UNIVERSITY OF TARTU
Institute of Computer Science
Computer Science Curriculum

Kristjan Korela

Spookytown – A 3D First-Person Horror Adventure Game

Bachelor's Thesis (9 ECTS)

Supervisor: Mark Muhhin, MSc

Tartu 2022

Spookytown – A 3D First-Person Horror Adventure Game

Abstract:

This thesis describes the development process of a first-person horror adventure game called *Spookytown*. The thesis delves into detail about different game mechanics and describes the design choices behind the game's level design. The thesis also describes how some of the more complex mechanics in the game were programmed. Testing was conducted among multiple people and the feedback was used to fix major flaws and improve the gameplay experience.

Keywords:

Computer game, game design, software development, Unity, usability testing, playtesting

CERCS: P170 Computer science, numerical analysis, systems, control

Spookytown – 3D esmaisikuline õudusmäng

Lühikokkuvõte:

Lõputöö kirjeldab esmaisikulise õudusmängu nimega „Spookytown" arendusprotsessi. Lõputöö süveneb erisugustesse mängumehaanikatesse ning selgitab erinevaid valikuid mis on selle mängu levelite disaini taga. Lisaks sellele kirjeldab lõputöö kuidas keerulisemad mehaanikad selles mängus on programmeeritud. Mängu katsetati mitme inimesega ning tagasisidet kasutati suurte probleemide parandamiseks ja mängimise kogemuse paremaks tegemiseks.

Võtmesõnad:

Arvutimäng, mängudisain, tarkvaraarendus, Unity, kasutatavuse testimine, mängu testimine

CERCS: P170 Arvutiteadus, arvutusmeetodid, süsteemid, juhtimine (automaat-juhtimisteooria)

Table of Contents

1 Introduction

Making a video game from scratch is a long and complicated process. Previously in history, only large companies had the resources to make a game. Mukkamala and others [1] state that recently, however, there has been a surge in the number of games created by independent video game creators. This is all thanks to the release of publicly available game engines, such as Unity Engine. Developers no longer have to make a game engine from scratch, as now we have the tools to make games easier and more efficiently than ever.

The goal of this thesis was to create a video game using Unity. It is called *Spookytown* and it is a 3D first-person story-based adventure game with a horror theme. The game combines immersive storytelling with interesting and engaging gameplay, and its uniqueness comes from the mystery of the world the player is able to explore. The game's main goal is to be interesting, intuitive, and provide an enjoyable gameplay experience. The game development process involved coming up with a story, designing and creating the game's world, designing gameplay elements and puzzles, programming, creating models, as well as creating and finding textures and audio.

Development of the game was started in the course MTAT.03.328 Computer Graphics Project. The milestones written on the project's web page[1] describe which parts of the game were developed for the course.

Chapter 2 gives an overview of other horror games similar to *Spookytown*. Chapter 3 covers the design of the game, and provides a walkthrough and analysis of the gameplay. Chapter 4 provides an overview of how the game was implemented. It goes into detail about the game's creation process and gives a brief overview about the primary tools used in the development process – Unity and Blender. The chapter also explains how certain things were programmed. Chapter 5 covers the testing of the game. It is primarily aimed at giving an overview about what players had difficulties with during gameplay and how things were changed in the game to solve the issues and improve the gameplay experience.

The appendix explains different terms used in this thesis. It also provides a link to a repository for the source code and the build of the game. In addition to that, the appendix lists all the controls in the game and what they do, and provides the license for this thesis.

[1] https://courses.cs.ut.ee/2021/cg-pro/fall/Main/Project-WalkingSimulator

2 Similar Games

This chapter will give an overview of a few other first-person horror adventure games that share similarities with *Spookytown*. When creating a game, it is important to analyze other similar games in order to gain inspiration from them and also see how they designed certain things. Taking a look at other games lets you know what features in a game are important, how it is best to design them, and whether other people liked them or not. By looking at the players' feedback for these games, you are able to see what people want from this type of game.

2.1 *Bendy and the Ink Machine*

Bendy and the Ink Machine[2] is a horror game that takes inspiration from old cartoons for the design of its characters and environments (Illustration 1). It features combat, puzzles, and encourages the player to explore the levels. The game is a good introduction to people wanting to try out horror games, since it features a deliberately slow pace and allows the player to discover information about the game through its environment[3]. This game provided inspiration for *Spookytown* through its storytelling aspect and the fact that it rarely takes away control from the player in order to convey story elements.

Illustration 1. *Bendy and the Ink Machine*

[2] *Bendy and the Ink Machine*
https://store.steampowered.com/app/622650/Bendy_and_the_Ink_Machine/
[3] Bendy and the Ink Machine is a Great Introduction to Horror Games
https://gamerant.com/bendy-and-the-ink-machine-horror-game-introduction/

2.2 *Poppy Playtime*

Poppy Playtime[4] is a puzzle-based horror adventure game where you must explore an abandoned toy factory and uncover the truth (Illustration 2). One of the main elements of this game is a backpack that allows you to grab and move objects or use a wire to conduct electricity. Using this backpack the player must solve different types of puzzles to progress throughout the game. *Spookytown* takes inspiration from this game from its interesting level design and unique puzzles that fit into their environment. The game also features chase sequences where the player has to run away from evil characters that are pursuing them. *Spookytown* has taken inspiration from the chase sequences and also from their surprise factor. For example at one point in *Poppy Playtime*, a monster appears from a dark doorway as the player is approaching it, and begins to chase them. In *Spookytown*, as the player is approaching a door to open it, the door breaks and a pumpkin monster comes through to chase the player. *Spookytown* has also taken inspiration from the puzzle designs in this game – the battery puzzle in *Spookytown* was inspired by the part where the player has to find batteries in *Poppy Playtime*.

Illustration 2. *Poppy Playtime*

[4] *Poppy Playtime*
https://store.steampowered.com/app/1721470/Poppy_Playtime/

3 Game Design

This chapter gives an overview about the overall design of the game. It describes different design choices that were applied during development and describes the meaning behind them. This chapter also covers the most important game mechanics seen throughout the game and describes them in detail. An overview is also provided of all the design choices regarding the game's level design.

The main method of creating mechanics in *Spookytown* was iterative. Iterative design means to test newly created mechanics multiple times throughout development and make changes based on the experience [2]. Doing this lets a developer directly experience everything they designed and get a good idea of what the game feels like and what improvements can be made. Throughout development, game mechanics and level design were continuously evaluated and adjusted to make the gameplay experience smoother and more intuitive.

3.1 Game Mechanics

This chapter describes the most important mechanics that are seen throughout the game that set the foundation for the overall gameplay experience. The following subchapters go into depth about scripted events, the user interface, the dialogue system, the save system, and game world interactions.

3.1.1 Scripted events

Scripted events have become a popular method in story-based video games, because they are able to convey the story to the player without interrupting the flow of gameplay – it would be exhausting to players if cutscenes were repeatedly shown whenever the developer wishes to convey events to the player [3]. Scripted events are sequences where something happens around the player, such as NPCs moving around or interacting with the player in some way. A scripted event can be a character talking to the player and then opening a door for them, for example. Many different scripted events appear throughout the game. They usually appear when the player enters a certain area which the developer has designed a scripted event for. Illustrations 3 and 4 show one of the scripted events that appears in the game.

Illustration 3. Friendly character notices evil NPCs approaching

Illustration 4. Friendly character falling over as evil NPCs break door open

3.1.2 UI

In first-person shooter games, the user interface is used to convey information to the player, such as their health and ammo. In *Spookytown* there is no UI on the screen, because it does not have such information that needs to be displayed. Muscat and others [4] say the purpose of the lack of UI is minimizing distractions, and it helps tune the player's focus towards the game's environments and story. Doing this allows the players to better immerse themselves in the game.

3.1.3 Dialogue system

For the dialogue mechanic, a system was chosen where the NPC is the one talking and the player's character does not say anything. This approach was chosen to give the player more freedom – to let them imagine that they are speaking to the character and that they have the liberty of saying whatever they want. This helps immerse the player more in the game world and make them feel like they are in control. When dialogue is initiated, a square appears on the bottom of the screen, and inside it the sentence said by the character starts appearing letter by letter to convey the fact that the NPC is in the process of speaking. The player can press E to skip this process and immediately display the entire sentence if they wish to do so. Once a sentence has fully appeared and the player has read it, they can press the dialogue key to go to the next sentence, or end the dialogue if there are no more things for the NPC to say. Illustration 5 shows what the dialogue system looks like.

Illustration 5. A character talking to the player

3.1.4 Save system

The player is not able to save the game manually – instead, the game saves automatically at certain points at the game specified by the developer. These save points were placed strategically around the entire game to provide the best experience for players. For instance, the game is saved right before the player enters an area where they could potentially die and have to restart.

3.1.5 Game world interactions

In *Spookytown* certain objects can be picked up. In order to do that, the player has to press E while looking at the object. When an object is picked up, it floats in front of the player and follows their view, instead of having physical hands appear that hold the object. This feature assists in making the world more dynamic and immersive by allowing the player to interact with it. The second purpose of this feature is to serve as a gameplay mechanic – at certain points in the game, the player has to pick up a key to open a door, or pick up batteries to power an elevator. Illustrations 6 and 7 show a gate before and after the player uses a key to unlock it.

Illustration 6. Closed gate

Illustration 7. Gate after being opened

The game features several puzzles that the player has to solve. Puzzles should be designed to fit into the environment and make sense instead of being incongruous – in some games there might be a puzzle that requires the player to do certain moves on a chessboard to be able to progress in the game, which can make little sense and be unintuitive [5]. *Spookytown* however has puzzles that are integrated seamlessly into the environment and are designed to be intuitive for the player, such as having to drop a container onto an enemy that is preventing the player from progressing to the next area.

3.2 Level Design

This subchapter gives an overview of *Spookytown*'s gameplay and story. In addition, insight is given into the creation of the game's levels and an explanation is given for the design choices behind different aspects of the game's level design. For a better reading experience, the following design analysis is divided into four chapters.

3.2.1 Chapter 1

The game starts with the player finding themselves on a boat that is moving towards an island in the middle of an ocean. The player starts with their view looking outside the front window of the boat to introduce them to the environment they are going to enter – it is a small town that contains convoluted streets and large buildings. A screen was added to the boat that is blinking red to attract the player's attention to it. The screen reads "low on fuel" and its purpose is to let the player know that the boat has run out of fuel and that they are stranded on the island. This communicates the game's main goal to the player, which is to find fuel for their boat to escape the island. Illustration 8 shows the player looking at the town they are approaching. The boat crashes into the dock and makes a loud noise. This little detail was added instead of just making the boat stop to help further immerse the player in the game world by having small things happen around them. Once the boat has stopped, the player is able to disembark and enter the town. Immediately the player is faced with a small puzzle that teaches them one of the game mechanics – jumping. To enter the town, the player has to jump across a gap in the broken bridge.

Illustration 8. Arriving to the town in a boat

While creating the game's world, a good sense of scale was maintained. Feil and Scattergood [6] say that making everything on scale with the real word is not a good choice, since everything on the screen seems smaller – areas will feel cramped and will be difficult to navigate. This means that you have to decide the scale of the world and objects in it yourself. In addition, during the level creation process you have to constantly run the game and walk around in it to ensure that the level's scale feels realistic and is comfortable to navigate.

The starting area has multiple paths and is designed to give the player some freedom to allow them to explore the game's world. The area is designed to focus the player's attention on the main building (Illustration 9), however it cannot be entered. One of the buildings in the beginning area can be entered, and inside it is a large window that highlights a tall tower in the distance. Below the window is an NPC that the player can interact with. The NPC knows the player is looking for fuel and tells them that it can be found in the main building the player saw earlier, but the key for it is located in the tower at the edge of the town, which the player is able to see clearly through the large window that is specifically designed to give a good view of it. This is designed to let the player know in which direction they are supposed to be headed. This encounter was designed to let the player know their objective – they need to go to the tower, get the key, and open the main building to get fuel for their boat so they can leave this island.

Illustration 9. The town's main building

When the player starts exploring the town further, they will eventually come across an alleyway with a street lamp shining on a door. The layout of the level and the buildings around the player are designed in a way that leads the player towards this specific spot and the bright street lamp further assists in attracting the player towards the goal. When the player walks up to the door, a scripted event begins – the door opens and a character walks towards the player to greet them. To have a logical reason for the player to go into the building, a closed gate was added next to the door. The gate leads straight to the tower, and the NPC tells the player that they have the key for the gate inside their house and that the player should stay inside while they are looking for it.

When the player enters the building, the NPC closes the door and starts searching for the key. This is where the game turns into a horror game and the first action sequence begins. The friendly NPC sees someone coming, and holds the front door shut while telling the player to run. The front door opens, the friendly NPC gets knocked back and falls over, and the enemy NPC will enter and start chasing the player (Illustration 10). To prevent the player from exiting through the door they entered from, several evil characters were placed there to block the player. Otherwise the player could miss the door at the back of the room and wander around outside without knowing where they have to go. Before the blocking NPCs were implemented, one of the playtesters experienced this. The door at the back lets the player escape through a fairly linear corridor all the way downstairs and out of a window onto the streets, where the

enemy will not follow. This first enemy encounter is moderately basic – it is easy to understand for the player and it serves as an introduction to the next encounters that happen throughout the game, which are harder due to a faster pursuer and more complicated level layout.

Illustration 10. An evil character pursuing the player

After the player has escaped the building, they find themselves in an alleyway, and the next small puzzle to proceed to the next area is to jump on a dumpster to get over the fence. This was added to keep the player engaged, and to minimize areas where the player is not doing anything other than walking forward. After that the player finds themselves in front of a locked gate with a large padlock on it. The aim of this slightly oversized padlock is to bring the player's attention to the fact that they now have a goal – they need to find a key to open the gate. This makes the player look around and find an alleyway that leads them to a door, which allows them to enter a building.

The linear hallways lead the player to a room with windows overlooking the street they were previously on, and a table under the middle window with a key on it, which is illuminated by a light to draw the player's attention towards it. When the player approaches the key, a scripted event is triggered. The player is looking towards the key while approaching it, and the room is designed so that they also see through the windows onto the street at the same time. The scripted event consists of several NPCs walking past the building, which all look like the same NPC that previously chased the player (Illustration 11). They are carrying a coffin, and inside it is

the friendly NPC the player met earlier. The evil NPCs walk past without noticing the player through the windows. This event is intended to give a player a resolution of what happened to the friend they previously met. In addition, looking out of the window and seeing the locked gate is designed to serve as a reminder that the player needs to backtrack and go outside again with the key in order to open the gate.

Illustration 11. Finding the key and watching a scripted event outside

Although the NPCs no longer appear in this area, the scripted event was also designed to keep the player vigilant, since they are led to believe that there might be evil characters lurking in the area. A linear path is designed to lead the player to the tower they have been seeing in the distance the entire time. The tower has an elevator in it and the player has to press a button to go up. At the top there is a balcony which overlooks the entire city from high above (Illustration 12). The player is shown areas they have traversed, as well as streets and buildings that they have not seen yet. It is also shown that the entire city is fully modeled despite the player not being able to walk through the entire city. The fact that it is so detailed helps with immersion and makes it feel like it is an actual city, and not just a video game where you walk through streets and everything beyond those streets are nonexistent.

Illustration 12. Looking at the town from the tower

When the player realizes the tower has no key in it, they have to return to the elevator and press the button to go down. The elevator starts moving down but then suddenly breaks. The elevator breaking was conveyed to the player by having it stop suddenly with a strange noise, and the light in the elevator blinking rapidly several times, followed by the elevator falling down very quickly, taking the player underground.

3.2.2 Chapter 2

The entrance to the underground was designed to look ominous – warning signs and a "do not enter" tape over the entrance which the player has to jump over (Illustration 13). The ominous setting was designed to keep the player alert – they will proceed with caution while trying to avoid the evil NPCs potentially appearing. The player will then have to traverse the underground, with the aim that they need to get back on the ground and escape, since the main goal of the game was to get fuel for their boat.

Illustration 13. The underground

The game features many areas that are blocked off from the player, instead of just having rooms connected with hallways. This makes the environment feel more realistic. The level design is still linear, but not too linear – the player will come across multiple doorways and hallways, but they will have to explore to find out which one is the right choice that lets them advance in the game. The wrong paths are blocked off by various methods, such as a closed door, or a bunch of objects in the way, for instance. Basic rooms with only one entrance and one exit would give the game an artificial feel, as if it was designed by a developer trying to tell the player where to go instead of feeling like an actual location.

The player will eventually find a hallway with a door at the end. This is where the game's first jumpscare was implemented, aside from the one that happens when the evil NPC catches you. When the player opens the door, there is a giant jack-o'-lantern behind the door, pointed towards the player. In addition to that, as soon as the player opens the door and sees it, an audio clip is triggered that plays a spooky sound effect intended to scare the player. This works well as a light jumpscare and is not too intense.

When the player proceeds to walk through the corridors, they will come across a closed gate, with seemingly no way to open it. The player will have to go the other way down the corridor, and will come across a window where they can see into a different room. In the middle of the other room, there is a spotlight illuminating a jack-o'-lantern on a chair. When the player opens

the door next to the window and goes into the room, they will notice the jack-o'-lantern is no longer there and the chair is empty. This event was designed to invoke fear in the player, since fear is one of the elements of a horror game. The path leads the player to a room with a lever and a window. The closed gate that they previously came across is visible behind the window. The room is designed so that when the player pulls the lever, they are able to see the gate opening, and will immediately know that they need to backtrack to the gate and go through. The window is important, because without the visual feedback the player would have no idea what the lever does. Illustrations 14 and 15 show the lever and the gate before and after the player interacts with the lever.

Illustration 14. Lever that opens a gate

Illustration 15. Lever after interaction

The next area contains prison cells, one of which contains the friendly character that the player met previously. As the player approaches, the NPC initiates dialogue with them. He tells the player to save him, and the player can do so by using the lever next to the prison cell to open the gate. After this, the NPC exits the prison cell, and tells the player to follow him. He walks to a closed door and opens it, waiting for the player to go through. This door here is important, because if the NPC was not designed to open the way to go forward, then the player could just skip the part where they have to release the NPC from prison.

When the player enters the hallway that the NPC has opened, dialogue is initiated again and the character says that they heard something. This was made to make the player look towards the NPC so that they would not miss what happens next: a giant pumpkin falls down from the ceiling and crushes the friendly character. This is accompanied by a loud noise to make the

event more intense. The doorway is too small for the pumpkin to fit through, so the brick wall around the doorway was turned into individual bricks that the pumpkin can jump through, causing all the bricks to fall (Illustration 16). The pumpkin starts chasing the player by hopping around quickly. This is the second chase in the game, and this time it is harder than the introductory chase the player encountered earlier, since the enemy is faster and the environment is more difficult to navigate. The player has to escape through the intertwined hallways and find a hole to jump into where the pumpkin can not follow.

Illustration 16. The pumpkin after jumping through the wall

3.2.3 Chapter 3

The next area has a large closed door with a lever next to it that opens the door. In the next room there is a grate on the ceiling and when the player walks through the door, they can see several evil NPCs walking above the room, but they do not notice the player. This was added to give a reminder about the evil characters and that they are looking for the player, in order to keep the player vigilant and engaged. When the player proceeds, they will come across a door at the end of a hallway, with a lever next to it. This looks exactly the same as the door the player had to open a bit earlier, so it is familiar to the player and they know what to do. However, when the player approaches the door, the giant pumpkin jumps through the door and immediately starts chasing the player. This was intended as a jumpscare to lead into another chase sequence. The hallway is not wide enough for the player to be able to go past the

pumpkin, so the area was designed so that the player could backtrack and run around the hallways, return to the door, and go through while the pumpkin is chasing them.

The next area has a large room, and at the end of the room is a large door, with a wheel a little bit further in front of it. The wheel is red to help bring the player's attention to it, and when the player holds down E on it, the door starts to slowly open. Letting go of the wheel makes the door close again fairly quickly, but once the door is fully opened it will not close again. The distance of the wheel and the door is set in a way so that the player has to fully open the door – if they do not open it fully, then it will close before the player is able to run up to it. When the player starts opening the door however, the pumpkin will catch the player before they are able to finish. This area is cleverly designed so that the player can open the door and proceed to the next area only when the pumpkin has been eliminated.

The way to eliminate the pumpkin is done through a puzzle. In the center of the room is a crane holding a large container in the air (Illustration 17). Next to the crane is a button that lowers the container rapidly to the ground, crushing anything beneath it. The player will hopefully figure out that this is the way to eliminate the pumpkin that is chasing them. The player has to lure the pumpkin by running away from it, and pressing the button right as the pumpkin is under the container. The pumpkin gets crushed with a loud explosion and the player will be able to open the wheel door without being interrupted.

Illustration 17. The crane puzzle

The next area has another door with a wheel next to it. At this point the player knows what the wheels do and interacts with it in order to open the door. However, when the player tries to interact with this wheel, it comes off the wall and falls to the ground. This is immediately followed by a noise from the other side of the hallway where the player came from, which is designed to attract the player to look in that direction, so that they would see the pumpkin that has returned. The purpose of the wheel was to have more interactive elements in the game to keep the player engaged the entire time. The player is cornered as the pumpkin approaches them, but the floor is designed to collapse from beneath the pumpkin and it falls down into a hole.

The design of the level intends for the player to go down the same hole which the pumpkin fell into, since there is no other way to go. Since jumping straight down will lead to death by fall damage, support beams were added to the edges of the hole which allow the player to jump onto them to slowly descend. Once the player has descended into the hole, they are in a cavern surrounded by darkness, intended to invoke fear in the player. Slightly further away is a door with a wheel next to it – a familiar concept to the player by this point, designed to keep the player engaged, since it requires them to be doing something other than just walking forward. Once the player opens the door and walks forward, a noise behind them was added with the purpose of alerting them about the fact that the pumpkin is pursuing them again.

The next goal of the player is to run down the fairly linear path and not get caught by the pumpkin. At the end of the hallway is an opened door with a wheel on the other side. While earlier the player was using wheels to open doors, this time they will have to use one to close a door. The previous interactions familiarized the player with the wheel mechanic, so that now they would be able to use a wheel in a tense situation. The player has to go through the door and close it before the pumpkin catches up to them (Illustration 18). In the same room is also a closed door with another wheel that opens it. The purpose of the second door is to delay the player – the area is designed so that if the player does not close the first door to block the pumpkin, then the pumpkin catches up to them before the player is able to enter the next area.

Illustration 18. Using the wheel to close a door to block the pumpkin

3.2.4 Chapter 4

Once the player enters the next area, the door behind them closes – this was designed to prevent the player from going back and avoiding the next event: a large boulder dropping down from the ceiling. The player has to run through the narrow corridor to escape the boulder and not get crushed. The player has to run across a bridge over a chasm, where their path will be blocked by a large closed door. At this point the player is now able to step to the side to avoid being in the boulder's path. Once they do so, the boulder crushes the door and breaks through the wall at the back of the room – the boulder opens up the path for the player to go. The next area was designed to test the player's jumping ability – they have to jump over a gap onto a thin support beam, and if they miss they fall into the chasm below. If the player successfully makes the jump, they will be able to proceed into the next area.

In the next area the player has to jump down a hole. The reason for that is to prevent the player from backtracking – the hole is too deep to jump back out from. Once the player jumps into the hole, the wooden floor below them creaks – this serves as auditory feedback to immerse the player and foreshadow the floor breaking later. In front of the player are bars that prevent them from going anywhere – this area is designed to have the player watch the scripted scene unfold. Behind the bars are the evil characters seen earlier in the game. This is where the next character is introduced – a green ghost. It can be seen emerging from an explosion and starts chasing the evil characters.

The floor below the player breaks and they fall into the next area. To proceed, the player has to crouch under some rocks and go through a door into a labyrinth. A scripted event was designed so that when the player steps forward, one of the doors breaks off and the ghost comes

out and starts to chase the player (Illustration 19). The player has to find their way out of the labyrinth while running away from the ghost. At the exit of the labyrinth is a puzzle: there is an elevator that requires four batteries to function. One of the batteries is already in its slot with the purpose of showing the player what they need to find. Next to this battery are three empty slots that each require a battery. The batteries can be found inside the labyrinth. Illustration 20 shows the battery slots, with one battery on the ground to show how they look before being placed in a slot. Once the player has successfully collected all the batteries while evading the ghost, they are able to start the elevator and go up.

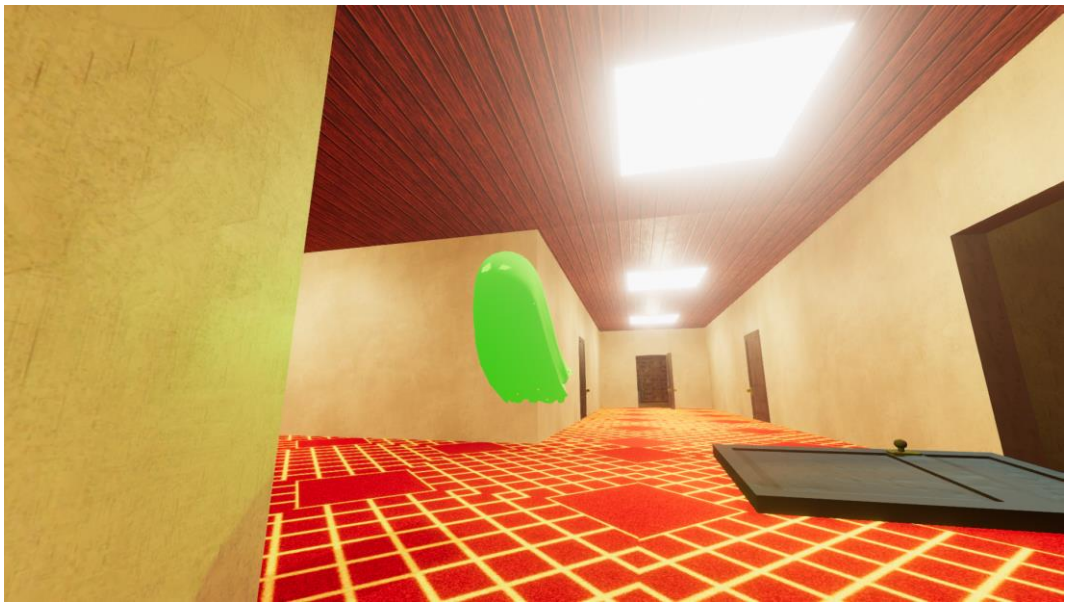

Illustration 19. The ghost pursuing the player in the labyrinth

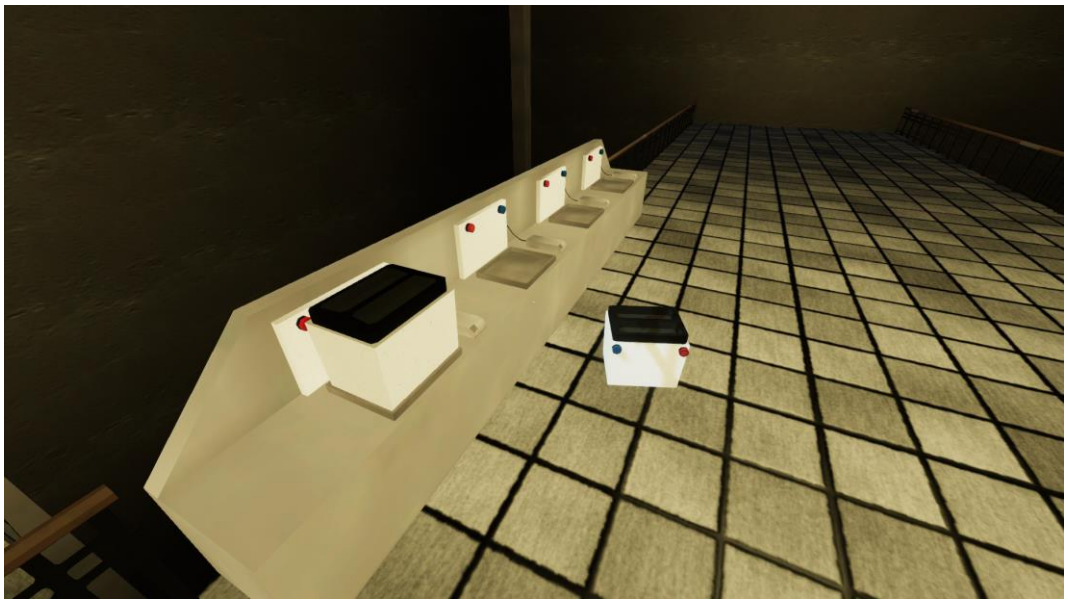

Illustration 20. The battery puzzle

Once the player exits the elevator, they have to pull a lever to open a door to keep the player engaged by interacting with the world. As the player walks into the next area, a scripted event designed to surprise the player is triggered and the ghost comes out of a vent. This is intended as one last chase sequence in the game, where the player has to run to the end of the hallway and enter an elevator. The elevator takes the player to a large house with three stories. The house is on the surface of the island and from the windows the player is able to see familiar areas that they were introduced to at the start of the game. The front door is locked and has a large padlock on it to let the player know that they need to find a key to escape. When the player explores the detailed and furnished house, they will eventually find the key in the bathroom inside the toilet. This was created with the intention that the player explores the building and carefully looks at their surroundings to appreciate the level of detail put into this world.

When the player uses the key to exit the building, they can walk to a street that they did not traverse at the beginning of the game, but from there behind fences they are able to see streets on which they previously walked. Street lamps are used here to illuminate the path the player needs to take, in order to guide the player towards the goal. On the floor the player finds the key they have been looking for the whole time – one that opens the main building in the town. The player has to jump on a pile of boxes and jump over one of the fences to get inside an alley that leads into the beginning of the game. The reason the player has to climb boxes over a fence is so that players would not be able to get over the fence from the other side, which would allow them to skip the majority of the game and immediately obtain the key. The exit of the alley is pointed at the main building, designed with the intent of driving the player towards it and reminding them that it is what they needed the key for.

The player is now able to unlock and explore the main building, which has a large main room with staircases leading up to a second level (Illustration 21). In the middle there is a staircase that leads into the basement. This area is designed to allow the player to explore the building, and in the basement they have to find their way around large boxes and find the goal of the entire game – a fuel can. This part is designed to remind the player that the entire goal of the game is to get fuel for their boat and leave this island. The player can pick up the can and go back up the staircase again. Once the player starts walking towards the main door, it is designed to close suddenly, making the player explore to find an alternative exit.

Illustration 21. The town's main building from the inside

Once the player explores the mansion and finds their way to the balcony, they have to jump over a gap to a bridge and then jump down onto the street. A bright light was added to illuminate the gap, because otherwise players would struggle to see where to go. The game ends when the player takes the gas can to the boat and places it into its slot. This activates the game's only cutscene, which shows the player leaving the island with their boat while being chased by several other boats containing characters the player met throughout the game (Illustration 22). The purpose of this was to conclude the game in a way that leaves the player thinking about what could happen next.

Illustration 22. The ending cutscene

4 Implementation

This chapter describes how the game was implemented. It gives an overview of the most important tools used during the development process and also delves into detail about how the most important game mechanics were programmed.

4.1 Unity

Spookytown was programmed using the Unity game engine[5]. Hussain and others [7] say that Unity is popular among many game developers and studios due to its flexibility and ease of use. Outside video game creation, Unity can also be used for films, architecture, construction, and engineering. It can be used to create two-dimensional, three-dimensional, virtual reality, and augmented reality games. It primarily supports scripting in C# and features a GameObject-based workflow. Something that *Spookytown* also utilizes is Unity's built-in navigation system that makes NPCs intelligent by allowing them to traverse terrain while avoiding obstacles.

The game's levels were created with ProBuilder[6], a tool designed for level creation inside Unity. Illustration 23 shows the manipulation of an edge of a mesh with ProBuilder. By creating and modifying meshes and using the different level creation tools provided by ProBuilder, the entire world of this game was created. It was then decorated with models created in Blender, which will be covered in the next subchapter.

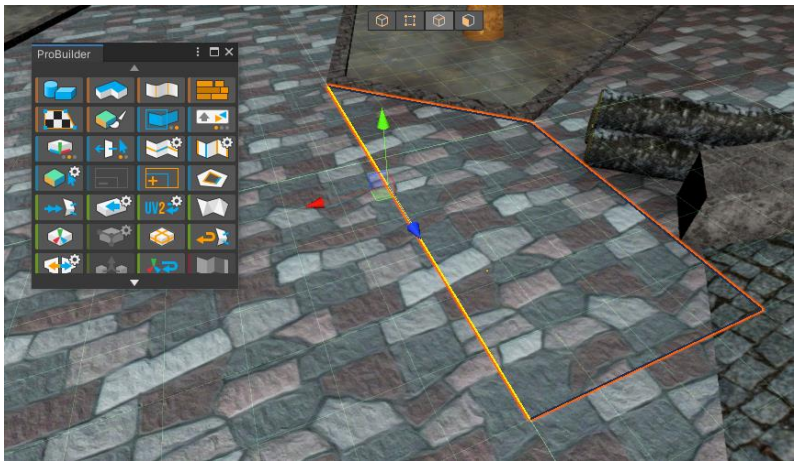

Illustration 23. Working with ProBuilder

[5] Unity
https://unity.com
[6] ProBuilder
https://unity.com/features/probuilder

4.2 Blender

All of the game's models seen in the environment were made using Blender[7]. Markham [8] describes Blender as a free and open-source application designed for making 3D models. Blender is constantly being updated and it is used for many different things, such as 2D and 3D art. It can also be used to create animations and is used in making films.

Blender was used to create the characters seen in the game, as well as all the different models that appear in the environment. Some models were created for decoration, such as trees or benches, and some were created with a purpose in mind, such as a key. Illustration 24 shows Blender's interface and one of the models created for the game – the friendly character.

Illustration 24. The friendly character inside Blender

4.3 Programming

Different programming techniques were used to create every interaction and event seen in *Spookytown*. The following paragraphs describe how different gameplay elements and mechanics seen throughout the game were programmed.

In order to make scripted events happen, the game uses triggers. Triggers are invisible volumes placed in a level by the developer, and when the player steps inside the volume, a certain event

[7] Blender
https://www.blender.org/

is triggered. For example, when the player steps near a door, there is a trigger that starts a scripted event, where an NPC walks towards the door and opens it. Illustration 25 shows what triggers look like in the Unity editor.

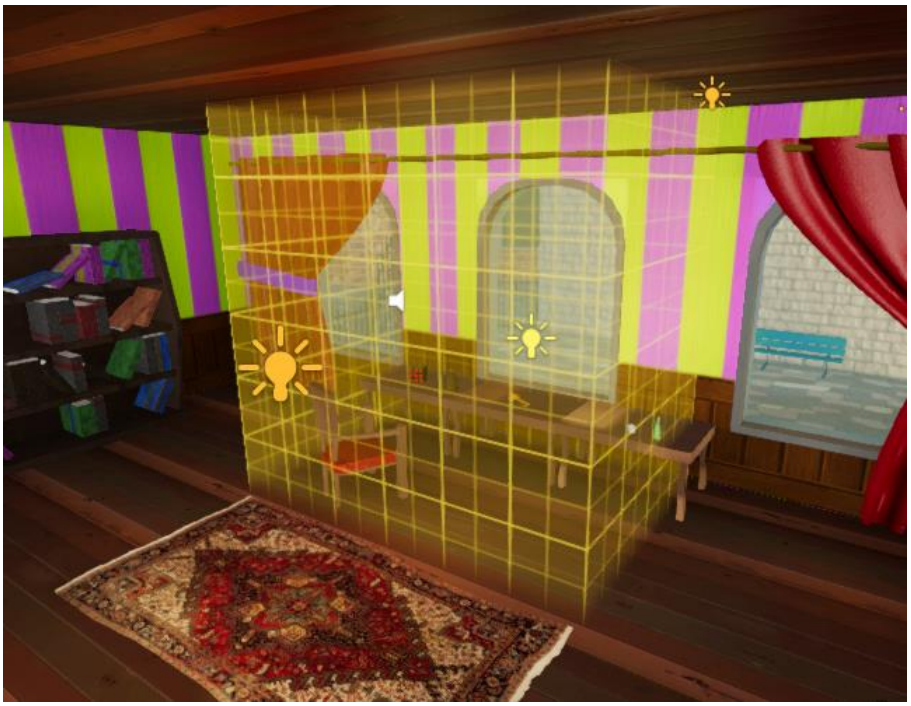

Illustration 25. A trigger volume as seen inside the Unity editor

In order to efficiently work with triggers during development, Unity Events were used. They make it possible to assign a certain task to a trigger volume, and they are useful for event-based behaviors – instead of programming everything separately, the code is modular and reusable [9]. Once the player is detected inside the trigger, the events that are assigned to it are automatically invoked. For example, in order to make a door open via a trigger, the door must be dragged to the trigger's Unity Event and the Open method must be called – when the trigger detects the player, it sends a signal to the door to make it open. Similarly to trigger volumes, buttons also invoke Unity Events. Multiple events can be added to a Unity Event – in addition to a door opening, an NPC can be made to walk towards it, for example.

Several times throughout the game the player has to use a key to unlock a door. The way it is designed is that the padlock has a trigger around it that is set to only detect the corresponding key. When the player picks up the key and puts it against the padlock, the trigger detects the key and invokes a Unity Event that removes the key and spawns a new key inside the padlock, giving the player the illusion that their key went in the padlock. The trigger also sends a signal

to open the door, with a slight delay. Illustration 26 shows how this was done using a Unity Event on a trigger.

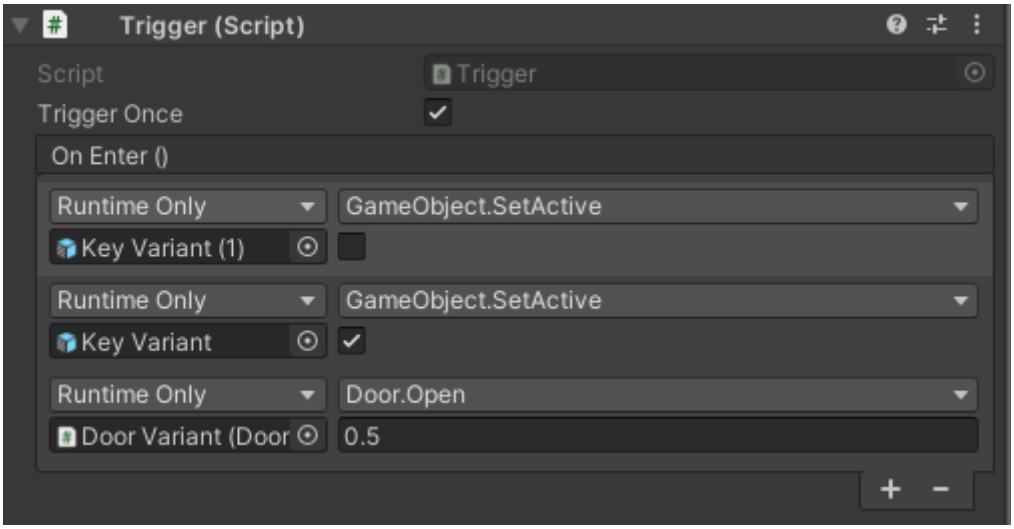

Illustration 26. A trigger with a Unity Event

The characters in the game all share the same NPC script, which another script uses to communicate with them – the NPC Action script. In order to make the characters in the game participate in scripted events, for example make them walk to a spot, an NPC Action node must be placed. The node has to be told which character it corresponds to, and then it is possible to trigger Unity Events when the NPC has reached its destination. When the NPC Action is invoked, the target character moves to the node and the events assigned to the node are triggered. An example of this is making an NPC walk into a specific spot and then initiate dialogue with the player by calling the NPC's method to start dialogue.

Similarly to NPC Action nodes, the game also has Move Path nodes. While the NPC Action nodes use Unity's navigation mesh designed for NPCs to walk along the ground, the Move Path nodes strictly make the target object go straight to the desired destination, even if it has to fly to it. Move Path nodes are used for the elevator in the tower, for example. When the player initially presses the button at the bottom, the elevator moves to the node at the top and stops. When the player presses the button again, the button tells the elevator to move to a node that is halfway down the tower, and once it is reached, a Unity Event is triggered that makes the light flicker, and after a delay the elevator moves down to the node in the basement.

The puzzles in the game also make use of the designed systems. As an example, an analysis is given of the puzzle where the player has to lower the container to crush the pumpkin chasing them. This puzzle makes use of triggers and Move Path nodes. A trigger is attached to the

underside of the container, and when it detects the pumpkin, it destroys it and activates an explosion effect at its location. The trigger is initially disabled however, since testing indicated that the pumpkin is able to jump high enough to touch the underside of the container while it is in the air, causing it to explode without the player having to drop the container. When the crane's button is pressed, it sends a signal to activate the trigger and it also sends a signal to the Move Path node on the floor under the container to make the container go to it. Once the node is reached, it sends a signal to the trigger to disable it again and a signal to the Move Path node above to make the container move back up to it. All of this happens each time the button is pressed, and the button can not be pressed again until the container has reached its initial spot, since it disables itself upon press and is enabled again only when the container's last Move Path node gives it a signal.

When designing the battery puzzle, creating only one simple script was required, and clever use of trigger volumes could be made for the rest. The created script is called Counter, which stores one integer value, and it has one method called Increment which increments the counter's value by one, and once that value reaches a set number – in this case it is three – it triggers the Unity Event attached to the script. The battery puzzle has three slots to place the batteries into, each battery fits into any slot, and once all slots are filled the elevator can be activated. Each battery slot has three triggers in it that each detect one of the three batteries. Once one of the three triggers in a single slot detects its corresponding battery, the battery and all the three triggers are removed, a new battery is spawned inside the slot, and the counter is incremented by one. The reason that each slot has three triggers is so the player is able to place any of the three batteries into any slot, instead of having each battery placed in a specific slot. Once all the three batteries are placed into a slot, the counter's value goes to three and a Unity Event sends a signal to the elevator's button to unlock it.

5 Testing

Testing of this game was conducted halfway through development at the Computer Graphics Projects Expo 2022 as well as near the end of development among multiple volunteers, and all of that provided valuable insights on how to improve the game. Sometimes there might be issues in the game that feel natural to the developer due to having played the game so many times already. Testing allows the developer to see things from a different perspective, since it is not always clear to them where mistakes and flaws in their game might be.

5.1 Feedback

At the end of development, testing of the game was conducted on five people. After playing the game, the testers had to answer questions by providing a rating on a scale of one to five. The purpose of these questions were to assess whether the development goals of the game were met.

The testers were asked to rate their overall gameplay experience (Illustration 27). The rating one stood for terrible, and five stood for excellent. The majority of the testers believed that there were no flaws in the gameplay and that it kept them interested and engaged the entire time. One of the testers found the gameplay to be good, but believed that it could be improved even further.

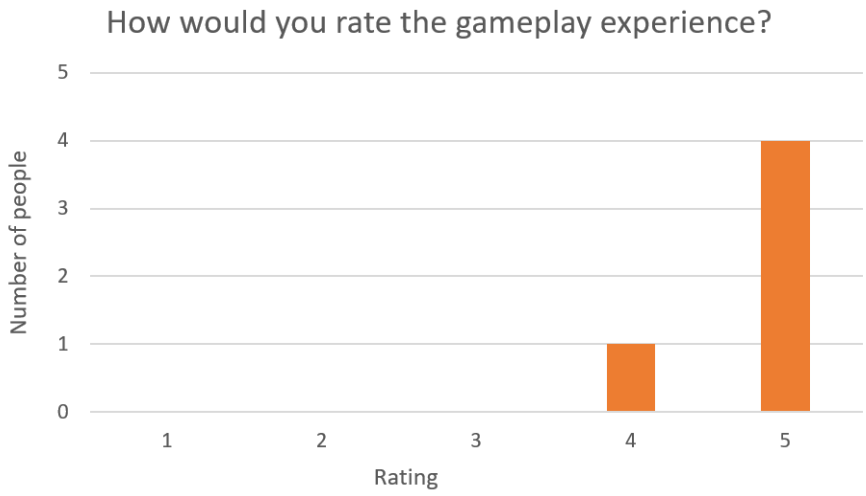

Illustration 27. Gameplay experience feedback

The testers were also asked to rate the difficulty of the game (Illustration 28). The rating one stood for very easy, and five stood for very hard. Most of the people found the game's difficulty

to be medium and easy, while one tester thought that it was a bit too hard. The tester described some jumping segments and the labyrinth as parts that they had difficulties with during playtesting.

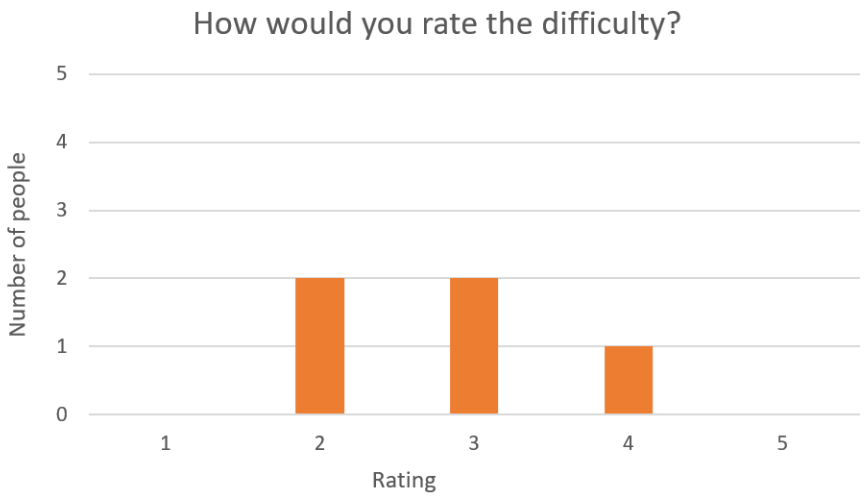

Illustration 28. Game's difficulty feedback

Furthermore, the testers were asked how intuitive the game was for them (Illustration 29). The majority of the testers found the game to be very intuitive and that their goals were always easy to understand. Two of the testers believed that the intuitiveness could be improved, but still agreed that most of the time they knew what the game intended for them to do.

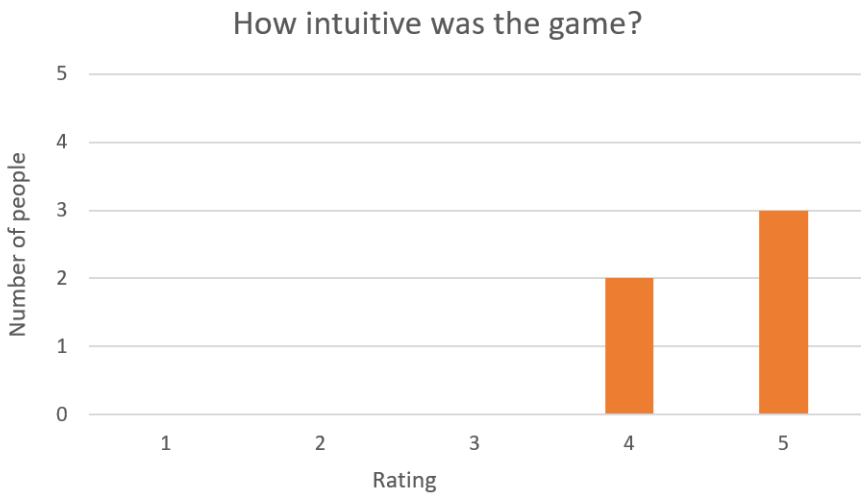

Illustration 29. Game's intuitiveness feedback

5.2 Improvements

The testers were also asked to give general feedback on different elements in the game that they were not satisfied with. Some players also discovered bugs that appeared in specific circumstances which had not been thought of or experienced. The feedback was used to fix major issues and improve the game to provide a better gameplay experience for everyone. The following subchapters cover the major problems players had during testing in the exposition that were fixed by the time of the final testing among five people.

Something all the testers pointed out is that the game was too dark – despite it being a horror game and being thematically dark, sometimes it was so dark it hindered the gameplay experience and progression. For example, it was hard to notice a door at the end of a hallway, and players did not think to press E to open the door because they could not see it, so they were unable to progress further without assistance. Fortunately, this problem was easy to solve and only required the addition of a light above the door so players could see clearly where they need to go.

Another issue that came up in testing was the fact that players could abuse mistakes in the game's level design. For example, in the part with a locked gate that the player had to find a key for, there was a bench next to the gate. Players could jump on top of the bench and then jump over the gate to skip a portion of the gameplay. This problem was easily fixed by moving the bench further away from the gate to make the jump impossible.

Something the testers pointed out, and which they could be seen struggling with during testing, is the fact that it was difficult to pick up objects. Some objects, like a key, had to be picked up in order to advance further into the game. Due to the design of the mechanic of picking up objects, the object the player wanted to pick up had to be in the center of their view. That was difficult to do since players struggled to see exactly where the middle of their screen was, and often had to press the button multiple times to pick something up. This issue was solved by adding a small crosshair to the center of the screen – by being one pixel in size, it was not obstructive and did not hinder the gameplay experience, and instead proved to be a very useful tool to help pick up objects without issue.

6 Conclusion

As a result of the thesis, a first-person horror adventure game called *Spookytown* was created using the Unity game engine. The main goal of this game was to provide an interesting and engaging gameplay experience that would be intuitive to players.

During the development of the game, the story and different game mechanics were designed, such as the scripted events, the dialogue system, and the puzzles. The 3D modeling software Blender was used to create models that are seen throughout the game. The world itself was created using ProBuilder. The C# programming language was used to program everything inside Unity. Throughout development the gameplay experience was designed to be engaging and intuitive.

The game was playtested halfway through development in an exposition by multiple people and also near the end of development among five people. The purpose of testing was to assess whether the set goals of the game were met. The testing results proved to be positive and revealed that the testers enjoyed the gameplay experience and that it was mostly intuitive and not too difficult or too easy. The testers' feedback was used to improve the game and solve major issues that players experienced.

Designing and developing an entire game proved to be an educational experience. I have never developed a game project of this size before. I was able to apply my knowledge in game development, and I learnt a lot from this experience. During the entire development process I was able to enhance my skills and knowledge in every single aspect of video game design, and also in Unity and programming.

References

[1] Mukkamala J, Srinivas S, Surendra T, Rao R, Chowdary R. A Study on Game Development Using Unity Engine. AIP Conference Proceedings 2375, 040001. 2021.

[2] Salen K, Zimmerman E. Rules of Play - Game Design Fundamentals. England, London: Massachusetts Institute of Technology. 2004.

[3] Rogers S. Level Up! The Guide To Great Video Game Design. United Kingdom, Chichester: John Wiley & Sons, Ltd. 2010.

[4] Muscat A, Goddard W, Duckworth J, Holopainen J. First-Person Walkers: Understanding the Walker Experience through Four Design Themes. Australia, Melbourne: RMIT University. 2016.

[5] Schell J. The Art of Game Design: A Book of Lenses. United States of America, Burlington: Morgan Kaufmann Publishers. 2008.

[6] Feil J, Scattergood M. Beginning Game Level Design. United States of America, Boston: Thomson Course Technology PTR. 2005.

[7] Hussain A, Shakeel H, Hussain F, Uddin N, Ghouri T. Unity Game Development Engine: A Technical Survey. Pakistan, Sindh: University of Sindh Journal of Information and Communication Technology. 2020.

[8] Markham R. What is Blender, and Is It Right for You? https://www.schoolofmotion.com/blog/what-is-blender-and-is-it-right-for-you (05.05.2022)

[9] French J. Events & Delegates in Unity. 2021. https://gamedevbeginner.com/events-and-delegates-in-unity/ (05.05.2022)

Appendix

I. Glossary

1. NPC – non-player character, a character in a game that is not controlled by the player[8].
2. Scripted event – a sequence where something happens around the player, such as NPCs moving around or interacting with the player in some way.
3. UI – user interface, screens and visual elements that allow a user to interact with a product or service[9].
4. Gameplay – the way that a computer game is played[10].
5. Game design – the act of designing and creating rules and structures that result in an experience for players. [2]
6. Game mechanic – a rule that guides the player's actions in a game[11].
7. Level design – the phase of game development that deals with creating the stages of the game, such as obstacles and elements that can be discovered[12].
8. Mesh – a structural build of a 3D model that consists of polygons[13].

[8] https://en.wikipedia.org/wiki/Non-player_character
[9] https://www.usertesting.com/blog/ui-vs-ux
[10] https://www.collinsdictionary.com/dictionary/english/gameplay
[11] https://en.wikipedia.org/wiki/Game_mechanics
[12] https://www.masterclass.com/articles/how-to-become-a-video-game-level-designer
[13] https://www.techtarget.com/whatis/definition/3D-mesh

II. Project Repository

The build of *Spookytown* can be found at:

https://drive.google.com/file/d/1Z_fWSMorde_FpoAlJSUitIb6gSTLdigY/view?usp=sharing

The source code can be found at:

https://drive.google.com/file/d/15-zA9_W7xb7lBP08YRQx3b9z8PHj5TtA/view?usp=sharing

III. User guide

Steps to launch the game:

1. Extract “Thesis Game Build.zip”
2. Open the folder “Thesis Game Build”
3. Open the folder “Build”
4. Run “Thesis Game.exe”

Controls:

Mouse	Look around
W	Move forwards
S	Move backwards
A	Move left
D	Move right
Space	Jump
C	Start crouching Press it again to stop crouching
E	Interact with the object you are looking at Can be used to interact with certain characters, advance dialogue, open certain doors, use buttons and levers, or pick up certain objects. In order to interact with a wheel, the button must be held down.
Escape	Pause the game

IV. Accompanying Files

- Thesis Game Build.zip – contains the build of *Spookytown*.
- /Build – contains the executable file “Thesis Game.exe” that can be used to run the game.

V. License

Non-exclusive licence to reproduce thesis and make thesis public

I, **Kristjan Korela**,

(*author's name*)

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright,

   Spookytown – A 3D First-Person Horror Adventure Game,

   (*title of thesis*)

   supervised by Mark Muhhin.

   (*supervisor's name*)

2. I grant the University of Tartu a permit to make the work specified in p. 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons licence CC BY NC ND 3.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.
3. I am aware of the fact that the author retains the rights specified in p. 1 and 2.
4. I certify that granting the non-exclusive licence does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Kristjan Korela

05/05/2022