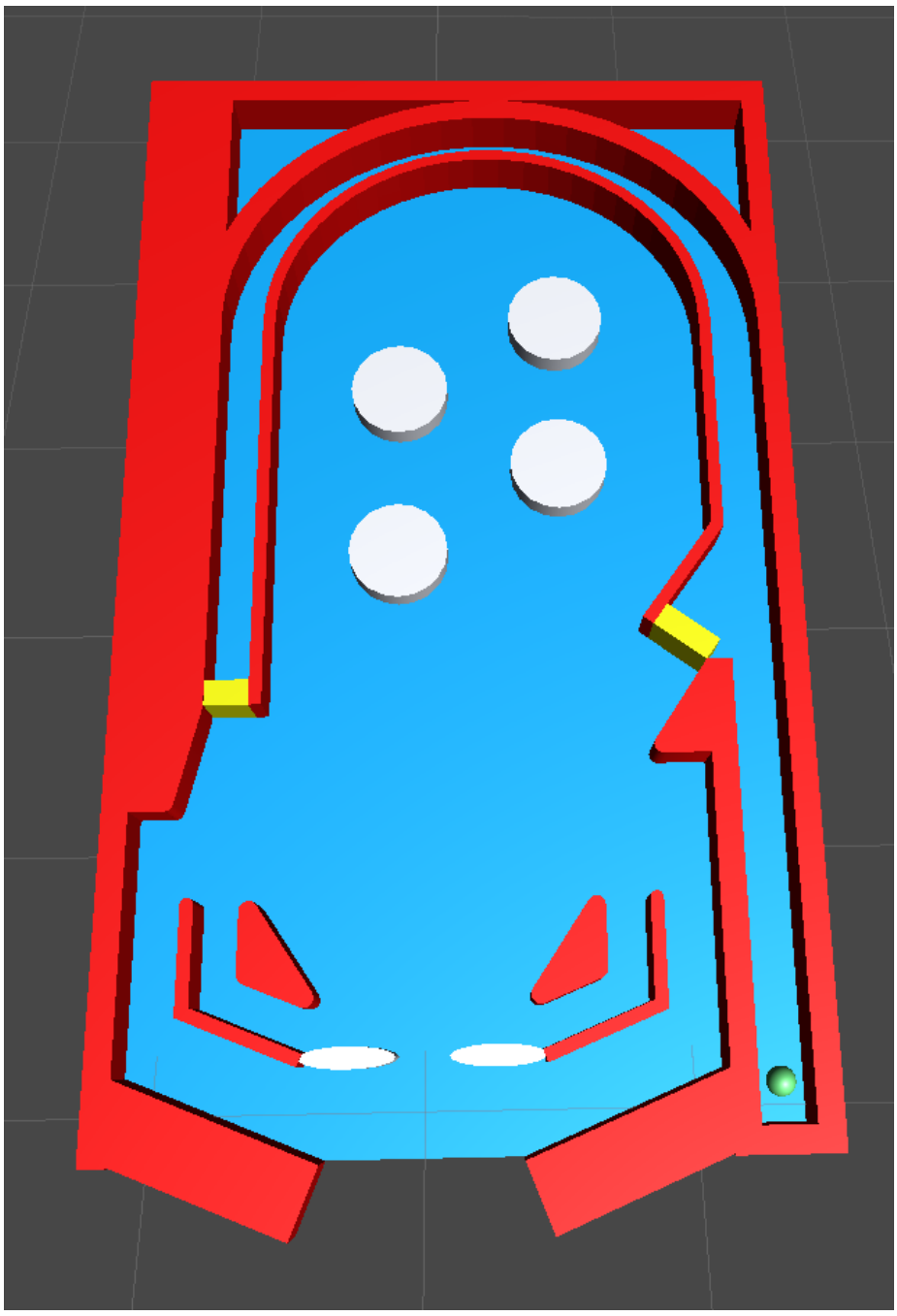


Figure 4.5: Map design using heavy inspiration from classic pinball machines.

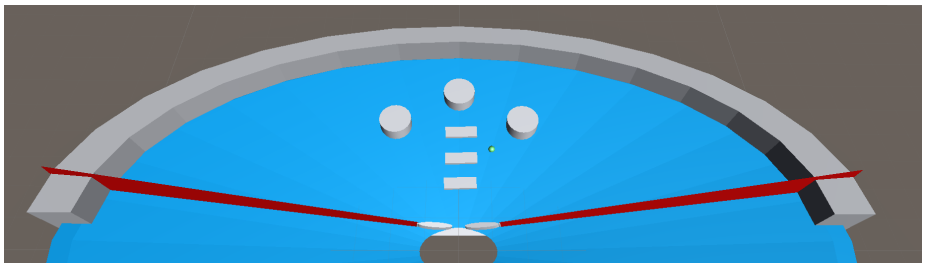


Figure 4.6: Cone map design.

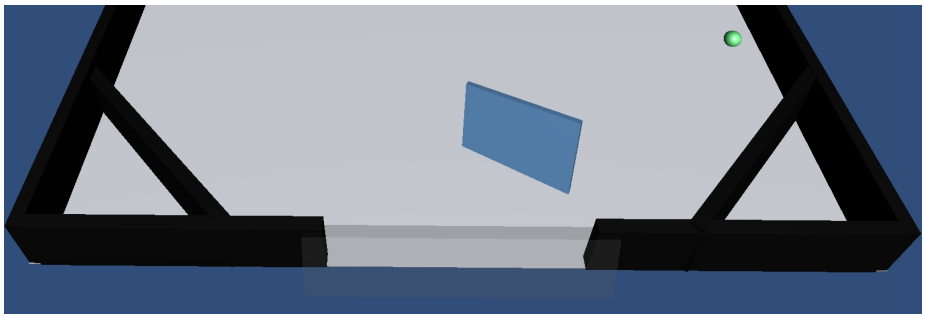


Figure 4.7: Force Flipper Design.

Figure 4.8: Classic Flipper Design.

easy, but gets hard when it's in the front of the listener. Because of this having a the ability to activate the middle section is also not that useful for the player and would just add unnecessary extra input controls.

4.6.4 Classic Flippers

The classic flippers (see figure 4.8) were used for most of the development phase but were later replaced with the static flippers (see 4.4.2). Classic flippers start in a resting positions on the bottom of the field near the drain. One is to the left of the drain and the other is to the right of the drain. They are typically slightly angled towards the drain when in a resting/inactive position. Each flipper can be independently activated by the user when tapping a certain region of the screen. For example touching the left side of the screen will activate the left flipper. When activated a flipper will rotate by a certain amount towards the top of the field and play a sound. Depending on the angle the user can hold touch to keep the flipper in an activated state and stop the ball from moving and rolling into the drain. When the ball is rolling on a flipper it will produce a sound that increases with pitch the closer the ball gets to the tip of the flipper. This helps the user know when the ball is touching the flipper and when the ball is close to the tip.

The pitched sound created when the ball is touching a flipper can be used to aim. For example, when the pitch is high the ball would be pushed more to the left or right (depending on the flipper position side) after activating the flipper. The problem is that the map has an angle that is bigger than what the classic flippers can support. The ball will never get to some sections that are placed to the far left or far right because of this. That is why the classic flippers were replaced with the easier to aim with static flippers. After this change players could easily get the ball to both far left and far right sections of the map using the pitch sound as their guide.

Chapter 5

Process

This section discusses the different methodologies and tools that were used during the development phase.

5.1 Scrum

The scrum development method was used for this project. There were multiple meetings every week between the development team members and a weekly meeting with the coach and the client. Amir acted as the scrum master and lead the flow of most meetings. Feedback and change proposals were given during the weekly meetings with the client and the coach. New issues were made for the sprint backlog at the end of these meetings. Each team member had the ability to choose which issue to work on afterwards. The other meetings during the week acted as our scrum meetings, where progress was checked on important features and/or changes.

5.2 Version and Quality control

The project used git as the method of version control and TU Delft's private Gitlab servers as the remote server. Every new feature or bug fix needed to be done in a new branch. These branches would then need to be merged into a development branch that acted as a secondary master branch. This was done to ensure that the latest stable release could always be found in the master branch and the current experimental changes in the development branch.

Code quality was a really important part of the project. Every merge request required at least 2 different team members that didn't work on the the proposed changes' approval to be accepted into the development branch. This ensured that every new change was scrutinized for every mistake and that the code is clean.

5.3 Continuous Integration

A continuous integration pipeline was used to verify the quality of the code. The pipeline ran on Gitlab CI/CD [7] using Gabriel Le Breton's docker image [8], with his project example [9] as a reference. After every commit to Gitlab, the pipeline would trigger. It would then create an Android build of the game and run the edit- and playmode tests that were written using the Unity Test Framework [10]. The pipeline used to have the ability to create a WebGL build, which was then published and playable on Netlify [11]. The web page was available via a button in the merge request, which was created using the Gitlab Review Apps feature [12]. This feature was eventually disabled, since the Oculus Spatializer plugin [13], which was used, did not work on WebGL.

The functionality of the example project by Gabriel Le Breton was further expanded upon, by determining the code coverage using the Unity Code Coverage package [14]. An HTML report would then be generated, published on Netlify [11, 15] and be available with a button in merge request using Review Apps [12]. The coverage was also visualised in the diff view of merge requests [16] and the coverage percentage was also displayed in merge requests [17]. To create the correct file formats, Report Generator [18] was used.

The test result were displayed using the JUnit test reports feature [19]. To create the correct file format, xsltproc [20] and a 'JUnit to NUnit' xsl file [21] were used.

5.4 Communication

Communication for this project was done exclusively online due to the corona virus pandemic. E-mail was used for contacting the client and the coach for setting up meetings. The meetings were done on a private Discord [22] server, a voice chat platform with file sharing capabilities. Mattermost [23] was occasionally used to send files to the team coach or to contact the course coordinators and supporting staff.

Chapter 6

Implementation

6.1 Engine

For the game, three development environments were considered: Android Studio, the Unreal Engine and the Unity Engine.

Android Studio had the advantage of allowing complete freedom with the project, at the cost of having to write all the code from scratch. Since the game will involve a rolling ball, a physics engine would have to be written, next to an audio engine. This would mean a lot of extra work. The game would also only run on Android. The assignment only specified that an Android version would be the only requirement. However, the client adviser explained in a meeting that most visually impaired people use iPhones, due to their better vision accessibility features [24]. Thus, an iOS build would be desirable.

The Unreal Engine was also considered. It provides a lot of desired features, such as a physics engine. It also is generally more customizable than the Unity Engine. However, it is also generally a lot harder to work with than the Unity Engine. This is in part due to a lack of documentation. Our TU Delft coach recommended to not use the Unreal Engine, as he has seen a lot of projects fail due to the difficulty of using said engine.

The Unity Engine was eventually chosen as the development environment. It was decided that a significant portion of the project would consist of trying out and testing new ideas, as none of the group members had any experience in creating audio games and audio games are also a relatively unexplored topic. The ability to rapidly prototype in the Unity Engine seemed like an advantage which could not be ignored. Furthermore, the engine can also build an Android and an iOS version from the same codebase, allowing the game to be played by a larger portion of the target audience. The only concern was if the Unity Engine would be able to supply the sophisticated audio spatialization that this project requires. Two other audio games, 'Legend of Iris' [4] and 'Loud and Clear' [5], both used the Unity Engine. Legend of Iris used the 'AstoundSound RTI' sound library, but it seems to no longer be available. Loud and Clear used

Unity's 'Audio Spatializer SDK' [25].

6.2 Spatialization

We use the Oculus spatializer for Unity [13] in this project. All audio sources in the game have an ONSP Audio Source component attached to them, to allow the spatialization to work. This component gives us a few additional parameters to tweak, but the only significant change we make is tweaking the range of the sources. Aside from the spatialization from Oculus, another important parameter given attention was the volume rolloff. Some audio sources are given logarithmic rolloff, and some have linear rolloff, to support audio prioritization at different distances.

6.3 Components

Levels are built out of a number of components. Some of these components are solid, and some are non-solid. Partially as a result of how Unity works, the code that powers these components is highly modular. This means that is possible to create an object that has the functionality of multiple components simultaneously. Here, we will go over the different components that these modules were made for and how they work.

6.3.1 The Ball

At its core, the ball is simply a spherical rigid body, with its general movement dictated by Unity's physics engine.

The ball has one main component attached to it, namely its rolling sound. The rolling sound is, as its name suggests, a sound played as the ball rolls around the field. The volume of the rolling sound increases asymptotically to a predefined maximum value the faster the ball rolls. A minimum speed can also be set, under which the ball makes no rolling sound. Along with simply changing the volume of the ball, there is also exaggerated Doppler compensation added to the pitch of the rolling sound. This increases the pitch as the ball rolls towards an object designated as the listener, and similarly decreases the pitch as the ball rolls away. The size of the pitch change depends on the component of the balls velocity in the direction of the listener. A faster ball will therefore change the pitch more dramatically, whether that pitch change is positive or negative.

The ball also has a repeating beeping sound for judging the distance to the flippers. A Unity GameObject is designated as the "listener". The minimum distance is then calculated between the ball and any colliders in the listener. Then, using this minimum distance and the last time that the sound repeated, the next appropriate time for the sound to play is calculated. The appropriate gap in time between each repeat of the sound is calculated every frame, as

calculating this once (after the previous playback) can lead to a situation where the gap is too large for a fast-approaching ball.

6.3.2 The Flippers

The flippers are one of the more technically complicated components of the game, as they are meant to, to a degree, simulate real flippers. This simulation gives us a much finer control of the angles and strengths the ball can be hit at. Simulating the flippers is achieved in a few ways. While the ball is in contact with the flipper, its position along the flipper is tracked. When the player activates that flipper while the ball is in contact, a force is added to the ball. The exact direction and strength of the force to add is however decided by how far along the flipper the ball is contacting the flipper, from the "hinge" of the flipper to the tip. The force and angle at any point of the flipper is precisely defined by two of Unity's AnimationCurves. While animation curves are made with animation in mind, they provide the ability to precisely define a curve, and evaluate it at any point. For the size and angle of the force to apply to the ball, the minimum and maximum force and angle are defined. The point of the ball along the flipper with a range of 0 to 1 is then used as input for each animation curve, which produces an interpolant to linearly interpolate between the minimum and maximum angle and force.

To aid in playability, there is also lenience in when you can hit the ball. This means that shortly after the ball has rolled off the flipper, there is a window of time where the ball can still be launched despite the ball not being in contact with the flipper anymore. This is achieved simply by logging when the ball is in contact with the flipper, and performing a check when the flipper is activated for whether or not the ball should still be launchable with a predefined lenience. There is also a constraint of space added to the lenience, to prevent the ball from being launched in exceptional circumstances. The last consideration with the lenience was to prevent the player from launching the ball multiple times off of a single contact by pressing the activation button in rapid succession. This is prevented simply by adjusting the stored last contact time after the ball is launched to some time outside of the range that would allow for the lenience to work.

Another functionality of the flippers is an indicator of when the ball is contacting the flipper, and how far along the flipper that contact is taking place. This is simply done by playing a looping sound when the ball comes into contact with the flipper, and stopping it when the ball leaves the flipper. Similarly to the force when launching, the pitch of this sound is decided by how far along the flipper the ball is contacting. In this case however, the pitch is decided simply through linear interpolation between a minimum and maximum pitch.

6.3.3 Bumpers

The bumpers are the simplest component of the game. When the ball comes into contact with a bumper, a constant force is added to the ball. This force is

directed in the normal of the point of contact, and is irrespective of the impact velocity of the ball (hence being constant). The bumper has a reference to the score tracker, as well as a section ID and a points value. Through this reference, score is added when the ball comes into contact with the bumper, and score is added for its section. Please see 6.4.2 on the section guider for more detail on the scoring sections.

6.3.4 Force Areas

Force areas are a non-solid game component. They apply a constant force to the ball when it is in contact with a force area. A few other variables apply to deciding whether or not to apply the force. In all cases, the component of the velocity parallel to the force vector is the only part of the velocity considered. Firstly, a maximum velocity can be set, above which no more force will be applied to the ball. This maximum velocity can be infinite. Secondly, there is the option for the force to only be applied when the ball is heading in the forward direction of the constant force. Setting this option to true means that the ball will be unaffected if it moves in the opposite direction of the applied force.

6.3.5 Gates

Gates do not have any unique components in and of themselves, they are instead referenced by the section guider to activate/deactivate them when it is called for. As such, they are simply a wall with no other special aspects. Please see 6.4.2 on the section guider for more details on their usage.

6.3.6 Sound Plates

Sound plates are a mostly passive component of the game, meaning that for the most part, they do not interact with any other objects or systems. Their only default functionality is to play a sound when the ball comes into contact with them. Along with this, they have the ability to award score as well, given a score tracker reference. They do not collide with the ball; the ball simply passes through them.

6.3.7 Rollovers

Rollovers work mostly the same as sound plates (6.3.6). The difference is that sound plates play a looping sound that stops when the ball is no longer touching them. Rollovers on the other hand play an audio clip once per collision and the audio clip does not get stopped until the entire clip is finished. This difference is important, since sound plates are used to give responsive feedback to the player about the position of the ball, while rollovers are used as targets with an accompanying victory sound.

6.4 Systems

6.4.1 Game Loop

The GameLoop component manages the game state. It handles launching the ball, resetting the ball and resetting the entire game when the player game overs. There are four possible game states: Waiting, Game, Gutter and End. The waiting state is for between game sessions (between game overs). When the player activates a flipper in this state, the ball is launched by placing it at the starting location, and activating it. After launching the ball, the game is put into the Game state. In this state, the game loop doesn't respond to any player input directly. Instead, the game loop tracks when the ball has fallen into the gutter, and resets it accordingly. Checking when the ball has fallen into the gutter is simply done by checking if the height of the ball has gone under a certain value. This also helps by guttering the ball in exceptional circumstances where the ball ends up outside of the play field. Resetting the ball simply places it back to the starting position and clears its velocity, and then deactivates the ball until the player launches it again. When the ball falls into the gutter, the remaining balls are announced (if that number is above zero), and the game is set into the Gutter state. The Gutter state simply holds the ball in the reset position until the player launches it again. If the ball falls into the gutter when the player has no balls left, the game transitions into the End state. When the game transitions into this state, the final score is announced, and information is given about the highscores and how the player performed relative to them. After announcing these things, the game is reset back to the Waiting state and it is ready to be played again.

6.4.2 Score

Score Tracker

Score is tracked by a ScoreTracker component. Objects such as bumpers can reference this score tracker to add points. As its name suggests, the score tracker only keeps track of score, it does not itself manipulate the score. The score tracker also comes with references to a score announcer and a section guider.

Highscores

The High Score Manager component saves and loads all scores to determine the high score and the player's ranking. When the player has game overed, the High Score Manager is called, which saves the new score in the ordered list of integers and returns the player's respective rank. The High Score Manager uses Unity's PlayerPrefs system to store the scores, which supports cross platform use.

Score Announcer

The ScoreAnnouncement component is used to read out the score a player has, including announcing the final score when the play has game overed and the number of balls the player has left when one falls into the drain. Scores are announced by cutting together a few prerecorded sounds. The most important of this to bring up is how numbers are read. Each number is split up digit-wise into groups of three. These groups are then read as separate three-digit numbers, followed by either "thousand" or nothing at all. Hence, a number such as 23,456 is read as "twenty three – thousand – four hundred fifty six". Each word is a separately recorded voice line. Voice lines are recorded for numbers 0-19 and all multiples of 10. Aside from those, voice lines denoting hundreds and thousands are also recorded. This allows numbers up until 999,999 to be read out loud, and can be expanded easily by adding lines for higher delimitations, if those numbers were to be realistic.

Section Guider

The Section Guider component is used to power the game mechanic of scoring in different sections in the correct order. Points added to the section guider are assigned to a specific section. There is one section selected as long as the minigame gates are closed. If points are earned in that section, they are added to a running tally. Once the player reaches a predefined score threshold, the next section is selected. When the desired score has been gained in all three sections, the section guider deactivates all objects it has referenced as gates.

The order the sections are selected in is decided when the game starts. To indicate the order, there are references to three audio sources, that the score guider then uses to play sounds representative of the three sections. The audio source of the next section to be completed is played when the ball enters an area around the flippers.

6.4.3 Minigames

Minigames are a relatively complex system in the game. There are a few aspects common to all minigames, decided by an abstract parent class. They must have a designated sound to play when the player triggers them, for a (short) explanation of the minigame. Minigames also get their input from the flippers, as such there are only two possible inputs for a minigame. When a minigame starts, there is also a designated game object for it to activate. What gets activated can differ from one minigame to another, but in general this is at least walls to hold the ball until the minigame is over. After playing the starting sound and activating the minigame object, the minigame can enter its internal game loop. When the internal loop has completed, the minigame ends and the minigame object is deactivated, allowing the ball to roll out of the minigame back into the play field.

Mash Minigame

When the mash minigame starts, the ball is frozen in place for the duration of the introduction voice line. The activated game object has, aside from walls and a roof, a number of bumpers placed around the perimeter of the minigame field. When the introduction line has ended, the ball is unfrozen and the player gains control for a predefined time. During this time, pressing either flipper gives the ball a random lateral force. This force causes the ball to rapidly bounce against the bumpers, gaining points. When the minigame is over, the ball is briefly frozen again to reset its velocity, and then rolls back into the play field.

Direction Minigame

The direction minigame is the second of the two minigames currently in the game. For this minigame, the only things activated at the start of the minigame are the walls and roof of the minigame play field. When the introduction voice has finished playing, the game begins its loop. First a direction is randomly chosen, and a sound is played directly next to the main listener in the according direction. This sound then continuously plays until the player activates a flipper. If the player activates the flipper of the corresponding direction, a success sound is played and points are earned. If the wrong direction flipper is activated, a failure sound is played instead and no points are gained. Regardless of success or failure, the game then waits a short time, picks a new direction, and repeats the process. This goes on for a set amount of time, after which the game simply ends and the ball is allowed to roll freely back into the play field.

Chapter 7

Evaluation

7.1 Testing

The game was tested by the team, the client and the coach during the development phase on a weekly basis. The feedback received from the client and the coach was crucial for fixing some of the bad design decisions in the game. Getting feedback from the visually impaired client gave us a better idea what the main issues were regarding sound design and the project coach gave valuable feedback on the map design.

To evaluate the game a questionnaire was created at the end of the development phase to hand out to random participants. The participants were asked to play the game on Android, but an option for Windows was included for participants that did not own an Android phone. The only difference between the Android and the Windows version is the input controls, which are simple enough to not have a major impact on the playability of the game.
The questionnaire begins with questions regarding the sound design and controls. These questions are used to evaluate the difficulty of tracking the ball, the difficulty of using the flippers, the usefulness of different sounds, etc. Then there are questions to help evaluate the design. And at the end of the questionnaire the participants have the option to leave some feedback for improving the game.

7.2 Results

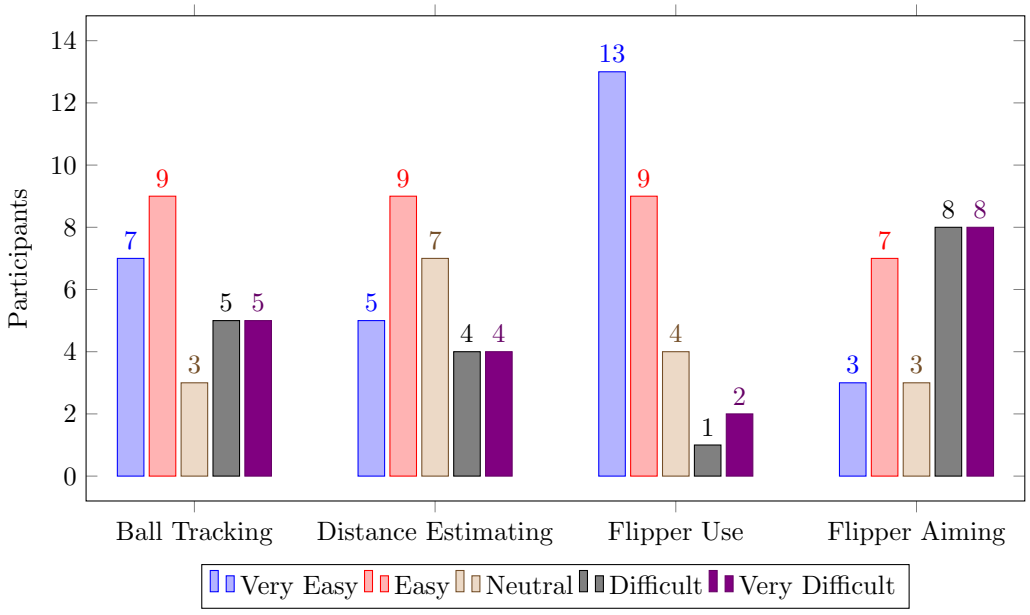
The findings of the experiments have been divided into four main topics. Each topic had multiple questions related to it in the questionnaire employed to the play-testers, and the results of the multi-choice questions are shown in accompanying plots. Twenty-nine participants answered the questionnaire employed for this test, which is a large enough sample size to be significant. The questionnaire itself can be found in Appendix C. There was one misunderstood question

about the sounds when the ball is rolling towards the drain. For that reason, the question is not taken into account for any feedback points.

7.2.1 Package

As a package, the game is well designed. 83% of players find the game fun to play, with 35% of players even choosing the option that they find the game "very fun to play". There are two main points of improvements for the final build. First, 45% of players want to hear their score more often, so milestones were added. For every 100 points the player scores, the announcer notifies them about their score. Second, 83% of players do not notice the sections, and the rotation through them that they are supposed to follow. This part of the game is now explained in the introduction.

7.2.2 Gameplay



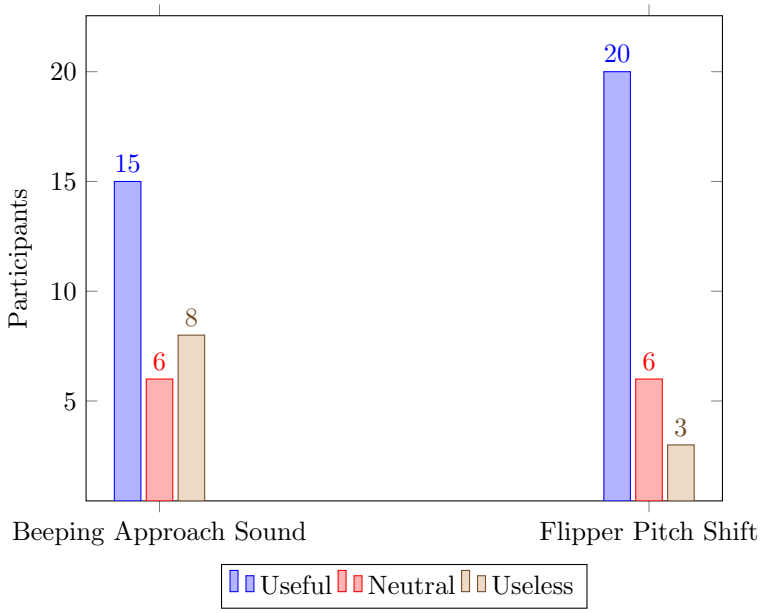
Tracking the direction the ball is rolling towards, and the distance the ball has to the drain appears to be easy enough for most players. On the other hand, aiming the flippers seems to be harder than expected. According to the open questions, players find it slightly too easy to keep the ball out of the drain, and would like to be challenged more in that regard. However, they still have difficulties with completing the rotating section tasks, as they have to practice for a long time with the flippers to be able to aim the ball towards a section. There is conflicting feedback on the ball's speed, with some players claiming it is perfect, while others always find it always too slow or too fast. Adding a

difficulty setting that adjusts the friction of the ball and gravitation pull on the ball should be considered in the future.

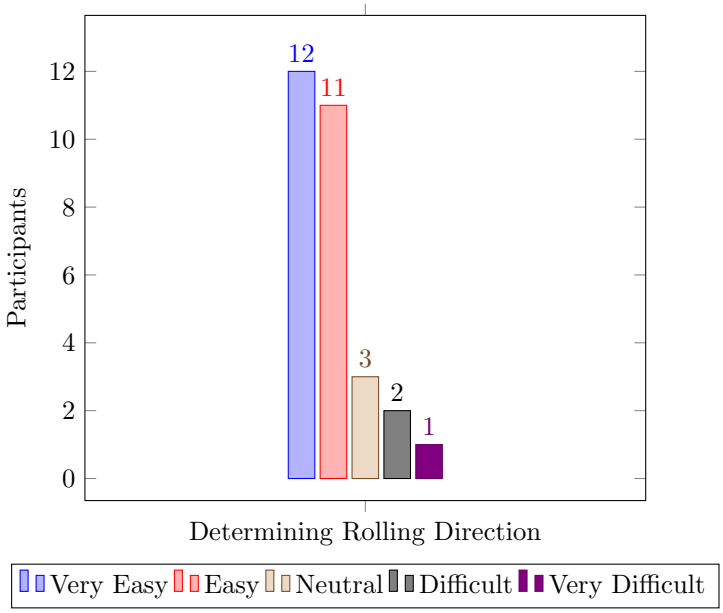
7.2.3 Map

Only 24% of players selected the correct 150 degrees angle of the map from the flippers, with 38% thinking the map was 90 degrees or less and the majority believing it to be 120 degrees. This should not have a negative impact on the playing experience, but does show the limitations of spatialization and accuracy of human hearing. 72% of players thought the map looked completely different from the picture they saw in the questionnaire, and merely 7% found it similar to their mental image of the map.

7.2.4 Audio



Both the beeping approach sound and the pitch shift on the flippers have been reported to be useful. This was not a surprise, as the results are similar to those from previous testers that tested older builds.



Players also find it easy to determine if ball is going away from or towards player, which is done by listening to both the changing pitch of the ball's friction sound effect and the volume reduction when the ball is further away from the player.

7.2.5 Mini-games

Even though 83% of players did not notice the sections and the way to unlock the gates, 52% of players managed to play at least one of the two mini-games, and 21% of players figured out how to play both.

7.2.6 Open questions

There were multiple open questions for play-testers to give feedback on the topics they wanted to discuss. Multiple testers requested for haptic feedback to be added to the Android build as an additional feedback mechanism when operating the flippers. Most importantly, the testers reported that they had fun, and liked both the main gameplay and the mini-games. Some clear-sighted people want to play this when their eyes are tired, to relax. This seems to be evidence that the game is fulfilling its original purpose: bringing an enjoyable experience in the form of a game without visuals.

Chapter 8

Discussions and Conclusion

8.1 Discussion and recommendations

The primary goal of this project was to develop a working audio-only implementation of pinball. As discussed before, this brought up numerous challenges that the team had not anticipated. This section discusses and compares the initial expectations of the team and the eventual findings and recommendations gathered during the span of development.

8.1.1 Capabilities of Spatial Audio

An important aspect of developing a playable game without visuals is using the right audio technologies. Spatialized audio is a very important technology for achieving the goals of the project, since it theoretically allows for more accurate positional estimation of audio sources. This was known among the team before development started, and played a significant role in deciding the right software for development. During the research phase, more research and experimentation was performed on spatialized audio, specifically on HRTF's. This yielded the discovery that with no visual inputs, even though much better than stereo audio, it was still challenging to locate the position of spatialized audio sources. As mentioned in the research report, HRTF's appear to perform well in improving directional perception, especially horizontally. It does not however, deliver such consistent results in distance perception of the audio source. When developing an audio-only game, it is therefore important to utilize the stronger aspects of HRTF's, especially in designing the player level. This should therefore utilize horizontal space as much as possible, and not rely on distance perception of the player. This discovery can be noted in the project's final map design, which is much more horizontally spaced compared to a traditional pinball setup.

8.1.2 Limitations of Audio-Only Interactivity

Another discovery made during development is the relatively limited amount of interactivity allowed before yielding sensory overload to the player. Since audio-only games limit the channel of communication exclusively to audio, the player will get overwhelmed fairly quickly when using multiple spatialized audio sources. This was much more of an issue than previously anticipated, and required a careful weigh-off between adding features and maintaining a playable product. Throughout the iterations of development, it was discovered that deciding on game play features required a different approach compared to conventional video-game design. When approaching feature/ game play ideation for audio games, limiting the information stream should always be a big consideration. In traditional video game development, this approach is not as important, since a gameplay feature can be tweaked in more aspects after initial playtesting. This is much more limited in audio game development, which means certain gameplay setups cannot be tweaked to work if these are fundamentally unfit for audio-only playability.

8.1.3 Development Strategies of Audio-Only Games

Throughout the development progress, it became apparent that designing the game (specifically the level), required some careful considerations. Even though the project's scope was to only use audio as a channel, it still deemed quite important to use visuals during the design. This was in part caused by pinball being very physics reliant. A common tendency among the team seemed to be designing the map and development visuals towards visual use. This tendency shifted as the team discovered a lack of progress when designing for visual purposes. It is important to use clear visual references when designing, but to strictly keep this as a tool for monitoring, and not for design decision making.

8.1.4 Testing of Audio-Only Games

User testing is well known to be very important in video games, but is even more important when developing an audio-only game. In earlier parts of the project, the team performed user testing mainly within the team and with the client and coach. This user testing always showed the visuals of the game to help the tester in case they were lost. This strategy proved to work to some extent, as it did give the player a better idea of what the game entails, but upon reflection, early user testing should have already been performed with no visuals. This was concluded during our official evaluation setup, where giving the players no visual feedback resulted in more critical feedback. This is most likely due to visual feedback giving the player a mental reference of a lot of information, which results in the player assuming multiple aspects before actually playing the game as intended. It is also highly recommended, if possible, to use a subset of visually impaired test subjects during user testing.

8.2 Discussion on ethical implications

A very significant factor that lead to the development this project was the objective of giving less-sighted people the opportunity to play regular and entertaining digital games. This is therefore a first obvious ethical implication of creating proper audio games such as this project. Currently there is no large availability of audio-only games, which brings less-sighted people to a rather significant disadvantage regarding their options of interactive entertainment. This is most likely due to the numerous challenging aspects (as discussed before) of developing a playable and entertaining audio-only game. If more research and experimentation were to be put into alternative development strategies for these kind of games, resulting in more standardized methods and setups, the supply of these experiences could increase.

On top of this, showcasing experiences which force sighted players to omit their visual senses while playing, could increase awareness of the limitations and current shortage of proper entertainment for less-sighted individuals.

8.3 Conclusion

The primary goal of this project was to create a playable audio-only pinball game. The end product, upon reflection, seems to have achieved this goal. Throughout development however, the team ran into multiple challenges that come with designing audio-only implementations of regular games. It became apparent that designing and implementing an audio-only game requires a different development approach compared to visual games. Implementing spatialized audio also did not relay as much information to the player as the team had hoped, and this lead to a need for more creative solutions to achieve proper play-ability. The field of audio-only games is certainly limited at the moment, but has good potential for a wider range of proper games.

Bibliography

[1] Marty Schultz. Blindfold Pinball, apr 2016. https://apps.apple.com/us/app/blindfold-pinball/id1093319916.

[2] iOS App Store. https://www.apple.com/ios/app-store/.

[3] Jaroslaw Beksa, Alexandra Garkavenko, Sonia Fizek, Shahper Vodanovich, and Phil Carter. Adapting Videogame Interfaces for the Visually Impaired: A Case Study of Audio Game Hub. *Information Systems Development: Complexity in Information Systems Development*, pages 117–124, sep 2016. https://aisel.aisnet.org/cgi/viewcontent.cgi?article=1104&context=isd2014.

[4] Kevin Allain, Bas Dado, Mick van Gelderen, Oliver Hokke, Miguel Oliveria, Rafael Bidarra, Nikolay D. Gaubitch, Richard C. Hendriks, and Ben Kybartas. An audio game for training navigation skills of blind children. *2015 IEEE 2nd VR Workshop on Sonic Interactions for Virtual Environments (SIVE)*, pages 1–4, 2015. doi:10.1109/sive.2015.7361292.

[5] Berend Baas, Dennis van Peer, Jan Gerling, Matthias Tavasszy, Nathan Buskulic, Nestor Z. Salamon, J. Timothy Balint, and Rafael Bidarra. Loud and Clear: The VR Game Without Visuals. *Lecture Notes in Computer Science Games and Learning Alliance*, pages 180–190, 2019. doi:10.1007/978-3-030-34350-7_18.

[6] David Schönstein and Brian Katz. Variability in Perceptual Evaluation of HRTFs. *Journal of the Audio Engineering Society*, 60:783–793, 10 2012.

[7] Continuous integration and delivery. https://about.gitlab.com/ci-cd/.

[8] Gabriel Le Breton. Unity3d docker image. https://hub.docker.com/r/gableroux/unity3d/.

[9] Gabriel Le Breton. Unity3d CI exmple project. https://gitlab.com/gableroux/unity3d-gitlab-ci-example.

[10] Unity Test Framework. https://docs.unity3d.com/Packages/com.unity.test-framework@1.1/manual/index.html.

[11] Netlify. https://www.netlify.com/.

[12] Gitlab Review Apps. https://docs.gitlab.com/ee/ci/review_apps/.

[13] Oculus Native Spatializer for Unity. https://developer.oculus.com/documentation/unity/audio-osp-unity/.

[14] Unity Code Coverage package. https://docs.unity3d.com/Packages/com.unity.testtools.codecoverage@0.2/manual/index.html.

[15] Latest coverage report. https://audio-pinball-coverage.netlify.app/.

[16] Test Coverage Visualisation. https://docs.gitlab.com/ee/user/project/merge_requests/test_coverage_visualization.html.

[17] Test coverage parsing. https://docs.gitlab.com/ee/ci/pipelines/settings.html#test-coverage-parsing.

[18] Daniel Palme. Report Generator. https://github.com/danielpalme/ReportGenerator.

[19] JUnit test reports. https://docs.gitlab.com/ee/ci/junit_test_reports.html.

[20] xsltproc. http://xmlsoft.org/XSLT/xsltproc.html.

[21] Alex Earl, Erik Ramfelt, Timotei Dolean, and Hein Couwet. nunit-to-junit.xsl. https://github.com/jenkinsci/nunit-plugin/blob/master/src/main/resources/hudson/plugins/nunit/nunit-to-junit.xsl.

[22] Discord. https://discord.com/.

[23] Mattermost. https://mattermost.ewi.tudelft.nl.

[24] iPhone Vision Accessibility. https://www.apple.com/accessibility/iphone/vision/.

[25] Unity Audio Spatializer SDK. https://docs.unity3d.com/Manual/AudioSpatializerSDK.html.

[26] Michael Littman. Brief History of Pinball. https://commons.princeton.edu/josephhenry/brief-history-of-pinball/.

[27] Internet Pinball Machine Database: Gottlieb “Baffle Ball”. https://www.ipdb.org/machine.cgi?gid=129.

[28] Henrik Møller, Michael Friis Sørensen, Dorte Hammershøi, and Clemen Boje Jensen. Head-Related Transfer Functions of Human Subjects. *J. Audio Eng. Soc*, 43(5):300–321, 1995. http://www.aes.org/e-lib/browse.cfm?elib=7949.

[29] Julián Villegas. Locating virtual sound sources at arbitrary distances in real-time binaural reproduction. *Virtual Reality*, pages 201–212, nov 2015. https://doi.org/10.1007/s10055-015-0278-0.

[30] Christopher Frauenberger and M. Noisternig. 3D Audio Interfaces for the Blind. 01 2003. https://www.researchgate.net/publication/215639387_3D_Audio_Interfaces_for_the_Blind.

[31] William G. Gardner and Keith D. Martin. HRTF measurements of a KEMAR, 1995. https://doi.org/10.1121/1.412407.

[32] Niklas Röber and Maic Masuch. Playing Audio-only Games: A compendium of interacting with virtual, auditory Worlds. 01 2005. http://www.digra.org/digital-library/publications/playing-audio-only-games-a-compendium-of-interacting-with-virtual-auditory-worlds/.

Appendix A

Project description

Audio Pinball
Bachelor project for CS Students

In the vast majority of (digital) games, their visual features play a crucial role in the game play and the entertainment experience:

There are almost 40 million fully blind people worldwide, and another 217 million people considered at least moderately visually impaired. Still, there are very few digital games suitable for blind or partially sighted people, let alone designed on purpose for them. Designing games without visuals is understandably challenging, but this situation is undesirable and unfair for such a large population.

Increasing effort is being dedicated to investigate how 3D sound and spatial hearing skills could be used as the main sense for entertainment. In this project, you will explore this design space in one particular direction: design and implement a digital game that conveys the core game mechanics of the classic pinball game, but purely based on spatial audio features (and possibly some haptics), without any visuals.

The game should require no complicated setup, and be playable by both blind and sighted people on a mobile phone, which would then drive the headphones and serve as gamepad as well. This game is meant to be tested and actually deployed at the Muzieum (https://muzieum.nl/), in Nijmegen, The Netherlands.

Workplace: mostly TU Delft (plus sporadic playtesting at the Muzieum or in Zeist) Target platform: Android smartphones

Commissioner: Lars Lommers (Muzieum) Audio tech supervision: Richard Hendriks (TU Delft, EEMCS) TU Delft advisor: Rafael Bidarra

A.1 muZIEum

A museum that makes you experience what it's like to live with a Visual impairment. Aside from the experience, it's also a place that offers people with a Visual impairment a place to work

Appendix B

Info sheet

Title: Audio Pinball
Client Organization: The muZIEum
Date of Final Presentation: 01-07-2020

B.1 Description

We were tasked with making a version of pinball that is played without any visuals whatsoever. During the research phase, we learned a lot about the options and capabilities of spatialized audio technologies, but did not make many findings regarding the development of audio-only games. The findings about spatialization affected a lot of the design decisions of the game, such as level design and game-play elements. The game was made in the Unity game engine. Unlike usual game development, we found that due to the lack of existing knowledge on the subject, creating this game had much more of an element of testing different ideas just to see what worked. In the end, we believe we succeeded in creating a game of pinball that is entirely playable without any visuals, that can be enjoyed by both sighted and non-sighted people alike. The game is both simplified in certain ways to accommodate the lack of visuals, but has made up for these concessions with alternative mechanics. The game was regularly tested together with our coach and the (sight-impaired) client representative. Towards the end of the project, a survey was sent out to collect feedback from outside of the development team. The game can be expanded on by a future team by adding new mechanics and further tweaking the existing ones. It will be available at the muZIEum in Nijmegen.

B.2 Members

B.2.1 Drew Berge

Drew is himself a gamer with an interest in game design. As such, his primary contributions to the project were the idea of various game mechanics, such as the minigame system and the flipper design.

B.2.2 Danilo Bettencourt

As a computer science student, Danilo is interested in computer graphics and low level programming. Danilo's main contributions were the prototype force flipper and various map designs.

B.2.3 Stanley Lageweg

As someone who has the ambition to work in game development and as a Computer Science student, Stanley is interested in game design and programming. His main contributions were the setup of the CI pipeline, level design, code testing and code refactoring.

B.2.4 Willie Overman

Within the field of computer science, Willie is interested in computer graphics and additionally has extensive experience with video game development. During this project, Willie's main contributions entailed work on the abstract game systems such as the game loop, section guider, score announcer and high score manager.

B.2.5 Amir Zaidi

As an hobbyist in Android development, Amir is the only member of the team with former experience in mobile development. However, during this project Amir's main focus has been improving the quality of sound source localization and using audio spatialization in the best way possible.

Client: Lars Lommers, muZIEum
TU Delft Coach: Rafael Bidarra, Computer Graphics & Visualization Group
Contact person: llommers@bartimeus.nl

The final report for this project can be found at: https://repository.tudelft.nl/

Appendix C

Questionnaire

Which build of the game did you use?

1. Android
2. Windows

How long did you play the game?

1. less than 2 minutes
2. 2 to 3 minutes
3. 4 to 5 minutes
4. 5 or more minutes

Do you have any significant visual impairments?

1. Yes, complete loss
2. Yes, significant loss of vision
3. No or nothing significant

How difficult is it to track where the ball is?

1. Very easy
2. Easy
3. Neutral
4. Difficult
5. Very difficult

How difficult is it to tell the distance of the ball?

1. Very easy
2. Easy
3. Neutral
4. Difficult
5. Very difficult

Do you consider the ball rolling too slow, too fast, or just right?

1. Too slow
2. Too fast
3. Sometimes too slow, sometimes too fast
4. Just right

How difficult is it to use the flippers to keep the ball out of the drain?

1. Very easy
2. Easy
3. Neutral
4. Difficult
5. Very difficult

How difficult is it to aim the ball with the flippers when shooting it?

1. Very easy
2. Easy
3. Neutral
4. Difficult
5. Very difficult

How difficult is it to determine if the ball was rolling towards or away from you?

1. Very easy
2. Easy
3. Neutral
4. Difficult
5. Very difficult

How useful is the beeping sound when the ball is approaching you?

1. Useful
2. Neutral
3. Useless

How useful is the sound with the changing pitch while the ball was rolling on a flipper?

1. Useful
2. Neutral
3. Useless

Do you notice that there are 3 sections?

1. Yes
2. No

Which themes do you think that the 3 sections have? Leave this open if you did not notice the sections.

Open response

Do you notice that there are sounds playing when the ball rolls towards the drain?

1. Yes
2. No

What do you think is the purpose of the 3 sounds that play when the ball rolls towards the drain? Leave this open if you did not notice the sounds.

Open response

There are two bonus sections with minigames. One lets you tap the flippers as fast as you can, and the other lets you press in the direction of the sound. Did you get to the two bonus sections?

1. Both
2. Only one
3. None

The game only tells you the score when the ball rolls in the drain. Is this sufficient, or do you want to hear the score more often?

1. Sufficient

2. Insufficient

What did you like about the map?

Open response

What didn't you like about the map?

Open response

How similar was your mental image of the map during the audio game compared to the image of the map?

1. The same
2. Slightly different
3. Completely different

How much fun did you have in this game?

1. None
2. A little
3. A lot

How likely is it that you would recommend the game to others?

1. Very likely
2. Maybe
3. Definitely not

What is the feature you enjoyed most?

Open response

What is the biggest flaw of the game that you would like to see fixed?

Open response

Appendix D

Research Phase Report

D.1 Research Introduction

During the 2020 Bachelor Project, the group will be developing an audio-only pinball game for Muzieum. This means the game needs to be playable without any visuals by both blind and sighted people. To achieve this, spatial audio will need to be implemented in order to aid the player in understanding what is going on within the game. Before development can start, a decision has to be made on a number of approaches. In this research report, the most important questions related to the project will be addressed.

One of the most important factors that needs to be researched is the capabilities and limitations of spatial audio as a medium. For example, how well can spatial audio be used to portray the position and movement of objects within a game? Or how precise is human hearing in locating the positional source of a sound? These questions will need to be answered before development can start, to ensure efforts are put into something that is actually achievable and comfortably playable.

D.1.1 History of Pinball

Pinball has gone through an extensive evolution in the past, but the most important aspects of the game have always remained. A ball is inserted into the system, which rolls down due to gravity. The goal of the game is to score points by controlling the ball in such a way that it hits targets, which are usually pins. If the ball enters a "drain", an area of the playing field where the ball disappears, the player loses the game. [26]

One of the first iterations of pinball, Baffle Ball, used a field with only four objects to hit. [27] Controlling the ball was impossible after launching the ball with the inject mechanism, so the only control given was during the injection. Flippers were added, which evolved into the flippers used today in pinball. Having this additional level of control allowed for a more interesting playing

field and consequently a significantly higher skill gap as well. Over time more aspects were added to the game to make gameplay more diverse. Bumpers, ramps, kickers and slingshots were placed on the field. Some variants of pinball added multiple balls, while others experimented with time trials.

D.2 Problem Definition and Analysis

Audio games come with many drawbacks that make it hard to adapt most games into the format. Pinball is no exception to this problem.

D.2.1 Problem Definition

Pinball has many variants and has gone through large changes since its conception. The main problem is adopting one of these pinball variants to the audio game format. To make the game work as an audio game, some changes will always have to be made to the gameplay. However, the goal is to adapt Pinball, and not to create a game vaguely resembling it. In the many existing variants of Pinball there could be one that works well as an audio game without major changes. If there is one, that variant should be the base of the audio game. If not, a new variant has to be developed. Testing would then be required to determine if players of the game still consider it a Pinball game.

D.2.2 Problem Analysis

There are various elements to Pinball that make adapting it into an audio game a difficult task.

First, giving constant feedback, such as with a score counter on the screen, is only possible with a constant sound. However, that might quickly become distracting. It is then important to decide what aspects of the game should provide feedback and when, as not all information needs to be constantly available to the player. Some information which the player is used to being able to view at any point, such as the score, requires more attention in an audio-only game than when there are visuals, as the ability for someone to change where they are looking does not trivially analogue to sound.

Second, object localization with only audio is hard, and becomes even harder when too many objects are added to the surroundings. Tracking the precise location of a moving object can probably only be done by those who have practiced that skill their entire life. Because of this, methods of tracking the ball and the game state in general should not rely entirely on spatialization. It becomes important to consider what information can be sonified, in ways that the player can learn without hindering them in other ways.

Third, the difficulty of the game has to be adjusted to account for the loss of visuals. Since localization with audio is much harder than with visuals, the game should be made easier than regular pinball. If the difficulty is too high, players might get frustrated by it and quickly stop playing the game. There are

a number of ways to adjust the difficulty of pinball, such as the speed of the ball and the layout of the play field, and attention needs to be put to adjust these to a desirable level.

D.3 Project Goal

The main goal of the project is to develop a pinball audio game for Android OS (and if possible iOS) that has intuitive controls. The player will need to easily be able to track where the ball is moving on the playfield so that player can activate the flipper(s) on time to deflect the ball. The goal is not however for the player to be able to pinpoint the precise location, but instead have a general idea of which direction the ball is moving towards compared to the player's location on the map.

D.4 Related Work

Blindfold Pinball [1] is a game on the iOS App Store™ [2] that claims it can be played blindfolded. The game is designed to be able to be played using only audio or visuals. When using visuals it is really easy to track the ball on the playing field, but when playing blindfolded the game fails in this aspect. The speed of the ball is too high and the amount of bumpers on the map produces too many sounds that overpower the sound of the ball. You can also not really tell when the ball is close to the flippers. This is due to the speed of the ball and because the sound of the ball does not get amplified enough when it's approaching the drain. A rectangular playing field also feels limiting for an audio-based game, as only a small range of spatial directions is used. It's difficult to differentiate a sound which is slightly to your left, from a sound that is straight in front of you.

Another related initiative to make a set of audio games is the Audio Game Hub [3], which is an assortment of audio games created to be playable both by sighted and non-sighted people alike. Here, both spatialization and sonification techniques are used to convey the game state. For example, in one of the games, Hunt, a target moves around the screen that the user must shoot. Here, the position of the target is conveyed in a few ways: as the user aims closer to the target, a targeting sound plays with increasing frequency; a stereo panorama is used to locate the object left and right; and the pitch of the targeting sound varies with the height of the target on-screen. While the target being visible makes this game trivial for a sighted person, it is entirely playable and more engaging when the player relies entirely on the audio.

Outside of the individual games in the hub, the creators also gave much attention to the menus. All important information is conveyed through voiceover, giving detailed tutorials of the games and their settings. The menu controls are also usable without seeing the screen. Changing selections is done through swiping the screen and other buttons, such as the back button or information

button, are placed relative to the corners of the screen, making them easy to locate.

Even compared to normal pinball, the games are relatively simple. They generally feature few individual game elements, letting the player use multiple audio cues for similar information, aiding in making the state of the game be extremely clear. The techniques used however are still applicable to Pingball.

Legend of Iris [4] and Loud and Clear [5] are two audio games developed at Delft, University of Technology. These games were made to gamify a regular task, while Pinball is a game with no additional purpose. However, the research done by their development studios is still useful for the development of Pingball by looking at their findings. For example, during their testing the Loud and Clear team discovered that "For many players, it was rather difficult to estimate how far a sounding item was. Indeed, detecting extreme relative distances is quite easy, e.g., when an item is really far or really close to the player; but distinguishing between two intermediate distances turns out to be rather difficult." [5, p. 9] Since determining the distance of the ball to the drain is an important aspect of pinball, this is important information to keep in mind.

D.5 Requirements Analysis

From the project description that can be found in Appendix A, multiple sub-requirements can be extracted.

> *There are very few digital games suitable for blind or partially sighted people, let alone designed on purpose for them.*

The game should be designed from the ground up for blind and partially-sighted people, and the requirements are made in the context of designing a game for them.

> *Design and implement a digital game that conveys the core game mechanics of the classic pinball game.*

The game should include the core mechanics of the classic pinball game. This refers to a moving ball falling down to a drain due to gravity, two player-controlled flippers to keep the ball out of the drain, and obstacles in the map that give the player points when the ball hits those obstacles.

> *Purely based on spatial audio features (and possibly some haptics), without any visuals.*

It should be playable without any visuals, and by default not have any visuals for that reason.

> *The game should require no complicated setup.*

When the player starts the game, he should be guided towards the basic gameplay loop within seconds. Any required setup could be hidden in a tutorial, but this should not become complicated. If play-testers have an issue with any aspect of this tutorial it must be removed.

> *Be playable by both blind and sighted people on a mobile phone.*
> *Target platform: Android smartphones*

This game should be compiled for Android smartphones, so there are limitations to which engines and frameworks can be used, as not all support the Android platform. User interface elements introduced in the game should be compatible with and navigable by Android's accessibility services.

> *Which would then drive the headphones and serve as gamepad as well.*

No external controller should be required, and the Android phone must be used as the input device. The sound mixing should be optimized for headphones, as that is the set-up the client wishes to use.

> *This game is meant to be tested and actually deployed at the Muzieum (https://muzieum.nl/), in Nijmegen, The Netherlands.*

As the game will be placed in a noisy environment, the sound mixing should not rely on any sounds with low loudness.

D.5.1 MoSCoW

Using the requirements analysis, the following MoSCoW requirements list was created. Must have items are requirements that cannot be missed in the final product, and have to be prioritized. Should have items are similar, but might be dismissed as infeasible. Could have items are interesting options that must be looked in to but have a low chance of appearing in the final product. Won't have items are ideas that were discussed previously and already determined to be excluded from the final product, but could be reconsidered at a later point in the project, when the other requirements have been met.

Must Have

- (Semi)circular playfield
- User input
- Flippers
- Sound when ball is rolling
- Spatialized audio
- Score points when hitting bumpers

Should Have

- Identifiable but not obnoxious sound when ball approaches flippers
- Sound when ball touches flippers
- High Score system

Could Have

- Ramps
- Slow down field near the flippers
- Force field type of flipper, throws a force towards a certain direction and pushes the ball back like that.

Won't Have

- Head tracking
- Custom HRTF audio engine (Unity already has this)

D.5.2 Requirements Elaboration

The previous subheader describes the requirements of the game in the MoSCoW system.

Some important parameters that need extra attention are the speed of the ball, the amount of sounds and map design. The speed of the ball is an important parameter that will need to be extensively tested and tweaked during the development phase. The ball cannot be too slow or it will make the game boring and not too fast or the player will have no idea what is happening.

The sound of the ball should be the most audible one, other components should have (slightly) lower volume. This will ensure that the player does not lose track of the ball.

D.5.3 Success Criteria

The project is considered successful if the following requirements hold. First, the must-haves mentioned in 5.1 are implemented. Second, players are able to effectively track the ball only using sound. Third, players find enjoyment in playing this game, as determined by a questionnaire.

D.6 Audio Spatialization

The human auditory system uses more than the difference between two ears to determine the location of sound. After all, two ears only give to measurement points, which would merely give one dimension of the location. Instead, sound entering the ear canal is processed to determine its direction. This is possible because soundwaves become distorted by the skull and outer ears in an identifiable pattern.

D.6.1 Head-Related Transfer Function

Every direction has its own head-related transfer function (HRTF) which determines how the sound is transformed when it is coming from that direction. [28] This HRTF can be emulated by a computer to trick a person into thinking sound is coming from a different direction than it is actually coming from. [29] If this is applied inside a game engine it can fully immerse a player with 360 degrees sound even when that player is only wearing stereo headphones.

According to Møller et al. the HRTFs "converge to 0dB for low frequencies", and "the main differences between directions are found as characteristic peaks and dips above 1kHz." This implies that overtones and higher pitched sounds are much better at being localized by the listener than the base tones and lower pitched sounds, which should be taken into account for sound design. "HRTFs for directions behind the subject have in general low amplitudes, especially at the highest frequencies", so placing sound sources behind the listener might be confusing and hard to localize. The report by Møller et al. describes from which directions humans can determine the location of sound well, and from which ones they cannot.

The capabilities and limitations of HRTF make a difference in how the playing field is set up. Schonstein and Katz evaluate the variability in perceptual evaluations of HRTFs in their paper Variability in perceptual evaluation of HRTFs [6]. The results of this study can be used to determine what properties of HRTF to utilize in the game for maximum effect. The study compares the consistency in test subjects based upon 3 attributes:

1. Sense of Direction: the ability of test subjects to determine which relative direction audio sources are placed.

2. Sense of distance: the ability of test subjects to define the distance between the sound source and listener.

3. Front image quality: the ability of test subjects to define the percept of a sound coming from in front of them.

The paper concludes that based on the subjective reports of the test subjects, the attributes sense of direction and front image quality seemed to be most useful. Sense of distance turned out not to be particularly reliable.

When looking at the results of the study in the aforementioned paper, it appears that the setup of the playing field should not rely too heavily on the player's judgement of the distance of the ball from the listener, but rather on the relative direction. This can be implemented by constructing the playing field to curve upwards in the distance, so the outer edge of the playing field is higher than the inner area.

D.6.2 Reverberation

Frauenberger et al. found that "early reflections are significantly contributing to the sense of space and sound source localisation". [30] Therefore, high quality room reverberation emulation is important to determine the location of objects in the playing field.

D.7 Unity Engine

For the game, three development environments were considered: Android Studio, the Unreal Engine and the Unity Engine.

Android Studio had the advantage of allowing complete freedom with the project, at the cost of having to write all the code from scratch. Since the game will involve a rolling ball, a physics engine would have to be written, next to an audio engine. This would mean a lot of extra work. The game would also only run on Android. The assignment only specified that an Android version would be the only requirement. However, the client adviser explained in a meeting that most visually impaired people use iPhones, due to their better vision accessibility features [24]. Thus, an iOS build would be desirable.

The Unreal Engine was also considered. It provides a lot of desired features, such as a physics engine. It also is generally more customizable than the Unity Engine. However, it is also generally a lot harder to work with than the Unity Engine. This is in part due to a lack of documentation. Our TU Delft coach recommended to not use the Unreal Engine, as he has seen a lot of projects fail due to the difficulty of using said engine.

The Unity Engine was eventually chosen as the development environment. It was decided that a significant portion of the project would consist of trying out and testing new ideas, as none of the group members had any experience in creating audio games and audio games are also a relatively unexplored topic. The ability to rapidly prototype in the Unity Engine seemed like an advantage which could not be ignored. Furthermore, the engine can also build an Android and an iOS version from the same codebase, allowing the game to be played by a larger portion of the target audience. The only concern was if the Unity Engine would be able to supply the sophisticated audio spatialization that this project requires. Two other audio games, 'Legend of Iris' [4] and 'Loud and Clear' [5], both used the Unity Engine. Legend of Iris used the 'AstoundSound RTI' sound library, but it seems to no longer be available. Loud and Clear used Unity's 'Audio Spatializer SDK' [25].

D.7.1 Audio Spatializer SDK

The Audio Spatializer SDK [25] is an extension of the native audio plugin interface of Unity. It allows developers to change "the way audio is transmitted from an audio source into the surrounding space." [25]. This sdk "allows replacing the standard panner in Unity by a more advanced one and gives access to important meta-data about the source and listener needed for the computation." [25]

Unity offers a free to use demo HRTF implementation based on HRTF measurements made by Gardner and Martin in the MIT Media Lab. [31] This covers the HRTF data measured from a KEMAR dummy head consisting of 710 positions sampled at elevations from -40 to 90 degrees. This data is sufficient for the planned Audio Pinball setup, since the majority of the directional audio is only required from above the player's horizontal line, which is more than sufficiently covered by the dataset. This means reimplementing a customized HRTF system within Unity is unnecessary effort.

D.8 Head Tracking

Using head tracking to improve sound localization and immersion for players can strongly enhance perception of 3D sound [32], but the decision to use it would have to be made at an early stage. The various options for this technology were discussed and decided against, to make the game accessible to everyone regardless of the hardware they have available.

D.8.1 Face Recognition

One way to perform head tracking is through the use of facial recognition technologies. This method however requires a more complicated setup than is appropriate for this project. Using face recognition for head tracking requires an additional camera to be trained on the subject, which complicates the setup of the game significantly, making the game less accessible both for usage reasons as well as cost reasons.

D.8.2 Google Cardboard

Google cardboard is a cheap and easy way to implement head tracking for phones. The problem is that it is not that easy to set up for the visually impaired and needing it to play the game limits the places you can play the game at.

D.8.3 Phone Gyroscope

Another possibility is to track the movement of the phone without attaching the phone to the user's head like with Google Cardboard. This then means

that for the best effect, the user would need to manually move their head with their mobile phone for the spatialization to still work. The advantages of this are that it requires no additional setup, and it frees up the screen of the phone for potential input. It however brings up concerns of comfort, especially if the game is played over a longer span of time.

D.9 Game Design

A very important aspect of developing an effective audio pinball game is of course the details of the game design. Getting the aspects of this right determines whether the game is actually fun and most importantly, playable. For game design, the most logical setup of the playing field, the movement and the gameplay controls will be evaluated.

D.9.1 Playing Field

As discussed in section 7.1, the limitations of HRTF determine how the playing field needs to be set up and shaped. This section will go into more detail on the shape and attributes of the playing field.

Position and Shape

In regular pinball, the playing field is positioned in front of the player and has a rectangular shape. This setup could also be used in the same way for an audio adaptation, like with Blindfold Pinball [1]. This way, however, the players only use a fraction of the available directions they could distinguish between using audio. A circular field, with both the player and the gutter positioned in the center, would allow the player to distinguish sound in 360° space. As discussed before, the field setup should take maximum advantage of the capabilities of HRTF, so should curve upwards away from the player, such that the outer edge is significantly more elevated than the gutter. This could prove to be disorienting for the player however, since sounds behind the player can be hard to localize (see 7.1). Thus further research and testing is required during development to determine the best approach.

Elevation

Players can not only locate sounds in a 360° plane, but they also can determine elevation. Apart from rails, the ball never really changes elevation in regular pinball. The team would like to experiment with different ways in which to introduce elevation aspects into the game, such as with multiple ground levels. Although, this will probably get disorienting rather quickly and might be difficult to incorporate into a level.

The ball does change elevation because the field is at an angle of course. Doing the same for the audio game could prove to be helpful in localisation. If the

playing field goes up from the gutter, then determining the elevation lets players estimate the distance from the gutter. Combined with the volume of the sound, it should help players determine the position of the ball more accurately.

Obstacles

The playing field will have to facilitate multiple obstacles to make the game comply with basic rules of pinball. The most important obstacles that will need to be in the game are bumpers, as these give the player points, and are a recognizable element of any pinball field. The bumpers will be activated when touched by the ball, and shoot the ball outwards. There should be multiple point classes for bumpers depending on the reachability and danger of bouncing the ball towards the pit.

Ramps, kickers and slingshots can be added too, but this will be evaluated after initial playtesting, since the addition of too many elements will be overwhelming and essentially require the player to memorize the playing field.

D.9.2 Ball

The speed of the ball is something which probably will require further testing, as the team suspects that this is one of the main determinants for the difficulty of the game. What makes it more complicated, is that the game should be playable for both sighted and visually impared players. If the ball moves too fast, it might be difficult for sighted players, while if the ball moves too slow, it might be boring for visually impared players.

To mitigate this problem, there are two possible solutions: Adding different difficulty settings or ramping up the difficulty over time. These solutions should be tested to determine the best approach.

D.9.3 Controls and Flippers

The flippers will be controlled using primarily the touchscreen of a phone and the optional use of a controller. There was the idea of using google cardboard for the head tracking function, but this would make setting up for the game be more complicated and would limit the places and/or time the game could be played, so the decision was made against it. There are two potential ideas for the flippers. The flippers are like the classic pinball flippers that you can find in the arcade or a custom flipper that works like a rotating force field that can push the bomb away when activated. During the development of the game some testing will be required to find the best option of the two.

D.10 Research Conclusion

There were some important points touched upon in our research that will guide the development of Audio Pinball. Unity will be the base of the game, sup-

ported by the Audio Spatializer SDK for the audio processing. There are some limitations in what can be done in an audio game, even with perfect audio processing. The game has to be simple to stay accessible to everyone. Too many objects creating sound at once could become too overwhelming for players. Localizing the distance of objects with only audio is difficult, so the game has to be designed in a way where the distance of objects can be identified in a different way. Using the theory about HRTFs the game can be designed from the ground up to work as an audio game, fulfilling the most important purpose of its development: bringing enjoyment to visually impaired people.

Appendix E

Software Improvement Group

Since the team was unaware that the Software Improvement Group would only do static analysis, all the code was submitted, under the assumption that it need to be executed. This included the Oculus Spatializer plugin [13], which caused the score to be heavily influenced by code that the team did not write (see Table E.1). The refactoring candidates relating to the project code were implemented wherever possible.

Metric	**Score**
Maintainability	2.7
Volume	5.5
Duplication	3.9
Unit Size	2.3
Unit Complexity	2.1
Unit Interfacing	2.5
Module Coupling	2.8
Component Balance	0.5

Table E.1: SIG feedback June 2nd

For the second submission, only the code which was written for this project was submitted. The score was significantly higher than the first submission (see Table E.2. Only the component balance metric was rather low. This is due to the way code is written in Unity. Game objects like a ball, bumper, flipper, etc. all need their own behaviour. In Unity, this is done with a script for each object which extends 'MonoBehaviour'. The component balance rating aims for about 9 equally sized scripts. Since the game already has more than 9 different types of game object, it is infeasible to achieve a good score for this metric.

Metric	Score	Change
Maintainability	4.4	+1.71
Volume	5.5	+0.03
Duplication	5.5	+1.63
Unit Size	4.8	+2.56
Unit Complexity	4.7	+2.67
Unit Interfacing	5.0	+2.61
Module Coupling	5.5	+2.56
Component Balance	0.5	0

Table E.2: SIG feedback June 22nd

Appendix F

Attributions

Author	Title	URL	Licence
RoganMcDougald	Metal Impact - Ceramic Piece in Sink	https://freesound.org/s/260435/	https://creativecommons.org/licenses/by/3.0/
bassmosphere	Instant Drain	https://freesound.org/s/384702/	https://creativecommons.org/publicdomain/zero/1.0/
InspectorJ	Marble, Rolling on Wood, A.wav	https://freesound.org/s/429104/	https://creativecommons.org/licenses/by/3.0/
JustInvoke	Success Jingle	https://freesound.org/s/446111/	https://creativecommons.org/licenses/by/3.0/
sonicboost.nl	Motion_Action_-Whoosh_Whistle-_MEDIUM_Dry_-48k24b.wav	https://freesound.org/s/476553/	https://creativecommons.org/licenses/by/3.0/
Michelle Souliere	Superman pinball	https://flic.kr/p/7LkhYW	https://creativecommons.org/licenses/by-nc-nd/2.0/

Alex Earl, Erik Ramfelt, Timotei Dolean and Hein Couwet	nunit-to-junit.xsl	https://github.com/jenkinsci/nunit-plugin/blob/master/src/main/resources/hudson/plugins/nunit/nunit-to-junit.xsl	https://github.com/jenkinsci/nunit-plugin/blob/master/LICENSE
Gabriel Le Breton	Unity3d docker image	https://hub.docker.com/r/gableroux/unity3d/	https://gitlab.com/gableroux/unity3d/-/blob/master/LICENSE.md
Daniel Palme	Report Generator	https://github.com/danielpalme/ReportGenerator	https://github.com/danielpalme/ReportGenerator/blob/master/LICENSE.txt
	xsltproc	http://xmlsoft.org/XSLT/xsltproc.html	https://opensource.org/licenses/mit-license.html