NOVA SCHOOL OF SCIENCE & TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE

ANDRÉ LINARD SANTOS AUXTERO

B.Sc. in Computer Science and Engineering

GAME ENVIRONMENT DESIGN CREATOR USING ARTIFICIAL INTELLIGENCE PROCEDURAL GENERATION

MASTER IN COMPUTER SCIENCE AND ENGINEERING

NOVA University Lisbon
December, 2023

NOVA SCHOOL OF SCIENCE & TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE

GAME ENVIRONMENT DESIGN CREATOR USING ARTIFICIAL INTELLIGENCE PROCEDURAL GENERATION

ANDRÉ LINARD SANTOS AUXTERO

B.Sc. in Computer Science and Engineering

Adviser: João Paulo Duque Vieira
Co-Founder, Skills Workflow

Co-adviser: Rui Pedro da Silva Nóbrega
Assistant Professor, NOVA School of Science and Technology

Examination Committee

Chair: Matthias Knorr
Assistant Professor, NOVA School of Science and Technology

Rapporteurs: Fernando Birra
Assistant Professor, NOVA School of Science and Technology

João Paulo Duque Vieira
Co-Founder, Skills Workflow

MASTER IN COMPUTER SCIENCE AND ENGINEERING

NOVA University Lisbon
December, 2023

Game Environment Design Creator using Artificial Intelligence Procedural Generation

This document was created with the (pdf/Xe/Lua)LaTeX processor and the NOVAthesis template (v6.10.12) [1].

Acknowledgements

I wish to express my gratitude to my advisor, CPO of Skills Workflow João Vieira, and co-advisor, Professor Rui Nóbrega, for their support and guidance throughout this entire journey. Their expertise and patience, significantly allowed for the successful completion of this thesis.

I extend my thanks to the Faculdade de Ciências e Tecnologias da Universidade Nova de Lisboa and all the professors that were present by any means during my whole academic path.

Another thank you to my colleagues, Duarte Guerreiro and Daniel Gavinho, who collaborated with me during this thesis. Their contributions, camaraderie and patience with me were invaluable.

I would like also to offer special thanks, in alphabetical order to my friends, André Fidalgo, Daniel Fidalgo, Frederico Pinheiro, José Ferreira, João Rodrigues, João Vaz, Sandro Pastor, Sofia Rocha and Tiago Mendes for their willingness to participate in the experiments for this thesis, as well as their encouragement throughout my academic journey.

Lastly, I extend a profoundly special thank you to my mother, Deolinda Auxtero, and my brother, Rafael Auxtero. Their belief in me and unwavering support have been the driving forces behind all my accomplishments and the strength to keep moving forward when facing any challanges. Without them, this journey would not have been possible, and I would not be the person I am today. Thank you from the bottom of my heart.

Abstract

Automatic generation is a prevalent technique in video games, applied to various elements like map design, character creation and level complexity. The current emergence of Artificial Intelligence (AI), such as Chat GPT and Large Language Models (LLM), has revolutionized problem solving in multiple domains, leading to the emerging of courses dedicated to mastering prompt based AI. This thesis investigates into the application of Artificial Intelligence, specifically LLM, to implement automatic environment generation within a system for interactive story creation in a video game format. The primary objective is to develop an intuitive system that allows users to effortlessly, create and experience different stories without requiring prior gaming or programming knowledge. This is achieved by simplifying complex game and story creation tasks, particularly, environment design, using an algorithm that automates the initial process while still allowing the user customization. This thesis explores multiple methods of Automatic Generation, drawing inspiration from Procedural Generation algorithms and Artificial Intelligence. Additionally, investigates multiple gaming related concepts, including Realistic and Semi-Realistic/Unrealistic design and the exploration of Open World and Modular Architecture game formats. The target audience are the individuals interested in creating and presenting their narratives in a video game framework. While the primary focus is on adventure games due to their tendency for structured storytelling, the system's versatility also allows for the creation of narratives including different genres, including educational narratives and re imaginings of well-known tales. The final prototype is a comprehensive system that uses LLM to automatically generate customizable environments. Stories can be experienced in a 3D point and click, topdown view, and the system is able to create diverse narratives, including a pizza restaurant story line and a re-creation of a segment from the game "The Secret of Monkey Island". This thesis exemplifies the potential of AI driven automatic generation, more specifically with AI-powered generative language models to facilitate accessible and imaginative storytelling in the realm of video games.

Keywords: Automatic generation, Artificial intelligence, AI-powered generative language

models, Video games, Customizable Environments, Interactive storytelling, Large Language Models, Procedural Generation, Game Design, Adventure Games, Game Exploration

Resumo

A geração automática é uma técnica prevalente em videojogos, aplicada a vários elementos, como o design de cenários, a criação de personagem e a dificuldade dos níveis. O surgimento da Inteligência Artificial, o Chat GPT ou *Large Language Models (LLM)*, revolucionou a resolução de problemas em diversas áreas. Começam a surgir cursos dedicados a dominar o *prompting* necessário para utilizar estes modelos. Esta tese explora a aplicação da Inteligência Artificial, especificamente LLM, para implementar a geração automática de cenários dentro de um sistema projetado para contar histórias interativas num formato de videojogo. O principal objetivo é desenvolver um sistema intuitivo que permita aos utilizadores cirar e experienciar estórias diversas sem exigir conhecimentos prévios na área de jogos ou programação. Isto é alcançado através da simplificação de tarefas complexas de criação de jogos e histórias, nomeadamente o design do cenário, por meio de um algoritmo que automatiza esse processo, ao mesmo tempo que permite ser customizável pelo utilizador. A tese explora múltiplos métodos de Geração Automática, inspirando-se em algoritmos de Geração Procedimental e Inteligência Artificial. Além disso, investiga diversos conceitos relacionados com jogos, incluindo abordagens de design Realistas e Semi-Realistas/Irrealistas e a exploração de formatos de jogos em Arquitetura de Mundo Aberto e Modular. O público-alvo é composto por indivíduos interessados em criar e apresentar as suas narrativas num contexto de videojogo. Embora o foco principal seja jogos de aventura devido à sua tendência para contar estórias bem estruturadas, a versatibilidade do sistema permite a criação de estórias de diferentes géneros, incluindo narrativas educativas e reinterpretações de contos conhecidos. O protótipo final é um sistema abrangente que usa LLM para gerar automaticamente cenários, permitindo ajustes pelo utilizador. As estórias podem ser vivenciadas numa perspetiva *3D point and click*, em vista *topdown* e o sistema é capaz de criar narrativas diversas, incluindo uma história num restaurante de pizzas e uma recriação de um segmento do jogo "The Secret of Monkey Island". Esta tese exemplifica o potencial da geração automática impusionada por modelos linguísticos generativos por IA, para facilitar a criação de estórias no domínio dos videojogos.

Palavras-chave: Geração Automática, Inteligência Artificial, modelos linguísticos generativos por IA, Videojogos, Ambientes customizáveis, *storytelling* interativo, Large Language Models, Geração Procedimental, *Design* de jogos, Jogos de Aventura, Exploração em jogos

Contents

List of Figures xi

List of Tables xiii

1 Introduction 1
- 1.1 Motivation and Problem Description . . . 1
- 1.2 Research Questions . . . 2
- 1.3 Objectives . . . 3
- 1.4 Solution Overview . . . 4
- 1.5 Contributions . . . 4
- 1.6 Document Structure . . . 6

2 State of the Art 7
- 2.1 Procedural Generation . . . 7
  - 2.1.1 Artificial Intelligence . . . 8
  - 2.1.2 Marching Cubes Algorithm . . . 9
  - 2.1.3 Wave Collapse Function . . . 10
  - 2.1.4 Noise Functions . . . 12
  - 2.1.5 Other Procedural Generation Algorithms . . . 13
  - 2.1.6 Summary . . . 16
- 2.2 Game Genres . . . 16
- 2.3 Game Map Architecture and Design . . . 17
  - 2.3.1 Realistic Design . . . 17
  - 2.3.2 Semi-Realistic/Unrealistic Design . . . 19
- 2.4 Game Exploration . . . 20
  - 2.4.1 Open World . . . 21
  - 2.4.2 Modular Architecture . . . 22

3 Analysis and System Design 25
- 3.1 Requirements . . . 25

3.2 Case Studies . . . 25
3.3 Architecture . . . 27
3.3.1 Create component . . . 27
3.3.2 Play component . . . 27
3.3.3 Cloud API component . . . 28
3.4 Algorithms Analysis . . . 29
3.4.1 Procedural Generation . . . 30
3.4.2 Artificial Intelligence . . . 30
3.5 Summary . . . 32

4 Implementation **33**
4.1 User Customization . . . 33
4.1.1 Object Database . . . 33
4.1.2 Tile Grid System . . . 35
4.2 Automatic Generation . . . 38
4.2.1 GPT 3.5 . . . 38
4.3 Story Gameplay . . . 40
4.4 Design . . . 42
4.4.1 Minimizing User Effort . . . 43
4.4.2 Avoiding Text . . . 43
4.4.3 Inspiration from Adventure Game Designs . . . 43
4.5 Summary . . . 44

5 Evaluation **45**
5.1 User Study . . . 45
5.1.1 Protocol and Specification . . . 45
5.1.2 Population . . . 46
5.1.3 Assignment A- User adjustments to an environment generated automatically . . . 47
5.1.4 Assignment B- Playing the story . . . 50
5.2 Results . . . 52
5.2.1 Assignment A: User adjustments to an environment generated automatically . . . 52
5.2.2 Assignment B: Playing the story . . . 58
5.3 Questionnaires . . . 65
5.3.1 System Usability Scale Questionnaire . . . 65
5.3.2 System Questionnaire . . . 67
5.3.3 Characterization Questionnaire . . . 67
5.4 Discussion and Comparative Analysis . . . 68

6 Conclusions **70**
6.1 Objectives . . . 70

6.2 Research Questions 71
6.3 Future Work 72

Bibliography **75**

Appendices

A Consent form for testing session **80**

Annexes

List of Figures

1.1 Create mode View 5
1.2 Play mode View 5

2.1 Algorithms applied in Procedural Content Generation papers [2] 9
2.2 Marching cubes process in a 3D cube 10
2.3 Brick Block example 10
2.4 Brick Block with floating blocks 11
2.5 Wave Collapse Function image examples 12
2.6 Adventure Games examples 18
2.7 GTA V vs Real Life side-by-side 19
2.8 Assassin's Creed Odyssey vs Real Life side-by-side 20
2.9 Not being able to jump over obstacles 21
2.10 Weapon Wheel and Large Inventory 21
2.11 Examples of games with Open World environments 23

3.1 Restaurant Case Study 26
3.2 Hensel and Gretel Case Study 26
3.3 The Secret Of Monkey Island Case Study 27
3.4 Architecture of Create feature 28
3.5 Architecture of Play feature 29
3.6 Architecture of Cloud 29

4.1 Object Database 34
4.2 Grid representation 36
4.3 Placement of an object 36
4.4 Multiple "Walls" on the same tile 37
4.5 Parameters used in the API connection 39
4.6 Example of a prompt to GPT 40
4.7 JSON format 41
4.8 3D Topdown View 42

4.9 Character animation while interacting 42
4.10 Inventory Object dragging 43
4.11 Design inspirations 44

5.1 Population demographics graphs 47
5.2 Assignment A - Average time per task 53
5.3 Assignment A - time to complete overview 54
5.4 Main Menu Design Evaluation 55
5.5 Object Preview System evaluation 56
5.6 Object types evaluation 57
5.7 Object Delete order evaluation 58
5.8 Identifiable Modes 59
5.9 Assignment B - Average time per task 60
5.10 Assignment B - time to complete overview 61
5.11 Character reaction evaluation 63
5.12 Item utilization Evaluation 64
5.13 SUS Scores 66
5.14 System Questionnaire Results 67

List of Tables

2.1 Noise Functions comparation [5] 13
2.2 Base Graph Complexities and Memory usages [17] 14
2.3 Time measurements for the main pipeline stages [17] 14
2.4 Summary of all the algorithms studied and their utilization in map generation for video games 15

3.1 Comparing the different hypothesis for automatic generation 32

5.1 Task A1 results 55
5.2 Task A2 results 56
5.3 Task A3 results 56
5.4 Task A4 results 57
5.5 Task A5 results 58
5.6 Task B1 results 60
5.7 Task B2 results 62
5.8 Task B3 results 62
5.9 Task B4 results 63
5.10 Task B5 results 64
5.11 Task B6 results 65
5.12 SUS questionnaire results 66

1 Introduction

The video game industry is an industry that has been growing through the years and is projected to keep growing in the next years to come[1].

Procedural Generation is a method that allows data creation automatically instead of manually, using human made algorithms[2] while artificial intelligence is the simulation of human intelligence by machines. Nowadays this can be used in a wide variety of sectors, including for content creation for video games. Inside that sector, it is used for a lot of different applications, such as map generation, level difficulty adjusting, missions generation and others [2]. The use of this type of generation allows the creation of more dynamic content, it saves in development time and increases replayability, however it can be taxing on hardware and can feel repetitive and unfair because it depends a lot on the quality of the algorithms [3]. In the context of map creating, the uses of Procedural Generation are nowadays accepted and normalized, to the point that many famous games, such as No Man's Sky(2016) and Minecraft (2011) implement this technique and it is becoming more common for video game creators to use this technique to improve their games. This technique is very common in Open World type games and despite being fairly common in many recent games it was also used in some older games, such as the first Civilization (1991) and Microsoft Minesweeper (1990) that used Procedural Generation to make the grid of hidden mines.

1.1 Motivation and Problem Description

In the industry of video games, storytelling is undergoing a constant evolution. As the methods of creation of narratives continue to expand, the demand for more accessible

[1]Global Video game market value from 2020 to 2025, Statistics regarding global video game market value from 2020 to 2025, https://www.statista.com/statistics/292056/video-game-market-value-worldwide/, Last Access: 2023

[2]What is Procedural Generation?, Procedural Generation overview, https://www.mit.edu/~jessicav/6.S198/Blog_Post/ProceduralGeneration.html, Last Access: 2023

[3]The pros and cons of procedural generation in *Overland*, Advantages and disadvantages of using Procedural Generation in a tactical survival game like *Overland*, https://www.gamedeveloper.com/design/the-pros-and-cons-of-procedural-generation-in-i-overland-i-, Last Access: 2023

and customizable interactive storytelling. With this in mind, this thesis explores the integration of artificial intelligence technology, to enhance the creation and play-ability of environments for storytelling.

Developing a video game is a task that requires a lot of knowledge in coding and in other aspects of a game, such as story writing, art, design, audio, among others.

The goal of this thesis is to create a system, that uses automatic generation to assist the user in the creation of the environment for a story. All parts of the map, both the structures and the asset placement will be automatically generated and the user must be allowed to make any adjustments to those objects. For this, the system must account for both automatic generations by the algorithm and manual interactions by the users. By using this method, we want to reduce the difficulty of one aspect of game development, allowing the user to create a realistic game environment with minimal effort.

The system's user interface needs to be simple and straightforward, to make it easier for the user to understand and utilize all the features provided and be able to create different scenarios with no difficulty and without requiring any prior knowledge regarding coding, design or gaming.

To make the stories captivating to the users, there should be all the common elements that usually exist in games, such as micro interaction, feedback, audio, dialogue, particle effects, shaders, animations, among others. This is generated along side the environment by the algorithm and allows the user to have all the elements that typically exist in a game, without the need for coding or prior knowledge, to create his own story. This thesis is being conducted concurrently with the work of Duarte Guerreiro [3], where the investigation if focused on the creation of the storyline based on a storyboard where it is specified the character dependencies. The junction of both works, should result in a system that enables to create and play any story, in an automatic way without requiring any prior knowledge from the user.

1.2 Research Questions

Based on the initial proposal of this thesis, it is expected to have to answer to some key questions that will arise. The research questions that this thesis have to answer are:

- **RQ1- How effectively can LLM be utilized to automate the creation of diverse and realistic environments for interactive storytelling purposes?**
  This question investigates the core capability of LLM to automatically generate environments. It seeks to understand how well the AI model can create a wide range of realistic environments that suit the storytelling needs.

- **RQ2- How to ensure user customization and minimal prior knowledge requirements in gaming and programming?**
  This question explores the extent to which users can customize the environments

created automatically, while always implementing a "no coding policy", meaning no required knowledge in regards to games or programming.

- **RQ3- In what way can the versatility of AI driven automatic generation accommodate different storytelling genres, including adventure games?**
  This question explores the system's adaptability and versatility by examining its capacity to accommodate to various storytelling genres. It seeks to understand how the AI driven automatic generation can be used to support different types of narratives, ranging from educational stories to reinterpretations of known tales. The goal is to access how this versatility enhances the system's appeal and potential user base, allowing a wide range of individuals to create and present multiple narratives in video game format, for different situations.

1.3 Objectives

This thesis' objective is to investigate and make a framework that allows the creator to tell any possible story in video game format. The following goals were defined to achieve this objective:

- **Develop an innovative System:** Creating a system that leverages Artificial Intelligence, specifically LLM, to automatically generate environments for interactive storytelling.
- **Ensure User-friendly experience:** The design of the system, must be intuitive and simple, allowing users to effortlessly create and play stories without requiring any prior knowledge in gaming or programming. The goal is to identify and implement efficient and user-friendly methods to simplify complex game and story creation tasks, especially environment creation, to enhance the accessibility and usability of the system.
- **Customization and versatility:** It needs to enable users to customize the automatically generated environments while also maintaining system simplicity. This includes accommodating various storytelling genres, including educational narratives and reinterpretation of existing known tales.
- **Scalability:** Ensure that the system is scalable by being possible to incorporate future rules with minimum effort required and accomodating for different types of stories and narratives.
- **Playable system:** The system needs to enable users to not only create narratives, but also be able to play and experience the stories within a video game format, with micro-interactions and animations to increase the sensation of realism, providing a complete and comprehensive interactive storytelling experience.

1.4 Solution Overview

Following the stipulated Objectives, in order to answer the previously stated Research Questions, this thesis can be divided into two main components: The Create component (Fig 1.1) and the Play component (Fig 1.2).

For the Create component, there are two subsystems that need to be implemented: the *World* and the *Storyboard*.

The world is the representation of the environment of the story. It includes all the available objects and characters, including player characters and non-player characters, each with their respective animations. This component is where the Artificial Intelligence is used to automate the creation of different realistic environments, based on the available assets. In addition to automatically generating environments, this component must also enable user customization. Users should be able to freely move, change or remove any object previously placed by the algorithm

The Storyboard comprises multiple steps that connect the objects and characters of the World using animations to facilitate storytelling. This component is concurrently developed by a colleague as part of another thesis [3].

The Play component is responsible to allow the user to interact with the storytelling. This subsystem allows the user to experience the narratives, in a 3D topdown point and click video game format. This will include all the elements commonly found in video games, such as micro-interactions, non-playable characters, main character, animations, etc... that allow to further progress the story. The end goal is to create a system that allows a user to create stories and worlds in video game format, where those stories can be used for various purposes, such as casual story telling or for other market uses, for example, in the educational market in teaching different subjects or in the advertising industry to make promotions, in situations such as formation, explanations or certifications. In short, the system will be an assistant in the creation of stories and worlds. With all this features implemented it is possible to create Case Studies to help evaluate if the Research Questions were answered.

1.5 Contributions

This thesis contributes to the fields of interactive storytelling, artificial intelligence in video games and user-friendly game development. These contributions are summarized below:

- **Innovative AI-Powered System:** The development of a system that utilizes LLM, specially GPT 3.5, to automate the generation of environments, opens new possibilities to ease the initial environment creation process.

- **User-Friendly design:** The study (which included review of current State of the Art) and implementation of a user-friendly interface design that enables the user to use

Figure 1.1: Create mode View

Figure 1.2: Play mode View

the system without prior gaming or programming knowledge to create their own interactive stories.

- **A user study:** The study that tested the system with the goal to evaluate the simplicity and intuitivity of the design, in addiction to evaluating the quality of the system and automatic generation in general.

- **Multiple Case Studies:** Some examples of stories that were created in order to validate the system and are already created within the system. These case studies are: A segment of the tale of "Hansel and Gretel", that represents a well-known tale, a segment of the game "The Secret of Monkey Island", that represents an entertainment story and a custom made "Restaurant making pizza" story, that represents a formation process in a restaurant.

1.6 Document Structure

This document will have the following structure: Chapter 2 explores the state-of-art that already exists and refers to related work that corresponds to the thesis themes. Chapter 3 presents an overview of the project design, going through its requirements, case studies, architecture and other algorithms analysed. Following, Chapter 4 will present details of the implementation of the system, stating all details on user customization, explaining the grid system and object database, followed by the implementation details of Automatic Generation of the environments using GPT 3.5. It is also stated all details regarding story gameplay and the explanation of all the design choices present in the system. Chapter 5 then presents the evaluation of the system made through system and user testing. The last chapter, Chapter 6, concludes this dissertation, stating the conclusions and proposing interesting future approaches that can take this subject further.

2

State of the Art

The state of the art is composed by four main sections, Procedural Generation, Game Genres,Game Map Architecture and Design and Game Exploration.

The first section will introduce what is procedural generation and how it will be used in the context of this thesis. It will also include examples of another works where procedural generation was used in a similar context. In this section, it is going to be explored different types of algorithms, such as the use of Artificial Intelligence, Marching Cubes Algorithm, Wave Collapse Function and others.

The second section will introduce different types of genres that characterize a game. It will have a bigger focus on genres that have more emphasis on the narrative, more specifically Adventure games. It will be explored what defines an Adventure game and what types of different Adventure games exist.

The third section will explain the different types of Game Architecture and Design that will be considered for this work, evaluating the advantages and disadvantages of each one, to be able to choose which one fits the objectives of this thesis the better. The types of Architecture and Design that will be evaluated are Realistic Designs and Unrealistic Designs.

The fourth section will evaluate different types of Game Exploration that exists, comparing them and analyse which one will be better to accomplishing the goals of this thesis. The different types of Game Exploration that will be evaluated are Open World games and Modular/Casual games.

2.1 Procedural Generation

In the process of creating a video game, one of the most important element that needs to exist and be given high priority in terms of importance is the creation of the environment[1]. The environment enhances the quality of the video game and has the ability to immerse the player making him more invested in the story and in the video game in general [4].

[1]Why map design is key to making game feel alive, Interviews to strategy game creators about the importance of map design, https://www.gamedeveloper.com/design/why-map-design-is-key-to-making-your-strategy-game-feel-i-alive-i-, Last access: 2023

Since the main objective of this thesis is to make a tool that allows users to tell a story in the format a video game, one of the main components is the creation of the map. The map and the environment need to be able to fit every story that the user wants to create and the process of creating it must be easy and intuitive for the user. For that reason, for this project the environment will be automatically created using Procedural Generation. Procedural generation is the creation of data using a random or pseudo-random procedure, which we will be using for the creation of the game environments. There are already multiple known algorithms and approaches for Procedural Generation, such as Noise Functions, Software Agents, Evolutionary algorithms and Physically-based erosion models [5].

2.1.1 Artificial Intelligence

Machine Learning in the creation of video games is a method that already was explored in situation such as NPC behavior [6] or adjusting the game difficulty [7]. It can even be used for play testing and bug finding, by exploring the game space, detecting key objects of the game and simulate human behaviors [8] and is a method that is rising in popularity in game development and it is predicted that the combination of Procedural Generation and machine learning will continue to show an upward trend in utilization in game development (Figure 2.1).

The use of AI to develop procedural generation content is a method that was already studied and tested in conjunction with the use of Virtual Reality to allow for unique game experiences [9].

For the player side, AI can be used to try to beat a specific game, beat a game in the fastest time or achieve perfect results in a game. For example, there are studies trying to create an AI that can solve chess [10] or an AI that performs at Pro-Level in fighting games [11].

Machine learning starts by feeding a group of several pre-made maps as a training set for the IA and then, letting the IA create the maps after learning from the training set. These method assures more variety in the final product because it does not follow a specific set of rules used by a traditional algorithm, however the disadvantage is that it requires a large amount of examples for the training set, that need to already be pre-made and increases the difficulty of having a diverse set of possibilities for the type of map to create because the AI needs to have training sets for each specific type of environment.

This type of approach using AI, was already experimented to integrate the story and the world in Computer Role Playing Games where the system finds an optimal game world configuration that supports a given story [12] and is used for generation of checkpoints for a Platform Game, as shown by Gao *et al.* [13], that aimed for generation of Super Mario Bros. levels. So, it is plausible that this method can be applied for map creation in Adventure Games.

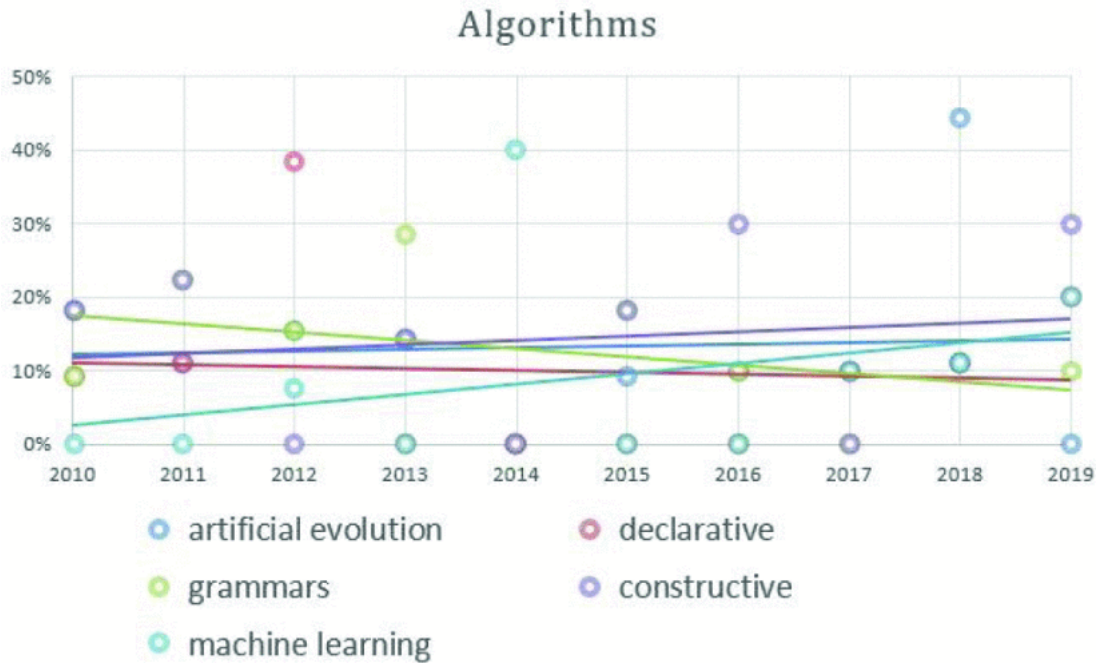


Figure 2.1: Algorithms applied in Procedural Content Generation papers [2]

2.1.2 Marching Cubes Algorithm

Marching Cubes is an algorithm that detects if an arbitrary point is within a given object. It does that by creating a triangle mesh from an implicit function. It iterates over a grid of cubes that is over a region of the function. Based on the vertices of the cube we get a triangle, and the final mesh is the union of all the triangles obtained from all the iterations[2].

It divides the space between the object and the bounds into an arbitrary number of cubes and checks for every cube if the corners are inside or outside the object. Then, there will be cubes that have some corners inside and some corners outside the object. That means, that the object must pass through those cubes, intersecting the edges that have corners of opposite classification. So, the algorithm takes those cubes and creates a point between every pair of two opposite corners and connects those points. The end result is the object[3] (Figure 2.2).

This algorithm has already been used for game creation, more specifically in map creation and design. It was used to construct 3D models of caves [14] where it was possible to produce different types of caves with intractability and different surfaces but with problems to make different shaped caves.

Other example where this algorithm is used is in a game called *Brick Block*[4]. This is the

[2]Marching Cubes, Marching cubes explanation, https://graphics.stanford.edu/~mdfisher/MarchingCubes.html, Last Access: 2023

[3]An implementation of the Marching Cubes algorithm, How the Marching Cubes algorithm works, https://www.cs.carleton.edu/cs_comps/0405/shape/marching_cubes.html, Last Access: 2023

[4]An Interview with Oskar Stalberg, Games made by Oskar Stalberg that lead up to Townscaper, https://www.gamedeveloper.com/blogs/how-townscaper-works-a-story-four-games-in-the-making, Last Access: 2023

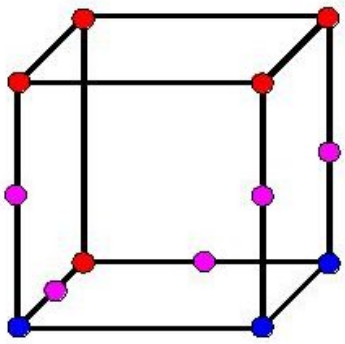


Figure 2.2: Marching cubes process in a 3D cube[5]

Figure 2.3: Brick Block example[6]

first iteration of a game called *Townscaper*, created by Oskar Stalberg, which was one of the main inspiration for the development of this thesis. In *Brick Block* the player is allowed to create a set of urban structures one block at a time in a 5x5x7 grid (Figure 2.3). For this, it is used a Marching Cubes Algorithm, that allows the creation of a polygonal mesh in a three dimensional space. By selecting a variety of points in the grid space it is possible to draw a 3D space. In this game, it is only allowed to build a new unit if the cell selected connects to a polygon already constructed in the neighboring cells. This manages to prevent the placement of rogue blocks, but can lead to structures that do not make structural sense and even floating structures (Figure 2.4).

2.1.3 Wave Collapse Function

According to Morris [15], the Wave Function Collapse is an image-based algorithm that uses constrains extracted from as input image to generate a similar, yet novel output. It

[5]An implementation of the Marching Cubes algorithm, How the Marching Cubes algorithm works, https://www.cs.carleton.edu/cs_comps/0405/shape/marching_cubes.html, Last Access: 2023

[6]Brick Blocks, 2017

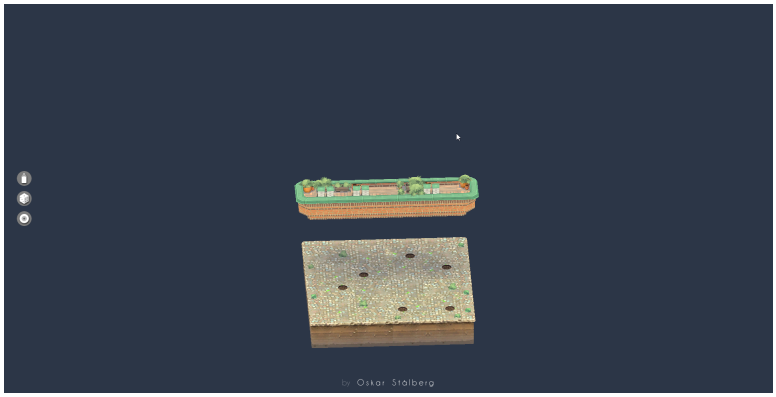


Figure 2.4: Brick Block with floating blocks[8]

transforms an initial input into many different outputs, allowing a great amount of variety from a single input set. The algorithm starts by analyzing each input pixel by pixel and generates a set of rules regarding color and local patterns. The idea is that the algorithm reads an initial image, and generates new images on is own, after learning what are the valid rules for creating new images[7] (Fig 2.5). This makes this algorithm pratical for a 2D situation since in an image it is easy to analyze the pixels and compare colors and vertex position but makes it hard to use in 3D. However, it is still possible to use it in 3D situations, for example by converting it to a non-grid shape and still having high controllability and scalability [16].

This type of function is used in a game called *Bad North*[7], which is the second iteration of *Townscaper* and it was released after *Brick Blocks*. Unlike the *Brick Blocks*, in this game having being aesthetically pleasing and realistic was not the only concern. It had the challenge of satisfying gameplay functions also. So, for this game, the solution needed the provide a lot of variety as well as insuring realism and logic. With that in mind, it was used an algorithm inspired from a Wave Collapse Function. It works in a way, where the algorithm tries to find a solution in which the variables can be assigned multiple values, but there are rules on what values can be assigned and the algorithm must reach a final solution where all the rules are being applied.

An example for game design is having assets being defined by rules, such as where can they be placed, closed to what other assets, if they can be on top of other assets, etc... The algorithm will take all those rules and try to get to a final design where all the rules are being followed and the final result will be realistic based on the rules that were implemented. When the algorithm is running, it is analysing for each asset, one at the time, the placement rules to create the world, so this type of algorithm should not be used for large worlds because the time to process will be considerably large.

The game *Townscaper* (which is the major inspiration for this thesis), is a combination of

[7] An Interview with Oskar Stalberg, Games made by Oskar Stalberg that lead up to Townscaper, https://www.gamedeveloper.com/blogs/how-townscaper-works-a-story-four-games-in-the-making, Last Access: 2023

[8] Brick Blocks, 2017

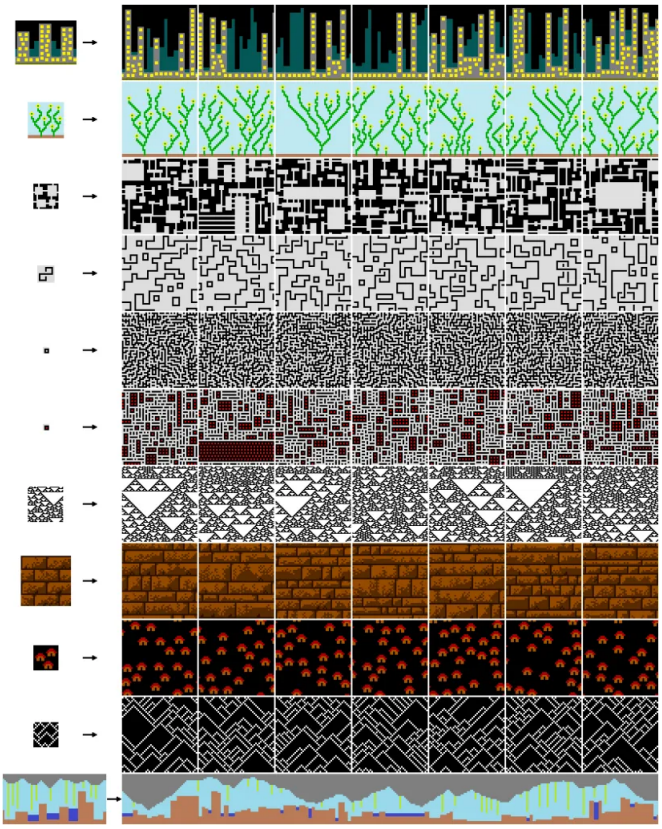


Figure 2.5: Wave Collapse Function image examples[9]

all of its previous iterations. The main difference is that uses an irregular grid to allow for more realistic and natural architecture, influencing how the tiles are built, changing them from chunks of a building to corner segments that can be more easily put together to create shapes that fit in the cells. In this game, it is used Procedural generation for the creation of the quadrilateral grids, the Wave Function Collapse that evaluates what building tiles are available based on the shape of the environment that the player has created and it used Marching Cubes to place the tiles in a way that they will fit the irregular grid.

All of the tiles besides the buildings, such as gardens,statues and stairwells are all part of the Wave Collapse Function of the game.

2.1.4 Noise Functions

Noise is a set of random numbers. In procedural generation, the noise is often added to produce more variation. The choosing of the noise is done by creating noise functions, that can create noise directly or take existing noise and modify it. There are already core noise functions that have been utilised in procedural game generation, such as Diamond-Square Algorithm, Value Noise, Perlin Noise, Simplex Noise and Worley Noise [5]. The

[9] An Interview with Oskar Stalberg, Games made by Oskar Stalberg that lead up to Townscaper, https://www.gamedeveloper.com/blogs/how-townscaper-works-a-story-four-games-in-the-making, Last Access: 2023

Table 2.1: Noise Functions comparation [5]

Algorithms	Speed	Quality	Memory Requirements
Diamond-Square	Very Fast	Moderate	High
Value Noise	Slow - Fast	Low - Moderate	Very Low
Perlin Noise	Moderate	High	Low
Simplex Noise	Moderate	Very High	Low
Worley Noise	Variable	Unique	Variable

Table 2.1 shows a comparation between these algorithms based on their speed, memory requirements and quality of noise.

This type of functions can be used in junction with other types of procedural generation algorithms, being able to generated 3D cave models using both OpenSimplex Noise and Marching Cubes algorithm [14].

2.1.5 Other Procedural Generation Algorithms

For the purpose of game development, that are several other algorithms that can be used, but are not as common. This algorithms applications can vary from map generation [5], character design, level generation [17] and others.

2.1.5.1 Minimum Spanning Trees

A minimum spanning tree is a tree that minimizes the weights of the edges of a given tree.

Using Procedural Generation creates a great challenge because it provides difficult controllability of the algorithms or limitations to specific level geometries. To overcome this problems, there was a study that used Minimum Spanning Trees to create complex indoor levels, by controlling a set of intuitive vertex and edge parameters [17]. This study was able to generate a 2D maze, a 3D maze, a Dungeon and a Space station environment and compare the memory usage based on the graph complexities of each environment (Table 2.2). They also compared time performance for each different environment for the main pipeline stages (Table 2.3):

1. Base Graph computation
2. MST computation (uses graph-weighting)
3. Level Graph Computation (includes graph cleaning and cycle-forming)

This method proves to be versatile and practical. The time measurements achieved are small enough to implement with this design approach. In the same way, the data structures require little memory resources, being almost negligible in the context of memory sizes for 3D models. It is concluded that this method of generation is viable because it is practical and versatile and do not use a lot of resources. This method still needs optimizations because it can bottleneck for very complex levels [17].

Table 2.2: Base Graph Complexities and Memory usages [17]

	Base graph		Memory Usage (in MB)
	Nodes	Edges	
2D Maze	49	64	0.04
3D Maze	216	258	0.29
Dungeon	89	185	0.12
Space Station	1056	2310	13.08

Table 2.3: Time measurements for the main pipeline stages [17]

	Base Graph	MST	Level Graph
2D Maze	47 ms	21 ms	9 ms
3D Maze	574 ms	43 ms	21 ms
Dungeon	236 ms	33 ms	13 ms
Space Station	43.75 s	381 ms	181 ms

2.1.5.2 L-Systems

The L-Systems is a versatile model that can be used for different applications, but it is mainly used for plant modelling [18, 19]. In video games it can be used for Music Composition for computer games [20], but it is mainly used for map generation. L-system is a type of system that can automatically generate "tree-like" objects, characterized by growth and branching. It contains instruction for how a single cell can grow into a more complex system. The system consists in a grammar and an Interpreter. The grammar contains an axiom, that is going to grow into a long string.

Using this system it is possible to create structure for cave paths with specified rules and be used in combination with other algorithms such as Marching Cubes to convert the volume data into mesh data [14].

This models can also be used for real time generation of 3D structures, having rudimentary game physics and interactability between the user and the L-System geometries.

There is an application that uses L-system rules, that allowing the creation of an environment for creation of 3D worlds. This application was divided in two different modes: the **play mode**, where the player is able to walk in the 3D world with controls similar to computer games, jump and rotate the camera and the **design mode**, where the development and visualization are made. In the design mode, the user is able to right-click with the mouse in a specific point of the world, to look at that point from the point of view of the user. The user can also zoom in and out and rotate the scene [21].

There is a study that uses L-system for encoding desired landscapes [22]. According to the study, the L-System allows for complex shapes, level of detail that can change rapidly and dynamically and allows to select various versions of a landscape without requiring large quantities of storage or time to render.

Table 2.4: Summary of all the algorithms studied and their utilization in map generation for video games

	Level Design	3D block generation	3D Model Generation	2D Map Generation	Indoor Mapping
Artificial Intelligence	x				
Marching Cubes		x			
Wave Collapse Function		x		x	
Noise Function			x		
Minimum Spanning Trees	x		x	x	x
L-Systems			x		x
Software Agents			x		
Evolutionary Algorithms	x				

2.1.5.3 Software Agents

Software agents can provide an alternative to noise functions for generating terrain. It was used in a system for Procedural Generation of Planetoid Terrain, using multithreading. In this case, the workers requests are spread per cloud computation instance, making it scalable per number of requests [23].

The main advantage comparing to noise functions, is that it is more controllable and can be used with designer-chosen parameters and not with random abstract number seeds [5].

Using a software agents based terrain generation allows for the user to have a greater influence on the end result of the generation, by modifying the agent constrains, being able to create terrain with different and interesting shapes, each one being unique [24].

2.1.5.4 Evolutionary Algorithms

Evolutionary algorithms are algorithms that are inspired by nature mechanisms and solve problems by emulating behaviors of living organisms. This algorithms use machine learning models to "evolve" the initial population and try to get a final optimized solution.

They are more difficult to control but provide a way of generating a large spectrum of game content types. They can provide a method for generating advanced game content by having a an initial content pool and improving it with successive generations using a fitness function [5].

This type of algorithms were already used for mobile platforms, where the main interest was implement interactive evolution based on user preferences, creating a game design that follows user preferences [25]. Interactive evolution algorithms can be divided in two types of feedback: **Reactive feedback**, which is the algorithm that requests feedback from the user or the user can intervene with the running algorithm and **Proactive feedback**, which is when the user can stop the algorithm and alter some of its parameters and then allow it to continue its process. Hui *et al.* [25], used the reactive feedback, where the user after finishing a game is required to assign a score to evaluate what rules should be kept.

2.1.6 Summary

In summary, it is concluded that different types of Procedural Generation Algorithms, can be used for different purposes and it is possible to join algorithms to take advantage of their individual qualities and reduce the disadvantages. In Table 2.4, it is displayed all the usages of the algorithms studied in this thesis, in the context of map generation.

2.2 Game Genres

Books, movies, video games and others are ways that allow a person to create a story. With the advancements of technology, the video game industry is increasing each year becoming, nowadays, one method that is used for multiple industries, such as the educational industry [26] and not only the entertainment industry. By being so versatile, it allows to achieve different purposes depending on the intention of the creator, going from education, competition, entertainment, advertisement and others. In this thesis, we are going to explore how can adventure games be used for story creation and be applied in those various industries.

Adventure games are games that focus on puzzle solving within a narrative. The main characteristics of this type of games are the narrative, puzzle solving, exploration and action.

Many studies have shown that educational games are effective tools for learning, and so it is positive to create guidelines for the creation of educational adventure games [27]. Multiple studies were made that show that adventure games are great for educational purposes, in fields like biology [28], engineering [29] and music [30]. Because they are have great focus on narrative, they are great for allowing a person to create is own stories and worlds. Usually, they do not require intensive and hard gameplay mechanics, which makes them a preferable type of game for more casual play, instead of competitive play. The definition of this type of games is very wide, so there are a lot of sub genres that can fit in the characterization of an adventure game:

- **Point and Click games:** Games where the player controls the character movement through a point and click interface, usually using a computer mouse. The player can move the character, interact with NPC's and interact with objects. Often, these games come down to explore the world and collect objects while interacting with NPC's and figure out when and where to use those objects to advance in the story (Figure 2.6a).

- **Escape room games**: These games are typically short, do not have a big space for exploration and almost no interaction with NPC's. Usually, they require the player to discover how to exit a room within a given time, by solving puzzles given clues (Figure 2.6b).

- **Narrative Adventure Games:** In these game, the main focus is the narrative. The narrative can be influenced by the choices made by the player in events throughout the game. While these choices, usually do not alter major plot points of the narrative, they can help personalize the story to the player, helping immerse the player into the game's story. This type of games, often, are characterized by conversation choices, puzzles and events with action choices for the player to choose. These games can have multiple endings and different character's deaths, depending on the player choices (Figure 2.6c).
- **Environmental Narrative games:** These are games that allow player to experience the story through exploration. These games are characterized by allowing the player to roam the environment and discover objects and interacting with NPC's. They usually do not force any time limits can even not have a final win condition, being possible to just explore the world without a final goal (Figure 2.6d).
- **Visual Novel:** These game are entirely focused on dialogue between characters. They are characterized by having dialogues trees, branching storylines and multiple endings (Figure 2.6e).

2.3 Game Map Architecture and Design

The level of detail and accuracy of the audiovisual elements in a game, can influence the player expectations on every aspect of the game, from the visuals to the story and core gameplay mechanics, however it is still really hard to quantify how much realism and immersion should a game have [4].

Because the main focus of this thesis is to allow users to create a world that allows him to play a story, one of the most important factors to consider is what kind of game architecture and design should be prioritized. Due to the fact that the map design is going to be implemented in an automatic way, using Procedural Generation, the choice of the architecture and design get extra relevance, to have a set of rules that allows the Procedural Generation algorithm to run efficiently and the user to understand how the map creation is being made and understand how to make adjustments.

The two main designs that were evaluated and compared are Realistic Design and Semi-Realistic/Unrealistic Design.

Some games choose to use Virtual Reality to create the environment [31]. In these cases, the design can be both realistic or unrealistic too, depending on the game creator intentions for the game.

2.3.1 Realistic Design

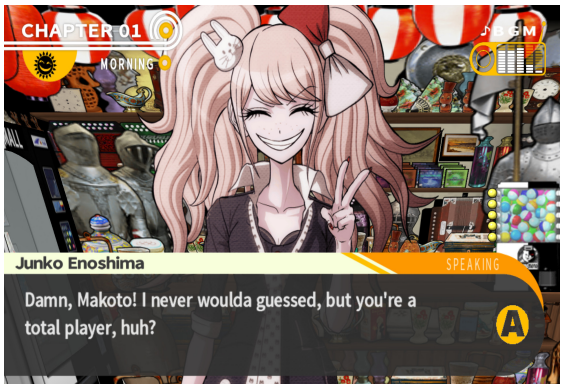
It's a realistic game design when the level of audiovisual elements and/or the mechanics incorporated in the game match to what the player can expect from the same audiovisual

(a) Point and click game (Source: Monkey Island 2 Special Edition: LeChuck's Revenge, 2010)

(b) Escape Room game (Source: The Room, 2014)

(c) Narrative adventure game (Source: Life is Strange 2, 2018)

(d) Environmental adventure game (Source: Firewatch, 2016)



(e) Visual Novel (Source: Danganronpa: Trigger Happy Havoc, 2010)

Figure 2.6: Adventure Games examples

element in real life. This type of game design is used in many of the most popular games in the world, such as GTA (Figure 2.7) and Assassin's Creed (Figure 2.8). There were already studies that even try to recreate real life environments using Google Maps integration to make game design, trying to make the game the closest possible to real world scenarios [32], projects to create automatic generation of worlds in miniatures using

Figure 2.7: GTA V vs Real Life side-by-side[10]

realistic architecture for more immersive virtual environments [33] and 3D modeling of cities that can be visualized in virtual environments, using the game engine Unity [34].

However, not only the visual aspect is important to consider. For this design, even the relationships for the characters, if made in a realistic way can give players a better, more immersive experience when playing the game [35].

For the purpose of this thesis, we need to evaluate the advantages and disadvantages of this design both for the creator and for the player. Regarding the players, it can help to make them get a connection to the game by associating it with real life events, making them be more invested in the game. For the creator it helps making him understand how the map is being created, because the rules of the Procedural Generation algorithm are based on real life designs and architectures [36], making it possible to recreate historic events and stories based on the quotidian life.

The downsides are: that having this type of game design possibly decreases the types of games/stories that the user can create, not being possible to create something that does not follow realistic rules, so making it even impossible to create some kinds of stories that are more fictional. For example, if a creator wants to create a world where every object in a room is upside down, he will not be able to because the Procedural Generation algorithm is based on realistic rules.

2.3.2 Semi-Realistic/Unrealistic Design

It's an unrealistic design when the audiovisual elements and/or mechanics of the game do not match the expectations that the players have based on those elements from real life. In the great majority of the games, the design is never full realistic or full unrealistic, the most common is to have semi-realistic designs regarding visual and audio elements and characters traits, but having some mechanics semi-unrealistic, such as not being able

[10]GTA V vs Real life side-by-side, Comparation between scenes in the game GTA V with real life locations, https://www.gtajunkies.com/gta-v-vs-real-life-side-by-side-part-2/, Last Access: 2023

Figure 2.8: Assassin's Creed Odyssey vs Real Life side-by-side[11]

to jump some obstacles (Figure 2.9), having a huge inventory when in reality it would be impossible to carry so many items and having a weapon wheel to choose the weapons (Figure 2.10).

From the creator point of view, this type of design allows for a more creative map creation because there are no definitive rules for the Procedural Generation algorithm to follow. However, it can easily get out of control and make it impossible to create intuitive and realistic designs.

From the player point of view, this type of design is a risk because it can make the player less invested in the game due to lack of familiarity with the environment of the game.

For those reasons, games almost never take a full unrealistic approach to the design, only allowing some small unrealistic environment and/or unrealistic mechanics.

2.4 Game Exploration

Besides the aesthetic and mechanic components, another component that needs to be given attention is the exploration component. In educational games, usually the aesthetic and mechanic components are present but the exploration component is not given to much attention and effort [37].

[11] Assassin's Creed Odyssey vs Real Life, Comparation between scenes in the game Assassin's Creed Odyssey with real life location, https://togetherintransit.nl/assassins-creed-odyssey-greece-comparison-to-real-life/, Last Access: 2023

Figure 2.9: Not being able to jump over obstacles[12]

Figure 2.10: Weapon Wheel and Large Inventory[12]

When creating a system it is important to consider both its scalability and interactivity. In the case of a video game the main question regarding this topics is if regarding to gameplay exploration the game should have an Open World approach or a Modular Architecture approach. This decision will greatly impact the complexity of the system and how scalable the system can be.

2.4.1 Open World

This approach in games, consists on having a large explorable world, where the player is free to explore wherever he decides and usually can approach the available objectives of the game in any order at any time. This type of approach is the complete opposite of having a linear and structure gameplay. Usually, in this type of concept, the game is

[12]Unrealistic mechanics in realistic games, Examples of unrealistic mechanics that can be found in realistic games, https://www.thegamer.com/unrealistic-mechanics-in-realistic-games/#not-being-able-to-jump-over-obstacles, Last Access:2023

setup in an outdoor environment typically lacking in structures such as walls and doors (Figure 2.11a). In the grand majority of the games that use this type of exploration, the world borders are defined by inaccessible geographic features such as oceans or mountains that fill in as background (Figure 2.11b). This type of environment can be achieved using Procedural Generation [13], although it is still a field with a lot of potential to explore.

The main advantage of having this type of game exploration is the ability to give more freedom to the player, by giving the ability him the ability to choose how to approach the game in the order that he wants, but still constrained by the gameplay rules. In this type of games, the player is free to explore the map even without a concrete goal, so for the creation of this game, there needs to be a lot of emphasis on map design, because the story is not the only main factor to make the player engaged.

Other important aspect to take into account, when developing an Open World game is the needed interactions between the player and the NPC's [38]. Because the player is free to explore the world, having NPC's and being able to interact with them is crucial for immersion and is useful because it provides extra entertainment for the player.

In Open World games, often the player needs to develop their character, fulfilling a unrealistic fantasy (Figure 2.11c) but in some cases, the game can focus on realistic aspects (Figure 2.11d), giving the world a realistic design.

The main disadvantages of the open world exploration is the fact that requires a lot of software processing to be able to generate a big world and the fact the world needs to be interesting enough to keep the players engaged and entertained.

To facilitate the hurdles of creating the world, sometimes Procedural Generation is used. This in an important asset to reduce game development time and opens the possibility of creating worlds with more variety in less time with few resources.

Nowadays, this type of games are very common, with some games having sold millions of copies such as No Man's Sky (Figure 2.11e) and Minecraft (Figure 2.11f). These type of game exploration can be used in educational games, where using an open world and virtual reality in conjunction, can help motivate the players to learn about the content of the game while exploring it [39].

2.4.2 Modular Architecture

This approach consists in dividing a large system in smaller parts, making it more easy to manage and extend whenever is needed. This can be very advantageous because it can provide a lot of scalabaility, and makes doing changes easier. It also, does not require so much resources because it operates in a smaller scale. There are many types of products that use this type of architecture, such as "Open Wonderland" that is a toolkit for building 3D virtual worlds, that is highly modular and focuses on extensibility [40].

In the context of this thesis, when we are talking about this type of architecture in a video game, it means dividing the map into smaller individualized parts, where each one is built independent of the other and then merged together. Doing this allows the usage

(a) Outdoor environment (Source: Elder Scrolls 5: Skyrim, 2011)

(b) Inaccessible world borders (Source: Ark Survival Evolved, 2015)

(c) Unrealistic fantasy (Source: Elden Ring, 2022

(d) Realistic game (Source: Grand Theft Auto: San Andreas, 2004)

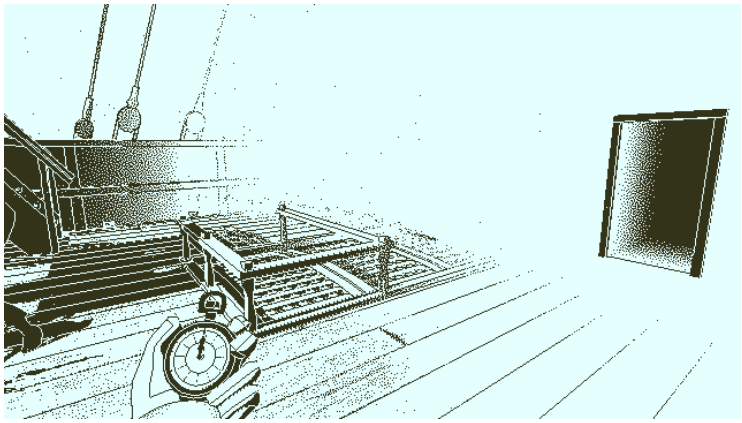
(e) No Man's Sky (Source: No Man's Sky, 2016)

(f) Minecraft (Source: Minecraft, 2011)

Figure 2.11: Examples of games with Open World environments

of the Procedural Generation Algorithms on a smaller scale, which also improves the run time of those algorithms improving the system in general. Using this architecture, it turns the system into a more casual game, because especially for creating the world, it makes it that the "creator" can create smaller parts of the world one at a time, which help the user because it can create is own story at a smaller scale.

Many games use this type of architecture in the gameplay, being very common in



(a) Mystery/Investigation game (Source: Return of the Obra Dinn, 2018)

(b) Older Point and click game (Source: Please Love my Computer Game, 2018)

mystery/investigation games (Figure 2.12a) and older point and click games (Figure 2.12b), where the player only access one room at a time and "teleports" between rooms to navigate.

3

Analysis and System Design

This chapters is an analysis of the system that is going to be implemented. The first section are the Requirements that we intend to achieve for the final product. The second section consists on the Case Studies that were done with the intention of evaluation the system. The third section is the explanation of the Architecture of the system, detailing what are the system components and how they interconnect with each other. The fourth section is the Algorithms Analysis, where it is explained the different algorithms that were tested before arriving at the final implementation of the system.

3.1 Requirements

This section defines the main requirements that the system needs to achieve in the end. Following the research questions, the framework needs to allow the creation of a wide variety of stories and also be able to play them. The requirements necessary can be divided in this categories:

User Requirements: Two types of users were identified as the users that will interact with the final product. A "creator" user, that is responsible to create the world of the game, being able to choose the initial setup of the game and adjust with Assisted Generation. The other is a "player" user, which is responsible for playing games created by the "creator", being able to choose a game and play it.

System Requirements: To achieve the goals of this thesis, the system will need to be composed by a framework to create a stories in a video game format, with interfaces to adjust the placement of elements generated by the Automatic Generation, which needs to be playable with characters, point and click movement, realism and fluidity achieved by animations and micro-interactions.

3.2 Case Studies

For this thesis, we proposed some particular case studies with the aim of evaluating the system's effectiveness. Specifically, the case studies revolved around the creation

Figure 3.1: Restaurant Case Study

Figure 3.2: Hensel and Gretel Case Study

and modification of a restaurant environment and the development of a story involving a cook preparing a pizza (fig 3.1), the recreation of part of a famous tale "Hensel and Gretel" (fig 3.2) and the recreation of a segment of a game "Secret of Monkey Island" (fig 3.3). To accomplish this, users were required to utilize the system's intend functionalities, including environment and storyboard creation, as well as gameplay. This case study was selected due to its simplicity and clarity, while also demonstrating the system's capacity for more complex scenarios. To achieve this goal, all intended functionalities were fully functional, and the required assets were readily available in the system.

Figure 3.3: The Secret Of Monkey Island Case Study

3.3 Architecture

The system's architecture is designed to empower interactive storytelling, facilitating the creation, playability, and storage of diverse stories. It is composed of three distinct components: **Create**, **Play**, and **Cloud API**. Each component plays a crucial role in enabling the creation of dynamic and creative narratives.

3.3.1 Create component

The Create component (fig 3.4) is the foundation for the ability to craft different captivating stories. It consists of two main components: **World** and **Storyboard**.

The **World** represents the core for storytelling. It encompasses all the objects, characters, animations and interactive elements. Objects range from all the different object types, while characters include the player controlled character and non-playable characters (NPC's), each with their respective animations and interactions.

The **Storyboard** comprises multiple steps that define the narrative's flow. Each step represents a interaction between objects, characters and animations. Animations play a pivotal role in conveying feedback and advancing the story-line. This steps form a dynamic sequence that guides the user and the story progression and interactivity.

3.3.2 Play component

The Play component (fig 3.5) is what allows to experience all the creative process from the Create component. It is divided into four main components: **Game Engine**, **Storyboard**, **World** and **Inventory**.

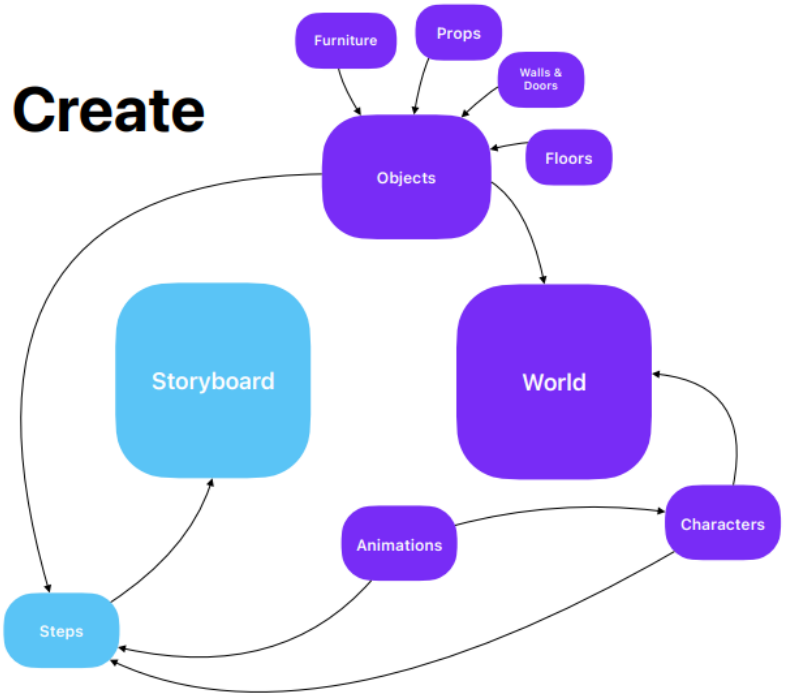


Figure 3.4: Architecture of Create feature

The **Game Engine** acts as the core system and it enables the executing of the story. It interprets the instructions from the **Storyboard** component to orchestrate all the events and interactions within the game. This components is what ensures the playability of the story, responding to user actions and choices.

The **Storyboard** acts with the similar function as in the Create component. It serves as the guide to the narrative's progression during the gameplay. It gives instructions to the **Game Engine**, dictating how the story should unfold based on the player's choices nad interactions.

The **World** components represents the playable characters, as well as the non-playable character's and objects. It is directly connected to the **Inventory**. Players engage with the environment through these elements, while making choices and decisions that are going to affect the narrative. The **Inventory** is the repository of objects that are available for interaction during the gameplay. It connects to the **World**, allowing players to manipulate and interact with objects as the story progresses.

3.3.3 Cloud API component

The **Cloud API** (fig 3.6) serves as the backbone for storing, sharing and accessing stories. It is built upon the combination of data collected from the **World** and **Storyboard**, encapsulating the entire narrative in JSON format.

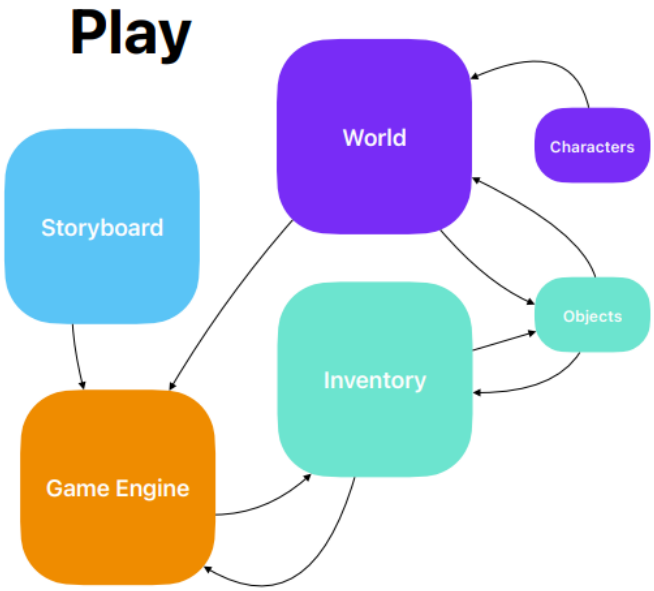


Figure 3.5: Architecture of Play feature

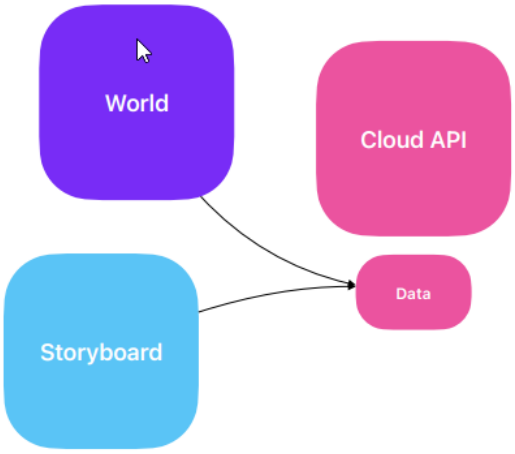


Figure 3.6: Architecture of Cloud

3.4 Algorithms Analysis

Prior to reaching the final system implementation, various methods were tried with the objective of creating an algorithm capable of automatically generating realistic and well-structured scenarios. The initial approaches commenced with the utilization of Procedural Generation techniques without Artificial Intelligence and later were changed to incorporate Artificial Intelligence methodologies. This section explains the diverse iterations of automatic generation that were analysed before arriving at the final implementation, which will be presented in Implementation.

3.4.1 Procedural Generation

3.4.1.1 Random generation

The first iteration of the automatic placement of objects. In this iteration, the objects were placed without any rules and constraints, making the generated rooms completely unrealistic and nonsensical. Having no real logic in the algorithm resulted in rooms with no logic.

3.4.1.2 Probability algorithm

The next iteration of the automatic placement of objects was using a probability based algorithm. To create better object placement, the algorithm categorizes objects into specific classes. For example, objects may be classified as "Chairs," "Desks," "Props," and so on. This categorization is used to allow different relationships between objects of different classes. Then, probabilities are assigned to each relationships between the object classes. For instance, it can be decided that the probability of a "Chair" being adjacent to a "Table" is 75 percent. These percentage are based on design principles and are pre-defined parameters, with the intention of guaranteeing realism and variety, while generating environments that should match user expectations. The algorithm evaluates each tile on the grid. It considers all the adjacent tiles and all the objects already placed on those tiles and use the object probabilities to determine if a certain object is suitable placement for that tile. This method based on probabilities is not practical, since the algorithm can be slow because it has to check all the tiles. Additionally, to ensure great variety and realism, there must be a lot of different classes for the objects, which exponentially increases the running time. Another limitation of this algorithm lies in the subjectivity of the criteria when choosing the classes and probabilities.

3.4.2 Artificial Intelligence

After deciding using Artificial Intelligence in the Procedural Generation for the Automatic Generation of the environments, the biggest obstacle to surpass was creating a training set of environments that needed to be big enough, comprising of thousands of different environments to use in a machine learning algorithm. These training set needed to be large and the environments needed to be realistic (because we follow a realistic design system), and so the challenge was how to create such a large data set in a conceivable time frame. To achieve this, many methods were analysed.

3.4.2.1 Manual Compilation for training set

The first Artificial Intelligence method analysed was to create all the environments required for the training set manually using the user customization system 4.1. While this approach, gives some advantages, it also poses some limitations. The main advantages of using this approach is that allows for system testing, since creating the environments, would

serve as a test to the system's capabilities allowing to identify any weaknesses or areas for improvement. This would also result in a proof of concept, because being able to successfully generate a significant number of environments manually, would demonstrate and prove the system's ability to create a large variety of environments in a simple and effective way. Despite the advantages, this approach also has poses limitations and challenges. Since machine learning algorithms require a large data set, this requires the creation of thousands of environments manually with the subjective creativity and environment understanding of the creator. This implies problems regarding large diversity and complexity. The data set needs to have a wide variety of environments, each with unique characteristics, layouts and sizes to be able to handle diverse scenarios. By creating the environments manually, the algorithm would be extremely biased towards the creator sense of realism and creativity, which would not result in many diverse scenarios, due to the possible lack of knowledge about architecture and design of the creator. In conclusion, while the manual creation of environments would offer several advantages, such as system testing and proof of concept, it was ultimately rejected due to not providing enough variety and quality to those environments, while being creator biased.

3.4.2.2 Chat GPT

Another solution analysed for the creation of the training data set involved utilizing a LLM: Chat GPT, for the creation of the required environments in JSON format. The process is to give the required format for the environments and explain the different types of objects and assets that are available to use. After, ask Chat GPT to give realistic environments (in JSON format), based on the object system and assets available. This approach offers potential benefits, such as assuring quantity and diversity, as Chat GPT's generation abilities allows for the fast creation of a large number of environments with different characteristics. However, despite the advantages, using this approach also comes with some challenges and limitations. First, it is required to give a definition of realism to Chat GPT. The subjectivity of this definition introduces an element of bias into the final results of the generated environments. The lack of the AI's understanding of realism, a little biased towards the user definition of realism, however it is still a viable solution comparing, since the biased is compensated by the constant improving "Intelligence" of the LLM. Other problem arrives when it becomes necessary to make changes in the object system or object database in the future. In that case, it is required, to re-engage with Chat GPT, making the adjustments manually to account for those changes and redo the training data set with those changes in consideration. This considerably reduces the scalability, not accounting for constant future changes, since the training set will always be heavily based on the objects system that are available on a particular time. In conclusion, this method offers a promising solution to the data set quantity and diversity requirements. However, challenges related to the definition of realism that can cause bias, and adaptability for future changes were introduced. While this approach, provides a valuable solution, it

also highlighted the importance of finding a method that addresses the subjectivity of defining realism, to ensure the database's robustness and improve scalability.

3.4.2.3 Overview

This section comprises a summary of all methods evaluated during this thesis to implement the automatic generation intended. The table 3.1 illustrates the different approaches tested and their correspondent advantages and disadvantages.

Table 3.1: Comparing the different hypothesis for automatic generation

	Realism	Speed	Variety	Non-user biased
Random Generation		x	x	
Probability algorithm	x		x	
Manual compilation for training set	x			
Chat GPT	x	x	x	

3.5 Summary

This chapter provides a comprehensive overview of the essential components required to address the research questions. It delineates the system's requirements and introduces the architecture of the entire system. Furthermore, it clarifies the various hypothesis previously analysed in the context of automatic object generation for environment creation, that ultimatly led to the final iteration incorporated in the system.

4

Implementation

This chapter will detail the implementation of the system based on the architecture described in Section 3.3. This implementation is separated in 2 main sections: User Customization that consists on the system being able to provide ways to manually create, delete and modify different environments using a set and scalable database of objects, to allow the user to be able to make adjustments that might be needed. And Automatic Generation that consists on the system using automatic placement of the objects without needing any input from the user. This was implemented using LLM, in particular GPT 3.5.

4.1 User Customization

The system needs to allow users to think of a story and be able to manually customize an environment that fits the needs to that story and be allowed to save it, changed it or delete it in a easy way. For this, the system needs to allow the placement and removal of objects, as well as having different animations and interactions that fit the desired theme.

4.1.1 Object Database

The object database (fig 4.1) is a fundamental feature for this system. Because the main goal is to allow the user to create any kind of story, the objects that are available must account for many different scenarios. The user is provided with several office and kitchen objects that allow for the creation of various environments specified in Case Studies. To account for the variety of environments that we wanted to introduce, one of the main focus was to allow the addition, removal and alteration of the objects in the database in a simple way, as well as give the objects specific characteristic to increase the realism and variety to the scene. Those characteristics include: name, size, type of object and scale. The objects can be of seven different types, and can only be one object of each type per tile on the map:

- **Furniture:** This type of objects are the default type for any new created object. This type can mainly be used for the most common items like tables, desks, etc...

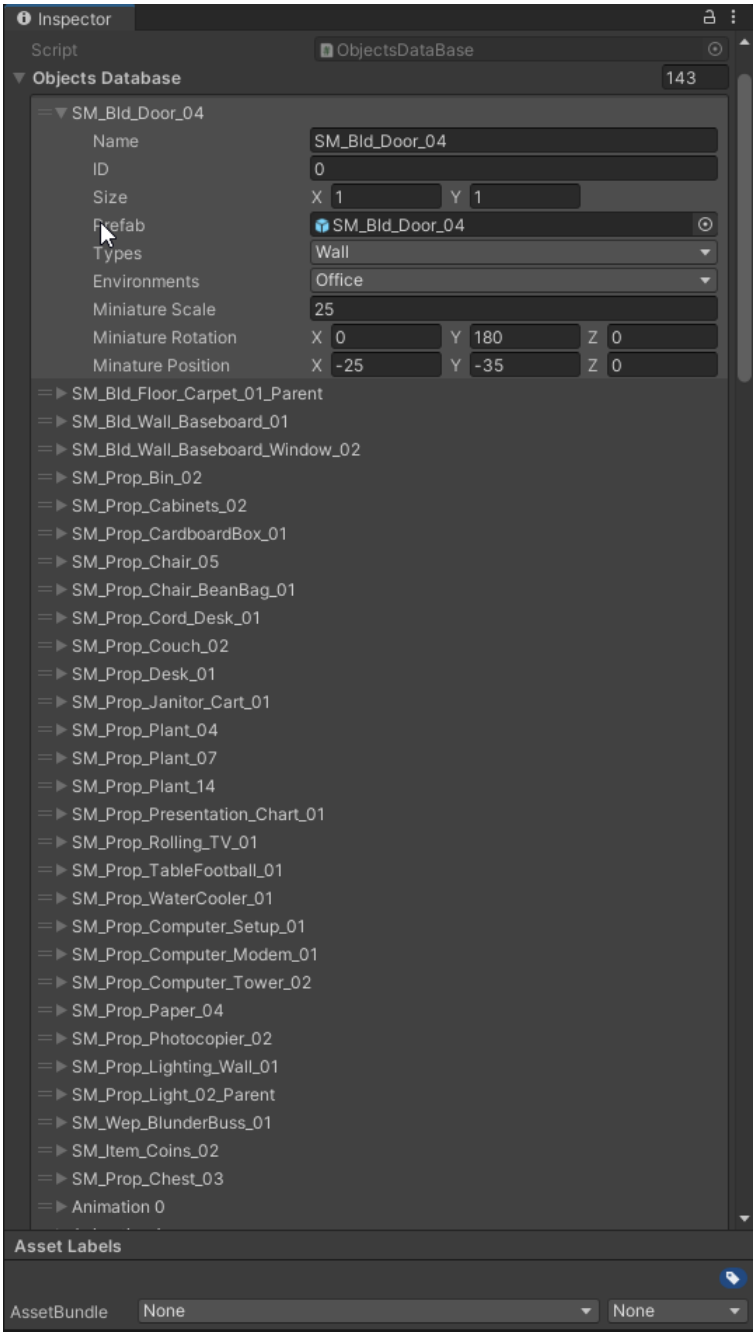


Figure 4.1: Object Database

- **Floor:** This type of items are for the flooring of the map. For example. can be carpets or floor tiles for indoor maps, or grass or dirt for outdoor maps. The map will always be created with a selected floor in all tiles, that can be removed later for manual adjustments.

- **Props:** This types of objects are objects that usually want to be placed in conjunction with another object, like on top or inside, such as a lamp on top of a table or a cup inside a bowl. In those cases one object should be "Furniture" type and the other

should be "Props" type. This type is always placed on top of last placed object on a tile that is "Furniture" or "Floor".

- **Wall:** This type of objects are the objects that need to be placed on the edge of a tile to create a border, such as walls, windows and doors. This type of objects are an exception in the sense that they are the only type that can be placed more that once in a tile.
- **Wall Prop:** This types of object are objects that can be placed on "Walls", such as paintings or wall lights.
- **NPC**: This represents the non playable characters that can be used as as figurants in the environment. These NPCs can have "Animations" and can be interacted with in the storyboard.
- **Animation:** This is an object that can be placed on a "NPC" to give it an animation during the gameplay or to put in the storyboard steps to determine what reaction the main character does while completing a step of the story.

To better distinguish the objects, they have a tag that identifies what type of environment that usually belong, to allow the user to filter objects based on that tag, showing the objects that are related on a specific environment. This allows for a big catalog of objects in the database to provide variety while still maintaining organization to help the user while creating an environment. With the objective of adding life to the scene, it is also possible to add living beings to the scene, such as other humans or animals. These act like the other objects, in the sense that they can be interactable. could be used as a part of the story, and can have animations. Each object can have animations, that can help the user creating a specific story or simply add realism to the environment even if the objects are not in direct relation with the story. These animations can be for normal objects, such as drawers opening, doors opening, lights turning on and off, or to the living beings, such as humans laughing, humans falling or pets moving.

4.1.2 Tile Grid System

The tile grid system (fig 4.2) is a fundamental aspect of the system. It serves as the canvas on which the users are able to place, move, and manipulate objects to design their desired environments. This section explains the functionality of the tile grid system, highlighting its adaptability to different grid sizes and the current implementation where each object occupies a single tile.

4.1.2.1 Object Placement

The tile grid system enables the users to place objects on the grid. Each object occupies one tile, each tile can only have one object of a specific type (Object Database) but can

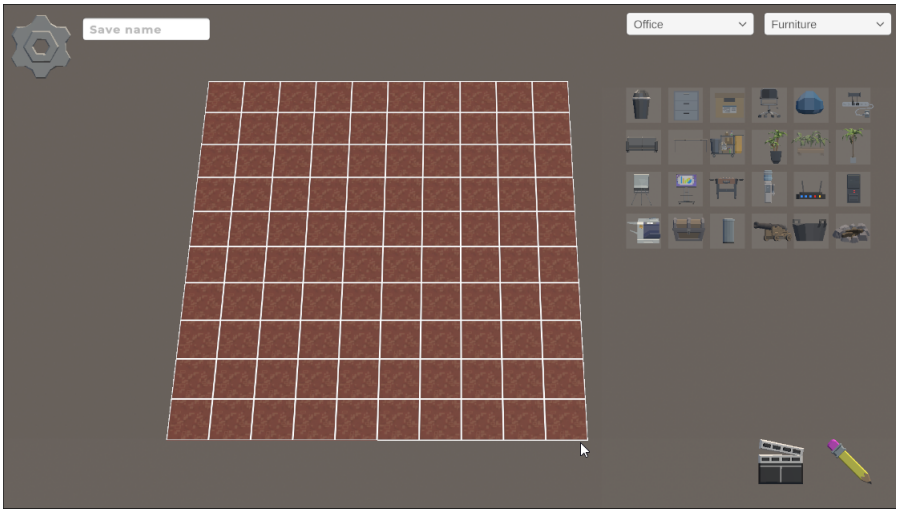


Figure 4.2: Grid representation

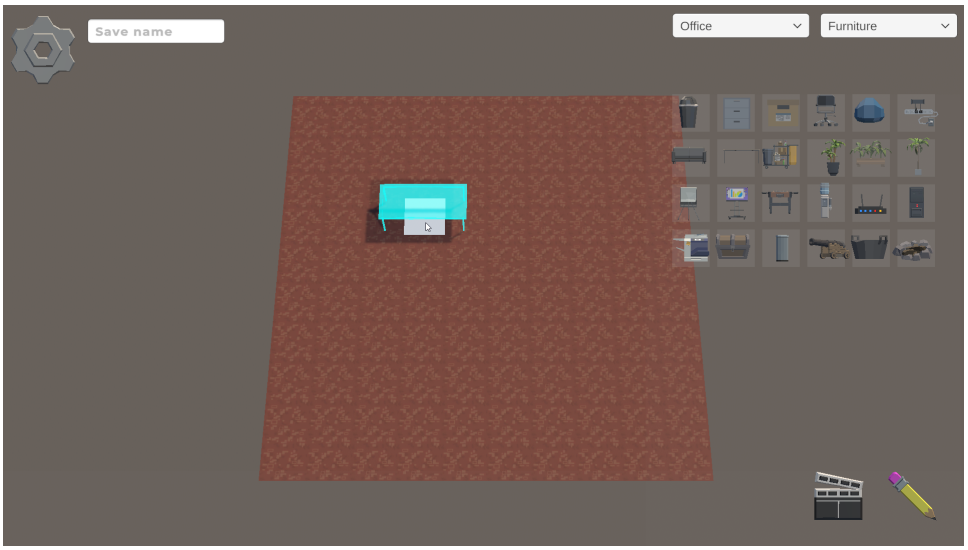


Figure 4.3: Placement of an object

have multiple objects if they have different types. This limitation ensures that users are allowed to easily create diverse and realistic environments while preventing unintended overlaps or conflicts between objects, as well as making the system more intuitive and easy without many complicated possibilities. Users can simply select an object from the available database and place it onto the grid by clicking on the object in the object menu and after clicking on desired tile (fig 4.3). The objects are all placed on the center of the tile, except "Wall" type objects that are placed on the edges of the tile. While placing the object, the users also have the ability to change its rotation by factors of 90º degrees, allowing for the placement of "Walls" on all the edges of a tile (fig 4.4). This allows the users to achieve more personalized environments that fit their needs. The process of object placement is also enhanced with the use of object previews. Before finalizing the placement of an object, a preview of the object will appear on the hovered tile, simulating an hologram

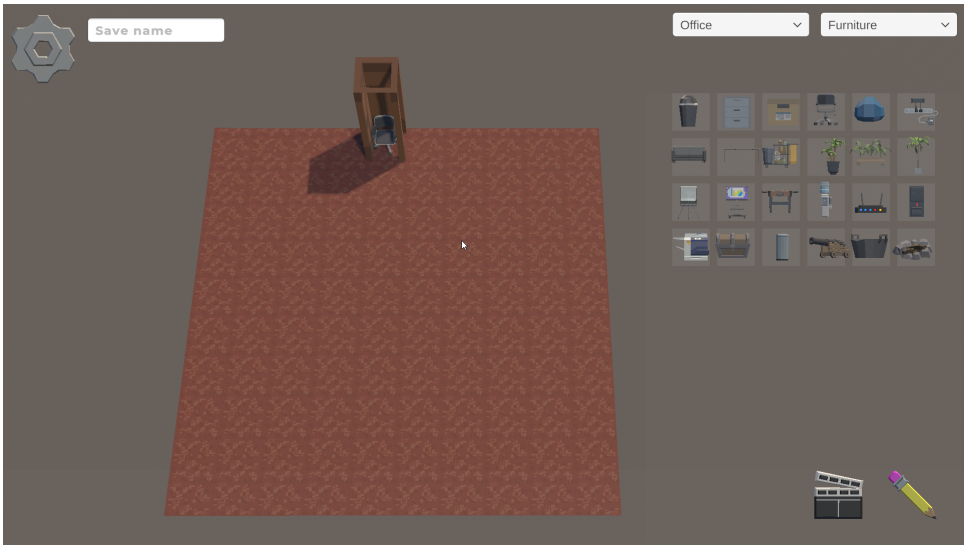


Figure 4.4: Multiple "Walls" on the same tile

that represents how the selected object is displayed on the grid. This object preview serves multiple important functions:

- **Validation of Placement:** The object preview allows users to visually assess whether the chosen object can be placed on the selected tile. If the hologram appears with the color blue, it indicates that the object can be placed there without conflicts or violations of the one-object-per-tile rule. However, if the hologram appears with the color red, that would signal to the user that the selected object is unavailable for the placement of the selected object, avoiding potential issues.

- **Preview of the object appearance:** The object preview also assists the user giving a visual representation of how the object will appear in the environment once placed. This allows users to assess the object size, orientation, and alignment relative to other elements in the environment. This helps the user on making informed decisions about object placement and ensures that their creative vision is accurately realized.

- **Real-time Feedback:** The object preview is interactive in real-time, responding to user input. Users can move and rotate the hologram before confirming the final placement of the selected object, giving them an opportunity to experiment with object positioning and orientation to achieve their desired outcome.

- **Intuitive User Experience:** This feature wants to enhance the overall user experience by providing instant feedback to reduce the likelihood of errors and facilitate the understanding of the whole object placement system. Users are allowed to experiment with object placement in an intuitive way, making the process of environment creation more engaging and user-friendly.

4.1.2.2 Object Deletion

Once an object is placed on the grid, users have the ability to delete it from its position. To achieve this, the user must select the delete mode and select the tile to delete objects from. The objects are deleted using a reverse order of placement, meaning that the last object placed on a tile is the first object being deleted. This works for every type of object, including walls, floors and NPC's.

4.2 Automatic Generation

In this environment creation system, the tile grid system not only facilitates for the manual placement and deletion of objects, but also facilitates and integrates with automatic object generation. This section explores how the objects are automatically generated in the system.

4.2.1 GPT 3.5

This was the solution adopted for the generation of environments in this thesis. It involved the integration with "GPT 3.5", a LLM developed by Open AI, that can take an input text and transform it into what predicts a useful results will be. Using this approach solved some problems that were raised in the other solutions, such as the adaptability to future changes and eliminating the need to create a custom machine learning algorithm, while maintaining the advantages provided by the previous solutions. Similar to the previous approaches detailed in **??**, it is needed to specify the available object types and database to "GPT 3.5". This input serves as the foundation for the environment generation. However, "GPT 3.5" as an essential advantage that is the ability to interact with an API. This allows for integration with Unity via code, facilitating direct communication between the application and the model. This addresses the adaptability issue, as the input required for the model, can update automatically because it is already connected directly with the application's code. Because "GPT 3.5" is already a machine learning model, by utilizing it, it is eliminated the need to create a custom machine learning algorithm, which also eliminates the need for a training set, as "GPT 3.5" already returns the final output for the application. A problem that emerged was the same problem identified in previous approaches: the definition of realism. Similar to previous approaches, defining realism remained a challenge. "GPT 3.5" also relies on user input to determine what constitutes a realistic environment, introducing subjectivity and potential bias. The process of establishing a connection with OpenAi's API involves creating a POST request with a specific URL endpoint, detailed in OpenAi's documentation [1]. This request includes essential components such as specifying an array of messages that serve as prompts for GPT and identifying the specific GPT model intended for the task. Each message is comprised of a "Role", indicating if it originates

[1]OpenAI API-reference, https://platform.openai.com/docs/api-reference/chat/create, Last Access: 2023

```
// Create a JSON object with the necessary parameters
ChatRequestData chatRequestData = new ChatRequestData
{
    messages = new Message[] { new Message { role = "user", content = prompt } },
    model = "gpt-3.5-turbo-1106",
    temperature = 0.5f,
    max_tokens = 200,
};
```

Figure 4.5: Parameters used in the API connection

from the "System" or the "User", and "Content", signifying the message's actual content. While other parameters are optional, this project utilized "Temperature" to express the randomness of the generated responses and "maxTokens" to delineate the token limit for the responses (fig 4.5).

4.2.1.1 Prompting

Upon establishing a connection with OpenAI's API, there is the need to make requests with maximum precision and clarity to instruct GPT effectively. This thesis aims to use GPT for the automatic creation of environments based on a specific list of objects and available space. To achieve this, three key components are necessary within the prompt (Fig 4.6).

- The list of objects available for the environment creation.
- The grid size that representing the space available for environment creation.
- The required response format.
- Rules to dictate the position of the objects.

The **List of objects** is dynamically acquired within the code, drawing from the pre-established database of the system. This setup enables the automatic generation of various types of environments. Any modifications to the database of available objects, will be automatically reflected in the algorithm, ensuring that it always utilizes the intended object database and not be outdated.

The **Grid Size** is manually specified and, for this thesis, was set as 10x10 tile grid. However, this can be easily adjusted to accommodate various grid sizes.

The **Format** for the response must conform to the JSON format utilized saving and loading environments in the system (fig 4.7). This JSON format includes an array of objects, each defined by a name, position (comprising row, column and height) and rotation (ranging from zero to four, representing the four cardinal directions). This array represents all the objects within the environment.

The **Rules** form the foundation of the algorithm, dictating object placement. These rules are established based on Object Types and include the following specifications:

- There are six object types: Furniture, Floor, Prop, Wall, Wall Prop and NPC.

```
"Create an environment for a grid of 10x10 size, using the specified objects and following the specific rules," +
" with the end result being in the required format:" +
"objects: SM_Prop_Desk_01 (Furniture), SM_Prop_Chair (Furniture), SM_Prop_Bowl (Prop), SM_Prop_Lamp (Prop)" +
"rules: Each tile can only have one object of each type, except Wall (that can be multiple);" +
"All tiles must have a floor in height 0;" +
"Some tiles should exclusively have floor" +
"Prop should be positioned on top of furniture" +
"Wall Prop must share a tile with Wall" +
"format: name: (x,y,z)";
```

Figure 4.6: Example of a prompt to GPT

- Each tile can only have one object of each type, except Wall (that can be multiple)
- All the tiles must have a floor in height 0.
- Some tiles should exclusively have floor.
- Prop should be positioned on top of furniture.
- Wall Prop must share a tile with a Wall.

This rules not only define the quality and realism of the environment but also offer adaptability based on specific requirements. The specific rules were successfully employed in the testing phase (Chapter 5), attesting to the creation of an acceptable environment. Importantly, this ensures scalability, allowing for incorporation of any future rules related to environment generation.

4.3 Story Gameplay

One of the core goals of the system is to also offer an engaging playing experience. To achieve this, we implemented a playing mode that allows users to experience the previously created worlds and interact with the narrative. This playing mode utilizes a 3D top-down point-and-click view (fig 4.8). This perspective is common in adventure games and some story based games, and provides users with a comprehensive overview of the environment while offering intuitive navigation and interactions. This view was used with the intention of creating a balance between realism and user-friendliness, allowing users to explore and be engaged with their stories with no difficulty. In this mode, users navigate their character through a point-and-click mechanism. They can guide their character by clicking in the desired location to explore or to interact with object and non playable characters within the environment. The object interactivity is much important, because it accounts for the story progression, that was created in the storyboard. To enhance immersion and give feedback to the user about their actions, it is implemented several micro interactions, animations and visual cues to inform users of the outcome of their choices (fig 4.9). This feature also has an inventory system that allows the user to collect items if interacted with (defined in the storyboard), and use those items to interact with others in the environment by dragging them with the mouse (fig 4.10). Users can save

```
"TaleWorld": {
                        "type": "object",
                        "properties": {
                            "ObjectsInWorld": {
                                "type": "array",
                                "items": {
                                    "type": "object",
                                    "properties": {
                                        "Name": {
                                            "type": "string"
                                        },
                                        "Position": {
                                            "type": "object",
                                            "properties": {
                                                "_row": {
                                                    "type": "number"
                                                },
                                                "_column": {
                                                    "type": "number"
                                                },
                                                "_height": {
                                                    "type": "number"
                                                }
                                            }
                                        },
                                        "Rotation": {
                                            "type": "object",
                                            "properties": {
                                                "_direction": {
                                                    "type": "number",
                                                    "enum": [
                                                        0,
                                                        1,
                                                        2,
                                                        3
                                                    ]
                                                }
                                            }
                                        }
                                    }
                                }
                            }
                        }
                    }
```

Figure 4.7: JSON format

their progress at any point of the story, allowing them to return to the narrative later. This ensures that the storytelling is flexible and can accommodate varying durations.

The implementation of this system is the solution to provide users an interactive storytelling experience. By integrating the creation and playing systems, we empower users to both craft, share and explore their narratives in a video game framework. This mode represents a crucial component of the system's vision, allowing for diversity and immersion for their stories.

Figure 4.8: 3D Topdown View

Figure 4.9: Character animation while interacting

4.4 Design

This section is about the User Interface (UI) design that was chosen and implemented for the environment creation system. The main guiding principles that was followed included prioritizing user-friendliness, efficiency, and accessibility, to ensure that even users that could be unfamiliar with this kind of applications find it easy and intuitive to use. To achieve this goal, some considerations such as usage with minimum amount of clicks, avoiding text overload, hard focus on iconography and micro-interactions were taken in account, as well as taking inspiration from adventure game designs and similar applications. The UI design for the Create Mode (fig 1.1) specifically caters to the creation of small, enclosed environments, because this application's purpose is more focused in those kind of environments, instead of bigger and open-world setups.

Figure 4.10: Inventory Object dragging

4.4.1 Minimizing User Effort

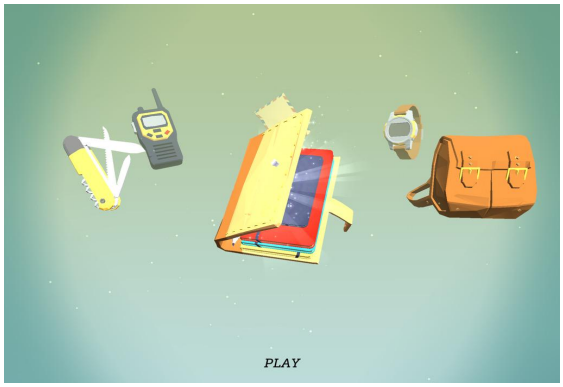
The main object objective was to reduce the number of clicks required for any action. Simplicity and intuitiveness were mandatory. The UI needed to be straightforward, ensuring that all types of users could uses it and understand with no difficulty, regardless of their familiarity with similar applications.

4.4.2 Avoiding Text

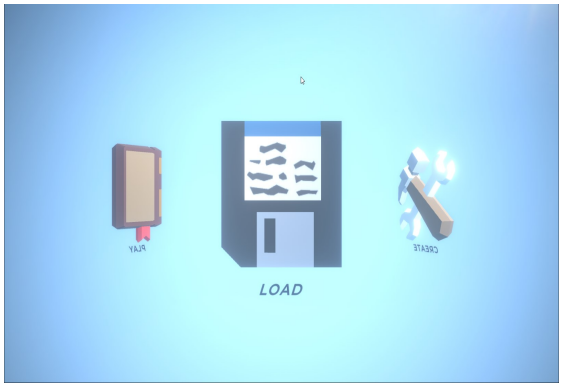
The approach aimed to avoid the use of very large amounts of text, focusing instead icons and micro-interactions to be able to give the information without overwhelming the users. Icons were pivotal for the separation and understanding of all possible actions that the user can do, while micro-interactions intends to engage users, providing feedback and guidance. One concern of avoiding the usage of text is the possibility of making the UI harder to understand in the sense that the user needs to recognize the icons, or at least, the icons need to be intuitive enough to the user to be able to infer what are the functionalities of each icon. To try to achieve this, the icons chosen were mainly icons that usually appear in other application or places, that have a correlation with the intended functionality.

4.4.3 Inspiration from Adventure Game Designs

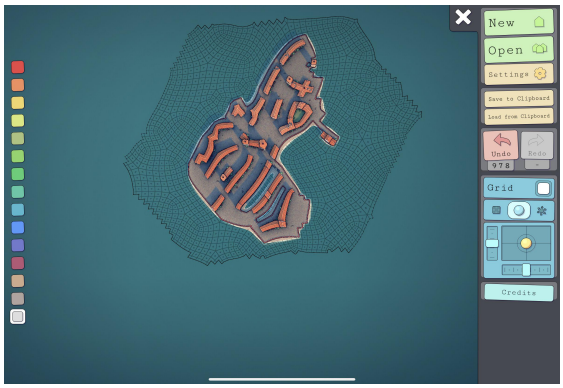
Regarding the UI's aesthetic, it was heavily inspired in other adventure game designs or other applications with a similar concept. The main menu and inventory system is inspired by games like "Lara Croft Go" (Fig 4.11a), while the UI for the Grid System is inspired by "Townscaper" (Fig 4.11c), which as similar functionalities.



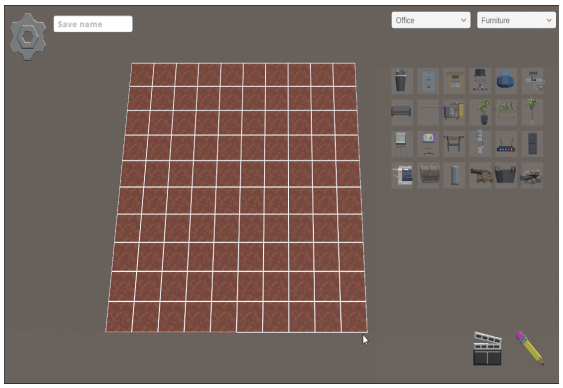
(a) "Lara Craft Go" Main Menu



(b) System Main Menu



(c) "Townscaper" grid system and interface



(d) System Tile Grid and interface

Figure 4.11: Design inspirations

4.5 Summary

This chapter provides an overview of the implementation of this thesis's system. It explains features such as user customization options, describes in detail the methodology for automatic environment generation, providing an insight into the utilization of GPT 3.5 for this purpose.

Next it shows the implementation of the gameplay component, providing insights into the user experience during gameplay, highlighting all the narrative interaction processes.

5

Evaluation

To address the research questions outlined in Section 1.2 and assessing the fulfillment of the objectives present in Section 1.3, a user study was conducted. This chapter explains the methodology for conducting the user study, providing details of the protocol, participant demographics and assignments implemented throughout the study. It also presents the conclusions taken for the study, with a dedicated section (Section 5.4) for the discussion of the results and a comparative analysis. The main objective of this user study was to assess the application's quality, by using a combination of customized questions and established questionnaires to evaluate all features presented within the system.

5.1 User Study

5.1.1 Protocol and Specification

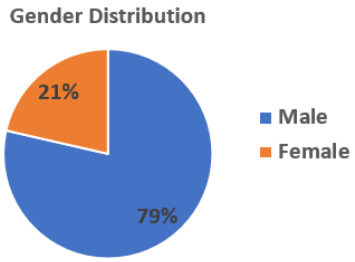
The test session required two people to be communicating, the researcher and the test subject. It could be in person or online via discord. The test subject starts by reading the informed consent, present in Appendix A and agreeing to it. After, the researcher will give the application containing the system to the test subject and give a some generic information about the goals of the application. Then, the user opens the application and if the test is being made online, the user shares his screen with the application open, to start the tasks. The tasks consist in actions that the user is asked to do interacting in the application to achieve general goals. These tasks are grouped into larger blocks of tasks, referred as Assignments, and are done sequentially. Each assignment tests a specific feature of the system. For each task, the researcher fills a form with small observations and indications if the user could or not finish the task and at the end of each assignment, the user also fills a form with some specific observations regarding the tasks and the assignment. All of the data collected is done manually with some of it being objective data, such as the time to completion or if the completion was achieved, and some being subjective data, such as realism and intuitivity of the system. All the assignments, help gathering data that contribute to get a better insight of what changes need to be done to the application and what is already working, to be able to answer with the most precision

the research questions related to this thesis. The first assignment evaluates the quality and functionality of the environment generation and user customization. The second assignment is related with the playing feature of this system. It evaluates the quality and functionality of being able to play with no difficulty any story that was previously created.

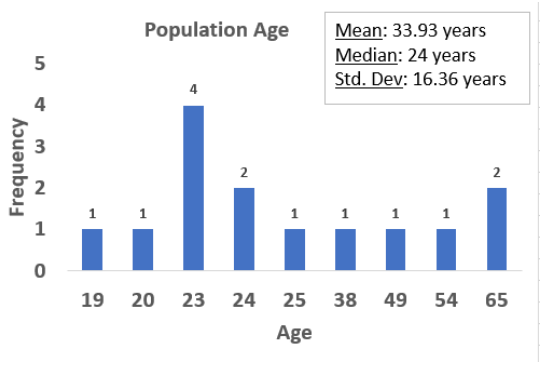
- **RQ1- How can we use adventure game frameworks to allow a user to create and tell his stories?**
- **RQ2- How to create a game in a fast and easy way?**
- **RQ3- How to assist in the world building?**

5.1.2 Population

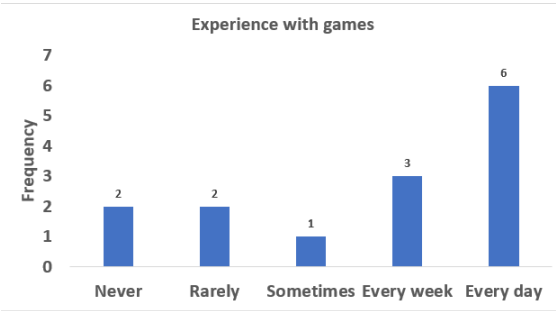
The population for the tests was composed of 14 participants. We evaluated 5 different demographic factors within this population: The gender, age, gaming experience, game genres preferences and game platform preferences. The objective was to encompass a wide range of characteristics, with particular emphasis on evaluating their familiarity with games, as our system's target audience includes individuals that don't have any gaming or coding knowledge. The analysis of game genre preferences and platform preferences aimed to understand and the participants' backgrounds in games to facilitate comparisons with our system's mechanics. Gender information for the population can be found in Figure 5.1a, while their ages are in Fig 5.1b. The majority of the population identified as male (79%), consisting of eleven males, with the remaining 21% representing females. The ages varied between 19 and 65 years, with a mean age of 33.93 years, a median of 24 years and a standard deviation of 16.36 years. Concerning the participants' experience with games, the majority of the population (43%), equivalent to 6 users, indicated having a lot of experience, as they claim playing games daily. On the opposite, 2 users reported no prior gaming experience, stating that they never play games. As shown in Fig 5.1c, the population showed a diverse range in gaming experience, with at least 1 participant per category. This align with the system's objective of encompassing a wide spectrum of gaming knowledge. Concerning game genres and platforms, our intention was to specifically assess the background of participants with at least some prior gaming experience and compare it with components of the system. Given that our system operates in a 3D topdown point and click interface and it is primarily designed for adventure games, this data allowed us to evaluate whether the participants were already familiarized with controls and features associated with this system. For this particular questions, the participants had the opportunity to select multiple options. As illustrated in Fig5.1e, Strategy games were the most frequently played genres, with 10 selections, followed by Action and Adventure games, with 8 selections. Fig 5.1d demonstrates that the computer was overwhelmingly the preferred gaming platform, with 11 selections, constituting 91.7% of the total choices.



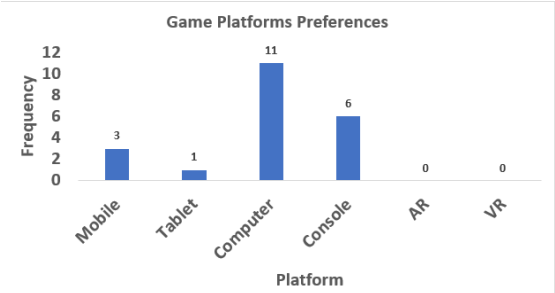
(a) Gender Distribution



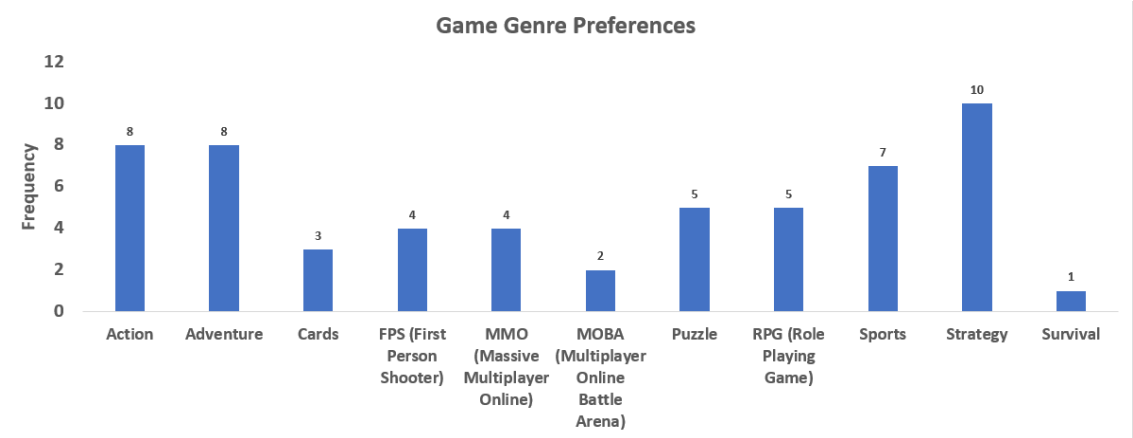
(b) Population Age



(c) Experience with games



(d) Game Platforms preferences



(e) Game genres preferences

Figure 5.1: Population demographics graphs

5.1.3 Assignment A- User adjustments to an environment generated automatically

The first assignment focuses on one of the main features of the system: The manual interactions. It teaches the user about the tile grid system and the object database. In addiction to evaluate this systems that constitute the manual interactions, this assignment also as focus on evaluating the quality of the automatic generated environments. This assignment will be create a base environment, to be used in later assignments, specifically

the creation of a pizza restaurant environment.

5.1.3.1 Task A1- Select the save "Test" and enter create mode

In this task, the user is instructed to select a specific save from a list of saves and enter the create mode. No more directions are given and the user needs to understand how to navigate through the options in the menu to achieve this goals. Some questions about the layout and design of the main menu are asked and some observations are written by the researcher.

A1Q1- Able to complete? The answer can be *Yes, without help*, *Yes, with help* or *No*.

A1Q2- Time to complete The answer is the time to completion in seconds, timed by the researcher.

A1Q3- Design of the main menu The answer can be *Very Bad*, *Bad*, *Neutral*, *Good* or *Very Good*. The answer is given by the user.

A1Q4- Observations The researcher can write some observations made by either person during the task.

This task's job is to understand how easy it is to navigate through the main menu and how easy and evaluate the design and interfaces utilized.

5.1.3.2 Task A2- Put three total objects on separate tiles of the room

In this task, the user is instructed to put 3 specific objects on different tiles of the room: a dinning table, a kitchen counter and a sink. The user needs to understand how to filter the objects to find objects of different types and also needs to understand how to put object on the grid. Questions about the preview system are asked in addiction to the usual observations of the researcher.

A2Q1- Able to complete? The answer can be *Yes, without help*, *Yes, with help* or *No*.

A2Q2- Time to complete The answer is the time to completion in seconds, timed by the researcher.

A2Q3- Was the preview system helpful? The answer can be *Yes* or *No*. Answered by the user.

A2Q4- Changes to the preview system? The answers are open but are related to the *color* and *transparency* of the preview.

A2Q5- Observations The researcher can write some observations made by either person during the task.

This task intents to evaluate the usability of the filter system and also evaluate the utility of the preview system.

5.1.3.3 Task A3- Put two objects on the same tile as other objects

In this task, the user is instructed to put two specific objects on top of other specific objects. The user is instructed to put a pizza with toppings on top of a free kitchen counter and

to put a box with sauces on top of a free dining table. Questions about the types of the objects are asked to the user.

A3Q1- Able to complete? The answer can be *Yes, without help*, *Yes, with help* or *No*.

A3Q2- Time to complete The answer is the time to completion in seconds, timed by the researcher.

A3Q3- Do the object types make sense? The answer can be *Yes* or *No (in this case, specify what why)*. Answered by the user.

A3Q4- Observations The researcher can write some observations made by either person during the task.

This task evaluates if the one object of one type per tile system and the quality of the object types that are already defined.

5.1.3.4 Task A4- Delete two objects

In this task, the user is instructed to two specific objects: the mob bucket and the table that has the sauce box on top. The user needs to understand how to delete objects. Questions about the delete system are asked to the user.

A4Q1- Able to complete? The answer can be *Yes, without help*, *Yes, with help* or *No*.

A4Q2- Time to complete The answer is the time to completion in seconds, timed by the researcher.

A4Q3- Does the delete order (in the same tile) make sense? The answer can be *Yes* or *Should be different (in this case, specify how)*. Answered by the user.

A4Q4- Observations The researcher can write some observations made by either person during the task.

This task evaluates if the delete system is intuitive and easy to use, as well as if the quality of the decision for deleting by reverse placement order of objects in the same tile. The chosen objects to delete were chosen specifically, to show how the feature of removal of objects works, by choosing an object that is the only one in a tile and other objects that is on a tile with multiple objects.

5.1.3.5 Task A5- Change between all the different functionalities of the create mode

In this task, the user is instructed to navigate between all the different functionalities of the create mode.Firstly, the user needs to go to the "Storyboard" mode, after go to the "Testing" mode and in the end return to the "Environment" mode. The user needs to understand how to navigate through the different modes, as well as be able to identify which icon represents which mode. Questions about the identifiability of each mode are asked to the user.

A5Q1- Able to complete? The answer can be *Yes, without help*, *Yes, with help* or *No*.

A5Q2- Time to complete The answer is the time to completion in seconds, timed by the researcher.

A5Q3- Which modes are easily identifiable? The answer can be *Environment Mode*, *Storyboard Mode* and *Testing mode*. Multiple answers can be chosen by the user.

A5Q4- Observations The researcher can write some observations made by either person during the task.

This task shows the user the different functionalities that are available in the create mode and evaluates the quality of the icons chosen for the design, judging by the user's ability to navigate between the different functionalities.

5.1.4 Assignment B- Playing the story

This assignment focuses on another of the features of the system: Playing a story. It teaches the user how to play any story that was previously created and teaches the different mechanics that are involved while playing, such as the inventory or object dragging. This also helps the user to understand how the create mode and the Play mode are connected, by seeing the changes previously done on the Create mode, being applied on the Play mode. In this assignment, the player will play a story that is about the creation of a pizza in a pizza restaurant, using the environment created in 5.1.3.

5.1.4.1 Task B1- Select the save "Test" and enter Play Mode

In this task, the user is instructed to enter the play mode with the save "Test" selected. No more directions are given and the user needs to understand how to navigate through the main menu. Some observations are written by the researcher.

B1Q1- Able to complete? The answer can be *Yes, without help*, *Yes, with help* or *No*.

B1Q2- Time to complete The answer is the time to completion in seconds, timed by the researcher.

B1Q3- Observations The researcher can write some observations made by either person during the task.

This task's job is to understand how easy it is to navigate through the main menu and how easy and evaluate the design and interfaces utilized.

5.1.4.2 Task B2- Move the character in the environment

In this task, the user is instructed to move the character within the environment. No more directions are given and the user needs to understand how to move the character.

B2Q1- Able to complete? The answer can be *Yes, without help*, *Yes, with help* or *No*.

B2Q2- Time to complete The answer is the time to completion in seconds, timed by the researcher.

B2Q3- Observations The researcher can write some observations made by either person during the task.

This task's job is to show the user how to move the character while playing and evaluate the quality of the controls, as well as the camera movement and positioning, while playing the game.

5.1.4.3 Task B3- Pickup the "Mushrooms"

In this task, the user is instructed to pickup "mushrooms" that are located somewhere in the environment. The user needs to be able to identify the "mushrooms" and understand how to pickup an object that was previously defined in the storyboard.

B3Q1- Able to complete? The answer can be *Yes, without help*, *Yes, with help* or *No*.

B3Q2- Time to complete The answer is the time to completion in seconds, timed by the researcher.

B3Q3- Observations The researcher can write some observations made by either person during the task.

This task's job is to evaluate the intuitiveness and utility of the picking up objects mechanic, while also evaluating the quality of the assets chosen, by assessing if the user could easily identify the specific object that needed to be picked up. This also, shows the user that it is possible to pickup items and store them in the inventory.

5.1.4.4 Task B4- Interact with the "Pizza oven"

In this task, the user is instructed to interact with the "pizza oven". The user needs to identify where and what is the "pizza oven" and understand how to interact with the object. Questions about the animations with be asked to the user.

B4Q1- Able to complete? The answer can be *Yes, without help*, *Yes, with help* or *No*.

B4Q2- Time to complete The answer is the time to completion in seconds, timed by the researcher.

B4Q3- Did the character animation show that the correct order of the story was not being followed?. The answer can be *Yes* or *No*.

B4Q4- Observations The researcher can write some observations made by either person during the task.

This task's job is to show the user the ability to interact with different objects, as well as show that different reactions may occur based on if the correct order for the story is being followed or not.

5.1.4.5 Task B5- Put the "mushrooms" of the inventory in the "pizza"

In this task, the user is instructed to put the "mushrooms" that are located on the inventory, on the pizza that is on the environment. The user needs to understand how interact between the inventory items and environment items. Questions about the intuitiveness of the item utilization are asked.

B5Q1- Able to complete? The answer can be *Yes, without help*, *Yes, with help* or *No*.

B5Q2- Time to complete The answer is the time to completion in seconds, timed by the researcher.

B5Q3- Is the item utilization intuitive? The answer can be *Yes* or *No*.

B5Q4- Observations The researcher can write some observations made by either person during the task.

This task's job is to show tho the user that it is possible to interact with environment objects with item that were picked up and are in the inventory. This also evaluates how intuitive is this mechanic and how difficult it is to use.

5.1.4.6 Task B6- Put the "Pizza" in the "Pizza Oven"

In this task, the user is instructed put the "pizza" in the "pizza oven". This task is similar to the previous and there are no new mechanics that the user needs to learn.

B6Q1- Able to complete? The answer can be *Yes*, *Yes, with help* or *No*.

B6Q2- Time to complete The answer is the time to completion in seconds, timed by the researcher.

B6Q3- Observations The researcher can write some observations made by either person during the task.

This task's job is show the ending to the story that was build previously in the storyboard.

5.2 Results

This section will go over the results and insights obtained during the user study. It will follow the order of tasks describer in Assignment A- User adjustments to an environment generated automatically and Assignment B- Playing the story. Each task will be analysed individually, and in the end there will be a general overview of the results taking in account all of the tasks in the assignment.

5.2.1 Assignment A: User adjustments to an environment generated automatically

Assignment A intends to evaluate one of features of the system: the environment generation and customization. The tasks are simple interactions that show the user the different types of customization that can be done do an environment, as well as show and evaluate a automatically generated environment created by the system's algorithm. Figure 5.2 shows the average time needed for the completion of each task in the assignment, with the respective standard deviation. As shown in Fig 5.3, the average time from the completion of the Assignment A in the totality was 298.5 ±132.82 seconds, which corresponds to almost 5 ±2.2 minutes. The median was 267.5 seconds or 4.5 minutes. The fastest time was by Participant 11, who did it in 154 seconds (2.6 minutes) and the slowest time was by Participant 9, who did it in 584 seconds (9.7 minutes).

This indicates that in average the process of creating an environment for a story and customize it with simple and small changes takes 5 minutes and at most 10 minutes. Due to the fact, that the changes implemented by the participants were pre-determined by the

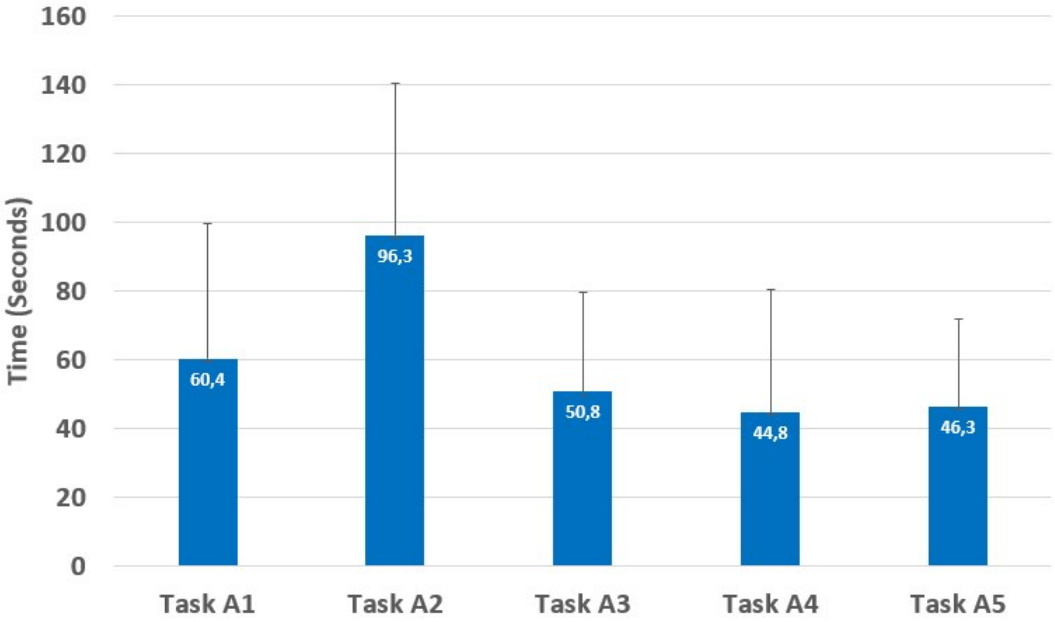


Figure 5.2: Assignment A - Average time per task

tasks and they were small changes, it is possible to assume that the process to create and customize a more complex environment would take longer, however assuming the user's starts getting use to the mechanics while using the system, it is possible to assume that the timings would not go much higher, comparing to the results from the testings. This can be corroborate, by the results of the tasks, considering that Tasks A2 and A3 had similar mechanics and in the majority of the cases the user took longer to do Task A2 than A3. This infers, that the user got used to the mechanic previously learned (placing objects) and used it in the next task, faster.

5.2.1.1 Task A1: Select the save "Test" and enter create mode

The first task introduces the participant to the system's main menu and the concept selecting saved stories and entering the create mode. The data collected from this task is synthesised in Table 5.1. The participants were observed if they could complete the task without help, with help or didn't complete it. The time necessary to complete the task was taken by the researcher. The participants were also asked to give an evaluation of the Main Menu design, from 1(Very Bad) to 5(Very Good). We conclude that the main menu's design might have some problems, observing in Table 5.1 that the majority of the participants only were able to do the task with help and in addiction, observing in Fig 5.4, the majority of the participants evaluated the design has a 3, evaluating it as average. The median is 3.5 and the mean is 3.58, which concludes that despite having some design problems, the main menu is on a decent state with just some tweaks to make it more intuitive. The average time to complete the task was 57.5 ±39.42 seconds, with a median of 57.5 seconds. Some observations made by the participants in this task, were the excessive

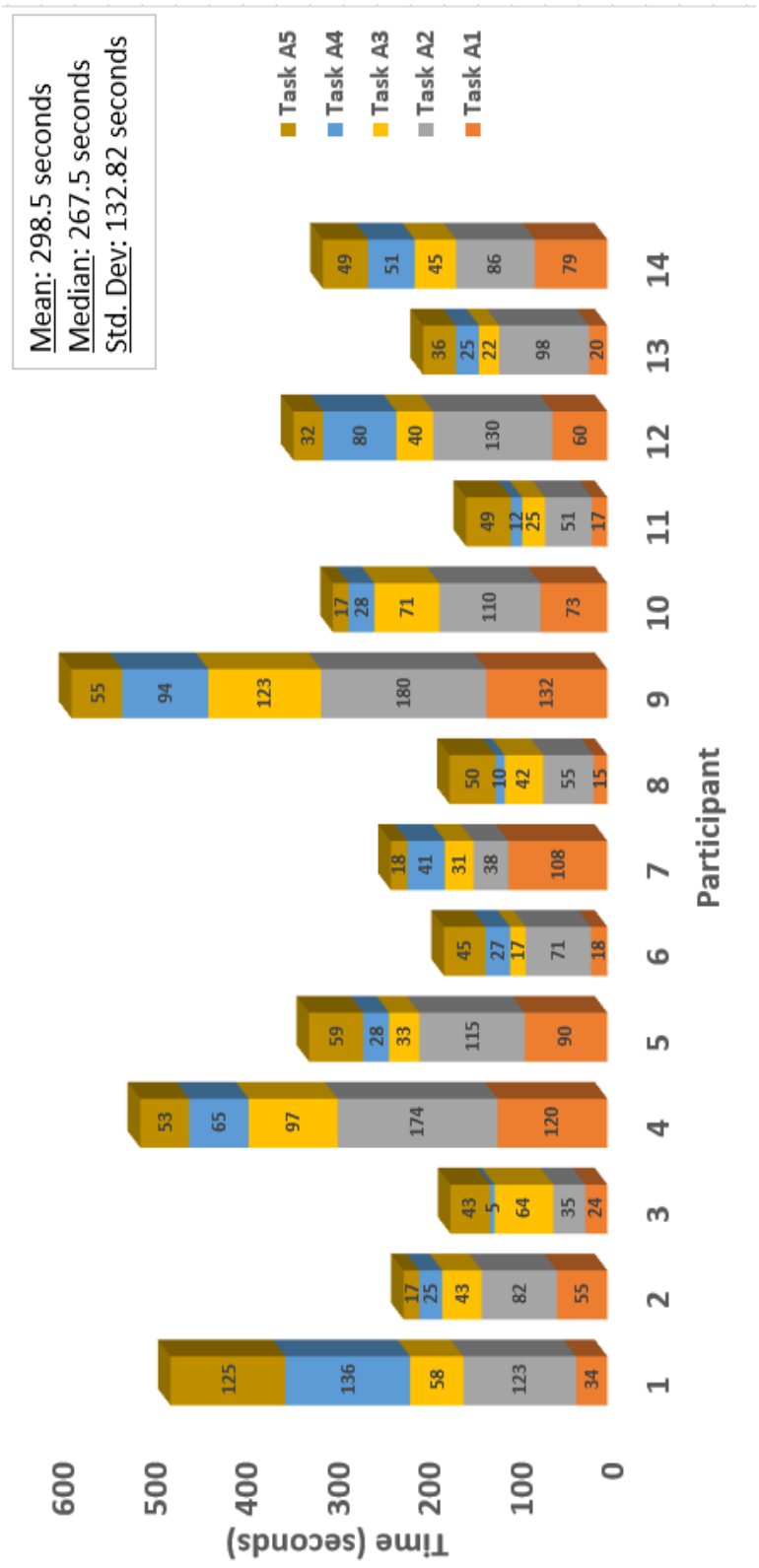


Figure 5.3: Assignment A - time to complete overview

brightness and low visibility of some User Interface that made the navigation through the menu harder.

Table 5.1: Task A1 results

Complete	Answer(#)
Yes, without help	5
Yes, with help	9
No	0

	Time (seconds)
Median	57.5
Mean	60.36
Std. Dev.	39.42

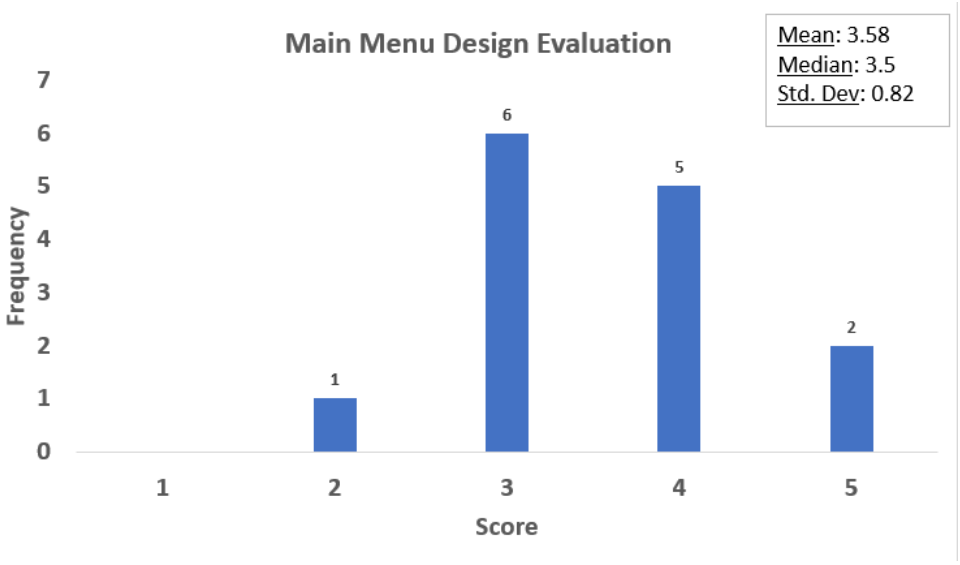


Figure 5.4: Main Menu Design Evaluation

5.2.1.2 Task A2: Put three total objects on separate tiles of the room

The second task introduces the participant the concept of adding objects to the environment. The data collected from this task is synthesised in Table 5.2. The participants were observed if they could complete the task without help, with help or didn't complete it. The time necessary to complete the task was taken by the researcher. The participants were also asked about the feature of previewing the objects before their placements in a hologram appearance, if it was useful or not and if they would make any change, such as changes to color or transparency. Based on the results in Table 5.2, we conclude that the placement of the objects in not an intuitive mechanic, since 50% of the participants were only able to finish the task with help from the researcher. However, regarding the Objects Preview System, the results were much positive, as only 2 of the participants didn't find it a useful feature (Fig 5.5).The average time of completion was 96.29 ±44.16 seconds and the median was 92 seconds. Some observations made by the participants in this task, included the hard understanding of the objects available displayed and suggesting dragging the objects to place them in the environment instead of clicking them.

Table 5.2: Task A2 results

Complete	Answer(#)
Yes, without help	7
Yes, with help	7
No	0

	Time (seconds)
Median	92
Mean	96.29
Std. Dev.	44.16

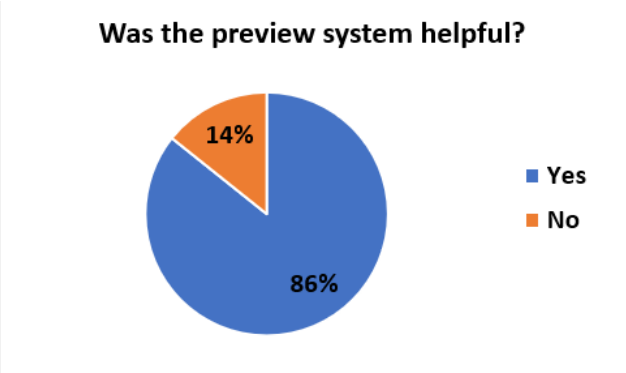


Figure 5.5: Object Preview System evaluation

5.2.1.3 Task A3: Put two objects on the same tile as other objects

The third task introduces to the participant the concept of objects types and the ability to add objects on tiles that already have other objects. The data collected from this task is synthesised in Table 5.3. The participants were observed if they could complete the task without help, with help or didn't complete it. The time necessary to complete the task was taken by the researcher. The participants were also asked about the feature and nomenclature of the different object types, if it was intuitive and easy to understand. Based on the results in Table 5.3 and in Fig 5.6, we conclude that the separation of the objects in different types is useful, mainly as a filter mechanic, as only 1 of the participants did not find the feature intuitive and the majority of the population (79%) was able to finish the task without help, while the others (29%) could finish it with help. The average time of completion was 50.79 ±28.81 seconds and the median was 42.5 seconds. The key observation made by the participants in this task, were focused around the nomenclature of the types of objects, especially the "Props" which some participants only could understand via process of elimination.

Table 5.3: Task A3 results

Complete	Answer(#)
Yes, without help	11
Yes, with help	3
No	0

	Time (seconds)
Median	42.5
Mean	50.79
Std. Dev.	28.81

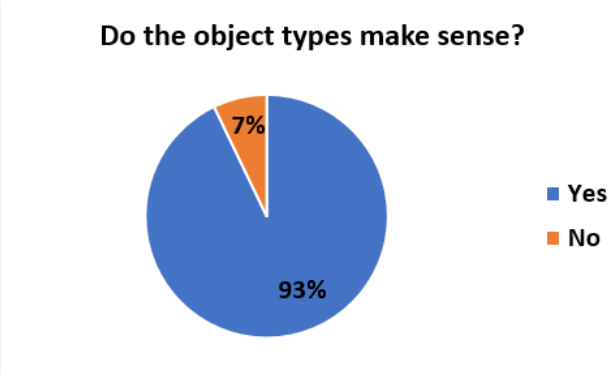


Figure 5.6: Object types evaluation

5.2.1.4 Task A4: Delete two objects

The fourth task introduces to the participant the concept of deleting objects from the environment and how it works. The data collected from this task is synthesised in Table 5.4. The participants were observed if they could complete the task without help, with help or didn't complete it. The time necessary to complete the task was taken by the researcher. The participants were also asked about the order of deleting objects from the same tile, if they understood it and found it intuitive. Based on the results in Table 5.4 we conclude that the process of deleting was not intuitive, as only 8 participants were able to finish the task without help and 6 with help. However, in Fig 5.7, we conclude that the order to delete the items is much intuitive and satisfactory to all the participants, except 1. The average time of completion was 28 ±35.83 seconds and the median was 28 seconds. Despite 8 participants not being able to finish the task without help, the average time for the conclusion of this task was relatively low. This is due to the fact that the participants could not understand how to enter delete mode, expecting an icon to click on the screen, but after being told the correct way to enter the mode, they managed to complete the task very fast.

Table 5.4: Task A4 results

Complete	Answer(#)
Yes, without help	8
Yes, with help	6
No	0

	Time (seconds)
Median	28
Mean	44.79
Std. Dev.	35.83

5.2.1.5 Task A5: Change between all the different functionalities of the create mode

The final task of the Assignment A introduces to the participant the different modes that exist inside the Create Mode: the environment, the storyboard and the testing modes. The participants were observed if they could complete the task without help, with help or didn't complete it. The time necessary to complete the task was taken by the researcher. The participants were also asked which of these three modes were easily identifiable

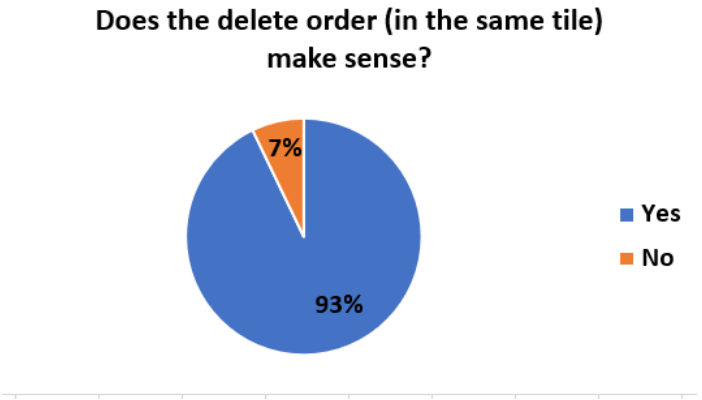


Figure 5.7: Object Delete order evaluation

and easy to understand what is their purpose. Based on the results in Table 5.5 we conclude that the process of changing between the different modes was not intuitive, as 8 participants were only able to finish the task without help and 6 with help. In Fig 5.8, we conclude that the environment mode is the most identifiable, with 6 selections, followed by the storyboard mode, with 4 selections, and the testing mode, with only 3 selections. This is mainly because, the environment mode is the default mode when entering the Create section, and the participants had already experience mechanics of the environment mode and not the other modes. However, having the 3 of the modes with less than 50% of selections, because each one could be selected 14 times (1 per participant) indicates that the different modes are not easily understandable and their functionalities are not intuitive. The average time of completion was 46.29 ±25.84 seconds and the median was 47 seconds.

Table 5.5: Task A5 results

Complete	Answer(#)
Yes, without help	6
Yes, with help	8
No	0

	Time (seconds)
Median	47
Mean	46.29
Std. Dev.	25.84

5.2.2 Assignment B: Playing the story

Assignment B intends to evaluate another of the features of the system: the gameplay. The tasks are simple interactions that show the user how to play a story that was created. It shows the different mechanics within the gameplay, such as object interactivity, inventory system, player movement and information through character's animations. This assignment's main goal is to evaluate, the intuitively and simplicity of the gameplay, since one of the objectives of the system is to give to the user, the ability to not only create but also play the stories in a video game format. This must be intuitive and simple enough, so that even users with no gaming knowledge are able to quickly experiment and play the stories

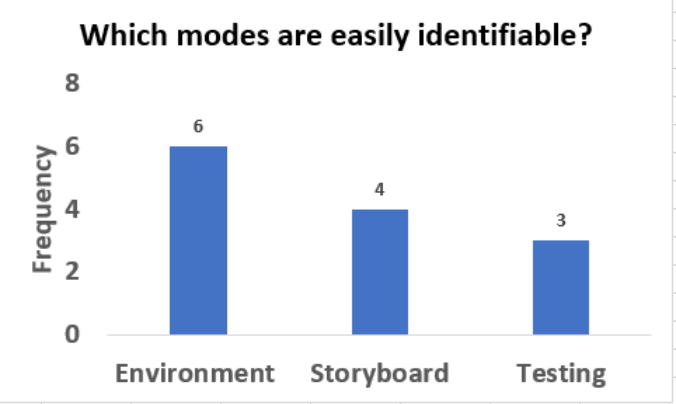


Figure 5.8: Identifiable Modes

with no difficulty. Figure 5.9 shows the average time needed for the completion of each task in the assignment, with the respective standard deviation. As shown in Fig 5.10, the average time from the completion of the Assignment B in the totality was 77.92 ±61.08 seconds, which corresponds to almost 1.3 ±1 minutes. The median was 50 seconds. The fastest time was by Participant 5, who did it in 24 seconds and the slowest time was by Participant 2, who did it in 229 seconds (3.8 minutes).

This indicates that the process of playing a story is very intuitive and fast to understand. Due to the fact, that the gameplay experiments were pre-determined by the tasks and it was represented by a small story, it is possible to assume that the process to play a more complex story would be longer, however the main concern and main focus of this assignment is to be able to evaluate the initial understanding of the gameplay by the user. Based, on the results obtained, it is possible to conclude the efficiency of this system regarding gameplay. It is also possible, to conclude that the users learn by their previous experiences, shown by the decrease of time to complete of tasks similar to previous ones, which also strengthens the stability and usability of this system.

5.2.2.1 Task B1: Select the save "Test" and enter Play Mode

The first task of the Assignment B is similar to the first task of the Assignment A. It ask the participant to again navigate trough the main menu and select a specific story, however this time the participant needs to enter the Play Mode after. The participants were observed if they could complete the task without help, with help or didn't complete it. The time necessary to complete the task was taken by the researcher. Based on the results in Table 5.6 we conclude that the majority of the participants (11) did learn from the previous task and were able to complete this task without help and only 3 participants needed help to complete the task. This shows that the learning curve of the system is small and the majority of users will be able to use the system based on previous interactions. The average time of completion was 18.71 ±15.45 seconds and the median was 12.5 seconds, which is considerably faster compared with Task A1: Select the save "Test" and enter create

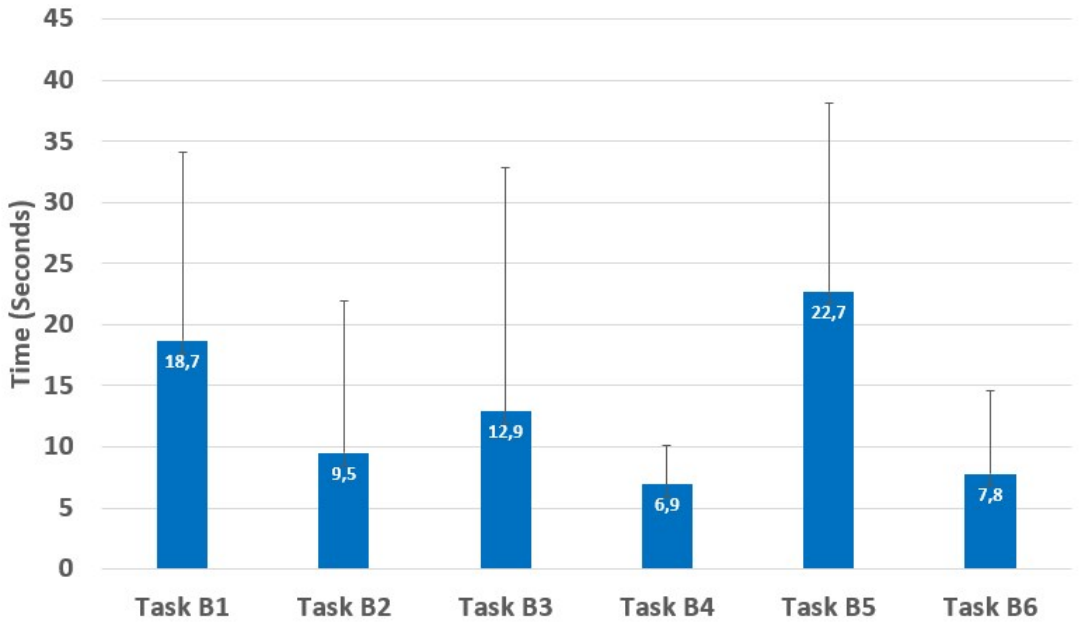


Figure 5.9: Assignment B - Average time per task

mode.

Table 5.6: Task B1 results

Complete	Answer(#)
Yes, without help	11
Yes, with help	3
No	0

	Time (seconds)
Median	12.5
Mean	18.71
Std. Dev.	15.45

5.2.2.2 Task B2: Move the character in the environment

The second task introduces the participant to the core mechanic of the gameplay, which is moving the character. The participants were observed if they could complete the task without help, with help or didn't complete it. The time necessary to complete the task was taken by the researcher. Based on the results in Table 5.7 we observe that the majority of the participants (12) were able to complete the task without help, being only 2 the participants that needed help. This is mainly to the fact that the first intuition to move a character in a game, was always using WASD on keyboard or pressing the right mouse button. This applied to the majority of the participants, not mattering their experience with video games. The average time of completion was 18.71 ±15.45 seconds and the median was 12.5 seconds, which also supports this conclusion, since is a very fast time to conclude the task.

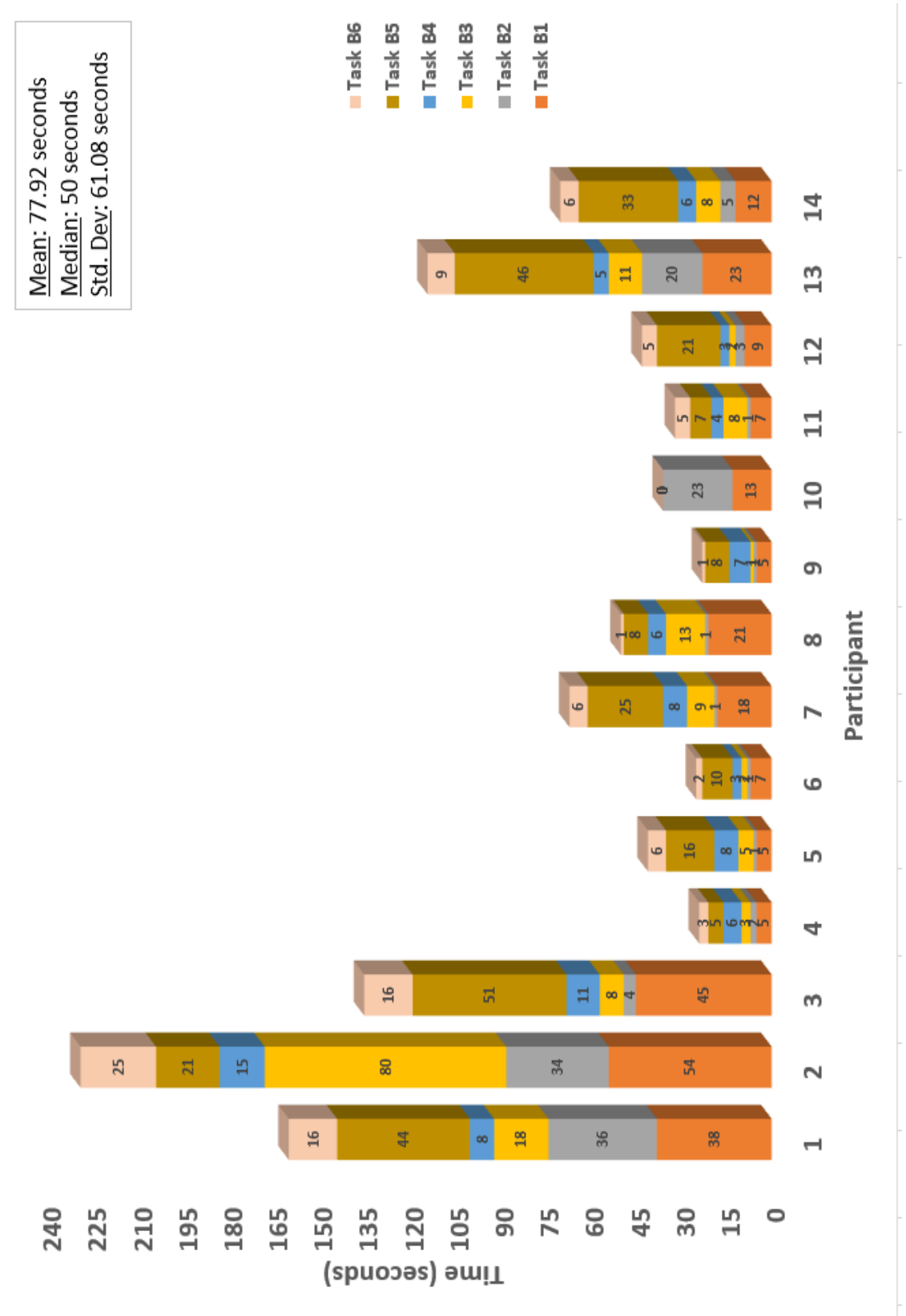


Figure 5.10: Assignment B - time to complete overview

Table 5.7: Task B2 results

Complete	Answer(#)
Yes, without help	12
Yes, with help	2
No	0

	Time (seconds)
Median	2.5
Mean	9.5
Std. Dev.	12.48

5.2.2.3 Task B3: Pickup the "Mushrooms"

The third task intends to show the participant how to interact with objects present in the environment and the storyboard. The participants were observed if they could complete the task without help, with help or didn't complete it. The time necessary to complete the task was taken by the researcher. Based on the results in Table 5.8 we observe that the majority of the participants (11) were able to complete the task without help, being only 1 the participants that needed help. This shows that interacting with objects is intuitive and validates the main gameplay mechanic, being easy to use. There was 1 participant that could not finish the task, this was due to the fact that there was a bug that prevented the save of the storyboard and despite having the right approach to the completion of this task, the participant was not able to do it. This bug was later fixed, but for this task the timing data is only based on the other 13 participants. The average time of completion was 12.92 ±19.92 seconds and the median was 8 seconds.

Table 5.8: Task B3 results

Complete	Answer(#)
Yes, without help	11
Yes, with help	1
No	1

	Time (seconds)
Median	8
Mean	12.92
Std. Dev.	19.92

5.2.2.4 Task B4: Interact with the "Pizza oven"

The fourth task intends to show the participant what happens when the player does not follow the storyboard. The participants were observed if they could complete the task without help, with help or didn't complete it. The time necessary to complete the task was taken by the researcher. The participant is asked about the animation from the playable character when interacting with the wrong object. Based on the results in Table 5.9 we observe that all of the participants were able to complete the task without help. There was 1 participant that could not finish the task, this was due to the fact that there was a bug that prevented the save of the storyboard and despite having the right approach to the completion of this task, the participant was not able to do it. This bug was later fixed, but for this task the timing data is only based on the other 13 participants. This completion rate is due to the fact, that the completion of this task is based on a mechanic already showed and tested before in Task B3: Pickup the "Mushrooms". The main objective of this task was to evaluate the animation to display that the user is interacting with the

wrong objects, but also serves the purpose to evaluate if the user learn from previous tasks, which can be verified by the 100% success rate without help, in addiction to the fast average completion time of 6.92 ±3.17 seconds with the median being 6 seconds, which is significantly faster compared to the similar task Task B3: Pickup the "Mushrooms". As shown in Fig 5.11, it is possible to conclude that using animations to transmit information to the player is a valid and intuitive solution, since only 1 participant didn't understand what the character's reaction was transmitting.

Table 5.9: Task B4 results

Complete	Answer(#)
Yes, without help	13
Yes, with help	0
No	1

	Time (seconds)
Median	6
Mean	6.92
Std. Dev.	3.17

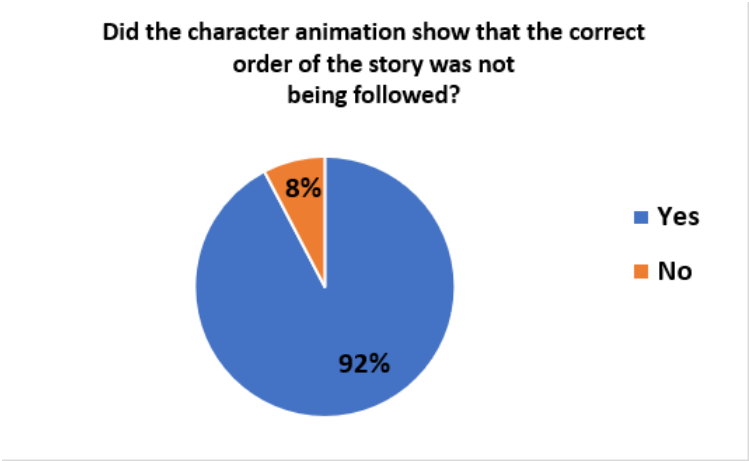


Figure 5.11: Character reaction evaluation

5.2.2.5 Task B5: Put the "mushrooms" of the inventory in the "pizza"

The fifth task intends to show the participant how to interact with objects of the environment using objects in the inventory. The participants were observed if they could complete the task without help, with help or didn't complete it. The time necessary to complete the task was taken by the researcher. The participant is asked about the utilization of the inventory items. Based on the results in Table 5.10 we observe that the majority of participants (10) were able to complete the task without help, while 3 needed help. There was 1 participant that could not finish the task, this was due to the fact that there was a bug that prevented the save of the storyboard and despite having the right approach to the completion of this task, the participant was not able to do it. This bug was later fixed, but for this task the timing data is only based on the other 13 participants. The average time of completion was 22.69 ±15.47 seconds and the median was 18.5 seconds. Based on the results obtained, it is possible to deduce that the mechanic of the inventory is not

quickly thought, verified by the increase of average time of completion, but is still intuitive enough that the majority of participants were able to complete the task without requiring help. This is also observed in Fig 5.12 where 11 participants (85%) find the mechanic intuitive, against 2 participants (15%) that do not find it intuitive. Some participants made the observation that the utilization of the items from the inventory was intuitive, but the inventory was hard to find, because it camouflaged with the background.

Table 5.10: Task B5 results

Complete	Answer(#)
Yes, without help	10
Yes, with help	3
No	1

	Time (seconds)
Median	18.5
Mean	22.69
Std. Dev.	15.47

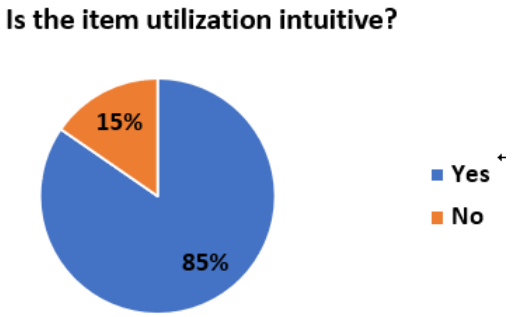


Figure 5.12: Item utilization Evaluation

5.2.2.6 Task B6: Put the "Pizza" in the "Pizza Oven"

The sixth and final task intends to show the completion of a story, closing the gameplay loop. The participants were observed if they could complete the task without help, with help or didn't complete it. The time necessary to complete the task was taken by the researcher. Based on the results in Table 5.11 we observe that all of participants, except 1 were able to complete the task without help. There was 1 participant that could not finish the task, this was due to the fact that there was a bug that prevented the save of the storyboard and despite having the right approach to the completion of this task, the participant was not able to do it. This bug was later fixed, but for this task the timing data is only based on the other 13 participants. The average time of completion was 7.77 ±6.83 seconds and the median was 6 seconds. This task is similar to Task B5: Put the "mushrooms" of the inventory in the "pizza" when it comes to mechanics. The main goal of this task was to complete the story that the user created and conclude the gameplay loop. However, this task also evaluates if the participant was able to learn from previous experiences, because it is similar to another. Based on the results, especially the time of completion being faster compared to the previous task, it shows that the participants can learn the mechanics from previous experiences.

Table 5.11: Task B6 results

Complete	Answer(#)
Yes, without help	10
Yes, with help	3
No	1

	Time (seconds)
Median	6
Mean	7.77
Std. Dev.	6.83

5.3 Questionnaires

After completing all the assignments the participants are required to answer three different questionnaires. The first one is the System Usability Scale Questionnaires (SUS), followed by a System Questionnaires that intends to evaluate the automatic generation algorithm and the user customization features for the environment generation and a Characterization Questionnaire.

5.3.1 System Usability Scale Questionnaire

This is the first questionnaire for the user to answer. This questionnaire was developed by John Brooke, and consists of ten questions with answers varying from a scale between one and five, meaning *Strongly Disagree* to *Strongly Agree*. This questionnaires in used to evaluate the quality and serviceability of a system. A quick explanation of how this questionnaire works is that higher answer in odd-numbered questions is better and lower answers on even-number questions is better. The questions and answers of this questionnaires are available in Table 5.12 and To calculate the final SUS score, we need to follow a set of rules [1]:

1. Each answer has a value from one to five, being 1 "Strongly Disagree" to 5 "Strongly Agree"
2. For odd-numbered questions; Subtract one from the user response
3. For even-numbered questions; Subtract the user responses from five
4. This normalizes the results to be between 0 and 4
5. Add up the converted responses for each user and multiply the total by 2.5. This will give a final score between 0 and 100.

The final scores, shown in Fig 5.13, give the average, which is 78. A SUS score above 68 is considered above average and anything below 68 is below average. In this case, the final score obtained is above average.

[1] Measuring Usability with the System Usability Scale (SUS), https://measuringu.com/sus/, Last Access: 2023

Table 5.12: SUS questionnaire results

Question	Median	Mean	Std. Dev
1- I think that I would like to use this system frequently	3.5	3.57	1.18
2- I found the system unnecessarily complex	1	1.64	0.89
3- I thought the system was easy to use	4	3.79	0.94
4- I think that I would need the support of a technical person to be able to use this system	1	1.14	0.35
5- I found the various functions in this system were well integrated	4	4.29	0.45
6- I thought there was too much inconsistency in this system	2	2.21	0.86
7- I would imagine that most people would learn to use this sytem very quickly	4.5	4.21	0.94
8- I found the system very cumbersome to use	1	1.50	0.82
9- I felt very confident using the system	5	4.36	0.97
10- I needed to learn a lot of things before I could get going with this system	2	2.43	1.40

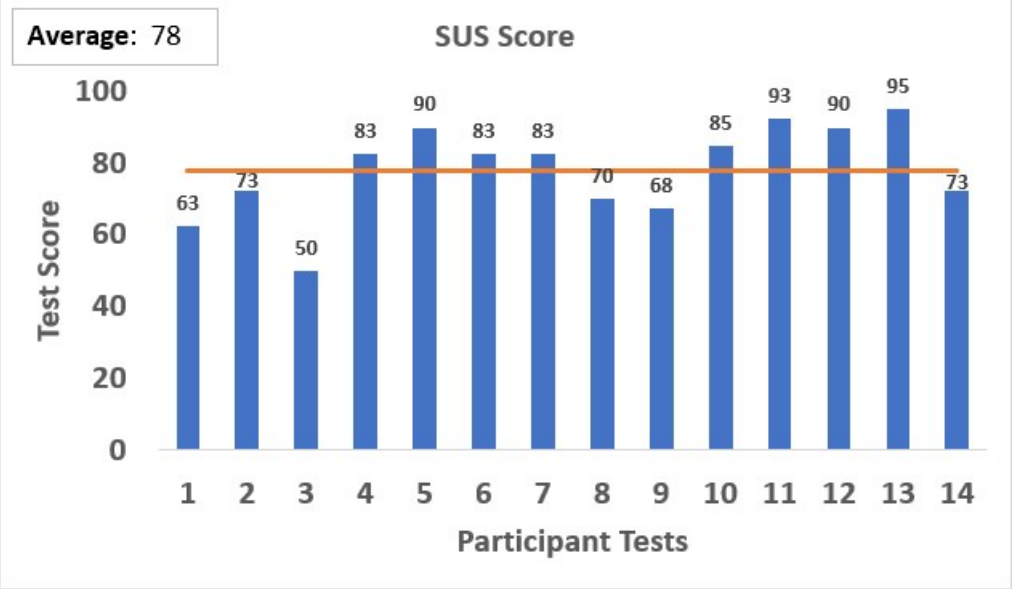


Figure 5.13: SUS Scores

5.3.2 System Questionnaire

The System Questionnaire that was asked has only two questions and serves to evaluate the quality of the Environment generation feature of the system. The first question evaluates the environment generated automatically with the system's algorithm and the second question evaluates the features of user customization, that allow the user to customize already generated environments. Both questions are an evaluation with values 1 to 5, being *Not Satisfied* and *Very Satisfied*. The results of the questionnaire can be observed in Fig 5.14.

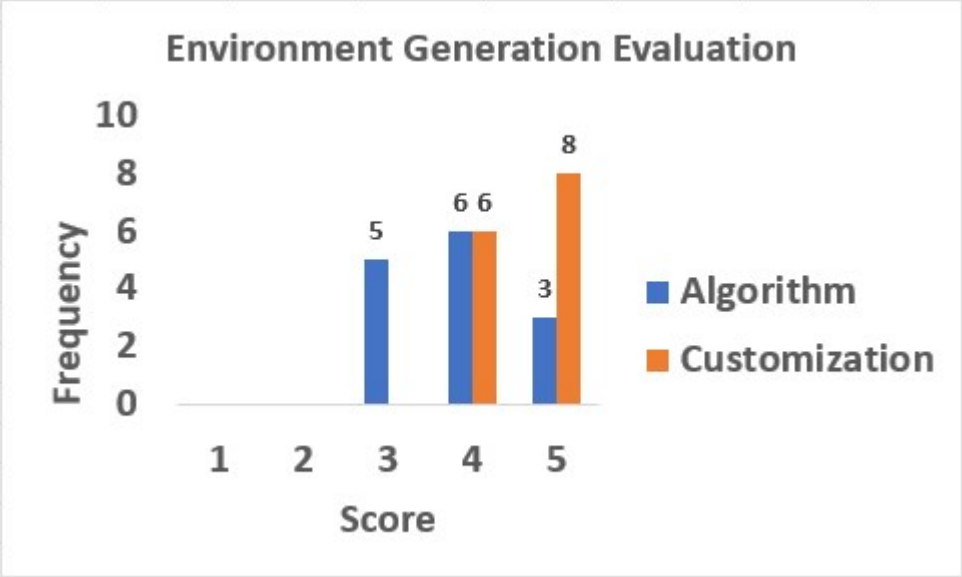


Figure 5.14: System Questionnaire Results

5.3.3 Characterization Questionnaire

The characterization questionnaire is the last questionnaire the user is asked to fill. This questionnaires asks for informations about themselves. The data that is asked is the data that allows to characterize the population and is data that is relevant is some form to the system. The questionnaire asks the participants the following data:

1. **Age:** The participant's age.
2. **Gender:** Multiple choice question with *Male*, *Female* or *Other* as options.
3. **Experience with games:** Multiple choice with *Never*, *Rarely*, *Sometimes*, *Every week* or *Every day* as options.
4. **Game Genre prefered:** Multiple choice with *Action*, *Adventure*, *Cards*, *Sports*, *Strategy*, *Puzzle*, *Massive Multiplayer Online (MMO)*, *First Person Shooter (FPS)*, *Role Playing Game (RPG)*, *Survival*, *Multiplayer Online Battle Arena (MOBA)* or *Other* as options.

The participant can select multiple answers from the options. If chooses *Other*, needs to specify.

5. **Game Platform prefered:** Multiple choice with *Mobile*, *Tablet*, *Computer*, *Console*, *AR*, *VR* or *Other* as options. The participant can select multiple answers from the options. If chooses *Other*, needs to specify.

The results of this questionnaire are available in Population.

5.4 Discussion and Comparative Analysis

The conducted experiments, both Assignment A and Assignment B, aimed to evaluate the functionality, user-friendliness and efficiency of the implemented system. These two assignment represent two main aspects of the system, being the automatic environmental generation and user customization (Assignment A) and gameplay with interactive storytelling (Assignment B). In this section, we analyse the implications and possible conclusions of the findings that resulted from the experiments of both assignments.

In **Assignment A**, participants were tasked with creating and customizing environments within the system. The results were an average completion time of approximately 5 minutes, with a maximum duration of 10 minutes. It is important to have in account, that participants were guided by predefined tasks and only made small adjustments to the environment. The relatively short completion times suggest that the system's interface and customization features are intuitive and user-friendly, even for users with limited or nonexistent gaming and programming experience. This aligns with our goal of enabling users to effortlessly create their own narratives within the system. Furthermore, the observed variations in completion times between Task with similar mechanical demands, such as Task A2 and A3 reveals a small learning curve required for the understanding of the system. The participants became more more proficient in the usage of the system and the mechanics, which led to a decrease of the completion times in subsequent similar tasks. This highlights that users can learn while using the system, continually decreasing the necessary time to perform the tasks desired. The evaluation from the System Questionnaire also shows that the participants were satisfied with both the automatic generation of the environment made by the algorithm, as well as the options to modify that environment manually. The results indicate that the final environment created by the algorithm is still not at perfect/high level, having still some unrealistic and inconsistent object placements, however the user customization options that are available to the user, compensate for the non perfection of the algorithm. This also, aligns with another goal, which is minimize user effort on creating environments by creating a sufficiently realistic environment automatically, but still allowing for user customization to better optimize the environment to the user's desire.

In **Assignment B**, the focus changes to the gameplay of the stories within the system. The participants were introduced to various gameplay mechanics, such as object interactivity, inventory managements, player movement and character animations. The results showed an average completion time of close to 1.3 minutes, showing high intuitiveness and simplicity of this feature of the system. Again, it is important to have in account, that the tasks were pre-defined and were based on a very small sample story, which reduces considerably the completion time of the assignment. However, despite the length and complexity of the tasks, some results indicate that the system is intuitive, regardless of there is a gaming background. Tasks such as moving the player (Task B2), can show by the short completion times, that despite the prior gaming background, the player movement is intuitive and easy to understand, which again aligns with the goals of this thesis. When analysing the results, it is safe to assume that the majority of the implementation follow the main goals of this thesis, such as intuitivety and realism, however it also shows based on the high number of participants that needed help to finish the task, in multiple tasks, that the design and interface still needs improvement and some feature might still need some adjustments. When comparing the two assignments, both highlight the participant's capability to learn from prior experiences, which is a promising indicator of the system's usability, which in addiction to the above average results obtained in the System Usability Scale Questionnaire questionnaire, with mostly positive results, suggest that the participants enjoyed to interact with the system, and the system shows potential. In conclusion, the experiment's goal was to showcase the system's effectiveness in facilitating environment creation and customization and gameplay interaction. While still having some problems, this was fundamentally achieved, showing the system's current capabilities and future potential.

6

Conclusions

This thesis' main goal was to redefine the realm of interactive storytelling in video games by introducing a system capable of both automatic environment generation and interactive gameplay experiences. The implementation of user customization, the usage of LLM and the integration with the narrative using a 3D top down point and click environment allowed for the exploration of this goal. To conclude this thesis, a revisit to the thesis' objective and research questions defined in the beginning must happen.

6.1 Objectives

The main **objective** was to create an intuitive system that empowers users to craft and experience narrative effortlessly, without the need for prior gaming or programming knowledge. The main goal was divided in multiple sub objectives:

1. **Develop an innovative system:** This was achieved by using the power of LLM, that allowed storytelling with dynamic, realist and automatic environment generation, enabling the creation of dynamic narratives. The system's capability for automatic environment generation, in junction with the user customization features, is an innovation in the world of storytelling and environment generation.

2. **Ensure User-friendly experience:** Another objective was to provide users with an simple and user-friendly experience while using the system. By implementing intuitive design choices and accessible interface, we ensure that users were able to engage with the system, irrespective of their prior gaming and programming knowledge. During the development process, user-friendly interfaces were a main focus point, trying to ensure that making narrative creation was accessible to all users.

3. **Customization and Versatility:** Besides the need to implement automatic environment generation, another objective was to allow the users with customization options of those environments. This was achieved through a wide array of features, including adding and removing objects, that enable users to change the environments to

fit their creative visions. The system's versatility is also accounted, since the system allows for diverse storytelling genres and narratives, including adventure games, as shown by the Case Studies.

4. **Scalability:** Another objective was to make the system scalable, allowing for flexibility to adapt to different storytelling needs, and making it suitable for a wide range of areas. The current implementation of the object database, the grid system and the usage of LLM, allows for an easy escalation of the assets database, giving the system a solid foundation for future scalability enhancements. This also allows for the usage of this system, in multiple different areas, such as Education or Marketing, along with the gaming and storytelling realms already identified.

5. **Playable System:** An uppermost objective was to create a system that is not only a narrative creation tool, but also provides an interactive gameplay. The introduction of the 3D topdown point and click Playing Mode, allows the user to bring their narratives to life, providing users with an immersive and interactive storytelling experience.

6.2 Research Questions

These objectives are in line with the research questions proposed in the beginning of this thesis.

1. **RQ1: How effectively can LLM be utilized to automate the creation of diverse and realistic environments for interactive storytelling purposes?** The utilization of LLM, notably GPT 3.5, has proven to be a big advancement in the realm of automatic environment generation. In this thesis implementation, it is showcased the great efficiency of LLM in automating the creation of diverse and realistic environments. With the usage of AI, we manage to redefine the storytelling realm, streamlining the environment creation process and helping creating many narrative possibilities.

2. **RQ2- How to ensure user customization and minimal prior knowledge requirements in gaming a programming?** Empowering users with customization options was always a core objective of this thesis. This system is a testament to the commitment to ensure that users, regardless of prior knowledge in either gaming or programming, can effortlessly create their on narratives based on their creative vision. The integration of user-friendly customization features made the creation process accessible to all users, breaking down barriers and inviting storytellers of all backgrounds to engage with this system.

3. **RQ3- In what way can the versatility of AI driven automatic generation accommodate different storytelling genres, including adventures games?** The versatility of AI-driven automatic generation has proven to be a pivotal factor in accommodating

for a wide variety of storytelling genres. Adventure games, in particular, can thrive in this system, benefiting from the dynamic environment creation, in junction with the Storyboard process creation [3]. The system's ability to adjust to various storytelling genres is also showcased by the system's adaptability to create diverse narratives.

In conclusion, the results achieved demonstrate the transformative potential of AI-driven automatic generation, while still ensuring that storytelling remains accessible, diverse and immersive for storytellers with all backgrounds. By addressing our objectives and research questions, we have demonstrated that the blend of automation and customization can redefine the art of storytelling.

6.3 Future Work

As we reflect on the achievements of this thesis and the system that was developed, we are aware of possible improvements and future work that can lie ahead. Future work holds the premise of expanding and enriching the system in multiple dimensions, trying to create a new era of interactive storytelling within video games.

Diverse Objects and Environment themes: One of the first possibilities of improvement is the inclusion of a more extensive array of objects and environment themes. By increasing our database of assets, it is possible to offer storytellers an even wider creativity with their narratives. This expansion can open doors to storytelling genres that were previously unexplored, allowing for more creativity and diversity.

Non-Playable Characters (NPCs) intentions and prompting: To enhance the depth of storytelling, it is possible to consider incorporating mechanism for NPCs to express their intentions and emotions more vividly. Enabling NPCs to convey their thoughts and motivations via prompting, can add to the narrative a new layer of realism and emotional attachment. This feature would not only elevate the quality of storytelling but also offer endless possibilities for storytelling creation.

Detailed micro-interaction and Realism features: The pursuit of realism in interactive storytelling is always an ongoing point of focus. Future work can experiment with the inclusion of more detailed micro-interactions and features that elevate the immersion. This can encompass elements such as dynamic time and day changes, character reactions to choices and other nuanced behaviors that mirror real world experiences. The additions can transform the current storytelling in a more immersive experience.

Automated storytelling steps: Looking forward, there is a possibility of using LLM to automated all the steps involved in storytelling creation, encompassing not only environment generation, studied in this thesis, but also allow for elements like dialogue scripting and character interactions to be done automatically. While always retaining the option for user customization, the goal is to provide a comprehensive solution where any storyteller can input their goals and themes for the story, and the system autonomously can generate a coherent narrative.

Goal-specific storytelling: One prospect of the future is the ability to tailor storytelling to specific goals and areas of application. Whether it is for entertainment, education or marketing, the system can adapt its generation algorithms to align with the desired goals. This adaptability will enable to cater to a vast majority of different industries and applications, offering a versatile tool for any storytelling needs.

In summary, despite the positive results achieved in this thesis, the can always suffer expansion and transformation. The future holds the promise of unlocking new dimensions in interactive storytelling, where this process can become an effortless and boundless experience for all storytellers.

In terms of implementation, there is still room to improve in some features of the system, such as:

1. **Grid Size Flexibility:** One of the features of the system that can be implemented in the future is the ability to adjust the size of the grid. This feature will ensure that users have the freedom to create environments of varying scales and complexities. Whether designing a small scene or a bigger environment, users can set the grid dimensions to suit their creative vision. However, the system is better suited for smaller environments, having the object sizes and camera positioning, better adjusted for smaller settings. To account for this, the user is allowed to change the camera positioning and zoom to be able to make adjustments in both big and complex, as well as small and simple worlds.

2. **Objects Occupying Multiple Tiles:** While the current iteration of the system accommodates for one object occupying one tile only, future enhancements are envisioned to allow objects to occupy multiple tiles. This feature will enable users to create more complex and detailed environments, such as large structures or terrain elements that occupy multiple grid tiles. This development will require adjustments in the grid system's logic, especially for rotation and collisions between objects. The current implementation for the user interface already works for this changes, having the holograms that preview the object placement with logic that already has those changes into account.

3. **Environment adjustment after user modifications:** In our vision for the system, we imagine the implementation of a continuous environment adjustment during the creation process. The concept is to make the system actively adapt and refine the environment throughout the entire creation process. The algorithm will leverage its knowledge to offer realistic suggestions, accelerating the creative process while still preserving user customization. In this envisioned system, the environment will evolve intelligently as users add, remove or modify elements. For instance, if a user adds a table to the environment, the system will understand the relationship between tables and chairs and dynamically adds chairs adjacent to that table. While this

feature may not be in the current system, it shows our vision to make environment creation a task more accessible, efficient and captivating.

Bibliography

[1] J. M. Lourenço. *The NOVAthesis LaTeX Template User's Manual*. NOVA University Lisbon. 2021. URL: https://github.com/joaomlourenco/novathesis/raw/master/template.pdf (cit. on p. ii).

[2] T. Gao and J. Zhu. "A Survey of Procedural Content Generation of Natural Objects in Games". In: IEEE, 2022-02, pp. 275–279. ISBN: 978-1-6654-5818-4. DOI: 10.1109/ICAIIC54071.2022.9722677 (cit. on pp. 1, 9).

[3] D. Guerreiro. "Adventure Game Creator - Criação e Representação de Histórias com Story Boards". Nova School of Science and Technology, 2023 (cit. on pp. 2, 4, 72).

[4] D. Wilcox-Netepczuk. "Immersion and realism in video games - The confused moniker of video game engrossment". In: *Proceedings of CGAMES 2013 USA - 18th International Conference on Computer Games: AI, Animation, Mobile, Interactive Multimedia, Educational and Serious Games* (2013), pp. 92–95. DOI: 10.1109/CGAMES.2013.6632613 (cit. on pp. 7, 17).

[5] T. J. Rose and A. G. Bakaoukas. "Algorithms and Approaches for Procedural Terrain Generation - A Brief Review of Current Techniques". In: IEEE, 2016-09, pp. 1–2. ISBN: 978-1-5090-2722-4. DOI: 10.1109/VS-GAMES.2016.7590336 (cit. on pp. 8, 12, 13, 15).

[6] P. García-Sánchez. "Georgios N. Yannakakis and Julian Togelius: Artificial Intelligence and Games". In: *Genetic Programming and Evolvable Machines* 20 (1 2019-03), pp. 143–145. ISSN: 1389-2576. DOI: 10.1007/s10710-018-9337-0 (cit. on p. 8).

[7] A. I. Rathnayake, I. K. Ganegala, I. S. Samarasinghe, S. B. Weerasekara, A. I. Gamage, and T. Thilakarathna. "Adjusting The Hard Level on Game by a Prediction for Improving Attraction and Business Value". In: (2021-03), pp. 310–311. DOI: 10.1109/ICTER51097.2020.9325474 (cit. on p. 8).

[8] G. Liu, M. Cai, L. Zhao, T. Qin, A. Brown, J. Bischoff, and T.-Y. Liu. "Inspector: Pixel-Based Automated Game Testing via Exploration, Detection, and Investigation". In: IEEE, 2022-08, pp. 237–244. ISBN: 978-1-6654-5989-1. DOI: 10.1109/CoG51982.2022.9893630 (cit. on p. 8).

[9] J. P. A. Campos and R. Rieder. "Procedural Content Generation using Artificial Intelligence for Unique Virtual Reality Game Experiences". In: IEEE, 2019-10, pp. 147–151. ISBN: 978-1-7281-5434-3. DOI: 10.1109/SVR.2019.00037 (cit. on p. 8).

[10] V. Kumar, D. Singh, G. Bhardwaj, and A. Bhatia. "Application of Neurological Networks in an AI for Chess Game". In: IEEE, 2020-02, pp. 125–130. ISBN: 978-1-7281-1769-0. DOI: 10.1109/INBUSH46973.2020.9392188 (cit. on p. 8).

[11] I. Oh, S. Rho, S. Moon, S. Son, H. Lee, and J. Chung. "Creating Pro-Level AI for a Real-Time Fighting Game Using Deep Reinforcement Learning". In: *IEEE Transactions on Games* 14 (2 2022-06), pp. 212–220. ISSN: 2475-1502. DOI: 10.1109/TG.2021.3049539 (cit. on p. 8).

[12] K. Hartsook, A. Zook, S. Das, and M. O. Riedl. "Toward supporting stories with procedurally generated game worlds". In: IEEE, 2011-08, pp. 297–304. ISBN: 978-1-4577-0010-1. DOI: 10.1109/CIG.2011.6032020 (cit. on p. 8).

[13] T. Gao, J. Zhang, and Q. Mi. "Procedural Generation of Game Levels and Maps: A Review". In: IEEE, 2022-02, pp. 050–055. ISBN: 978-1-6654-5818-4. DOI: 10.1109/ICAIIC54071.2022.9722624 (cit. on pp. 8, 22).

[14] K. Satyadama, R. F. Rachmadi, and S. M. S. Nugroho. "Procedural Environment Generation for Cave 3D Model Using OpenSimplex Noise and Marching Cube". In: *CENIM 2020 - Proceeding: International Conference on Computer Engineering, Network, and Intelligent Multimedia 2020* (2020-11), pp. 144–148. DOI: 10.1109/CENIM51130.2020.9297889 (cit. on pp. 9, 13, 14).

[15] Q. E. Morris. "Modifying Wave Function Collapse for more Complex Use in Game Generation and Design". In: (2021) (cit. on p. 10).

[16] H. Kim, S. Lee, H. Lee, T. Hahn, and S. Kang. "Automatic generation of game content using a graph-based wave function collapse algorithm". In: *IEEE Conference on Computatonal Intelligence and Games, CIG* 2019-August (2019-08). ISSN: 23254289. DOI: 10.1109/CIG.2019.8848019 (cit. on p. 11).

[17] B. von Rymon Lipinski, S. Seibt, J. Roth, and D. Abe. "Level Graph – Incremental Procedural Generation of Indoor Levels using Minimum Spanning Trees". In: IEEE, 2019-08, pp. 1–7. ISBN: 978-1-7281-1884-0. DOI: 10.1109/CIG.2019.8847956 (cit. on pp. 13, 14).

[18] B. Sun, L. Jiang, B. Sun, and S. Jiang. "Research of Plant Growth Model Based on the Combination of L-system and Sketch". In: IEEE, 2008-11, pp. 2968–2972. DOI: 10.1109/ICYCS.2008.250 (cit. on p. 14).

[19] K. Guan. "The research of the improved 3D L- system and its application in plant modeling". In: IEEE, 2008-09, pp. 718–723. ISBN: 978-1-4244-1673-8. DOI: 10.1109/ICCIS.2008.4670976 (cit. on p. 14).

[20] M. Fridenfalk. "Algorithmic music composition for computer games based on L-system". In: IEEE, 2015-10, pp. 1–6. ISBN: 978-1-4673-7452-1. DOI: 10.1109/GEM.2015.7377230 (cit. on p. 14).

[21] M. Fridenfalk. "Application for Real-Time Generation of Virtual 3D Worlds Based on L-System". In: IEEE, 2015-10, pp. 73–78. ISBN: 978-1-4673-9403-1. DOI: 10.1109/CW.2015.16 (cit. on p. 14).

[22] D. Ashlock, S. Gent, and K. Bryden. "Evolution of L-systems for Compact Virtual Landscape Generation". In: IEEE, pp. 2760–2767. ISBN: 0-7803-9363-5. DOI: 10.1109/CEC.2005.1555041 (cit. on p. 14).

[23] Y. Levus, R. Westermann, M. Morozov, R. Moravskyi, and P. Pustelnyk. "Using Software Agents in a Distributed Computing System for Procedural Planetoid Terrain Generation". In: IEEE, 2022-11, pp. 446–449. ISBN: 979-8-3503-3431-9. DOI: 10.1109/CSIT56902.2022.10000868 (cit. on p. 15).

[24] J. Doran and I. Parberry. "Controlled Procedural Terrain Generation Using Software Agents". In: *IEEE Transactions on Computational Intelligence and AI in Games* 2 (2 2010-06), pp. 111–119. ISSN: 1943-068X. DOI: 10.1109/TCIAIG.2010.2049020 (cit. on p. 15).

[25] O. J. Hui, J. Teo, and C. K. On. "Interactive evolutionary programming for mobile games rules generation". In: IEEE, 2011-10, pp. 95–100. ISBN: 978-1-4577-0444-4. DOI: 10.1109/STUDENT.2011.6089332 (cit. on p. 15).

[26] N. P. Zea, J. L. G. Sanchez, and F. L. Gutierrez. "Collaborative Learning by Means of Video Games: An Entertainment System in the Learning Processes". In: IEEE, 2009-07, pp. 215–217. DOI: 10.1109/ICALT.2009.95 (cit. on p. 16).

[27] P. Sommeregger and G. Kellner. "Brief Guidelines for Educational Adventure Games Creation (EAGC)". In: IEEE, 2012-03, pp. 120–122. ISBN: 978-1-4673-0885-4. DOI: 10.1109/DIGITEL.2012.32 (cit. on p. 16).

[28] V. Zafeiropoulos, D. Kalles, and A. Sgourou. "Adventure-Style Game-Based Learning for a Biology Lab". In: IEEE, 2014-07, pp. 665–667. ISBN: 978-1-4799-4038-7. DOI: 10.1109/ICALT.2014.195 (cit. on p. 16).

[29] R. Morsi and S. Mull. "Digital Lockdown: A 3D adventure game for engineering education". In: IEEE, 2015-10, pp. 1–4. ISBN: 978-1-4799-8454-1. DOI: 10.1109/FIE.2015.7344072 (cit. on p. 16).

[30] R. A. Bordini, J. L. Otsuka, D. M. Beder, L. F. Fonseca, P. A. G. de Freitas, A. A. Nunes, D. L. Santiago, G. L. A. Santiago, and M. R. G. de Oliveira. "Musikinésia – An Educational Adventure Game for Keyboard Learning". In: IEEE, 2015-07, pp. 142–146. ISBN: 978-1-4673-7334-0. DOI: 10.1109/ICALT.2015.90 (cit. on p. 16).

[31] C. Ji. "Sakura: A VR musical exploration game with MIDI keyboard in Japanese Zen environment". In: IEEE, 2020-08, pp. 620–621. ISBN: 978-1-7281-4533-4. DOI: 10.1109/CoG47356.2020.9231808 (cit. on p. 17).

[32] A. Predescu, M. Mocanu, and C. Chiru. "A case study of mobile games design with a real-world component based on Google Maps and Unity". In: *Proceedings of the 13th International Conference on Electronics, Computers and Artificial Intelligence, ECAI 2021* (2021-07). DOI: 10.1109/ECAI52376.2021.9515013 (cit. on p. 18).

[33] A. Bonsch, S. Freitag, and T. W. Kuhlen. "Automatic generation of world in miniatures for realistic architectural immersive virtual environments". In: IEEE, 2016-03, pp. 155–156. ISBN: 978-1-5090-0836-0. DOI: 10.1109/VR.2016.7504700 (cit. on p. 19).

[34] C. Davis, J. Collins, J. Fraser, H. Zhang, S. Yao, E. Lattanzio, B. Balakrishnan, Y. Duan, P. Calyam, and K. Palaniappan. "3D Modeling of Cities for Virtual Environments". In: IEEE, 2021-12, pp. 5587–5596. ISBN: 978-1-6654-3902-2. DOI: 10.1109/BigData52589.2021.9671898 (cit. on p. 19).

[35] Y. Cai, C. Miao, A. H. Tan, Z. Shen, and B. Li. "Creating an immersive game world with evolutionary fuzzy cognitive maps". In: *IEEE Computer Graphics and Applications* 30 (2 2010-03), pp. 58–70. ISSN: 02721716. DOI: 10.1109/MCG.2009.80 (cit. on p. 19).

[36] R. Huijser, J. Dobbe, W. F. Bronsvoort, and R. Bidarra. "Procedural natural systems for game level design". In: *Proceedings - 2010 Brazilian Symposium on Games and Digital Entertainment, SBGames 2010* (2010), pp. 189–198. DOI: 10.1109/SBGAMES.2010.31 (cit. on p. 19).

[37] M. Moradi and Y. Mahdavinasab. "Investigating the consideration of explorer player style in Iranian educational games". In: IEEE, 2021-11, pp. 81–85. ISBN: 978-1-6654-0936-0. DOI: 10.1109/ISGS54702.2021.9684767 (cit. on p. 20).

[38] D. Panzoli, C. Peters, I. Dunwell, S. Sanchez, P. Petridis, A. Protopsaltis, V. Scesa, and S. de Freitas. "Levels of Interaction: A User-Guided Experience in Large-Scale Virtual Environments". In: IEEE, 2010-03, pp. 87–90. ISBN: 978-1-4244-6331-2. DOI: 10.1109/VS-GAMES.2010.27 (cit. on p. 22).

[39] D. Sedlacek, O. Oklusky, and J. Zara. "Moon Base: A Serious Game for Education". In: IEEE, 2019-09, pp. 1–4. ISBN: 978-1-7281-4540-2. DOI: 10.1109/VS-Games.2019.8864540 (cit. on p. 22).

[40] J. Kaplan and N. Yankelovich. "Open Wonderland: An Extensible Virtual World Architecture". In: *IEEE Internet Computing* 15 (5 2011-09), pp. 38–45. ISSN: 1089-7801. DOI: 10.1109/MIC.2011.76 (cit. on p. 22).

A

Consent form for testing session

The testing session required participants to read and consent to participate in the session. The participants are required to accept the consent to participate. Below is the consent form present to the participants before the start of the session.

To participate in this test, you must be aware that data will be handled and collected. You can only participate in the tests if, after reading this information, you agree to take part in the study.

Data Handling All data collected will only be used in the context of the thesis. The data will be anonymized before processing. The information that will be collected includes all the information present in this questionnaire, along with possible observations from both parties during the session.

At any point during the session, you can ask questions about the session.

By choosing "Accept" below, you voluntarily agree to participate in this academic study, acknowledging that you are aware of and agree to the collection and processing of the aforementioned data.