Bachelor Project

Czech
Technical
University
in Prague

F3

Faculty of Electrical Engineering
Department of Computer Graphics and Interaction

3D logic game

Sára Veselá

Supervisor: Ing. Ladislav Čmolík, Ph.D.
May 2023

BACHELOR‘S THESIS ASSIGNMENT

I. Personal and study details

Student's name:	**Veselá Sára**	Personal ID number:	**499046**
Faculty / Institute:	**Faculty of Electrical Engineering**		
Department / Institute:	**Department of Computer Graphics and Interaction**		
Study program:	**Open Informatics**		
Specialisation:	**Computer Games and Graphics**		

II. Bachelor’s thesis details

Bachelor’s thesis title in English:

3D logic game

Bachelor’s thesis title in Czech:

3D logická hra

Guidelines:

Get familiar with principles of game design. Analyze game principles used in 3D logic games, especially in logic games where the player is changing the game environment. Further, get familiar with modular components and with their role in the design of game levels. Based on the analysis, create design document for 3D logic game in which the player is changing the game environment. Create modular component, from which you will compose the game levels, their materials, and textures. Allow loading and generation of the game levels from description in a file. In accordance with your design document, create at least five game levels. Test your game using a qualitative test with at least six players.

Bibliography / sources:

1) R. Koster. Theory of Fun for Game Design, 2nd edition, O'Reilly Media, 2013.
2) J. Schell. The Art of Game Design: A book of lenses. CRC Press, 2008.
3) B. L. Mitchell. Game Design Essentials, John Wiley & Sons, 2012.
4) S. Rogers. Level up! the Guide to Great Video Game Design, John Wiley & Sons, 2014.
5) E. De Nucci and A. Kramarzewski. Practical Game Design: Learn the art of game design through applicable skills and cutting-edge insights, Packt Publishing, 2018.

Name and workplace of bachelor’s thesis supervisor:

Ing. Ladislav Čmolík, Ph.D. Department of Computer Graphics and Interaction

Name and workplace of second bachelor’s thesis supervisor or consultant:

Date of bachelor’s thesis assignment: **16.02.2023** Deadline for bachelor thesis submission: **26.05.2023**

Assignment valid until: **22.09.2024**

Ing. Ladislav Čmolík, Ph.D. Supervisor’s signature	Head of department’s signature	prof. Mgr. Petr Páta, Ph.D. Dean’s signature

III. Assignment receipt

The student acknowledges that the bachelor’s thesis is an individual work. The student must produce her thesis without the assistance of others, with the exception of provided consultations. Within the bachelor’s thesis, the author must state the names of consultants and include a list of references.

.	
Date of assignment receipt	Student’s signature

Acknowledgements

I would like to thank Ing. Ladislavu Čmolíkovi, Ph.D., for his guidance, patience, willingness to help, and valuable advice. I am also grateful to my boyfriend who supported me throughout the whole period of writing this thesis. Special thanks to the testers who voluntarily helped me with the development, provided new points of view, and discovered many bugs.

Declaration

I hereby declare I have written this Bachelor thesis independently and quoted all the sources of information used in accordance with methodological instructions on ethical principles for writing an academic thesis.

V Praze 26. May 2023

Podpis:................................

Abstract

This thesis analyzes the game design and the game principles of 3D logic computer games, particularly logic games that alter the game environment. Based on this analysis, I designed 3D logic game from a first-person perspective and the design and implementation of the game is included in this thesis. In addition, the creation of modular components, models, and their materials was described. The game was implemented in Unity and then qualitatively tested with six players.

Keywords: 3D logic game, Unity, game design, modular components, level generation

Supervisor: Ing. Ladislav Čmolík, Ph.D.

Abstrakt

Tato práce se zabývá analýzou herních principů 3D logických počítačových her. Zejména logických her, které mění herní prostředí. Na základě této analýzy byla navržena 3D logická hra z pohledu první osoby a její design a popis implementace je obsažen v této práci. Pro hru byly vytvořeny modulární komponenty, modely a jejich materiály pro implementování výsledné hry v Unity. Hra obsahuje originální návrh logických hádanek, které byly následně ve hře kvalitativně otestovány s šesti hráči.

Klíčová slova: 3D logická hra, Unity, herní design, modulární komponenty, generování levelů

Contents

Figures

Chapter 1

Introduction

In the last decade, the trend of computer games has become very widespread, mainly due to the world's digitalization. People play computer games mainly for fun in their free time or to make money from computer games, either through esports, streaming, or by creating lets plays. On the other hand, there are also educational games, and their main goal is to teach the player something. Thus, many genres are emerging, and games are no longer of interest only to teenagers but to all ages.

On the opposite side of the players are the developers who create the games. Developing a video game involves many aspects, including game design, creating graphics, game world and story, programming logic, and more. In game development, it is impossible to concentrate exclusively on one of these elements because they all influence the player's experience.

This thesis provides a view of game design from the perspective of logic games. It is divided into two parts, theoretical and practical.

The theoretical part consists of the analysis of game design and principles of logic games. Logic games are one of the classic genres also appearing in non-computer games.

The first chapter focuses on the game design itself. I will ask questions such as what a game is, what the essential elements of a computer game are, and what they mean to the player. It is followed by research on multiple topics in the second chapter, called Game principles in logic games, where I will analyze a specific genre: logic games. I will clarify its subgenres, how to create puzzles and levels, and above all, I will describe the already created 3D puzzle games.

The practical part is based on the previous chapter's research results and describes the design and implementation of a game. It is a 3D first-person puzzle game with original puzzles based on changing the game environment. A more detailed specification is described in Chapter 4, Design of the game, which explains where the game will take place, what the story will be, passing the levels, mentioning controls and game elements. In addition to the game description, I will look at the 3D models and modular components used in the game.

In the following Chapter 5, Implementation, I briefly describe the logic implementation in Unity [UT]. I will describe the movement of the player, the first-person camera, and creating levels. I also give a specific puzzle example and explain how to solve it. In Chapter 6, Testing is represented by tests and the results. The game will be tested in two rounds with several volunteers to improve the development and get feedback. These tests should lead to changes in game logic to better suit the player's requirements and bring the necessary experience.

This thesis aims to develop a first-person 3D logic video game in Unity step by step on my own. Furthermore, I will also create visuals and models that will be used within the game. Nevertheless, before I begin, let us look at the research done in preparation for developing the game.

Chapter 2
Game design analysis

Game design at its core means deciding about what a game should be. But in larger scope it is more than that. Many sources state that game design is an art form that is mixed with technical and functional design. Being a game designer is not just about making ideas up and creating the whole game in head, it is about understanding players as human beings. It is important to acknowledge the psychological mechanisms that govern human mind, as well as cultural aspects of people. Learning about these angles of view, helps game designers to gain insights and put themselves into player's place. Mentioned aspects and many more are considered as game studies. However, game studies are not study of video games. Only they encompass also different types of gaming such as board games or card games.

As mentioned, game design is not only concerned with aesthetics and the perception of players, but also with the technical and functional sides, which are the content of field of game theory. Game theory studies the mathematical models of optimizing strategies among agents. Game-theoretic models analyse conflict situations and use calculations to find the best possible strategies for specific participants. Game theory is also applied in many areas such as science, economics, sociology, or biology.

In this chapter I will look at the definition of a game. After that, I will focus on player's experience and how to make a good and unique game. And finally, I will describe individual elements of game design and how they are connected together.

2.1 What is a game

Game starts with a designer, her/his ideas and motivation. From the start there is arising game design document which is the first part of process of game design. On the other hand, many of the decisions made in the first part of creating a game, can be changed in the process, or can't be answered at the beginning at all. Throughout designing a game is the game designed document amended and overwritten several times, so many versions are produced. The first version of document is followed by the actual implementation and coding the game where are developed game mechanics, technology, and visuals. Here also comes the interface which is the crucial component between the game and the player. And then there is the point where game ends, by the user. That is the reason why the whole game is created. To be a fun and engaging activity for user.

Despite that, game should meet some basics to be a game at all. Game should have goals, conflict, rules, challenge and should be interactive and closed formal system. Also, could be won and lost and could create their own internal value. The last but not least games should engage players. [Sch08]

Many of these points could be covered up as a problem-solving activity, which is in cognitive psychology attempt of achieving a goal by overcoming obstacles and finding a solution to the problem. While solving a problem in a game, there starts a process in player's mind. First of all, it is necessary to identify the problem and shape it. Then player starts to come up with possible solutions and tries to line them up. After that s/he chooses one solution and confirms the result after s/he applies it if s/he achieved the goal. [FF95] Reaching the goal brings pleasure and excitement to the player and for many of them it is obviously the motivation or reason, why they play videogames. The four main reasons of playing videogames are; fantasy factor, escape and entertainment, competition and challenge and socio-emotional activation factor. [GVI18] The reason to play videogame is also affected by what player expects from the game and which genre s/he chooses to play.

While playing, solving problems, and going through the story of the game, if the game has one, player gains experience. The game experience has every player little bit different, but they have something in common.

2.2 Game experience

You never truly know the experience of players because everyone senses the interaction with game different and interprets it from his view. You should think about game in overview. The game design is not about game designers and what they like, but what could be generally fun and full of enjoyment for the players. Game designers are trying to create an experience; however, game is not the experience it just enables experience. [Sch08]

There is difference in experiences when it comes to single game session or repeated gaming. It also depends on player's previous experience with gaming if s/he is incidental player or an experienced gamer. As mentioned, socio-emotional factor could be reason, why to play games, and it also influences the experience. If player's social life is closely tied to the game, the experience is likely to mean more to her/him. Overall, the experience of the game is mainly connected with emotions, which are most often associated with achieving the goals of the game. [TJ08] That's why one of the most important things about designing a game is knowing which essential experience therefor emotions, you want to pass on the player.

2.3 Elements of game design

The elements of game design could be many different parts of the game but if I try to split them I can create categories which are mechanics, story/world, aesthetic, and technology. These create so called mental tetrad. [Sch08]

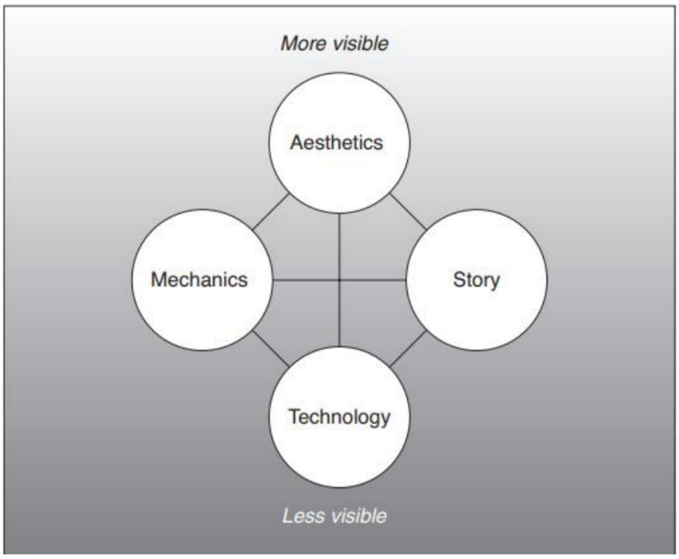


Figure 2.1: Mental tetrad from element of game [Sch08]

Mechanics make bridge between player's actions and goal of the game. What lead and what don't to complete the main challenges of the game. Mechanics are also very specific for games unlike other types of entertainment. Game mechanics can also be perceived as set of rules and game's response to player's actions. The more various mechanics in game are the more complex the game is. Game designers have to choose which mechanics and how much of them are best for certain game and this also influence the choice of technology, aesthetics and story to make it believable for the player.

Technology is on the other hand the way how to mediate the mechanics and aesthetics. Technologies are all materials and interactions that are feasible in the game. They define what you can and cannot do in the game environment. You give the player some freedom, but at the same time you can guide him in his actions so that too many choices do not demotivate and frustrate him.

Story is probably the best natural method how to keep player on the way to the goal of the game. It is an account of incidents and events which lead to some end. Not every game has a story, but always is there something what player tries to achieve, for example best score or solving a puzzle. Story should be supported with appropriate aesthetic so that it creates right atmosphere to the player.

Aesthetic contains visuals, music and everything that affects the player's senses. This element is the most visible one and has the most direct relationship to player. It creates the atmosphere, underlines the story, makes the game world in which are mechanics and technologies to interact with.

All these elements define the theme, goals, interactions, rules and the environment of the game. They affect each other but also complement. For every game genre is crucial to have all four elements well specified. In next chapter I will take a closer look at logic games, their elements and designs.

Chapter 3

Principles in logic games

Logic games also known as puzzle games are evidently about solving puzzles and testing player's cognitive skills. Due to the fact that many players like the sense of accomplishment that comes from solving a mystery, riddle or other type of problem, puzzle games are a significant part of the game industry. Many other genres often implement some puzzle elements in gameplay to decrease the pace of the game or to make the game special or interesting thanks to this activity.

Logic videogames are not only about testing player's cognitive skills but can also improve player's cognitive abilities. According to research from Adam C. Oei and Michael D. Patterson, after playing physic-based puzzle game [Zep11] for 20 hours, players have significantly better higher-order executive function skills, which are cognitive skills to coordinate thought, organize task to direct toward obtaining goals and also after starting one of the tasks to stay focused on them. [OP14]

Like Cut the rope game, most puzzle videogames are 2D and do not tell a complex story. They are single-player and the conflict stands between player and the system, in most cases, not between two players. Typical examples are puzzle mobile games that have become a trend due to their accessibility and simplicity. They are mostly built on simple levels that slowly get harder and during a gameplay are added new elements. Individual levels can be solved in few minutes, for example while waiting for a bus.

My game heavily relies on puzzles, thus it is important to understand how to design the best puzzles that are favortite among the player. In this chapter I will look at designing these levels, principles of creating puzzles and subgenres that have arisen due to different types of puzzles. In the last third I describe few famous 3D logic games, because in second part of this thesis I am developing game based on these characteristics.

3.1 Puzzle principles

Creating a good puzzle is harder than one think. Game designers deal with the problem of dominant strategy for puzzles and their replay-ability. Once the player finds a dominant strategy for the puzzle, it is not fun for him anymore. It is necessary to come up with an idea how to make the puzzle dynamic unless you want the player to find the dominant strategy. One of the options could be playing against other player or adding some elements that change the dynamic while replaying the same puzzle.

Moreover, there are some described puzzle principles I can discuss in this paragraph. A good puzzle has an easily understandable goal. When you look at it, you know what the final state should be. Player is then interested and starts to think about possible solutions. When he comes up with some ideas, it should be easy for him to start. The first step should not be too complex with many conditions. Player should have the option to try easy solutions that might be wrong but could help in further brainstorming. While solving it is great to show a progress player has made. One of the alternatives is to create some smaller puzzles below the main one or make the main puzzle decomposable. An appropriate example is Rubic's cube, where player gradually adds colours to one side until it is finished and continuous with next side. With each side becomes the puzzle also more difficult. It is important to increase the difficulty, so that player does not get bored. On the other had you do not want him to become frustrated, so you should give him a sense of solvability. [Sch08]

There are also options how to design a sequence of puzzles because you should avoid making the player feel stuck. One way is parallelism, which gives the player option of solving another puzzle when he fails in one to make progress. This could be supported by pyramid structure of puzzles. You create it with smaller puzzles that give us clues to some final puzzle.

Probably the most important aspect of designing puzzles is to create ideal environment for the player so that s/he does not feel frustrated, demotivated, or lost.

3.2 Puzzle subgenres

There are subgenres of puzzle games, which are divided according to the type of puzzles. One of the first logic videogames is Breakout developed by Atari, which laid the foundation for the subgenre called ball-and-paddle. Player controls block called paddle and tries to bounce ball towards bricks, that disappear after hitting them.

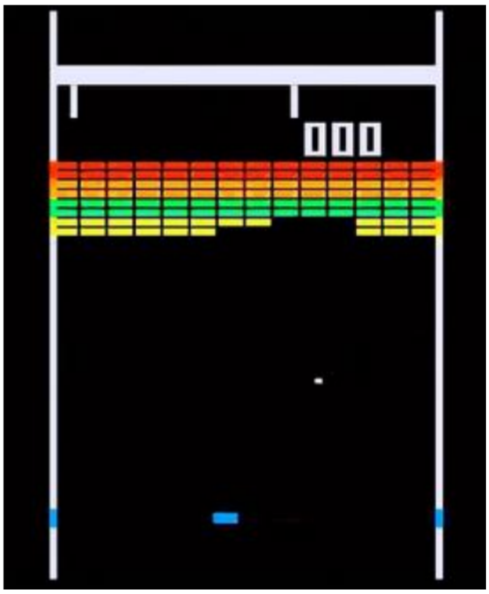


Figure 3.1: Atari Breakout [Spe]

Other subgenre are games based on environmental puzzles using physics of the game world. Best examples are probably Portal [Val07] and The Talos principle [Cro14]. They use knowledge of real-world physics and there is no need to explain complicated rules and principles of the game to the player.

The subgenre that is most typical for mobile puzzle games are tile-matching games, where player need to create a sequence of identical shapes to make them disappear according to certain rules. A clear representative is the mobile game Candy Crush Saga [Kin12], which belong to the match-tree games.

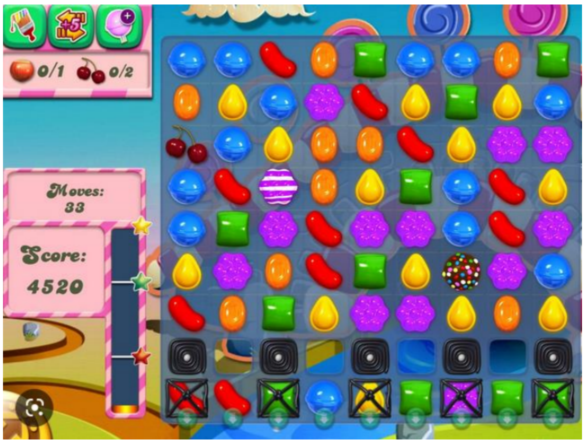


Figure 3.2: Candy Crush Saga [Kin12]

Some mobile puzzle games are just a reworking of traditional puzzles into digital form such as Sudoku, chess, etc.

There are also logic games with adventure elements, which are based on exploring the surroundings. The player usually has to understand several different mechanisms and the games are not designed with typical levels. Puzzles are more naturally in the game environment and are a medium for story and adventure. One such game is called Myst [Inc21], which combines adventure and logic in an original and very sophisticated way.

Other subgenre which is associated with exploring are escape room videogames where the player is usually locked in a room or building from which they must escape by solving individual puzzles. Well-known games of this character include The Witness [The16], where the player starts inside a bunker and The Room [Gam14] here the player should more pick up items, put them together or analyse many objects to solve the puzzles.

The subgenre that is probably the biggest trend in modern logic games are cooperative puzzle games. They can be divided into two basic types. In one, both players are together and also two players are logically needed to solve the puzzle, which is usually based on physics of the environment. Examples include Portal 2 [Val11] in co-op mode or Human Fall Flat [Sak16]. The second type are games where one player describes puzzle procedure and gives instructions to the other player who is trying to solve it. These include We Were Here [Gam17] or Keep Talking and Nobody Explodes [Gam15].

Most of these games contain consecutive levels of puzzles that get more difficult as the gameplay and player's skills progress.

3.3 Level design

Level design has two directions. Firstly, I will focus on possibilities how levels can follow each other and then on design of the levels themselves.

3.3.1 Alleys and Islands

Since the game I am aiming for is in 3D, I will be describing two types of three-dimensional video game-level design: alleys and islands, which differ in their layout in space. [Rog10] Alleys are built based on direct gameplay, so the player has a clearly defined path and goal, which, thanks to the levels, can achieve in a predetermined order. The biggest advantage for designers is that they know when and how the player will enter the level and how he will move in the level. They have more options to create scripted or triggered gameplay.

On the other hand, the island type gives rather more options to the player, to whom the designers have to adapt. The player can choose the order they want to experience the gameplay or go through levels. It awakens in players the desire to explore the world beyond levels. This, in turn, allows designers to create secret levels or objectives. Both ways have their benefits and disadvantages. Their use depends on the level goals and how you want to guide the player through the game. In Figure 3.3 you can see the island level design from game Myst, which has already been mentioned in Chapter 3.

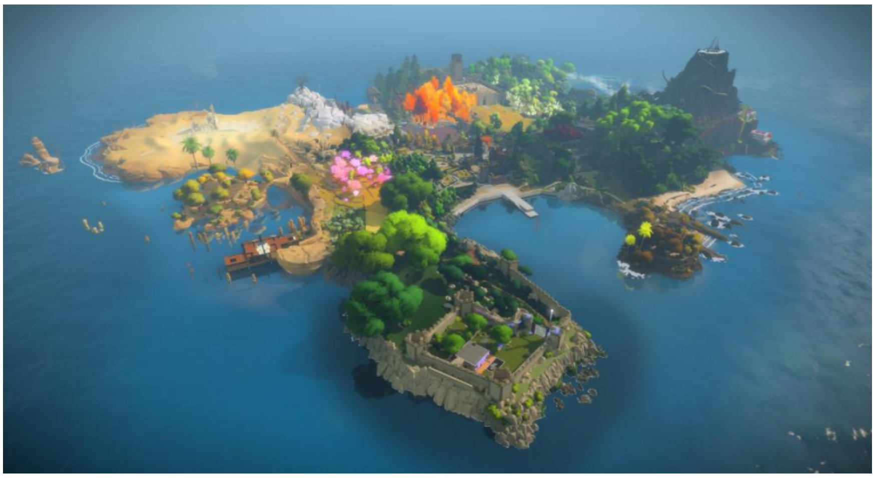


Figure 3.3: Island design in Myst [Inc21]

3.3.2 Elements of level design

However, the levels are more than just guiding movement of the player through the game. There are many aspects that game-level designers must take into account. The first thing that needs to be considered is the goal or purpose of the level. Should the player escape from the level, survive something, teach some new abilities, or get a moral lesson.

The main level elements are location, aesthetics, gameplay, objectives, story beat, new abilities, enemies, mechanics, NPCs, bonuses, and could also be many others. [Rog10] In the following paragraphs, I will discuss these elements in detail.

Location and level aesthetic go hand in hand because they create the essential atmosphere and theme together. You can often reuse assets or models by re-texturing or re-coloring them through levels. Different colors of the environment are used to differentiate each part of the game. Therefore, you have the possibility to use the same models across the levels but with different texture or color. These models are called modular components and they give us the ability to seemingly change the environment with only small details, so the flow of the visuals stays the same. They are also great if they are repeated several times within the levels and you can divide a seemingly complex model into simpler parts, which you then just repeat. For example, instead of modelling a whole room, you just create parts such as blank wall, a wall with window or door and then place them side by side to form a room. The game thus creates a sense of uniform but not dull environment.

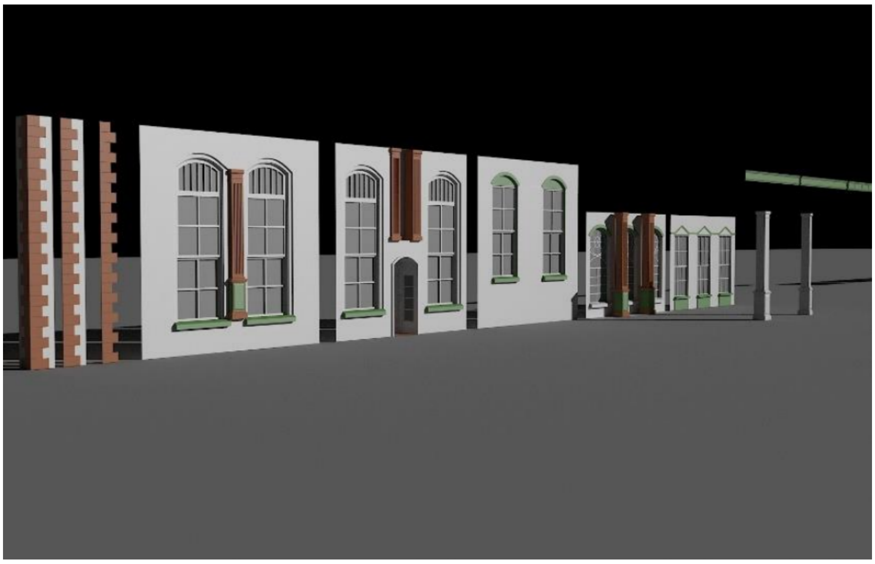


Figure 3.4: Example of modular components [Tho]

The size of the space is also an important aspect. The player feels uncomfortable in too large spaces over which he may not have as much control. In puzzle levels, it can also be a problem that the level does not fit on the screen in its entirety, and the player constantly has to move from place to place to solve the puzzle.

The level should feel like a real place as much as possible. Wherever a player can go, he should be able to go there. Places, where the player cannot get to, must be marked better than just with an invisible wall. Ideally, the player should learn that, for example, a fence or a stone wall is the imaginary boundary of the location in which he is. On the other hand, you are not constrained by realism in game design. Instead, you have to focus on making sure that something does not feel strange and unnatural to the player.

As far as the gameplay in logic games is concerned, it is rather straightforward and is, therefore, about solving logical puzzles. In other level-based games, there may be signs of fighting, shooting or jumping to the goal or exit. Logic games can also combine such elements in their levels, but the main focus of the gameplay should be solving a puzzle.

The objective is the goal or reward to get to using the gameplay and mechanics in the level. As I mentioned before, a clear and predetermined goal is vital to motivate the player and reassure him that his path leads to some rewards and that his efforts are not in vain.

A side goal of a level may be to advance the game's story. There are several ways to convey the story to the player. These include cutscenes, dialogs, direct and indirect gameplay, narration, or environmental storytelling. Cutscenes, dialogs, and narration are more of a passive way to show the story. The dialogs could be created interactively, but the gameplay is more interactive, where the player can change the story with his actions in the direct case. In the indirect gameplay, the world around him changes, or the change stands on NPCs. [KDN18] Nevertheless, if the level has no story at all, then in any case, it advances the player's story by collecting experience and possibly new abilities. After a player gets a new ability, it is important to let him try the ability with impunity and see how it works or guide him to use it correctly. New mechanics in the level may also need to be introduced sufficiently.

Game or level mechanics are a controversial topic in game design, because of its unclear definition. However, the most common definition is that mechanic is something that connects the player's actions to the goal of the game. A more general definition might be that mechanics are interaction of the player with the game. [Sic08] So, by the term mechanics I mean for example grabbing, jumping, shooting, hooking etc. The mechanics, in general, should be the right amount in the level so that it is simple enough, and it is, of course, essential in logical games to build levels according to mechanics to be progressively more difficult. On the other hand, enemies and NPC are not so crucial in levels of puzzle games because there are not the basis of this game genre and they are not even used.

3.3.3 Process of level design

The level-based games that are created in the game industry are very different, Therefore, their creation may also differ, but I can describe the general steps of level production. [KDN18] First, you create the premise, which is defined as elements of the level, such as the aesthetics, the goal, and possibly the story of the level. Then you create the sketch. The sketch describes a more detailed gameplay, the objectives of the level and the timeline. The level should be designed in detail before it is implemented in the game, which is the purpose of the game design document, which I will describe in the next chapter. Then a basic playable prototype could be created using grayboxing. Grayboxing is an opportunity to try out our proposal, test it and then modify it thanks to feedback. In this step, you also create modular components and basic models. You define yet undefined level elements such as new abilities, enemies, NPCs and other things that need to be placed.

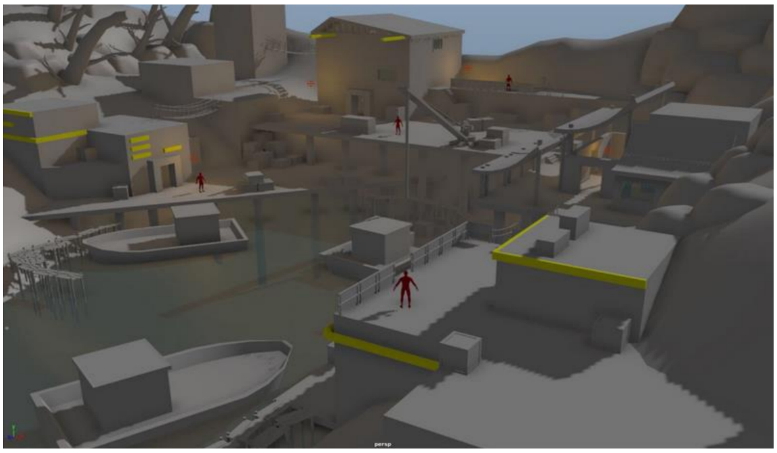


Figure 3.5: Example of level grayboxing [KDN18]

The final polish contains detailing the models, creating their materials and textures, fixing bugs, and making gameplay smooth.

Having discussed how to design level, I can see parallels with the game design as described in Chapter 2. Levels can sometimes be seen as little games within a game. I will discuss what level-based games look like and the characteristics of levels in these games in the following subchapter, where I focus on 3D games because of the specification of the topic of this thesis.

3.4 3D logic games

The game developer should search the gaming market to ensure that the game s/he plans to create has not previously been made. If not, it is wise to examine games that are comparable, draw inspiration from them and be sure to avoid their mistakes.

In Chapter 3 about subgenres of puzzle games, I mentioned some examples of puzzle games, but many of them are 2D. The most famous 3D games in this genre include Portal, The Talos Principle, and Human Fall Flat. In the following paragraphs, I will discuss them in more detail.

3.4.1 Portal

Portal was published by Valve in 2007. [Val07] It is a first-person logic game where the player goes through a series of test chambers representing the puzzles.

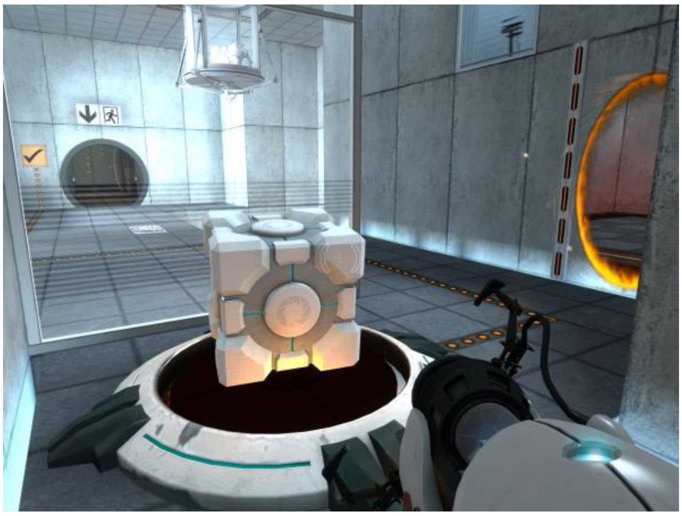


Figure 3.6: Portal [Val07]

The game's main mechanic is laying the portals with a so-called portal gun, which creates physically connected portals. The player could place just two portals on flat planes in the room with a puzzle and teleport himself or other objects through the portals. The teleport does not change the speed and direction of the object that entry the portal on one side.

In 2011 was released a sequel called Portal 2 [Val11] brought the prevalent cooperative mode. Portal uses the environment and physics rules and adds its mechanics to it, making it natural and intuitive for the player. The game is divided into levels where each level implements a new mechanic, for example laser bridges, turrets and more. Level contains several sub-levels, which are in the common part of the map and the players passes them linearly in a row. They also gradually escalate their difficulty.

3.4.2 The Talos Principle

The Talos Principle published by Croteam in 2014 [Cro14] is another first-person puzzle game with the essence of philosophical science fiction. The player plays as a robot with human mindset and explores the world around him. The levels are built on the principles of islands and the player can choose the order in which he will go through them. The player solves puzzles that involve exploring enclosed spaces and overcoming difficulties there in order to gather so-called sigils.

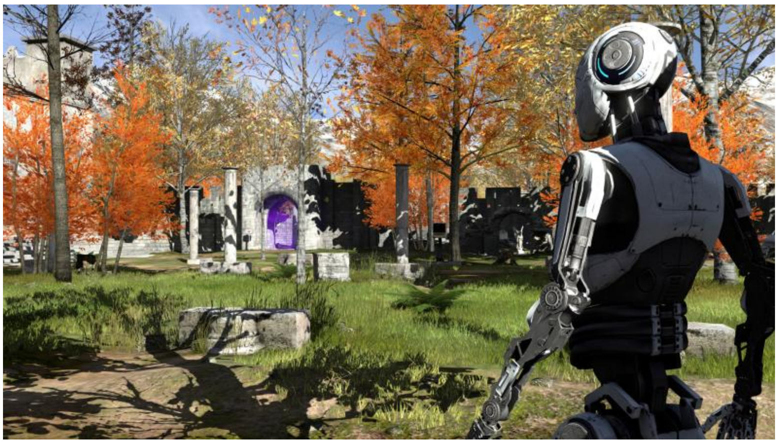


Figure 3.7: The Talos Principles [Cro14]

Certain puzzles require the player to use portable refactors to turn on light-based switches, and other puzzles put the player in danger. The robot ought to show intelligence by solving puzzles and free will by refusing to follow instructions. Depending on the player's choices, the game can have several endings.

3.4.3 Human Fall Flat

Human Fall Flat developed by Tomas Sakalauskas and released in 2016 [Sak16] is a third person puzzle game. The player plays as a simple male character named Bod, who solves puzzle levels built naturally behind each other in open space. The game uses only the environment and simple physics to create levels. The character has no special abilities and the player may control him to use both arms, look up with his head, thanks to grabbing climb-up ledges or interact with objects in the environment. The game does not introduce many new mechanics in walkthrough. The levels are separated from each other mainly by aesthetics and linearly increasing complexity, which is created by the fact that the levels get bigger and the player needs to do simple actions in right order to get to solve the puzzle.

Figure 3.8: Human Fall Flat [Sak16]

Each level has a distinctive theme and offers multiple solutions of the puzzle. The player can learn how to play the game and eventually solve the problems by finding several remote controls concealed throughout the game.

3.5 Summary

To sum up, in this chapter I described the games that were the biggest inspiration for the game, which is the result of the practical part of this thesis.

In general, puzzle games all share the same characteristic. They begin with simple levels and later they use more challenging tasks to keep the player interested and motivated into the game. The games I showed are based on interaction with the environment and, for the most part, touch on themes around robots, testing and artificial intelligence, which will also play a role in the game being created. On the other hand, every game should be unique and original. Even though logic games have a certain pattern that has proven to be effective and workable, it does not mean that stepping out of it or changing it will lead to a bad game. However, it is important to understand what the game developer wants to target and should decide what original element to have in their game accordingly.

I describe my game in more detail in the following chapter, which is a description of game design document.

Chapter 4

Design of the game

The game design document describes the details of the game and is written by the game designers. It should contain everything that the programmers need to implement in the game. Game design documents may be made of text, images, sketches, or diagrams.

It specifies the central theme, goal, game world, story, models, controls, characters but also user interface. The document defines every level in detail as well as the attributes of characters or enemies. The game balance should also be accurately designed in this phase. Understandably, the final balancing is after game testing, which is almost the last part of a game development, but it is crucial to know the elements and factors of the game that need to be adjusted. It would help changing them until the whole balanced game generates the required experience you want players to have, which is one of the most important things I mentioned in Chapter 2.

In this chapter I will describe the game design document which was created for the game.

4.1 Game overview

It is a 3D logic game from a first-person view. The player must pass levels by himself to complete the main puzzle between him and the exit. The playthrough is linear, meaning the player must solve the puzzle in the room to access the next one.

The path to the exit consists of three levels that test the player's various skills and try to raise questions about where s/he is, why s/he is here, and, above all, why s/he has to pass the levels. The player may come across hints that answer these questions during the game. Nevertheless, what really stands behind the walls of this maze...

4.2 Game mechanics

The main purpose of the game is to entertain the player by solving the puzzles. These are based on using the environment, one of the subgenres of puzzles described in Chapter 3. Each puzzle is one level, and the task is simple, to open the door and get to the next room. Each room is full of cubes on the walls and floor. On the floor are generally cubes that can be moved, and on the wall are those that can be interacted with. The player can build a path to the exit by moving the floor cubes. In order to move a floor cube upwards, the player has to press the interactive cubes in the way shown in Figure 4.1.

The game is 3D, and the player can move the floor cube from the bottom up gradually on the z-axis. If you imagine the room in the Cartesian coordinate system and the floor cube is the point [2, 3, 0], then you have to activate the cubes [2, 0, 1] and [0, 3, 1]. This moves the moving cube to [2, 3, 1]. When you add one to the z-axis of the wall cubes, you move the floor cube up again. The player must not stand on the cube when moving it; otherwise, it will not move up. This prevents the player from taking advantage of the cube movement to get up faster. At the same time, the wall cube cannot be activated from an infinite distance. The player has to reach the cube. This distance is set to 1 meter to match reality and not be unnatural for the player. It is also not possible to activate a cube in a jump, so a player can activate at most one cube above him/her. When standing on the ground, the maximum cube height s/he can reach is three on the z-axis.

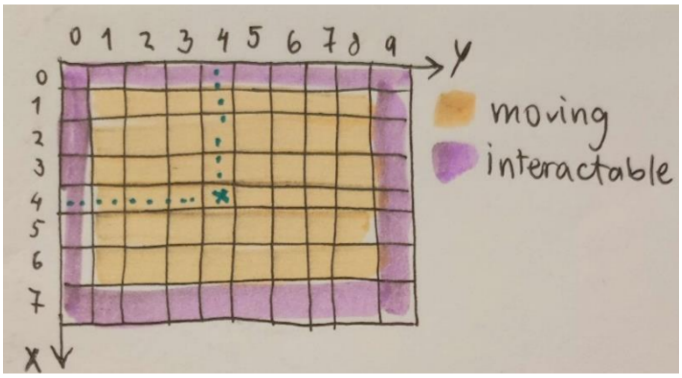


Figure 4.1: Example of mechanic in 2D

Every wall cube can hold only one floor cube. If the player tries to break this rule, the first held cube is automatically deactivated. If the player activates cubes [2, 0, 2] and [0, 3, 2], the floor cube move to [2, 3, 2], then this cube is held by four cubes: [2, 0, 2], [0, 3, 2], [2, 0, 1] and [0, 3, 1] to prevent the player from using them again. These activated cubes are marked with different colors to show they are already used.

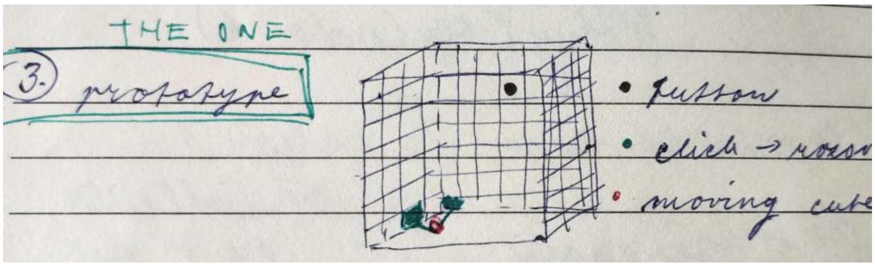


Figure 4.2: Example of mechanic in 3D

Another mechanic is the cube, which deactivates itself after 7 seconds. In the game, it is called an unstable cube and has a different color.

The following special cube is the elevator cube, which is not activated similarly. The player needs to stand on it and press it, and the cube will automatically go up and then down.

4.3 Game world

The game occurs in indoor spaces stylized with a futuristic, even laboratory and sci-fi feel. At the same time, the environment tries to be minimalistic without unnecessary details.

The puzzle rooms are laid out linearly behind each other, thus following a type of the 3D-level design is called the alley.

4.3.1 Game Story

The player is a subject in the simulation but does not know about it. Artificial intelligence tests this subject and its intelligence in 3D environmental puzzles. The player tries to get through the simulation and unlock the way home.

4.3.2 Controls

The player needs to use W, A, S, and D or arrow key commands to move and can use the mouse to modify which direction the player is facing when walking. By pressing Space player can jump and by pressing Left Control crouch. The interaction with cubes is done by mouse click. The left click activates the wall cube, and the right click deactivates it.

4.3.3 UI

Menu should be simple and clear for players. The pointer is locked in the screen's center unless the menu is opened. After then, it is unlocked, and the pointer no longer causes the camera to move. This makes it possible to select a menu item. When the player hits the Escape key, the menu appears. Also, the game is stopped, and the player cannot use or rotate the camera. There are two types of menus: The Main menu and the Pause menu. Upon opening the game, the player is taken to the Main menu where s/he can start playing or exit the game. The Pause menu can be entered into the game using the Escape button. The player can return to the game by pressing the Resume button on the menu. The last save is loaded by pressing the Reload button. The save was done when entering a new room. This can be helpful if the player gets stuck or wishes to start the puzzle from the beginning. The player can also access simple help with fundamental information about the cube's movement and types. The last button in the menu is the Menu button, which enters the Main menu.

4.3.4 Game saving

Saving the game is done using checkpoints located at the entrance to the puzzle room, so the game saves right after the player steps into a new room. However, the player is unable to save independently and overwrites the save. It might make the solution irreversible so that the player would not be able to reload the room and start again.

4.3.5 Models

The base model of the whole game is cubes. These are distinguished only by the colors that indicate the property of the cube. The levels will be built using modular components of walls, floors, and ceilings. Point light placed on the ceiling will produce the lighting in the rooms. Automatic doors will connect the levels. The door can be opened after pressing the button, which is always located somewhere on the wall. Whether the door opens or not will be indicated by a colored button next to the door, which will glow either red for closed or green for passage. There are occasionally short corridors between levels to connect levels of different sizes. In these corridors are often panels on the wall explaining the game's mechanics.

4.4 Level Design

There are three sections in the game, which get linearly harder. Each section contains from five to six levels. Based on the analysis of other games, the goal was to gradually increase the game's complexity. Because of this, the first section primarily serves as an introduction to the game's concepts and tests if the player understands them well. The following two sections then take advantage of the player's acquired knowledge and expand on it with additional smaller concepts.

The analysis also shows that the player seeks a dominant strategy. Therefore, it is important to introduce a new element of the game in each section that will change the dominant strategy and the passage through the puzzles.

The player begins in the Initial room, the gateway to all further levels. After completing the section, the player obtains a Puzzle stone and is teleported to this room again. The Initial room also supports a sense of progress because it contains all the stones that are currently gained.

In the following section, I discuss each section, describe the additional features they add to the game, and give examples of the puzzle rooms.

4.4.1 First section

The first section is about the introduction of the player to the mechanics. In the Initial room, the player learns the game controls. Each essential game element is explained to the players by panels. Since the game's basic mechanics may be complicated for some players, it is explained in two steps in rooms 1.1 and 1.2. I introduce the player to the mechanics in 2D first so that the cubes are not higher than one in z-axis. Then the player moves to a level where s/he is working more in 3D and tries to put the cube to height two.

This important tutorial follows the first proper puzzle in level 1.3 in Figure 4.3, where the player tries out the mechanic on higher walls. In the three first levels, I also teach the player the sequence in which s/he has to go through the puzzle. In order to open the door to the next room, the player has to press a button located somewhere in the room.

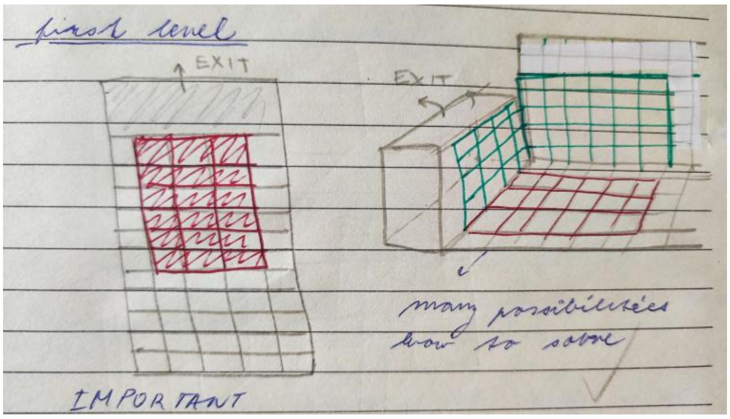


Figure 4.3: Prototype of room 1.3

Puzzles 1.4 and 1.5 are connected. The player has to get to the button, which opens the door and moves the floor with all the moving cubes upwards, showing the player a new room puzzle and allowing him to climb up again to get to the mentioned door. The button is located on the opposite wall where there are no cubes. The player must activate the cube as close to the button as possible. The player must reverse the path. From the highest moving cube to the lowest one, since the highest one cannot be activated from the ground and must build a path to it. At the same time, s/he must be careful not to deactivate cubes that already lead to the button. Then s/he needs to press the button. This also opens the door where the player still has to get to. The solution to this part is very similar to 1.3, but the interact cubes are on the opposite wall.

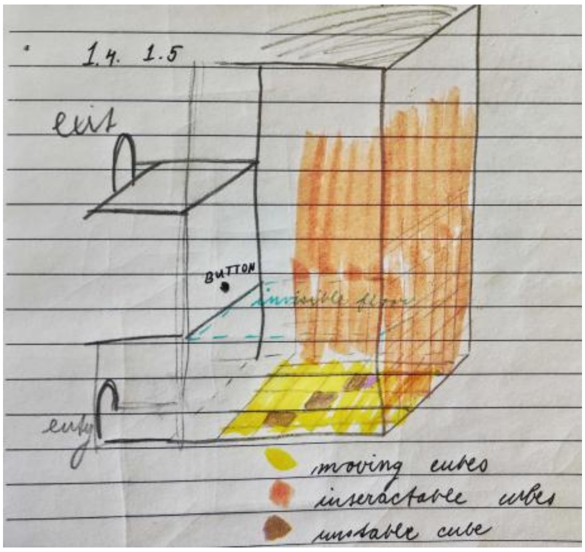


Figure 4.4: Prototype of room 1.4

The last room, 1.6, teaches the player how to work with a new type of unstable cube. S/He has to use it to solve a complex puzzle because since it is a simple mechanic to understand, there is no need to make extra room for learning. The player has to press the button again and then figure out the way to the door. The unstable cube pushes players to think ahead and activate them last in order to follow the created path to the goal.

Any knowledge gained in this section is then considered obvious in the following levels.

4.4.2 Second section

In the second section, right at the beginning, I introduce a new cube that acts like an elevator when activated. The player can try it out in the first level 2.1. The player needs to use the elevator cube to press the button because it is too high, and the player cannot get to it otherwise. The room is higher than in the previous section, so the elevator needs to be used even when going to the door. At the same time, moving cubes are only in some places on the floor. The player has to activate the highest possible cube and then use the elevator to get to it without building a path.

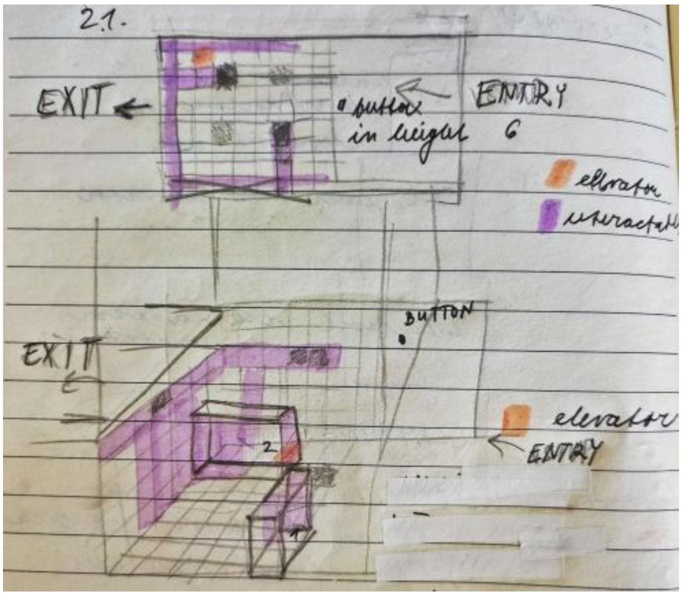


Figure 4.5: Prototype of room 2.1

The next room combines then the elevator and an unstable cube. It has the same logic as the previous level. Only it forces the player to work with the cubes quickly. S/He can only get to the door again by activating the highest possible unstable cube and letting the elevator take her/him out, but the player has to make it before the unstable cube falls down.

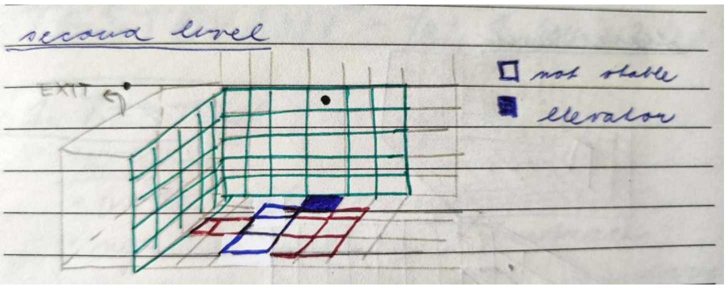


Figure 4.6: Prototype of room 2.2

The last part is again the climbing part, which contains three subparts. The room contains three buttons that, when pressed, move the floor up so it creates a new subpart, and the player can progress further. Only the last button opens the door. This section combines all types of cubes and forces players to think more in 3D because the buttons are placed on opposite walls. The floor is not full of moving cubes, so the player has to go a certain way and has less freedom. This makes the puzzles more challenging and tests whether the player understands all the rules and mechanics.

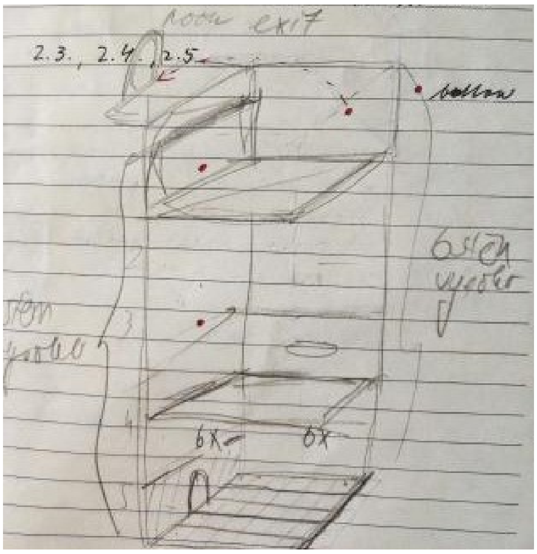


Figure 4.7: Prototype of climbing part 2.3 - 2.5

4.4.3 Third section

The last section builds on all the knowledge gained through the game. The player should already be very familiar with the moving of cubes environment. In the first room, 3.1, the ceiling has been replaced by the floor, so the cube is moving from the top. The logic of the moving cube from above and below is exactly the same, so it is perfect for introducing the concept that the cube just does not have to be on the floor. It contains only basic moving cubes so it is simpler for the player.

Similarly, the other two levels have cubes moving out of the wall. They both look the same as in Figure 4.8. They have a different placement of the floor. The player must figure out how it works and how to find his way to the button and exit.

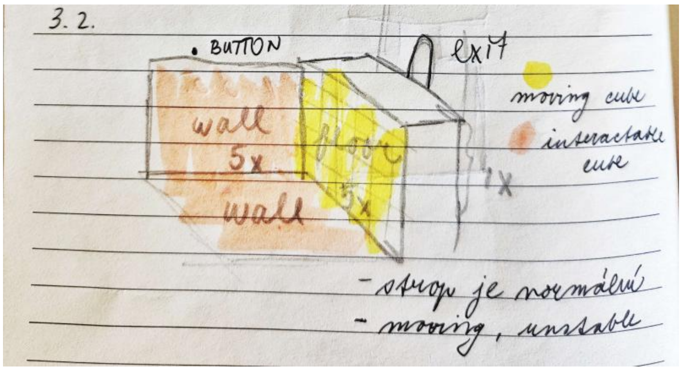


Figure 4.8: Prototype of room 3.2

The last part is again the climbing section, where the cubes come only from the walls, not the floor. This changes logic quite a bit and is a refresher for the player without adding a new cube type. Floors with moving cubes alternate against each other on the walls. The player must always figure out a path from one side to the other. This is also aided by the elevator, which travels horizontally and not vertically because it is also on the wall. The player can then also activate the cubes on the other side and work with them. The button again moves the bottom floor, allowing the player to climb further. The last button opens the door.

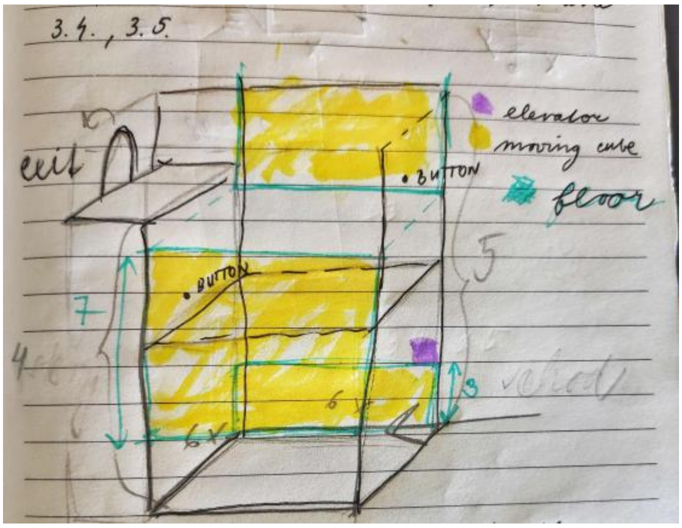


Figure 4.9: Prototype of room 3.4

All the features and mechanics of the game were then implemented in Unity. I will describe the implementation of each issue in detail in the next chapter.

Chapter 5
Implementation

After defining many aspects of the game and writing all ideas on paper, I can begin to implement the game itself. In this chapter, I will focus on the implementation using Unity Engine[UT], version 2020.1.10f, but in addition, I will also use Blender[ble], version 2.90.1, to create models for my game. I used Blender and Unity because I have previous experience, and many tutorials are available, and both software is free.

5.1 Modular Components

As mentioned in Chapter 3, level design benefits from using modular components. It can significantly simplify and speed up the overall process, and also, in terms of efficiency, modular components preserve consistency and saves memory.

Without having to texture and build each component separately, modular components are merged and utilized to create a variety of rooms and levels. A list of all components and crucial features that create the environment is also required. It is better to start with a base unit from which we may create variations later than create detailed prefabs. To do that, you can give a single asset more uses. However, it is essential to accessorize the environment by adding a few components to break up the repetition and improve the appearance. The last component type is a custom object, which can occasionally be divided into more parts that can be used again.

In this section, I specify which modular components I determined were required to create the puzzle game described in Chapter 4.

5.1.1 Cubes

The cubes are the basic elements of the game, and their creation was simple in terms of modeling when it is a primary shape. As already mentioned, based on the type of cube, the color of the cube varies. The material can be seen in Figure 5.2. The moving cube is grey, the interactive is blue, the unstable is red, and the lift is green.

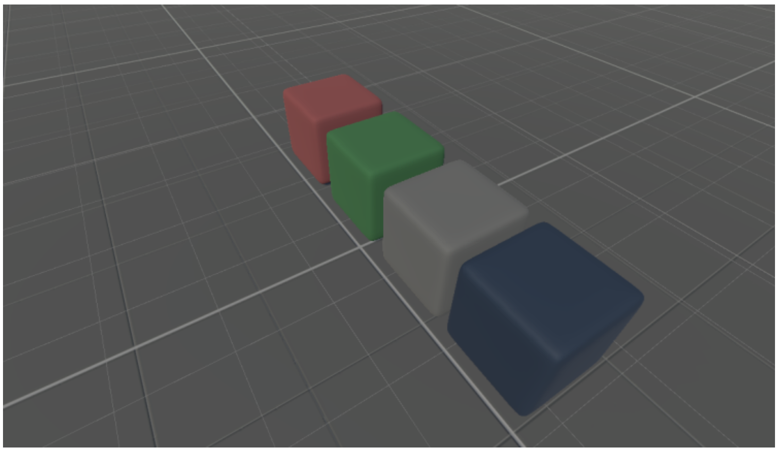
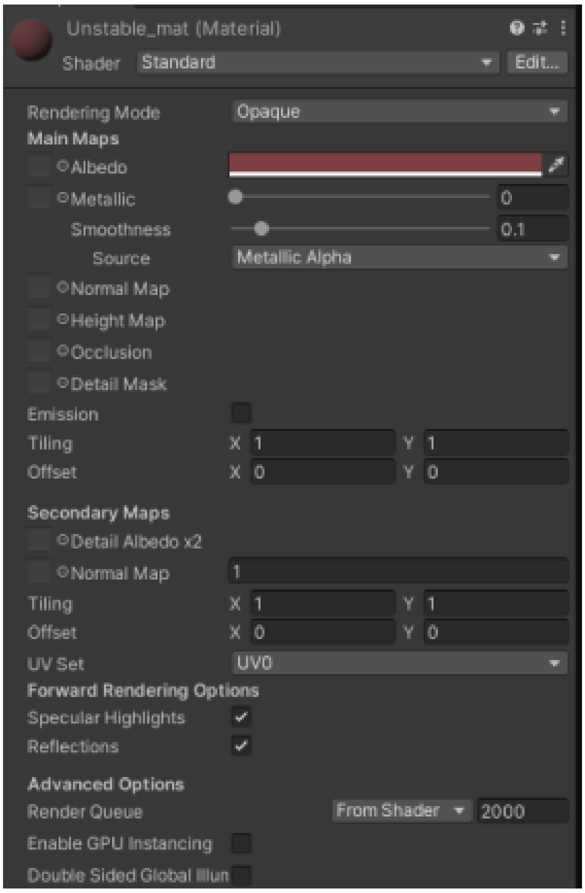


Figure 5.1: Models of cubes



Unstable_mat (Material)

Shader Standard Edit...

Rendering Mode Opaque

Main Maps

Albedo

Metallic 0

Smoothness 0.1

Source Metallic Alpha

Normal Map

Height Map

Occlusion

Detail Mask

Emission □

Tiling X 1 Y 1

Offset X 0 Y 0

Secondary Maps

Detail Albedo x2

Normal Map 1

Tiling X 1 Y 1

Offset X 0 Y 0

UV Set UV0

Forward Rendering Options

Specular Highlights ✓

Reflections ✓

Advanced Options

Render Queue From Shader 2000

Enable GPU Instancing □

Double Sided Global Illun □

Figure 5.2: Material of cubes

5.1.2 Environment

Because the game is set inside rooms, one of the most crucial modular components is the walls and ceiling. The simplest type of wall is a plain block, and the ceiling and floor as a plain plane. I used some materials and models as modular components from the Unity Asset Store package [Ass]. You can see them in Figure 5.3.

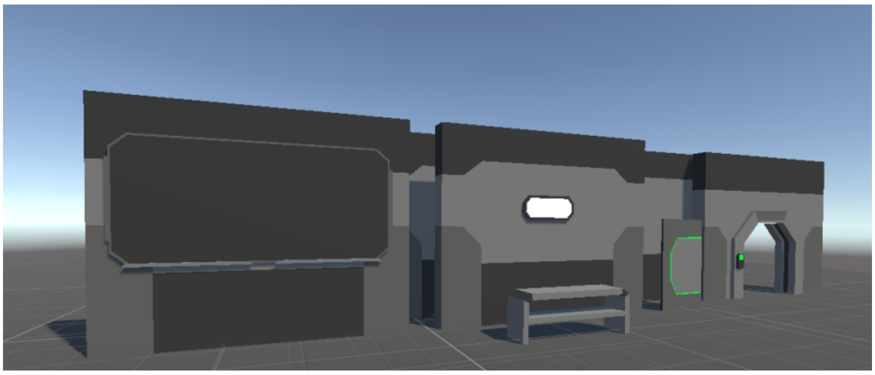


Figure 5.3: Models from Unity Asset Store

As I mentioned before, only planes were used for the floor and ceiling, where material was added to the ceiling and texture was added to the floor. The texture had a grid on it to assist the player in solving puzzles and make the rooms more clear.

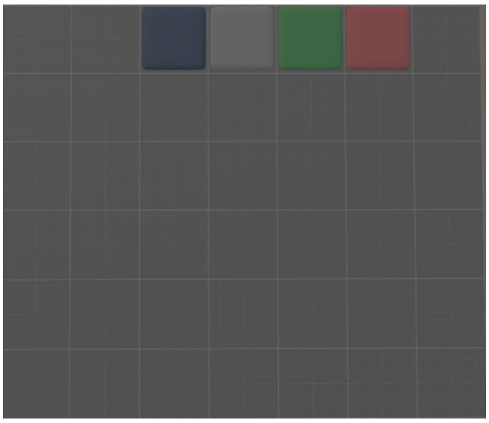


Figure 5.4: Floor as a grid for cubes

I used models of the panel, lamps, and table to distinguish the uniform environment. At the same time, it was not possible to use something like this often because the walls are primarily full of interactable cubes.

On top of that, I created automatic doors, buttons, and Puzzle stones that were color coded. The first level is blue, the second purple, and the third yellow, as shown in Figure 5.5.

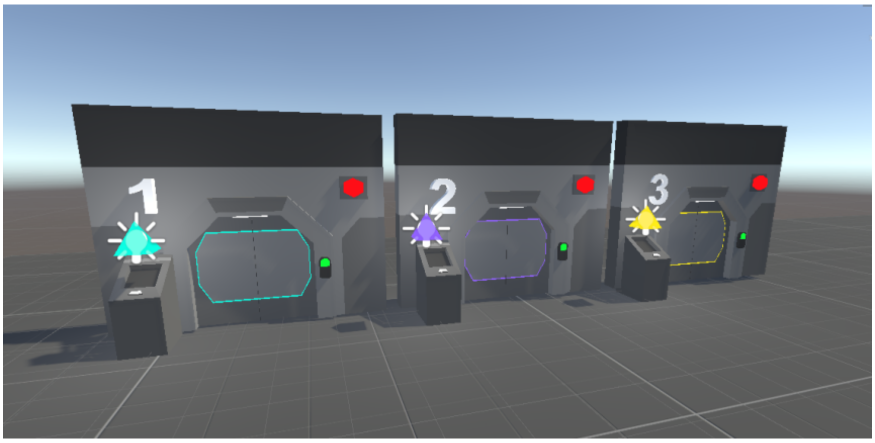
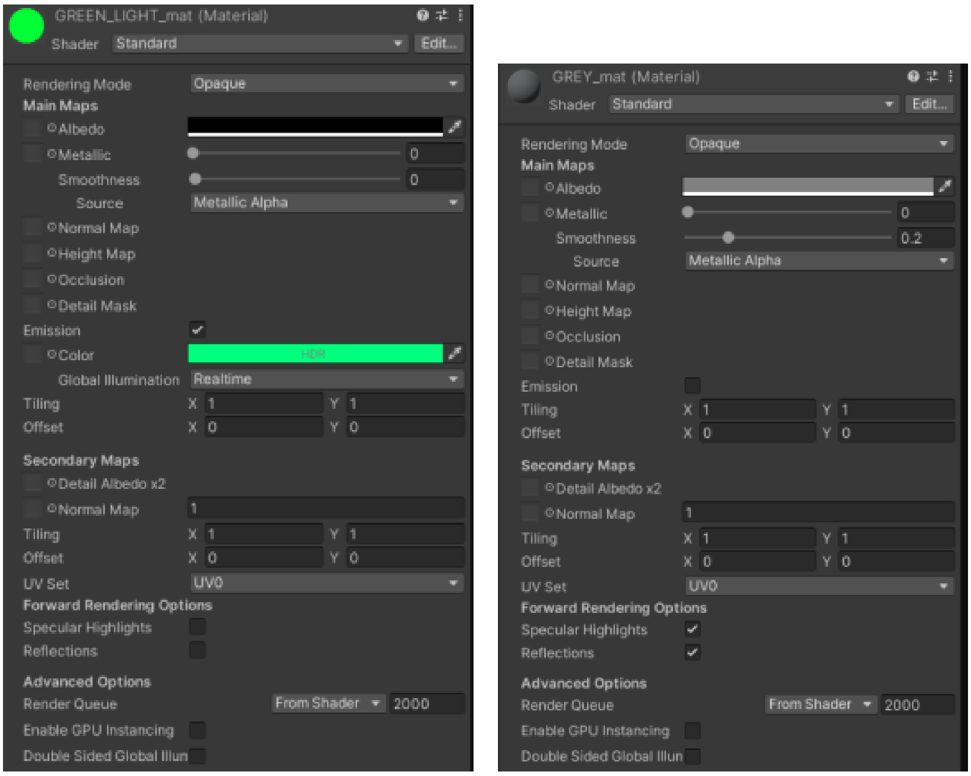


Figure 5.5: Doors and stones models

After creating the models in Blender, I imported them into Unity, added them to their material, and created a prefab for each modular component. All of the materials were created in Unity. There are three types of material. There is a light material, cube material, and wall material. The light material has an extra emission color, and the walls, like cubes, have higher smoothness, as shown in Figure 5.6. I then varied the albedo component for various materials to change the base color.



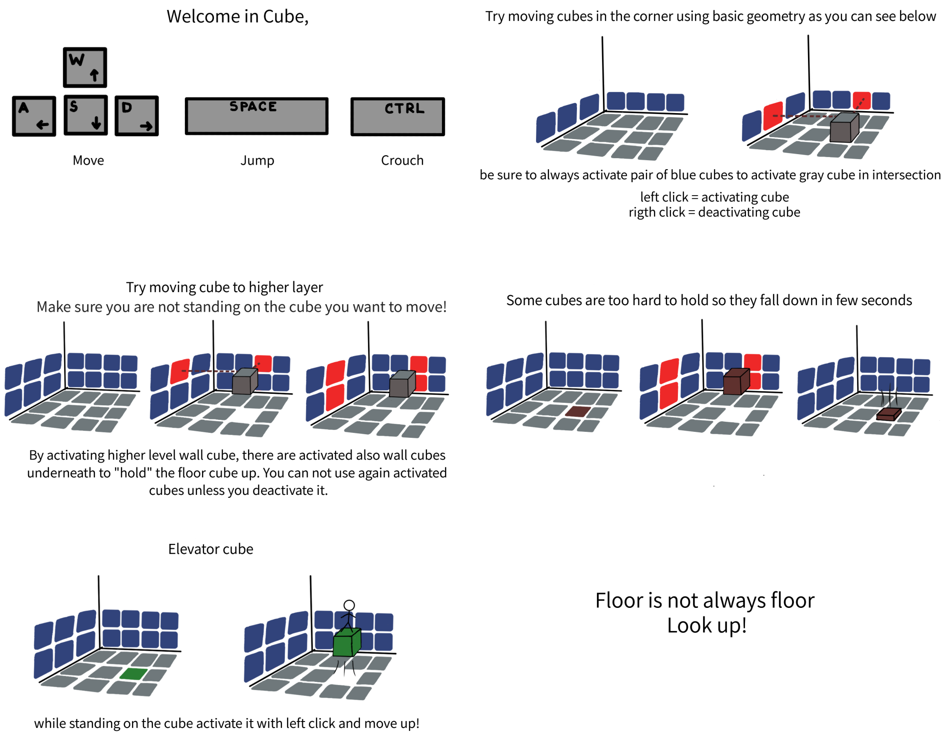
GREEN_LIGHT_mat (Material)
Shader Standard
Rendering Mode Opaque
Main Maps
□ Albedo
□ Metallic 0
Smoothness 0
Source Metallic Alpha
□ Normal Map
□ Height Map
□ Occlusion
□ Detail Mask
Emission ☑
□ Color HDR
Global Illumination Realtime
Tiling X 1 Y 1
Offset X 0 Y 0
Secondary Maps
□ Detail Albedo x2
□ Normal Map 1
Tiling X 1 Y 1
Offset X 0 Y 0
UV Set UV0
Forward Rendering Options
Specular Highlights □
Reflections □
Advanced Options
Render Queue From Shader 2000
Enable GPU Instancing □
Double Sided Global Illun □

GREY_mat (Material)
Shader Standard
Rendering Mode Opaque
Main Maps
□ Albedo
□ Metallic 0
Smoothness 0.2
Source Metallic Alpha
□ Normal Map
□ Height Map
□ Occlusion
□ Detail Mask
Emission □
Tiling X 1 Y 1
Offset X 0 Y 0
Secondary Maps
□ Detail Albedo x2
□ Normal Map 1
Tiling X 1 Y 1
Offset X 0 Y 0
UV Set UV0
Forward Rendering Options
Specular Highlights ☑
Reflections ☑
Advanced Options
Render Queue From Shader 2000
Enable GPU Instancing □
Double Sided Global Illun □

Figure 5.6: Material of light and walls

5.1.3 Tutorial Panels

To explain and help the player, I chose a graphic form in the form of panels on which I created descriptions and pictures. These should help the player when new mechanics occur. The posters on the panels were created in a program called MediBang Paint Pro [Inc].



Welcome in Cube,

Try moving cubes in the corner using basic geometry as you can see below

be sure to always activate pair of blue cubes to activate gray cube in intersection

left click = activating cube
rigth click = deactivating cube

Try moving cube to higher layer
Make sure you are not standing on the cube you want to move!

By activating higher level wall cube, there are activated also wall cubes underneath to "hold" the floor cube up. You can not use again activated cubes unless you deactivate it.

Some cubes are too hard to hold so they fall down in few seconds

Elevator cube

while standing on the cube activate it with left click and move up!

Floor is not always floor
Look up!

Figure 5.7: Posters for panels

These pictures were imported as 2D textures in Unity. The material was then created with the help of physically based standard shaders from Unity. Specifically, a diffuse shader was used that computes a Lambertian illumination model. The lighting on the surface decreases as the angle between the surface and the light decreases. It does not depend on any other angle. [Uni]

The created materials from the textures were then assigned to the prefab of panel which you can see in Figure 5.3 on the most left wall.

5.2 Levels

When all the components were created, I made prefabs from them. Prefab is a reusable asset that contains all the components of a GameObject, its values, and even its children. The objects created from this prefab also change when we change the prefab. Then I added colliders to the walls, floors, and other assets. Colliders are bounds that are used to detect collisions between objects.

After that, I started building rooms based on a design from Chapter 4.4. I used grid snapping to move the objects in the scene. This allowed us to align the walls precisely next to each other. I also placed the floor, ceiling, and lights and duplicated them. Then I connected the rooms with automatic doors and added buttons for opening the doors.

At the end of this chapter are all created rooms shown in Figures.

After that, I modeled the Initial room, which I also made a prefab from since it was used at the beginning of each scene. In the Initial room, only the Puzzle stones acquired were changed.

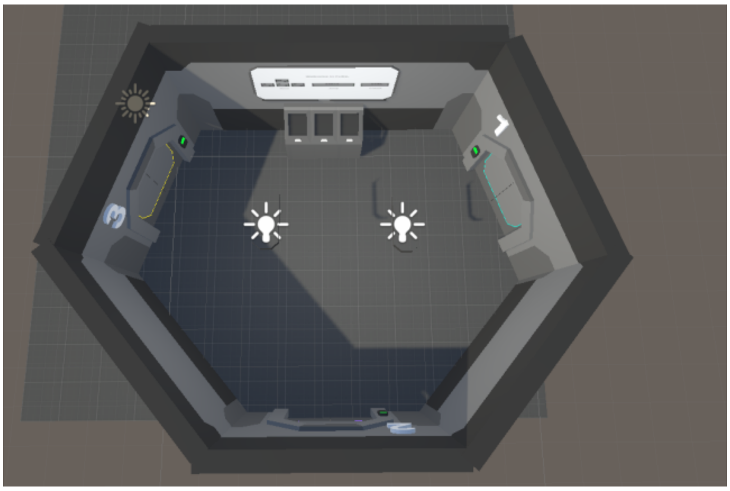


Figure 5.8: Inital room

The End room with the Puzzle stone is at the end of each level. In the End room of the first level, I added a UI panel saying that the player should press E to navigate him/her what to do and how to enter the next level. The color of the stone changes according to the current level.

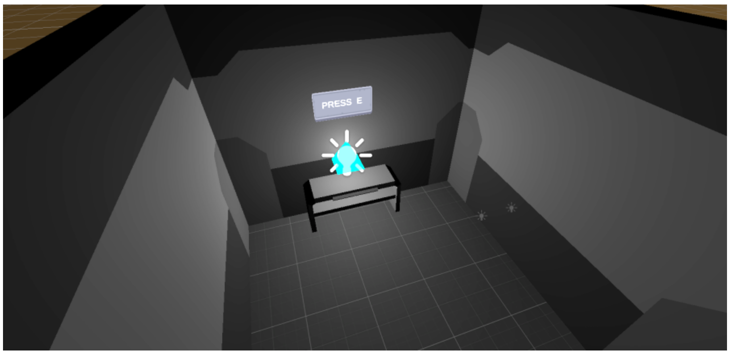
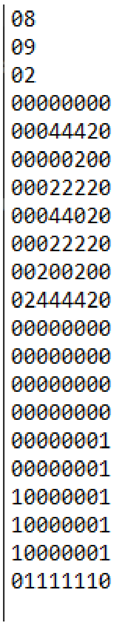


Figure 5.9: End room

5.2.1 Generating levels

The levels are generated from text assets in the form of .txt files, which you can see in Figure 5.10. In the first lines of the text file are written the width, height, and rows of the level, and according to these values, I am going through three for loops of the algorithm. Every element is an integer that stands for the type of cube that should be placed.



```
08
09
02
00000000
00044420
00000200
00022220
00044020
00022220
00200200
02444420
00000000
00000000
00000000
00000000
00000001
00000001
10000001
10000001
10000001
01111110
```

Figure 5.10: Example of level generation

- 0 - no cube,
- 1 - interact cube,
- 2 - moving cube,
- 3 - invisible moving cube,
- 4 - unstable cube,
- 5 - elevator cube,
- 6 - invisible unstable cube,
- 7 - invisible elevator cube,
- 8 - invisible interact cube.

In for loops, I am filling a 3D array of game objects, and I place the prefabs of cubes in the scene on an axis that corresponds to the indices of the element in the array. It fits the right way because each cube is in every dimension 1 meter long.

Invisible cubes are used because of the climbing levels, where the cubes only become visible with the press of a button. So the cube array is ready; I need to initialize the prefabs in the scene after that.

5.2.2 Interaction with cubes

The player activates the cube with a left click and deactivates it with a right click. The program immediately checks whether the cube is not activated yet and changes the cube's color to red or back to blue.

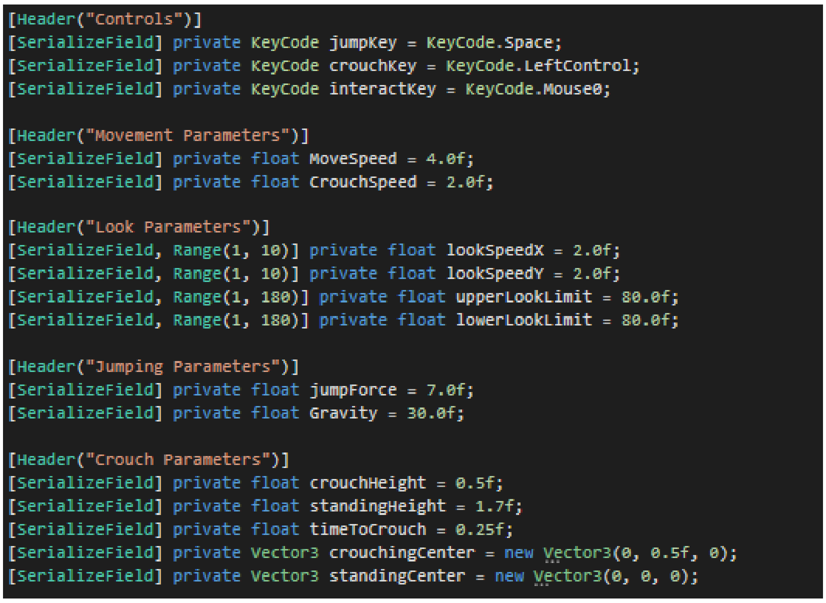
The next, more challenging step was to create an algorithm that would scan the 3D array every time a player activated to check the cubes' movement. In the first attempts, there was an effort to traverse the entire array and connect cubes that could be possibly related, which could have been more efficient. It needed to be made clear how the player was activating the cubes in sequence, and some strategies would have to be developed to select the correct moving cube. So I rather switched to the option of evaluating the move only after two cubes are activated. Only that currently activated pair is ever evaluated in the algorithm. Firstly I check if the player does not stand on the cube that wants to be moved. The moving cube prefabs contain a box collider as a child. This box collider is used as a trigger and watches whether the player stands on the cube. The longest time is then spent checking whether there is any obstacle between the moving and activated cube.

At the same time, of course, the movement of the floor cube in the game is reflected in the stored 3D array in the code. Each moved cube also keeps all the wall cubes that were activated because of it so that when it is deactivated, I do not have to go through the whole array again to find all the related cubes.

5.3 Movement and camera

The first implemented thing was player movement and controller. The game is from first-person view; therefore, the camera and the player are one object called FirstPersonController.

In this class's Update method, the movement is handled if the game is not paused. The player can move to the left, right, forward, or backward based on the direction of the forward vector and the defined movement speed. The movement speed is different when the player is crouching. Crouching is also done by animation to smooth the transition from standing to crouching. The height and center of the player change in animation thanks to linear interpolation. Jumping is done by changing the movement direction in the y-axis by the jump force and gravity set in the script. All set variables are in Figure 5.11. SerializedField allows me to change the value in Unity while debugging the game without changing the script.



```
[Header("Controls")]
[SerializeField] private KeyCode jumpKey = KeyCode.Space;
[SerializeField] private KeyCode crouchKey = KeyCode.LeftControl;
[SerializeField] private KeyCode interactKey = KeyCode.Mouse0;

[Header("Movement Parameters")]
[SerializeField] private float MoveSpeed = 4.0f;
[SerializeField] private float CrouchSpeed = 2.0f;

[Header("Look Parameters")]
[SerializeField, Range(1, 10)] private float lookSpeedX = 2.0f;
[SerializeField, Range(1, 10)] private float lookSpeedY = 2.0f;
[SerializeField, Range(1, 180)] private float upperLookLimit = 80.0f;
[SerializeField, Range(1, 180)] private float lowerLookLimit = 80.0f;

[Header("Jumping Parameters")]
[SerializeField] private float jumpForce = 7.0f;
[SerializeField] private float Gravity = 30.0f;

[Header("Crouch Parameters")]
[SerializeField] private float crouchHeight = 0.5f;
[SerializeField] private float standingHeight = 1.7f;
[SerializeField] private float timeToCrouch = 0.25f;
[SerializeField] private Vector3 crouchingCenter = new Vector3(0, 0.5f, 0);
[SerializeField] private Vector3 standingCenter = new Vector3(0, 0, 0);
```

Figure 5.11: First person controller variables

5.4 Scripts

Scripts control everything in the game, including how objects move and interact with one another. Scripts are text documents containing code representing the game's logic. C# is the programming language used by Unity. I already described the FirstPersonController, LevelGeneration, and Cube scripts. In this section, I will focus on other important GameObject, such as buttons, doors, stones, elevators, etc.

Let us start with the door model. It has two functions. The first is an automatic opening, again using the collider as a trigger. You can see the collider in Figure 5.12 as an invisible rectangle.

The opening is then done by an animation whose state diagram can be seen in Figure 5.13.

The animation transitions are linked to the condition controlled by the script. Another function is changing the door's button material from green to red and vice versa. This function is used by the button that is connected to the door.

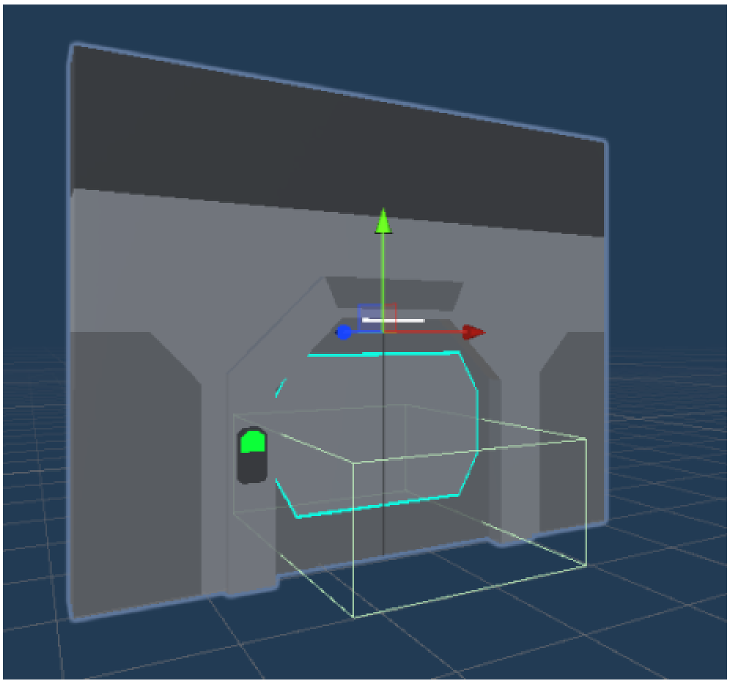


Figure 5.12: Prefab of doors

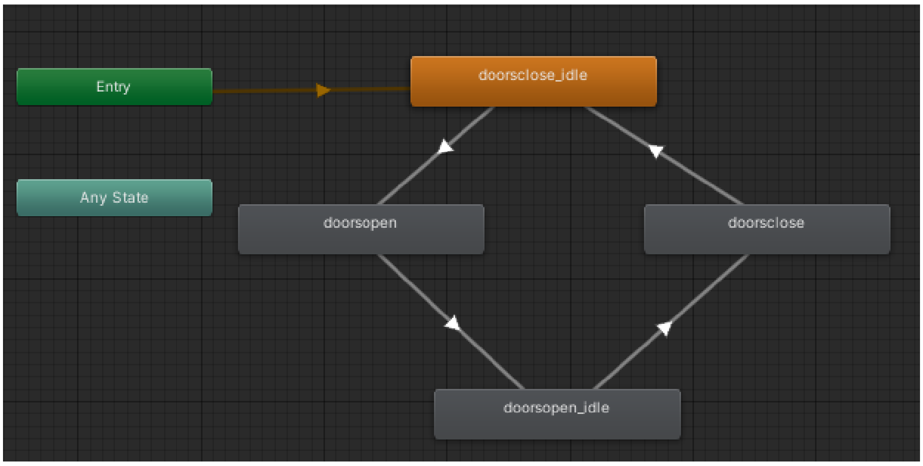


Figure 5.13: State diagram of door's animation

The button script responds to whether the player clicks on it and when they stop clicking. These conditions are again used to animate the button press.

If the press occurs, the script has a reference to either a linked door, which changes the state to open or triggers a cutscene where the floor is raised and a new puzzle appears. When pressed, it changes its color to green.

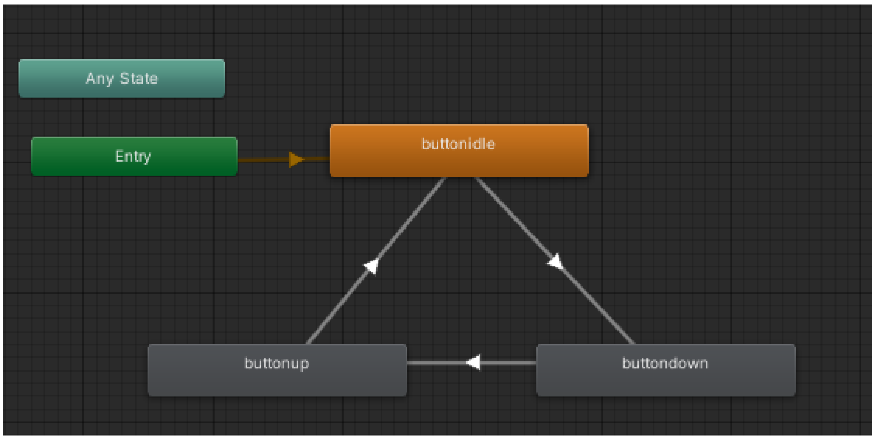


Figure 5.14: State diagram of button's animation

A cutscene is an empty GameObject that references the moving objects in the cutscene and calls their movement when triggered. My cutscene contains two levels and a floor. I use Coroutines to move objects within multiple frames. I first move all the cubes from the bottom to the top within 3 seconds. I use linear interpolation to calculate the current position between the start and end positions. Then the floor and the rest of the level, which were previously invisible, appear. Last, I delete the previous level and its cubes.

Coroutines were not used to move basic moving cubes so that they could be quickly deactivated, and the player did not have to wait for the end of the animation. However, where animation was used was in the dropping of the unstable cube and when the elevator moved. The speed of these cubes is set to one, and their transformation is calculated relative to the elapsed time since the last transformation.

The last animation used in this game is the Puzzle stone animation. This is to simulate the levitating stone the player gets at the end of each level. This animation is again done through an animator and not Coroutine. This animation is infinite, and the movement is not restricted by condition, as shown in Figure 5.15. The script handles if the E key is pressed. When this key is pressed, the player is taken to the next level and, thus, to the next scene. In the initial room, this Puzzle stone appears.

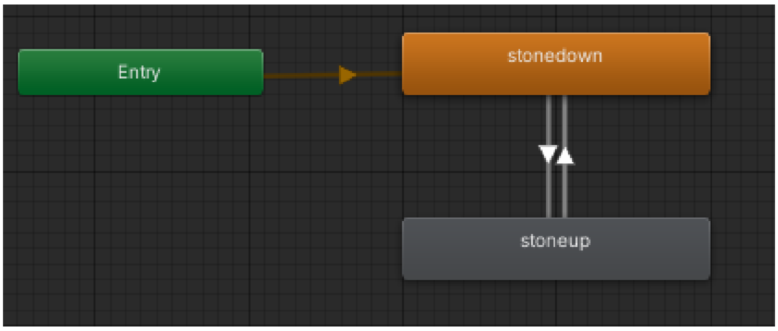


Figure 5.15: State diagram of stone's animation

The last implemented functionality is the saving system. In each level, there is an empty object called Track, which contains a container full of all checkpoints in the level. Checkpoints are again created as invisible collider triggers. I use a Track script to add these to the list. There is another script for the checkpoint itself, which contains a reference to Track. If the player passes a checkpoint, the checkpoint calls Track, and it notes the player's position and saves it to PlayerPrefs. This is used because they can be stored even though scene switching. Reloading the scene restarts it so we would lose the reference to the checkpoint. Similarly, the player position is handled when switching scenes after picking up the Puzzle stone.

5.5 Puzzle examples

In this section, I will describe a proper walkthrough of three puzzles that are significantly different from each other. First, I chose 1.4 rooms. This puzzle aims to press the button placed on the opposite wall. The player must activate the cube marked yellow in Figure 5.16.

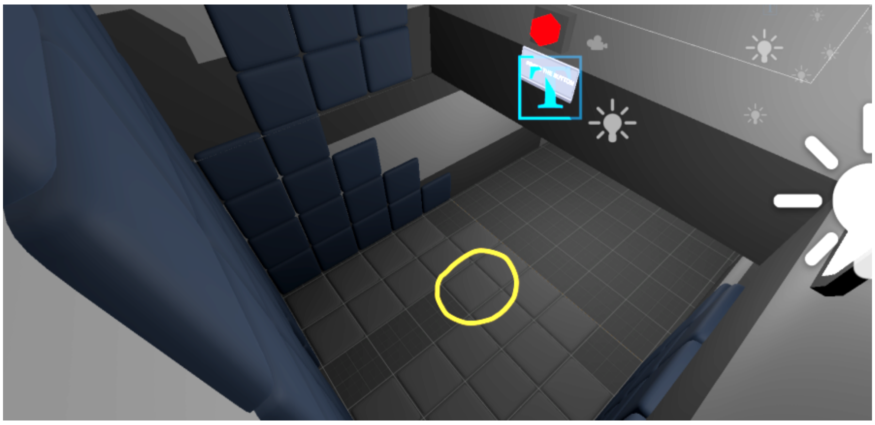


Figure 5.16: Room 1.4 example of solution 1/3

To reach the button, this floor cube must be at a height of four. Since the player cannot reach that high from the ground, he must move the cubes to activate the necessary cubes on the walls, which we can see in Figure 5.17 marked in yellow, along with the correct solution. The player must put the cubes in the correct position so that he can activate both of these wall cubes at the same time in one turn.

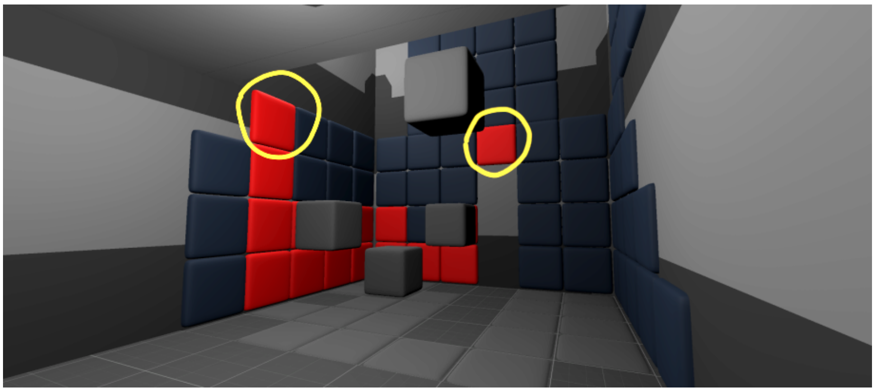


Figure 5.17: Room 1.4 example of solution 2/3

Then the player must build a path to this cube. Thus, there must be at least one cube in each height. For example, the solution can be the cube's position in Figure 5.18.

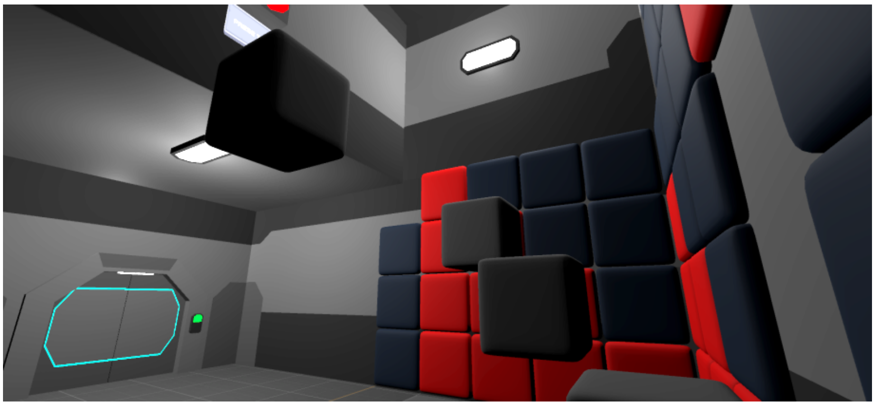


Figure 5.18: Room 1.4 example of solution 3/3

After pressing the button, a new puzzle appears, and the player climbs further.

The next puzzle I chose is in room 2.2, where is an unstable elevator cube. The goal is to press the button and get to the door. First, the player must build a path to the button, as shown in Figure 5.19. The unstable cube has to be activated as the last one so the cube does not fall.

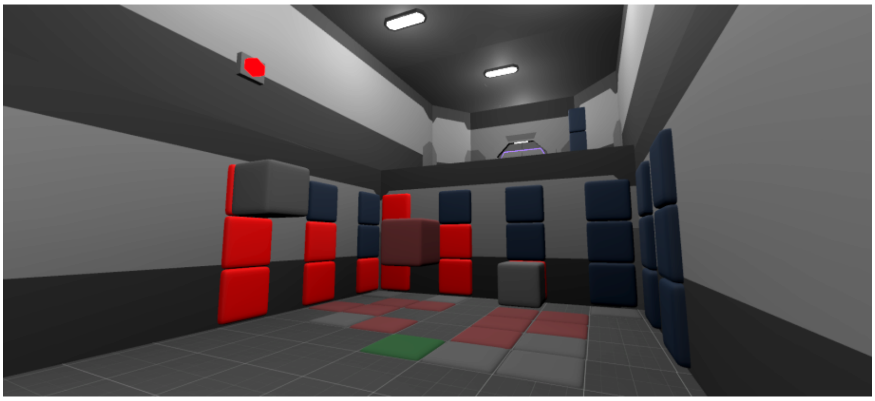


Figure 5.19: Room 2.2 example of solution 1/2

The next step is to get to the doors. There is an elevator cube that could help with that. If the player notices the elevator at the beginning, s/he can use it to get up to the cube. The player activates then one cube near the wall where the door is, activates the elevator, goes up, jumps on the cube next to the wall, and goes to exit.

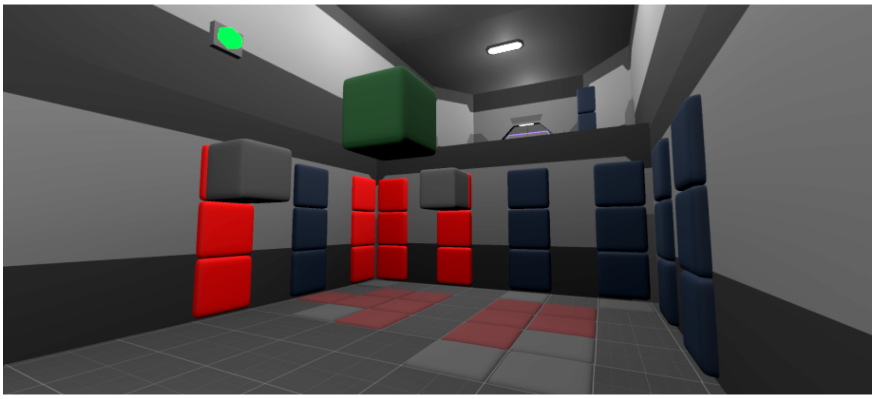


Figure 5.20: Room 2.2 example of solution 2/2

The last chosen puzzle is room 3.1, where the moving cubes are on the wall. The easiest way to get to the button is the elevator on the opposite wall. So the first thing to do is to build a path to the elevator. The player can make simple stairs along the wall with grey cubes.

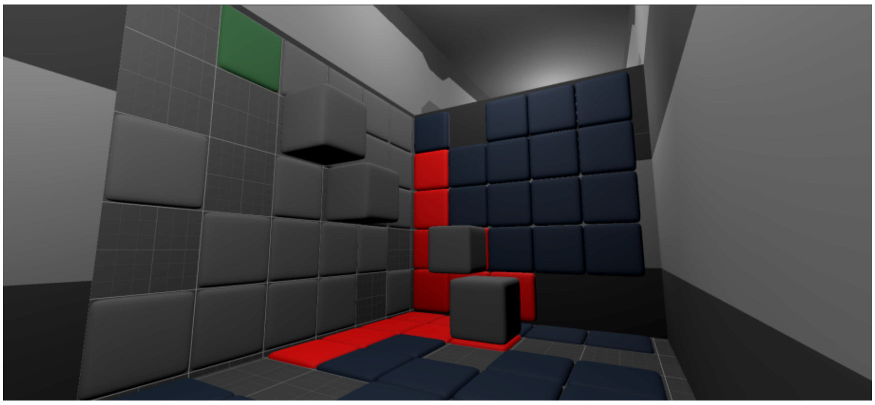


Figure 5.21: Room 3.1 example of solution 1/2

Once the button is activated, the player must figure out a path to the door. There are a few simple solutions, such as taking the stairs along the wall again but to the opposite side, activating one cube by the wall leading to the exit, and jumping on it using the elevator. I have demonstrated the diagonal solution in Figure 5.22, which may build the player because this was always an effective way in previous rooms where the cubes were on the floor.

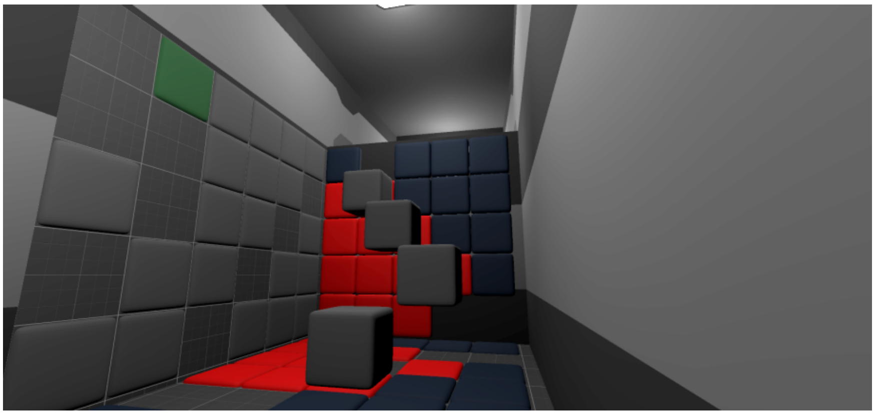


Figure 5.22: Room 3.1 example of solution 2/2

5.6 User Interface

The game opens with a simple menu. The design of the user environment was described in Chapter 4.3.3. MainMenu and PauseMenu and their functions are also implemented through scripts. Each button is then assigned a function from the script. When the player opens the MainMenu, all his progress is lost. The game does not have a save system through multiple play sessions, but only when navigating through the PauseMenu. Therefore, for testing purposes, the ability to run additional levels individually has been added to MainMenu. PauseMenu is transparent to make the player feel like he has not left the game session. The background of MainMenu was downloaded and used because it connects the cube and Unity as two main elements of this game. [cub]

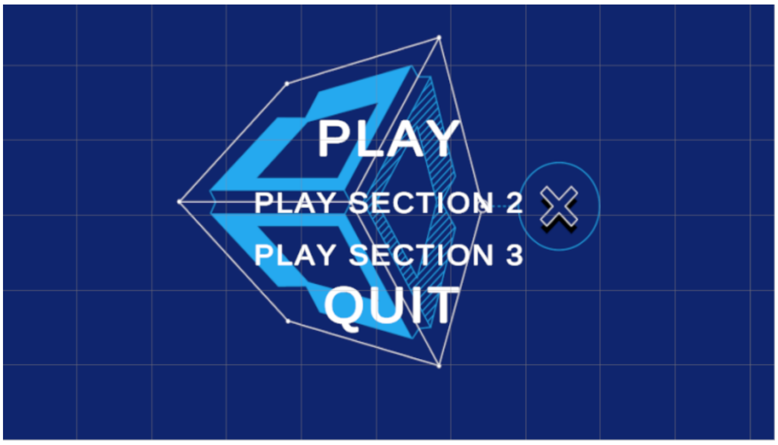


Figure 5.23: Main menu

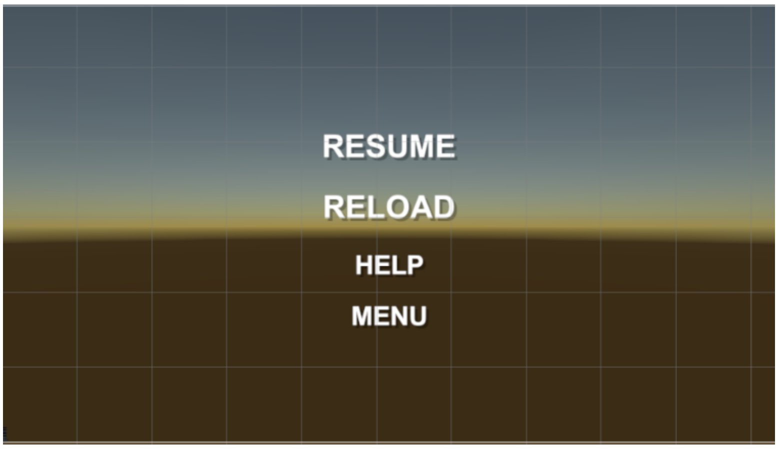


Figure 5.24: Pause menu

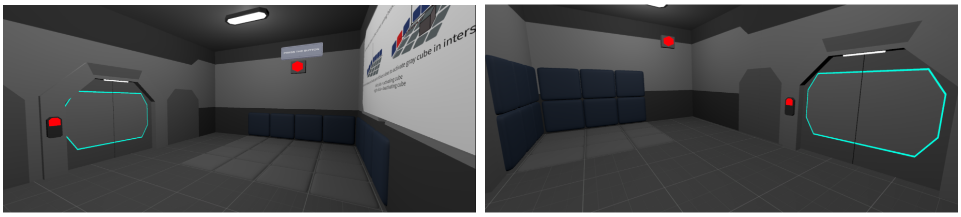


Figure 5.25: Level 1.1 and 1.2 is the introduction to mechanics

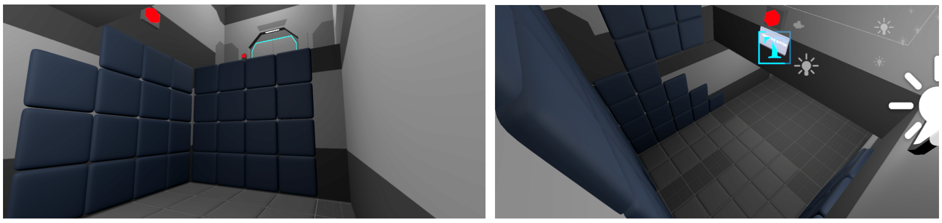


Figure 5.26: Level 1.3 and 1.4 use learned mechanics and presents climbing

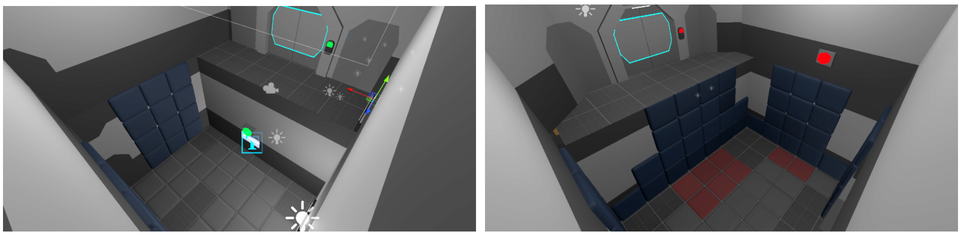


Figure 5.27: Level 1.5 is harder for the player and 1.6 introduces unstable cube

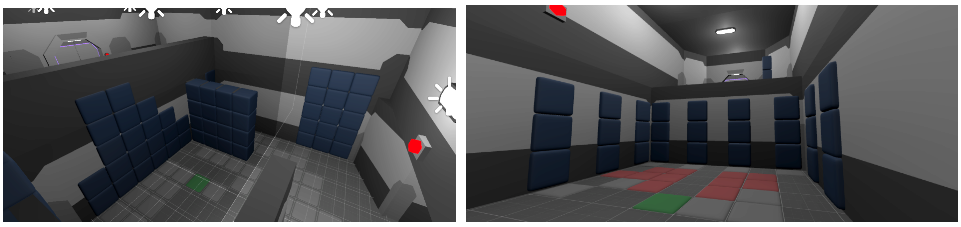


Figure 5.28: Level 2.1 and 2.2 introduce elevator cube and connect it with unstable cube

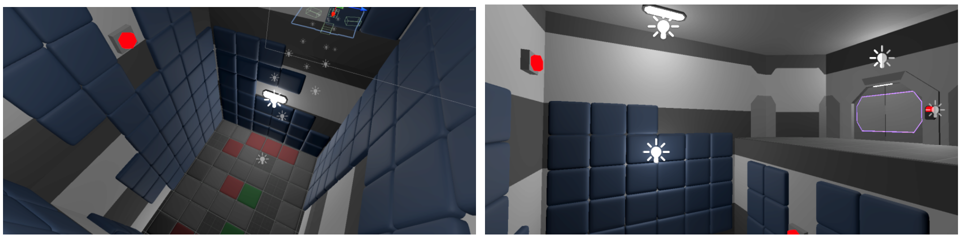


Figure 5.29: Level 2.3, 2.4 and 2.5 use learned mechanics in climbing

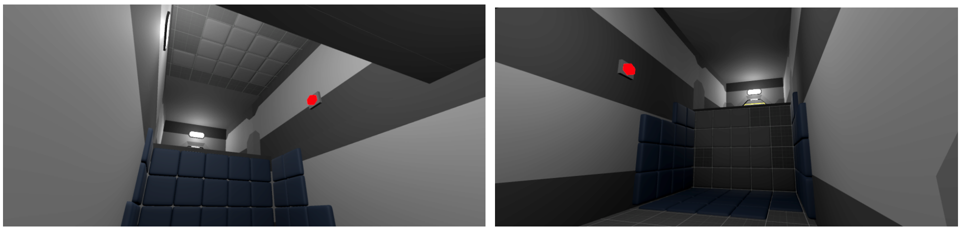


Figure 5.30: Level 3.1 and 3.2 introduce rotated levels

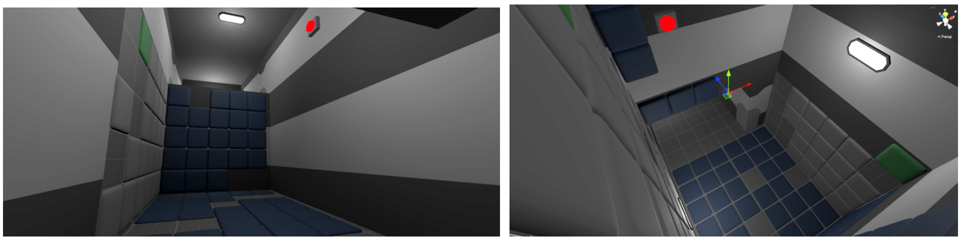


Figure 5.31: Level 3.3 and 3.4 is climbing part

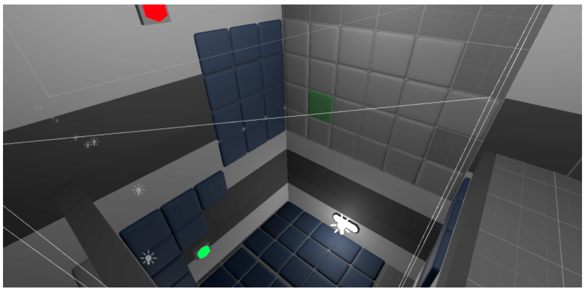


Figure 5.32: Level 3.5 final of climbing part

Chapter 6
Testing

Testing is a crucial part of the process of developing a game. It is the best way to get feedback before the game is finished. Even during testing, the game design document may often change to improve the game concept regarding player experience.

The main objective of the testing was to check whether the puzzles were correctly designed and conformed to the principles described in Chapter 3. In the tests, we mainly focus on whether the players like the puzzles, whether they enjoy them, and whether they understand what to do and how to do it. We also need to test how challenging the puzzles are and how quickly players find a dominant strategy.

Participants should download the game on their devices and share a screen when passing. They were also instructed to verbalize their thoughts. I spoke as little as possible to them in the walkthrough so as not to affect the testing flow.

In this chapter, I will describe the testing process, methodology, and the test results. There were two rounds of qualitative testing conducted during the development.

6.1 First test

The first testing was done after the tutorial but before the next set of levels were created. The game contained one entry tutorial room and then two puzzles. There were no sounds and no details in terms of graphics.

This qualitative test was done with two participants and they were required to go through every level in the tutorial. The test objective was to test the main mechanics and theme of the game, to determine whether the puzzles are straightforward, and if the tutorial is sufficient enough for the player to understand the rules and be able to solve the two created puzzles. Another concern was whether the game preserves logic and is not overly chaotic. Another aspect of testing was to check bugs and other problems with the gameplay.

6.1.1 Appearance of the game

The game appearance was not important, but the mechanic was. The game has also been modified before testing so that, in the event of a bug, testers can still play and were not forced to shut the game down. A saving system has not been implemented yet, but I created simple UI on Escape key to reload or quit the game.

The entry level contained panel where the basic logic was explained and in the corner of the room the player could try it out. At the same time, player needed to activate at least one cube in order to press a button and be able to go to the next room. The color of the button was red as well as the color of the button next to the door, to give the player the context that the button opens the door. When the button is pressed, its color changes to green just like the color next to the door.

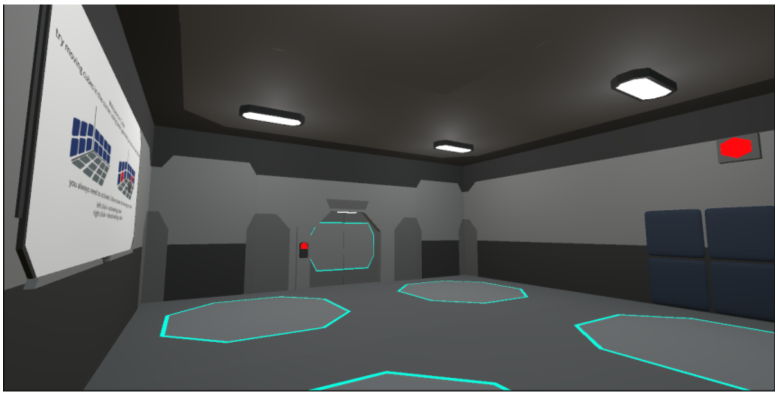


Figure 6.1: Tutorial level

After the player reached the first level, his/her task was to use the cubes to get to the top floor in order to proceed further. There were plenty of cubes on the rooms walls so the player could choose their own path and try out the mechanics without limitations. There is no button to open the door, so the door is already green.

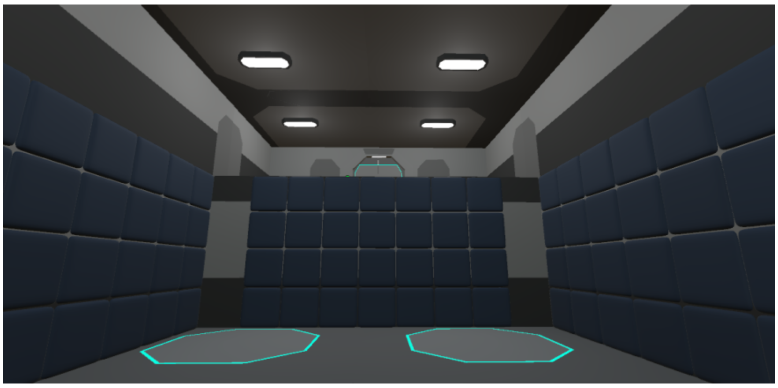


Figure 6.2: First level

The second level was different mainly in the placement of the cubes and making sure the player understands the function of the red button that opens the door despite not being able to see the door. The player has to get to the button first from where he/she can see that there us a door beyond the cubes. The player must also use certain cubes to create the path and does not have that freedom.

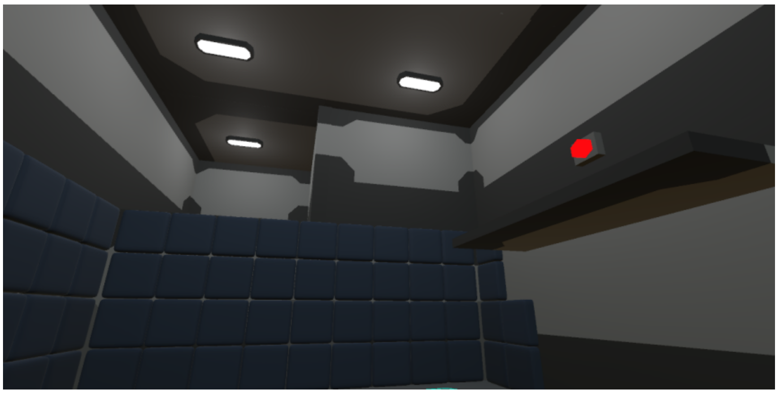


Figure 6.3: Second level

6.1.2 Test results

Both participants were 24 years old and identified as a male. Their experience with computers and games was little bit diverse. One of the participants rates his experience with computers and games as high, while the second one is rather average.

The two testers did not really understand the instructions in entry level and it took quite a while to understand the game logic and get through the tutorial. The second participant also had problem of not being able to navigate in a 3D environment and did not understand that activated cubes had to form an intersection where the moving cube would appear.

Nobody encountered any difficulty controlling the game. It was clear how to move and perform basic tasks. Despite the fact that the panel with tutorial showed the cube and their function, the players did not understand that the floor cubes were under the floor in their default position and they needed to activate the cubes on the wall in order to get the cube up from the floor and use the activation to move them up the z-axis.

On the other hand, in all the rooms they both understood what the goal was and where they should go. At the same time, after they figured out how the game works, they claimed that the theme is interesting enough to make them enjoy the game. Once they became familiar with game rules and how specific things interacted, they did not have issues with following the gameplay and solving the puzzle.

Testers discovered a bug with controlling the player where they were able to jump over multiple cubes due to jump and crouch. They also noted that the player should be narrower in order to to be able go between the two cubes.

Finally, I made a list of things that should be changed based on the feedback from the first test:

- Change the width of FirstPersonController.
- Make a better tutorial.
- Make the floor cubes visible.
- Change logic of the game.
- Change controller of the player.

These points were implemented into the game through the development and were then included in the next testing.

6.2 Second test

The second testing was done with the first section finished and also with three levels of the second section. The game together contained the Inital room, two tutorial rooms and seven puzzles. This qualitative test was done with six participants and they were required to go through every room. The test objective was to test the modified main mechanics and introduces new types of cubes. To determine whether the puzzles are straightforward and if the tutorial is sufficient enough for the player to understand the rules, new cubes and changing environment. The purpose of the test was also about testing if all puzzles are solvable, and finding if the game is generally playable and enjoyable.

6.2.1 Appearance of the game

The game appearance was finished so it looked just like you could see in the Chapter 5. The only difference was that in section 1.1 and 1.4 was no Press the button image. A saving system was implemented, as well as a UI that contained both MainMenu and PauseMenu to resume, reload or quit the game. In other words, the rooms used for the testing were: 1.1-1.6, 2.1, 2.2 and 3.2. you can see the rooms in Figures 5.25-5.27, 5.28, 5.30.

6.2.2 Test results

All six participants were aged 20-25 and identified as a male. Three of the participants rated their experience with computers as average, two as high and the last one as very high. On the other hand, two rated their experience with games as high, another three as very high and the last participant as low.

Four of the more experienced players had also played a 3D puzzle game before, and I'll describe their findings first. All four players were clear on the geometry and mechanic after going through the two tutorial rooms and the first puzzle. They all understood how the unstable cube and the elevator worked and had no problem using them in the puzzles. At the same time, two of them remarked on the balance between the unstable cube tying them down more but the elevator giving them more options and freedom.

Regarding the difficulty of the levels, two players commented that level 1.4 was perhaps too challenging to follow almost after the tutorial On the other hand, 1.6 was easier, as three players found a way around the puzzle without using the unstable cube. Room 3.2 was quite complicated for two of them and it took them a long time to get their bearings as the moving cubes were not on the floor but on the wall. The other two picked it up quickly and found it a very good diversion.

The two less experienced players took longer to navigate in the 3D environment and it took them longer to understand all the rules. Both felt that they only started to understand the game after completing the entire first level. One of them remarked that the elevator helped him a lot in solving the puzzles afterwards because he was able to look at the puzzle from above.

After testing, I asked all the participants what they found fun about the game and what bothered them. They mostly enjoyed finding an efficient way through the puzzles and that there are often multiple ways to solve them. One of them also mentioned that he liked the fact that you cannot die in the game. On the other hand, half of them were bothered that the cubes could not be activated from an infinite distance. Two mentioned that it was not always clear to them where to go in the puzzle and that they had to press a button. Or they could not see the door at that moment and this made them nervous.

Two bugs were also found during testing. Some players did not see the cursor in the game, or it showed up after a reload of the game, or a reload of the game made the cursor invisible. The saving system behaved similarly unstable. Some players were not saving checkpoints and when they reloaded, they appeared at the very beginning of the level and not in the room they reloaded from.

Finally, I made a list of things that should be changed based on the feedback from the second test:

- Make the room 1.3 easier.
- Delete some basic moving cubes in the room 1.6.
- Add text about pressing the button.
- Fix the disappearing cursor.
- Remake the saving system.

These point were implemented in the game during the development. Based on the feedback, the following levels were then completed and implemented and the game was finished.

6.3 Summary

The tests have fulfilled their purpose and have helped in the development of the game. All the outputs from the tests have been incorporated because they had a major key result. The game mechanics and puzzles are complicated and challenging, but once the player fully understands them, s/he enjoys solving them. At the same time, given the principles of the puzzles, I ensured the players saw the puzzle's goal and what they had to do. They also enjoy looking for a dominant strategy and find introducing new game elements fun and exciting. All of this was the main goal of the design.

After testing, I concluded that the game is more suitable for regular players who may already have experience with a puzzle game.

Chapter 7
Conclusion

This bachelor thesis aimed to design and implement a 3D logic game that changes the game environment. I researched the game design, principles of logic games, and level design process. The literature suggests four main elements to focus on in game design: technology, mechanics, aesthetics, and story. At the same time, we need to think about the player's experience and connect these elements to create the essential experience we want the player to get from playing.

My game heavily relies on puzzles. Thus, a comprehensive literature review was conducted to identify the main principles of logic games. The main goal of a good puzzle design is an understandable goal, a simple start, and a sense of progress. The difficulty should increase as the player progresses through the levels. I also analyzed appearance of the levels. The levels should not be uniform and should look like a place with specific repetitive rules and looks to feel like a real place. The level must contain the right amount of new elements to not overwhelm the player.

I then analyzed 3D logic games with similar features to my game. Thanks to this, I confirmed that these principles work and are used in games to create good puzzles and levels.

Based on this research and analysis, the game design document of a 3D first-person logic game was written. Original puzzle design and mechanics were devised. Then I created modular components and models, some of which I also created textures and materials. Components were imported into Unity and were used to build the levels based on the design. The puzzles were written into the text files to form a 3D array generated into the rooms. The game logic was implemented using scripts.

After implementing the basic logic, a prototype of the puzzle mechanics was tested with two users. The test result was that the original mechanics were too complex and that more tutorial levels were needed.

Subsequently, the game design document was changed, and the mechanics were re-implemented. The game was expanded with more models, levels, and new elements.

Finally, I again qualitatively tested the new game mechanics with six users. One significant outcome was that the game's principles and puzzles are challenging and complex, but once the player fully understands them, they become enjoyable to solve.

Overall, the findings from the theoretical part were used in the practical part to create the game and the thesis has fulfilled the goal that was set in the Introduction.

For the future, I would like to model the character and animate it so the hands can be seen all the time and the interactions with cubes are more visible. Unfortunately, the game does not have an implemented story, which is essential. I want to add some voice lines for the main character and artificial intelligence accompanying the player. For a better game experience, I would like to add background music or sounds when interacting with the cubes.

Working on the game improved my skills with the Unity software. After this experience, I would implement the game differently, so I will continue in the development of the game and may even publish it one day.

Appendix A

Bibliography

[Ass] *Unity asset store*, https://assetstore.unity.com/packages/3d/environments/sci-fi/free-lowpoly-scifi-110070#releases, ([Online] Accessed: 20.04.2023).

[ble] *Blender*, https://blender.org/, ([Online] Accessed: 20.04.2023).

[Cro14] Croteam, *The talos principles*, https://store.steampowered.com/app/257510/The_Talos_Principle/, 2014, ([Online] Accessed: 20.04.2023).

[cub] *10 common unity mistakes*, https://www.toptal.com/unity-unity3d/top-unity-development-mistakes, ([Online] Accessed: 20.04.2023).

[FF95] Peter A Frensch and Joachim Funke, *Definitions, traditions, and a general framework for understanding complex problem solving*, Universitätsbibliothek der Universität Heidelberg, 1995.

[Gam14] Fireproof Games, *The room*, https://store.steampowered.com/app/288160/The_Room/, 2014, ([Online] Accessed: 11.05.2023).

[Gam15] Steal Crate Games, *Keep talking and nobody explodes*, https://store.steampowered.com/app/341800/Keep_Talking_and_Nobody_Explodes/, 2015, ([Online] Accessed: 11.05.2023).

[Gam17] T.M. Games, *We were here*, https://store.steampowered.com/app/582500/We_Were_Here/, 2017, ([Online] Accessed: 11.05.2023).

[GVI18] Alejandro González-Vázquez and Juan-José Igartua, *Why do adolescents play videogames? proposal of a scale of motives to play videogames based on the theory of uses and gratifications?*, Cuadernos.info (2018).

[Inc] MediBang Inc., *Medibang paint pro*, https://medibangpaint.com/en/pc, ([Online] Accessed: 04.04.2023).

[Inc21] Cyan World Inc, *Myst*, https://store.steampowered.com/app/1255560/Myst/, 2021, ([Online] Accessed: 11.05.2023).

[KDN18] Adam Kramarzewski and Ennio De Nucci, *Practical game design: Learn the art of game design through applicable skills and cutting-edge insights*, Packt Publishing Ltd, 2018.

[Kin12] King.com, *Candy crush saga*, https://www.king.com/game/candycrush, 2012, ([Online] Accessed: 11.05.2023).

[OP14] Adam C Oei and Michael D Patterson, *Playing a puzzle video game with changing requirements improves executive functions*, Computers in Human Behavior **37** (2014).

[Rog10] S Rogers, *Level up. the guide to great video game design.-a john wiley & sons, ltd*, John Wiley Sons, 2010.

[Sak16] Tomas Sakalauskas, *Human fall flat*, https://store.steampowered.com/app/477160/Human_Fall_Flat/, 2016, ([Online] Accessed: 11.05.2023).

[Sch08] Jesse Schell, *The art of game design: A book of lenses*, CRC press, 2008.

[Sic08] Miguel Sicart, *Defining game mechanics*, Game studies **8** (2008), no. 2.

[Spe] IEEE Spectrum, *Atari breakout: The best video game of all time?*, https://spectrum.ieee.org/atari-breakout, ([Online] Accessed: 04.04.2023).

[The16] I. Thekla, *The witness*, https://store.steampowered.com/app/210970/The_Witness/, 2016, ([Online] Accessed: 11.05.2023).

[Tho] R. Thorgmorton, *Modular architecture for game project*, https://www.artstation.com/artwork/6wYZ6, ([Online] Accessed: 18.01.2023).

[TJ08] Ed S Tan and Jeroen Jansz, *The game experience*, Elsevier, 2008.

[Uni] *Unity user manual*, https://docs.unity3d.com/Manual/shader-NormalDiffuse.html, ([Online] Accessed: 07.05.2023).

[UT] ed. Unity Technologies, *Unity*, https://unity.com/, ([Online] Accessed: 04.04.2023).

[Val07] Valve, *Portal*, https://store.steampowered.com/app/400/Portal/, 2007, ([Online] Accessed: 11.05.2023).

[Val11] ______, *Portal2*, https://store.steampowered.com/app/620/Portal_2/, 2011, ([Online] Accessed: 11.05.2023).

[Zep11] ZeptoLab, *Cut the rope game*, https://www.cuttherope.net/, 2011, ([Online] Accessed: 11.05.2023).