DM

Customizing Experiences for Mobile Virtual Reality

MASTER DISSERTATION

José Gabriel dos Santos Gomes Agrela
MASTER IN INFORMATICS ENGINEERING

UNIVERSIDADE da MADEIRA
A Nossa Universidade
www.uma.pt

September | 2023

Customizing Experiences for Mobile Virtual Reality

MASTER DISSERTATION

José Gabriel dos Santos Gomes Agrela
MASTER IN INFORMATICS ENGINEERING

ORIENTATION
Mónica da Silva Cameirão

CO-ORIENTATION
Sergi Bermúdez i Badia

Faculdade de Ciências Exatas e da Engenharia

Mestrado em Engenharia Informática

Customizing Experiences for Mobile Virtual Reality

Gabriel Agrela

Orientado por:

Prof. Mónica Cameirão

Prof. Sergi Bermúdez i Badia

Constituição do júri de provas públicas:

Karolina Baras (Professora Auxiliar da Universidade da Madeira), Presidente

Filipe Quintal (Professor Auxiliar da Universidade da Madeira), Arguente

Mónica Cameirão (Professora Auxiliar da Universidade da Madeira), Orientadora

11 de setembro de 2023

Resumo

A criação manual de conteúdo para um jogo é um processo demorado e trabalhoso que requer um conjunto de habilidades diversificado (normalmente designers, artistas e programadores) e a gestão de diferentes recursos (hardware e software especializados). Dado que o orçamento, tempo e recursos são frequentemente muito limitados, os projetos poderiam beneficiar de uma solução que permitisse poupar e investir noutros aspectos do desenvolvimento. No contexto desta tese, abordamos este desafio sugerindo a criação de pacotes específicos para a geração de conteúdo personalizável, focados em aplicações de Realidade Virtual (RV) móveis. Esta abordagem divide o problema numa solução com duas facetas: em primeiro lugar, a Geração Procedural de Conteúdo, alcançada através de métodos convencionais e pela utilização inovadora de Grandes Modelos de Linguagem (normalmente conhecidos por Large Language Models). Em segundo lugar, a Co-Criação de Conteúdo, que enfatiza o desenvolvimento colaborativo de conteúdo. Adicionalmente, dado que este trabalho se foca na compatibilidade com RV móvel, as limitações de hardware associadas a capacetes de RV autónomos (standalone VR Headsets) e formas de as ultrapassar são também abordadas. O conteúdo será gerado utilizando métodos actuais em geração procedural e facilitando a co-criação de conteúdo pelo utilizador. A utilização de ambas estas abordagens resulta em ambientes, objectivos e conteúdo geral mais re-jogáveis com muito menos desenho. Esta abordagem está actualmente a ser aplicada no desenvolvimento de duas aplicações de RV distintas. A primeira, AViR, destina-se a oferecer apoio psicológico a indivíduos após a perda de uma gravidez. A segunda, EmotionalVRSystem, visa medir as variações nas respostas emocionais dos participantes induzidas por alterações no ambiente, utilizando tecnologia EEG para leituras precisas.

Keywords: Geração Procedural de Conteúdo · Co-Criação de Conteúdo · Optimização Móvel · Realidade Virtual Móvel · Personalização de Experiências · Pacotes Unity3D

Abstract

Manually creating content for a general game is a time-consuming and labor-intensive process that requires a diverse skillset (usually designers, artists and programmers) and managing different resources (specialized hardware and software). Since budget, time and resources are usually very limited, projects could benefit from a solution where these are saved and invested somewhere else in the development. In the context of this thesis, we address this challenge by suggesting the creation of packages tailored for customizable content generation, focused on mobile Virtual Reality (VR) applications. This approach bifurcates the problem into a dual-faceted solution: Firstly, the *Procedural Generation of Content*, achieved through conventional methods as well as innovative utilization of Large Language Models. Secondly, the *Co-Creation of Content* which emphasizes collaborative content development. Additionally, because this work focuses on compability with mobile VR, the limitations of hardware related with standalone VR headsets and how to overcome them are also addressed. Content will be generated by utilizing the current methods in procedural generation, and by facilitating the co-creation of content by the user. The usage of both these approaches results in environments, objectives, and general content to be more replay-able with much less designing. This approach is currently being employed in the development of two distinct VR applications. The first, *AViR prototype*, is designed to offer psychological support to individuals following pregnancy loss. The second, *EmotionalVRSystem*, aims to measure the variances in participants' emotional responses induced by changes in the environment, utilizing EEG technology for accurate readings.

Keywords: Procedural Generation of Content · Co-Creation of Content · Mobile Optimization · Mobile Virtual Reality · Customization of Experiences · Unity3D Packages

Acknowledgements

Leading the list of those to who gratitude is owed are my parents for their unwavering support throughout this journey. I also extend my sincerest gratitude to my advisors, whose expert guidance and mentorship profoundly enriched this endeavor. Genuine appreciation goes out to my esteemed colleagues, whose collaborative spirit was pivotal in advancing this work. My deepest thanks are reserved for the members of the NeuroRehabLab; their invaluable contributions in testing and validating were indispensable.

Acknowledging all your efforts, the presence of such a supportive community during this academic venture is truly appreciated.

Table of Contents

List of Figures

List of Tables

List of Acronyms

AI Artificial Intelligence

CC Co-Creation

GnT Generate-and-Test

HMD Head Mounted Display

LLM Large Language Model

LOD Level of Detail

ML Machine Learning

NPC Non-Player Character

PCG Procedural Content Generation

SVR Standalone Virtual Reality

1 Introduction

1.1 Background

Video game development is a multifaceted domain demanding a wide arsenal of tools and expertise. Despite the immense efforts and resources poured into projects, developers often face the stark reality of an oversaturated market. The fierce competition frequently culminates in many games failing to generate a substantial profit. The unpredictability of the industry, coupled with the significant costs associated with marketing, sales, resources, and personnel, further exacerbates this challenge [1].

Virtual Reality (VR) games, while promising a revolutionary interactive experience, introduce added layers of complexity. The nascent and sometimes unstable nature of VR technology, combined with the heightened hardware costs, amplifies the challenges faced in traditional game development [1].

1.2 Objectives, Motivation, and Research Questions

Recognizing the challenges in optimizing the game development process, particularly for mobile VR platforms, there emerges a pressing need to devise innovative solutions. A pivotal aspect of this challenge is content creation—an area that often consumes significant resources and time. In this light, the primary objective of this project is to harness the power of customizable packages for Unity. These packages, enriched by procedural generation and co-creation techniques, aim to alleviate some of the burdens associated with content creation.

Against this backdrop, this research aims to answer the following overarching questions:

- **RQ1:** Can the use of customizable packages for Unity that generate content through co-creation and procedural generation improve the efficiency of game development projects?
- **RQ2:** Can Unity packages for custom content generation engage users in gameplay?

Moreover, given the burgeoning domain of artificial intelligence (AI), and the emergent capabilities of Large Language Models (LLMs) as tools for crafting unique AI personalities, this research also delves into the realm of AI-driven avatars. This leads us to our third research question:

- **RQ3:** Can users noticeably differentiate between the personalities of AI agents in a Unity package during gameplay?

2 State of The Art

2.1 Introduction

This chapter conducts a comprehensive review of the literature central to the themes underpinning this thesis. It begins by exploring the transformative potential of procedural content generation (PCG), a technique capable of providing unique experiences by algorithmically generating in-game material. This not only enhances replayability but also ensures that every gaming session is distinct. The discourse then shifts to the co-creation of content, a participatory design strategy that allows users to contribute to game design in a myriad of ways. This aspect extends from the ability to create custom levels and game modes to participating in dedicated community forums and hubs. Lastly, the focus transitions to the implementation of standalone VR headsets. These devices, also known as all-in-one Head-Mounted Displays (HMDs), are self-contained, plug-and-play systems that do not require a physical connection to a PC. They have evolved to provide a higher degree of mobility and freedom, despite their limitations in image quality and refresh rates. By traversing these interconnected domains, the ensuing sections aim to provide a holistic understanding of the current landscape and the interplay of these technologies in shaping the future of immersive and interactive experiences in the gaming industry.

2.1.1 General Definitions

- **Content**: Generally speaking, content refers to a particular form of information that can be presented in various forms, such as text, images, audio, or video. It is the substance or material that is presented to an audience and can take many different forms depending on the context. Content can also be described as a form of creative work, as it often involves the expression of ideas or concepts in a unique and engaging way. This can include things like articles, blog posts, videos, and other forms of media that are designed to capture the attention of the audience and provide them with valuable information or entertainment [2]. In the context of gaming, content refers to the various elements of the game that make up the player's experience. This can include things like the game's story, characters, and environments, as well as any additional content that is created by the players themselves. Overall, content is an essential part of any form of communication and can take many different forms depending on the context in which it is presented. It is a powerful tool for conveying information, ideas, and emotions, and can help to engage and inspire audiences of all kinds.
- **User**: To have a user, you need both technology and a person. The technology provides the platform or interface through which the user is able to interact with the system, while the person is the individual who is using the technology to perform a specific task or accomplish a goal [2]. Without technology, it is not possible to have a user, as there would be no means for the person to interact with the system. Similarly, without a person, the technology would not be used, and would therefore be useless. Together, technology and the user form a symbiotic relationship, where the technology provides the means for the user to accomplish their goals, and the user provides the intelligence and creativity that drives the technology forward. This relationship is at the heart of many of the most successful and innovative products and services in the world, and is the driving force behind the rapid pace of technological advancement.
- **Virtual World**: Software platforms that provide users with access to a 3D world in-game are commonly known as virtual worlds. These platforms allow users to create and customize their

own avatars, explore virtual environments, and interact with other users in real-time. Virtual worlds are often designed to be highly immersive and engaging, with rich and detailed graphics, engaging gameplay mechanics, and extensive social features. This allows users to feel as if they are truly transported to another world, where they can explore and interact with others in ways that are not possible in the real world. Some popular examples of virtual world platforms include Second Life, The Sims Online, and World of Warcraft. These platforms have millions of active users and have become a significant part of the online gaming landscape [2]. Overall, virtual world platforms are an important part of the gaming industry, as they provide users with a unique and engaging way to interact with others and explore virtual environments.

2.2 Procedural Content Generation

The gaming industry may undergo a revolution if procedural generation is used in game design. Games can provide gamers with a distinctive experience every time they play by creating material algorithmically. This not only increases replayability but also guarantees that no two sessions will be identical. Players can benefit from a new and exciting experience every time they play because of the variation, which still preserves a level of familiarity that enables sharing of experiences with other players. Procedural generation is a very wonderful tool that can significantly improve the gaming experience [3].

Another use of procedural generation implementation is the incorporation of dynamic components in educational serious games, which significantly improve the learning opportunities of students. This technology can assist students' engagement and interest in the information being delivered by exposing them to a number of various surroundings and circumstances. This can be especially helpful for subjects that are typically regarded as boring or uninteresting since it enables teachers to design engaging and interactive learning activities that can enhance students' comprehension and recall of the subject matter. As students must adjust to new circumstances and obstacles to succeed in the game, the usage of changing elements can also aid in the development of critical thinking and problem-solving skills [4].

2.2.1 A Taxonomy

A good starting point for this technology is looking at its taxonomy. It is important to note that taxonomies in this area are very scarce and the following should be seen more as a spectrum, rather than a binary comparison [5]:

- **Online versus offline**: Content does not always need to be generated in real-time. For some of it, it might be a good idea to generate it at the developing phase of the application. Let's consider that we need to create a huge map for a multiplayer game. It is a tedious job to patch every field with grass, create every mountain, etc. We can instead apply PCG to create it once and use it in the development (offline). In contrast, we might need PCG to create a different weapon every time an NPC (Non-Playable Character) drops it (online).
- **Necessary versus optional content**: Content may not need to be obligatory, this means, if an NPC drops a weapon that was created as PCG (we see this example of procedural generation, on modular weapons in borderlands [6]), a user probably does not need to pick it up and continue with his/her journey; he/she may just ignore it and find a better one. However, it is if there is a PCG maze or dungeon that stands between the user and the end.

- **Random seed versus parameter vectors**: How much can something be customized? Content can be generated with a certain degree of control. For example, there is this algorithm that creates a dungeon. At one extreme, we could have a PCG that takes a random seed and generates a dungeon, where we do not have any control over its contents (other than the constraints of the algorithm itself of course), at the other extreme, the algorithm takes a vector of values, one for its height, another for its length, number of bosses, rooms, etc.

- **Stochastic versus deterministic**: Content could be deterministic, reproducible. This means that if we have a seed, and we put it in the algorithm as a parameter, the algorithm's output should always be the same, as long as the seed does not change. However, stochastic algorithms do not have reproducible outputs, basically random in the eyes of the player. Deterministic algorithms are very useful and may be seen as a form of compression, i.e. instead of having the model of the map (that could very much be in the GB's range), the algorithm can generate it (KB's range).

- **Generic versus adaptive**: Content can be generated without taking the user's current behavior into account (generic), for example, the algorithm just spawns enemies at a rate based on time. An adaptive PCG algorithm would consider the user's behavior before producing an output. With the previous example, the algorithm could instead spawn the enemies at a rate related to the player's behavior (in this case, how many enemies are still alive, the number of enemies slain, or some other player performance metric), we see this happening in left2dead thought their "Director AI" [7] [8]. We see another good example of an adaptive approach in PCG in the Emotional Labyrinth application, where the maze changes based on the emotional state of the player, measured by bio-metric signaling hardware [9].

2.2.2 Techniques

Like in the case of taxonomies, procedural generation of content techniques are also vaguely and broadly described, and classified/categorized. The following are the most settled techniques in this area (Fig. 1). It is worth noting that a certain technique or example of procedurally generated content can fit into most of these taxonomies, given they are not exclusive:

- **Search Based Approach**: The basic metaphor is that the algorithm keeps iterating results based on a score until that score is acceptable [8]. It is one of the many ways you can apply the generate-and-test approach. It's divided in:

  - The test function - grades a candidate content, using one or a vector of real numbers (other evolutionary/genetic algorithms call this a fitness function).

  - The generating function - generates a new candidate depending on the previous candidate's fitness values. The objective is to generate a higher fitness value [5].

- **Constructive Generation Methods**: We may consider as an example a generation of a dungeon. The procedural generation of dungeons usually refers to a topology. Typically consists of a representation model (a simplified representation of the dungeon), a method for constructing the representational model and a method for creating the actual geometry [10].

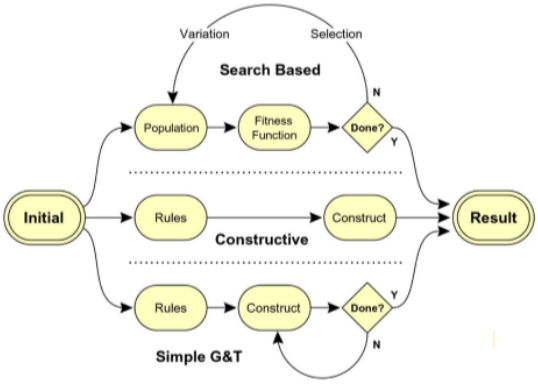


Fig. 1. Three ways of procedural generating content [5].

– **Height-maps or intensity maps**: A constructive method, usually it is closer to the random seed taxonomy spectrum, although most functions used (like perlin noise Fig. 2) are pseudo-random [11], making its output replicatable and therefore deterministic. Using some kind of coherent noise function [12], a noise map can be created, where every coordinate is related to its adjacent neighbors, value-wise. The values of each coordinate could be, for example, the height of that point, creating a terrain [8]. An approach more in the "parameter vector" side of the spectrum would have parameters defining the max height of the terrain, size of the terrain (X and Y axis), colors (based on height for example), etc.

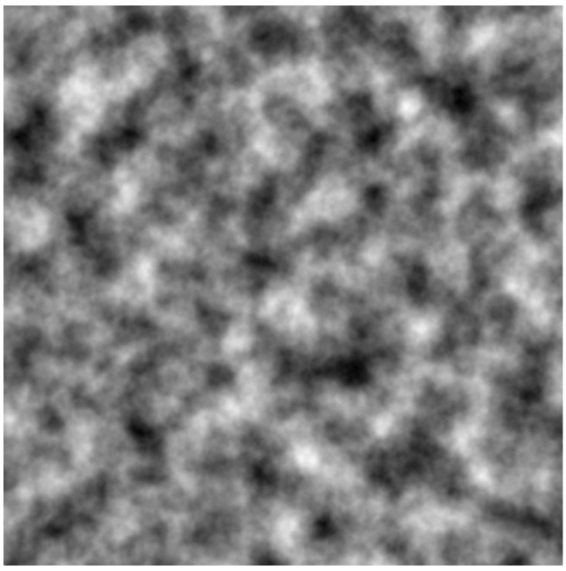


Fig. 2. Perlin noise example.

– **Random terrain**: There are different interpretations of a random terrain, however, following an extreme random-seed approach (Fig. 3) will result in a map with unrelated coordinate values [13]. A way to come around this is by interpolating the values of each coordinate with its adjacent [8]. This can be done by simply making the map/matrix X-times bigger and then

interpolating by replacing the new values with the average between the neighboring coordinates. It's also a constructive method and stochastic.

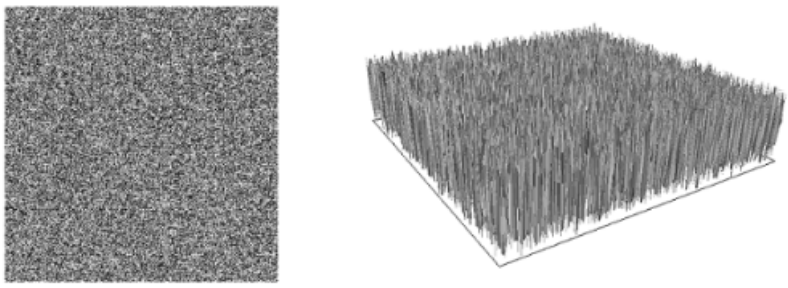


Fig. 3. Example of how random noise, without interpolation, produces a "spiky" and uninteresting terrain.

– **Fractal**: A constructive method that can be based on height maps and therefore mostly inherits its taxonomies. However, it is a bit more into the "parameter vector" spectrum because this technique takes as parameters a vector of values that multiply height maps with different scales [14] [8]. Shaker's book [8] mentions the previous technique on the creation of landscapes, however we have made a prototype in python more directed towards object instantiating (trees, grass, etc). The script creates an image of a visual representation of perlin noise, where the potency of every coordinate is adjustable. See Fig. 4 for an example:

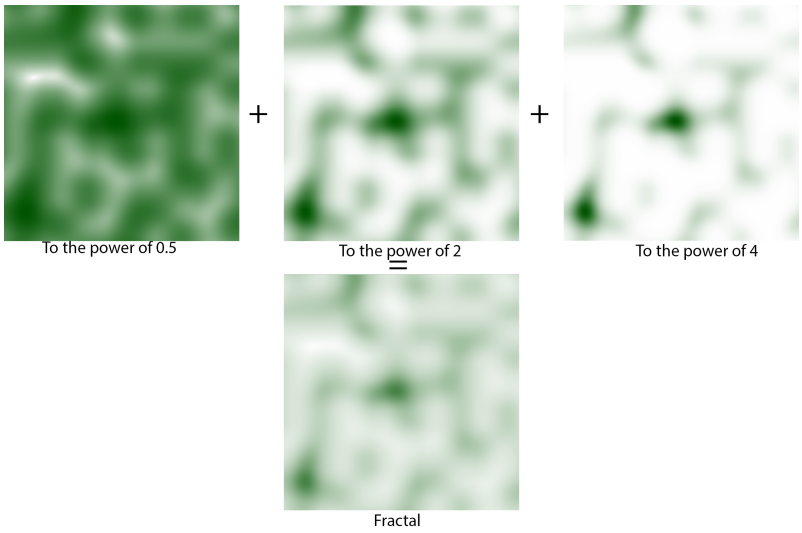


Fig. 4. Example of an implementation of fractals. White represents 0 and green represents 1. Implementing for instantiating, of trees for example, would create more natural looking forests.

– **Agent based**: Although in previous methods it is possible to implement a more "parameter vector" approach, they are still rather limited in terms of degrees of control [8]. Agent-based approaches are usually at the end of this spectrum. With this Generate-and-test approach, there are different agents doing different things. In this city creation example, each agent can work as an iterator of the generation of the city [15] (Fig. 5).

Fig. 5. Multiple agents creating a city evolution

- **Custom PCG through common algorithms**: There is a plethora of algorithms that can be used to generate whatever a developer can imagine. Recursive backtracker, Kruskal's and Prim's algorithms, recursive division, binary trees, and depth-first search are all examples of algorithms that are not necessarily procedural generation related but can be used as such. As an example, we have the emotional labyrinth, where the depth-first search was used as a constructive, parameter vector (for the number of rows and columns for example), adaptive (changed size depending on the user's emotional state), online (created at loading) and stochastic [9].
- **AI Techniques**: The advent of AI and machine learning (ML) has opened up new possibilities for PCG. These techniques can be used to learn from existing content and generate new content based on the learned models. This is often referred to as PCG via ML (PCGML) [16].
  - **Neural Networks**: Deep learning, a subset of ML, has shown promise in various fields, including PCG. Deep learning models, particularly various types of neural networks such as long short-term memory networks, autoencoders, and deep convolutional networks, can learn complex patterns from large amounts of data, making them suitable for generating high-quality content. These models can be trained in a supervised or unsupervised manner, depending on the availability of labeled data. They can learn to generate content that is similar to the training data, making them useful for tasks such as level design, story generation, and card generation in collectible card games [16, 17].
  - **Large Language Models**: LLMs have been instrumental in the field of PCGML, which involves the generation of game content using ML models trained on existing content. These models are used to generate a wide range of game content, including platformer levels, game maps, interactive fiction stories, and cards in collectible card games. They can also be used for autonomous generation, co-creativity, mixed-initiative design, and compression. One of the key advantages of PCGML is that it can be used for repair, critique, and content analysis due to its focus on modeling existing content. Various PCGML methods are used, including neural networks such as long short-term memory (LSTM) networks, autoencoders, and deep convolutional networks; Markov models such as n-grams and multidimensional Markov chains; clustering; and matrix factorization. However, PCGML also faces challenges such as learning from small datasets, lack of training data, multi-layered learning, style-transfer, parameter tuning, and PCG as a game mechanic. In addition to these applications, LLMs can be used to address challenges in narrative text generation by implementing high-quality content planning, which involves learning good plot structures to guide story generation. This approach results in stories that are more relevant to a given prompt and of higher quality. Furthermore, LLMs can also be used to generate educational content, such as

programming exercises and code explanations, providing a valuable tool for instructors and potentially improving the quality of the educational experience for students [16, 18, 19].

2.2.3 Summary

In conclusion, PCG presents a potentially revolutionary tool for the gaming industry, offering unique experiences for gamers, improved learning opportunities in serious games, and a broad taxonomy and variety of techniques for content generation. The taxonomical spectrum ranges from online vs offline generation, necessary vs optional content, random seed vs parameter vectors, stochastic vs deterministic, and generic vs adaptive generation. These methods can be leveraged to create different gameplay environments, encounters, and challenges, increasing replayability and engagement. Moreover, several techniques can be utilized for PCG, including search-based approaches, constructive generation methods, height maps, random terrains, fractals, agent-based generation, and various other algorithms. The advent of AI and ML, particularly deep learning and neural networks, has further broadened the horizons for PCG, allowing for high-quality content generation that mimics human creativity. LLMs have been instrumental in the field of PCGML, generating a wide range of game content. While challenges persist, such as learning from small datasets and multi-layered learning, the potential benefits and advancements these models provide, are clear. In educational contexts, these techniques can generate interactive and engaging content, enhancing student comprehension and overall learning experience.

As we look forward to the future of game development, the role of the player becomes increasingly significant. The next level of immersion and personalization in gaming could well be the co-creation of content. This approach fosters a sense of ownership and creativity among players, allowing them to contribute to the game world in meaningful ways. Be it through creating levels, modes, or even new in-game items and events, the possibility for players to co-create content opens a new chapter in gaming. It shifts the perspective from gaming as a purely consumptive activity to a creative, communal, and participatory one, enriching the overall gaming experience. In the following section, we will explore the various ways in which games have started to embrace co-creation, highlighting the benefits and challenges this approach brings.

2.3 Co-Creation of Content

The contemporary digital realm bears witness to a remarkable revolution in video game design, driven by the concept of co-creation. This innovation fundamentally shifts the roles of game developers and players, nurturing an interactive, immersive environment where content is not merely consumed but also created by the players. Co-creation, defined broadly, allows players to contribute and influence game content, be it through the creation of unique levels, game modes, characters, events or items, or through interactive forums and community hubs. This expansive framework of co-creation, while applicable across a multitude of digital domains, becomes particularly intriguing within the context of gaming. Consequently, our focus will revolve around the intricate dance of co-creation within this dynamic space, shedding light on how the collective intelligence of gaming communities shapes the game's narrative, social interactions, and the very fabric of its virtual world [20] [21] [22].

2.3.1 Animal Crossing Example

An example of a game where user-generated content is valuable, we have Animal crossing. Summarizing the findings from the analysis by Kim et al. [20]'s analysis:

– **User Generated narratives**: Players are awarded several types of in-game cash or tokens when they interact with the game. The gathering of these tokens is a crucial component of the game since it not only gives players a sense of accomplishment but also allows them to access additional things and features inside the game. One example is the museum, a crucial aspect of the game, where players may display artifacts they have gathered via gameplay (Fig. 6). This not only provides a competitive aspect as players attempt to obtain the most stunning things, but it also stimulates them to explore the game environment and try out new techniques.

Fig. 6. User donating items to the museum, exposing the tokens to admiration.

While completing pre-programmed tasks that are vital in the game, it is the players' imagination and inventiveness that allow them to really immerse themselves in the gaming world. As a result, it is critical for game designers to provide constraints or obstacles that drive players to think outside the box and come up with unique solutions to difficulties. This not only increases the pleasure of the player in the game but also contributes to the gameplay remaining new and exciting.

– **Game as a labor and vice-versa**: Planning and creating one's own route within a game may quickly develop into a habit, thereby ruining the overall gaming experience. To get around this, game designers could include challenges that encourage players to think creatively and try out novel strategies. In the next example, maintaining a spotless city for a set period of time, say two weeks, could be a fun and rewarding task. The player may receive in-game awards or other incentives if they successfully maintain a clean city for the entire two weeks. However, if they neglect to keep the city clean, even if only by skipping one day of work, they may fall far behind schedule and must restart the project. In contrast to more traditional, heroic games that generally focus around rescuing a princess or beating a boss, this style of challenge allows players to generate their own content and feel a sense of success. While some may find this style of gaming boring, for those who can fully immerse themselves in the game environment, it can be immensely entertaining and gratifying. It may even be thought of as a "second life" for the gamer, providing a unique and gratifying gaming experience.

– **User collaboration**: It is not a flaw when a game lacks pre-written plots; rather, it is a special quality that encourages players to actively shape their own gaming experience. Designers may promote a sense of ownership and interest in the game that can significantly improve the overall player experience by allowing players the opportunity to build their own pathways and narratives inside the game. Additionally, the pursuit of one's own objectives and tactics might benefit other users as well as game developers. Item gathering (Fig. 7) has long been a common narrative technique in video games.

Fig. 7. Example in a newer animal crossing for the Switch where the user collects items.

2.3.2 Minecraft Example

– **Co-creation experiences and values**: The user base of Minecraft (MC) can influence and alter the game's gameplay, creating distinctive and individualized co-creation experiences and values [21]

– **Gameplay**: There are different game modes that have their own set of rules and features. The three most popular game modes are survival, adventure, and creation. The survival mode is about collecting items, fighting enemies, and crafting in order to thrive and survive, while the adventure mode is about exploring player-created worlds that have certain game-play features limited to prevent disrupting the created world. The creative mode is the most popular mode, which allows players to freely construct or destroy blocks, mechanisms, or structures, and enables the survival and adventure aspect, such as health bars, but no damage can be taken by the player. The game modes can be switched depending on the server's settings. The game also has an infinite number of ideas and customized inputs such as self-made or downloadable mods or plugins, which can create novel game mechanisms or even mini-games, making the game potentially never appear repetitive. MC can be played as a single player or with multiple other people in multiplayer mode, using user-generated content, which results in customer-to-customer and customer-company co-creation. Players can work together on dedicated projects, create buildings or new tools, and communicate through messaging and voice talk. The game also offers the opportunity for players to socialize and interact with others who share similar interests [21].

- **Social Environment**: In a virtual environment, players may explore, make things, and engage with one another. The chance to interact with other players and make friends is one of the game's key charms. Players can compete with one another through a variety of player-versus-player modes or unique minigames made by players themselves in MC. Players can also converse with one another through messaging and voice chat. Numerous gamers also create clans, which are compact associations of people with like ideals, gaming preferences, and objectives. Players may have to submit their experiences, talents, and personalities as part of an application procedure to join a clan; this process may even involve a Skype encounter. Players can look for others through forums or social media in addition to connecting with people through the game itself. The game is renowned for having a sizable and engaged player base that frequently communicates with one another both within and outside of the game. As it enables players to cooperate to achieve a shared objective and have fun with one another, working with other players in the game may be engaging and fun. In MC, playing with others may be a gratifying experience since it lets users collaborate on projects, engage in friendly competition, and develop lasting relationships. Players may work, play, and interact with others in the virtual environment of the game, which acts as a sort of second world for them. As a result, except while playing in single-player mode, it is quite uncommon that a player is not engaging in some type of interaction with other players during the game. Overall, MC is regarded as a socially networked game that has attracted a large and active community of players [21].

- **Skill Development**: The third identified MC co-creation experience is the assistance and development of game-relevant abilities. To fully use the game's features, players may need to obtain certain resources or expertise, such as programming, design, and in-game knowledge. These abilities may be gained and improved through a variety of methods, such as asking for assistance in forums, reading guides and articles, watching video tutorials, and using tools and software made by other players or small businesses. The game and its community also provide opportunities for players to learn and improve their skills outside of the game, such as through "Minecraft Education," an educational version of MC that promotes creativity, collaboration, and problem-solving skills through the game's open-world setting. In order to assist others in learning and developing their talents, players may also produce and share their own video material and live gaming broadcasts. These video productions and live streams may be published on websites like YouTube and Twitch and can cover a variety of subjects, such as tutorials, reviews, and gaming highlights. Many gamers also produce material expressly to amuse and captivate their audience, and they may enrich their films with creative components like music, commentary, and special effects. In general, making and sharing videos and live broadcasts is a frequent and well-liked hobby among MC players, and it may be a useful tool for others to learn from and hone their abilities [21].

2.3.3 Learning through game co-creation, Kauko School Example

- The idea of "knowledge co-creation" describes the procedure of building and developing knowledge via cooperation and discussion with others. The co-creation and validation of information through social interactions that entail action, discourse, and emotions are regarded as key components of creative and fun learning. Playfulness and the production of fictional worlds through role-playing are considered as being facilitators of the knowledge co-creation process, which is said to occur through creative activities and the fabrication of artifacts. In general,

the idea of knowledge co-creation implies that learning involves not just obtaining information but also negotiating and co-creating it with others [22].

- The construction of coherent tales and narratives that aid learners in making sense of their experiences and the world around them is regarded as a component of co-creation in connection to narratives. Children are said to benefit from co-creation because it allows them to formulate, develop, and confirm their common understanding of a subject and organize their thoughts into a more digestible format. The use of narratives is said to have a vital role in the process of knowledge co-creation as well as in creative and fun learning. It is said to be especially helpful for connecting students' imaginations to the subject matter being learnt and inspiring their creativity via fiction. Regarding narratives, the idea of co-creation often implies that learning entails working together to develop and shape tales and narratives to more fully comprehend and make sense of experiences and knowledge [22].

- The function of co-creation in connection to possibility thinking and creative learning is stated as giving learners the chance to build various and flexible viewpoints and creatively apply their knowledge. It is proposed that participating in design projects, such as developing hypothetical gaming worlds, can offer a platform for applying scientific knowledge and evaluating ideas of reality against potential worlds via thought experiments. The creation of hypothetical worlds is regarded as vital for creativity, and the explanations and consequences advanced by students throughout this process are supposed to indicate their comprehension of a curriculum topic. The emphasis of this study, however, was not on academic accomplishment or science learning, but on children and teachers' experiences of creative and fun learning processes [22].

2.3.4 Conclusions

Upon a closer inspection of each example of co-creation of content previously discussed, three main points can be highlighted:

- **Co-created Narratives**: Narratives can be co-created and explored by the users. This allows greater freedom and a lesser chance of failure since the user creates their own.

- **Leverage the social aspects**: The social aspect of co-creation is discussed in both studies. The social environment can significantly impact the player's experience and the overall engagement of the game. Players are more likely to collaborate and co-create content by themselves, leading to a more engaging and rewarding experience. This happens because players may like to share their work or explore other players' experiences for ideas. Although this explanation indicates a multiplayer approach, it is not necessarily the case. Both examples explain how this is also achieved in offline environments like forums, sharing your knowledge, or offline save.

- **Construction of a world/space**: In the Minecraft and Animal Crossing's examples, it is explained how the creation of a space or world is the prime example of the co-creation of content. The game should empower users with tools for personalizing their virtual world. In addition, it is crucial to incorporate engaging mechanics that prompt regular player interaction. Examples include NPCs that can damage structures or the sporadic appearance of garbage in a city, requiring the player's attention to maintain or expand their environment. This is often accompanied by a reward system to further exacerbate the sense of accomplishment of engaging with the game mechanics.

With these critical points of co-created content in mind, it is essential to understand the platforms where these interactions could be fully realized. One significant and growing platform is Standalone Virtual Reality (SVR) headsets, which have the potential to create immersive and interactive environments, further enhancing co-created content.

2.4 Optimization Challenges and Techniques for Standalone VR Headsets

In our endeavor to advance the state-of-the-art in SVR game development, our primary objective is to comprehensively analyze the constraints specific to SVR headsets. We particularly focus on the architectural limitations of the Meta Quest 2, a dominant device in the SVR market. Our overarching goal is to ensure smooth gameplay at a minimum of 72 frames per second (fps), which is pivotal for a seamless VR experience.

To achieve this, we delve into a myriad of SVR constraints, such as draw calls, triangle count, texture management, shadow generation, shader processes, vertices simplification, and the overall system geometry. Our proposal encompasses a suite of optimization strategies to counteract these constraints. These include efficient texture mapping, intelligent shadow calculations, strategic shader utilization, merging vertices based on proximity, and enhancing system geometry through techniques like Level of Detail (LOD) and billboarding.

Within the context of the Unity gaming engine, we dissect the application of these strategies. Our discourse will be underpinned by industrial case studies and academic research, reinforcing the merit of our proposed techniques. Through this holistic examination, we aspire to equip developers with the knowledge to craft immersive and high-fidelity experiences on SVR headsets, transcending the hardware's intrinsic limitations.

2.4.1 Mobile/Standalone Virtual Reality Headset

Standalone VR (SVR) headsets, also known as all-in-one HMDs, are plug-and-play systems that do not require a physical connection to a PC. These headsets are equipped with built-in custom CPUs, sensors, power sources/batteries, storage, and displays, allowing for higher mobility and movement freedom. SVRs typically offer lower-quality images and lower refresh rates, but recent advancements in processors such as the Qualcomm Snapdragon XR2 Platform, have improved their processing and rendering capabilities. VR technology and headset vendors such as Meta, Google, and HTC, are focusing more on developing SVR headsets, which can operate both tethered to a PC and wirelessly. This trend is likely to make SVRs the dominant next-generation VR headsets [23]. Several sources indicate this, such as Omidia's report [19] and the evolution seen in Steam's Hardware Surveys [24] and the evolution seen in Steam's Hardware Surveys [25], [26], [27]. However, in these cases, they represent the use of SVR's as tethered to the computer, essentially functioning like a standard VR headset. Nevertheless, they still indicate an obvious increase in SVR headsets. But, even though advances have been made in the processing capabilities of SVR headsets, it is still rather important to look into optimization and other techniques to reach, at least, 72 fps gameplay, since low frame rates in HMD are known to cause nausea and other negative symptoms. The Meta Quest 2, being the most popular SVR headset on the market [27] and a representative exemplar of typical SVR capabilities, will serve as our primary focus in this part of the thesis.

2.4.1.1 Constraints and bottlenecks overview

To understand and optimize the performance of SVR systems, it is essential to explore the constraints and bottlenecks of these devices. The data referenced here are drawn from the Meta

developer documentation. Two critical factors that play a significant role in the performance of these headsets are the number of draw calls per frame and the triangle count.

- **Draw Calls:** The number of draw calls that can be used per frame depends upon different factors. Switching pipelines, like changing shaders, textures, meshes, etc.. The space available on the main and render thread and graphics API and its usage are all points that must be looked up closely. A busy simulation would be an application with many simulations, VoIP, networking, animation, etc. (i.e., multiplayer FPS). A medium one would be a medium sized world without too many players or NPCs, like a story driven FPS, while a light simulation does not have many pipeline state changes, like an escape room, or a puzzle game.

Table 1. Simulation Draw Calls.

Platform	Draw Calls	Description
Quest 1	50-150	Busy simulation
Quest 1	150-250	Medium simulation
Quest 1	200-400	Light simulation
Quest 2	80-200	Busy simulation
Quest 2	200-300	Medium simulation
Quest 2	400-600	Light simulation

- **Triangle Count:** Triangle budgets are an important consideration for optimizing the performance of a game or application. They are influenced by factors such as triangle size, memory access patterns, and the number and precision of vertex attributes. To improve performance, it is important to monitor and adjust your triangle budgets regularly and stay up-to-date with the latest rendering technology.

Table 2. Triangle Count by Platform.

Platform	Triangle Count
Quest 1	350k-500k
Quest 2	750k-1.0m

2.4.1.2 Optimizations and development techniques

- **Texture**: A visually beautiful and entertaining game requires the use of colors. In Unity, colors are applied to objects using meshes and materials, which include shader references, information on colors and textures, and other data. By creating a UV map of the model in Blender, it is possible to employ color palettes as a helpful resource-saving strategy. By doing so, we may employ a single texture for an object with multiple sides [28] (Fig. 8).

  The technique involves pre-processing the texture to create multiple copies at different sizes, which allows for better rendering of fine details and produces a better overall effect. When using MipMaps, the 3D card can detect the scaling factor and adjust the texture accordingly. In other words, the farther away a texture is, the fewer details it needs. Using pre-processed copies of a texture with less resolution will, therefore, save resources, while the perception of lower quality will be hardly noticeable [29].

- **Shadow Generation**: The calculation of shadows in a game engine depends heavily on the performance of the GPU. Soft shadows are more demanding on the GPU than hard shadows

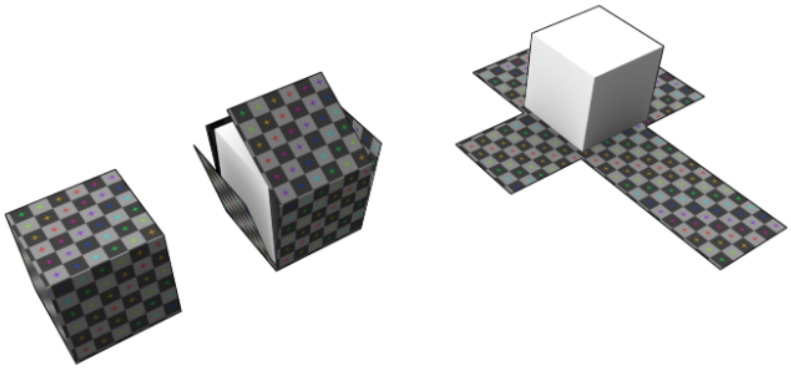


Fig. 8. Visual example of how UV mapping works.

(Fig. 9), but the CPU and memory requirements are the same. To improve performance, it is recommended to put small objects such as stones or rubble, in the same layer and use the Camera.layerCullDistances function of the script to control the distance at which they are displayed. This can help to reduce the workload on the GPU and improve overall performance [29].

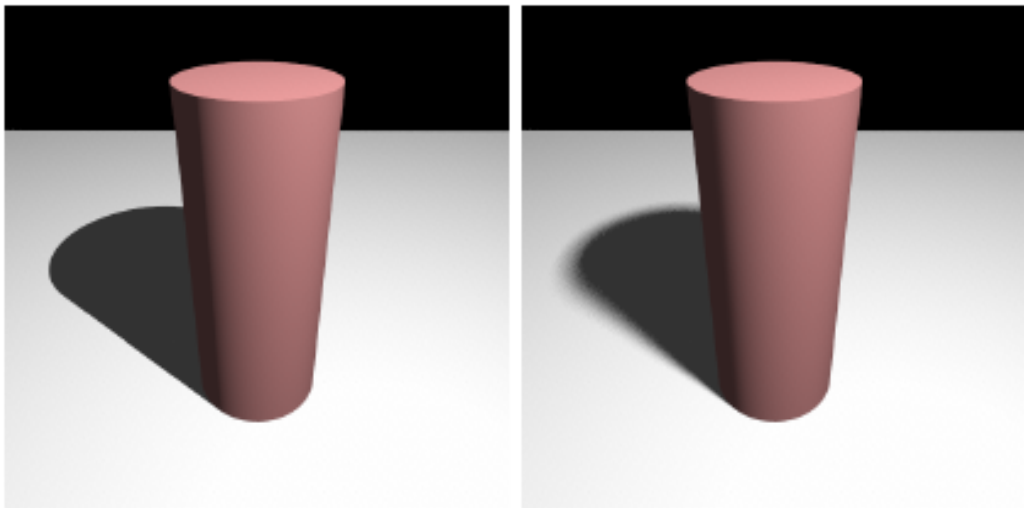


Fig. 9. Hard shadow vs soft shadow.

– **Shaders**: Generally speaking, shaders perform calculations determining the color and position of the pixels that will show up on a screen, which means the shadows, textures, how the reflections impact the look of the texture (if they do at , see Fig. 10), etc. Because shaders are fully GPU-sided, different devices have very different capabilities for running them. In the following example, mobile GPUs have support for half precision support, a number type (along with float and fixed) that requires less power and is usually faster to do calculations. Transcendental mathematical functions should be avoided since they are quite resource-intensive, as well as custom operations that are built-in on the engine, such as in Unity since they tend to be much more efficient. In general, we should use Unity's built-in mobile shader for mobile devices whenever possible because they are already optimized and should only be substituted if any problem or incompatibility arises, or if we need an object to look better [30].

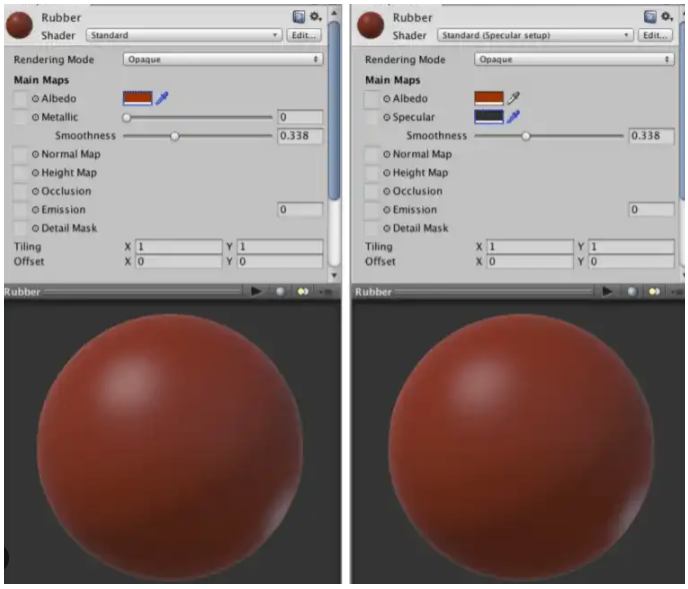


Fig. 10. Different shaders.

– **Vertices**: To reduce vertices count, merge by distance can be used. The "merge by distance" tool is similar to the decimate modifier in that it allows for the simplification of a mesh by joining vertices that are close to each other. This can be useful in model design to avoid problems such as overlapping faces, which can cause issues with texture application and other elements. The "merge by distance" tool can be used with a distance value of 0 to deal with duplicate vertices, or with a value greater than 0 to achieve a result similar to the decimate modifier. Overall, this tool provides a useful way to simplify meshes and improve the performance of game models.

– **Geometry of the system**: Game models are made up of polygons, with a higher number of polygons resulting in more detailed and realistic assets. However, this can be resource-intensive during rendering. To be drawn on the screen, all the polygons in a model must be converted into triangles, as GPUs can only render objects made of triangles. To improve performance, game developers should use models with a low number of polygons, which are easier to draw on the screen. They can then use textures and shaders to add more detail. The number of models used in a game can also affect performance, with Unity's documentation recommending that mobile games use meshes with between 300 and 1500 polygons per mesh. However, these numbers may vary depending on the number of models in a scene. Polygon reduction can be achieved using tools such as Blender's decimate modifier. Nevertheless, it is more efficient to design low-poly models from the start rather than reducing the polygons in an existing model [28].

  A technique to deal with this is the Level of Detail (LOD) system, a technique used in game development to optimize performance by dynamically changing the number of polygons rendered on the screen based on the distance of the object from the observer. The LOD system works by having multiple versions of each model, with each version using a different number of polygons (Fig. 11). When an object is close to the camera and requires more detail, the LOD system will use a version of the model with a higher number of polygons. When the object is further away and requires less detail, the LOD system will use a version of the model with fewer polygons.

This allows the system to be effective in saving resources and improving performance by only rendering the amount of detail that is necessary at any given time [29].

A step further would be the billboarding technique. This basically consists of representing a 2D object in a 3D world. Usually, the illusion of the 3D effect is created by having the object rotate in a way that it is always facing the camera. This can be achieved by shaders (drawing the texture in the direction of the camera) or scripts (rotating the object to face the camera) [31]. This is very frequently used in grass and low vegetation, as well as trees when they are far away.

Fig. 11. Visual example of LOD working on a wheel model.

2.4.2 Conclusion

The state of the art provides a detailed analysis of SVRs' constraints, specifically focusing on the Meta Quest 2. Through this analysis, several optimization techniques have been proposed to mitigate these constraints, ranging from efficient texture mapping to system geometry optimization using LOD and billboarding techniques. Despite the inherent limitations of SVR headsets, there is substantial potential for the application of these strategies. These methods can push the boundaries of VR gaming experiences, contributing to the evolution of next-generation VR gaming landscape.

2.5 General Conclusions

PCG has emerged as a powerful tool that offers unlimited potential for creating diverse and unique gaming experiences. Its capacity to algorithmically generate in-game material not only enhances replayability but also ensures that no two gaming sessions are identical. However, caution is advised when implementing this technique, as over-reliance on procedural generation can potentially result in a lack of cohesive design and narrative consistency. Co-creation of content, on the other hand, promotes a participatory design approach that increases player engagement and sense of ownership. It is an approach that transcends traditional game design by empowering players to contribute to the creative process. This participatory design strategy, while innovative, requires careful design of tools and interfaces that allow players to express their creativity while maintaining game balance and integrity.

SVR headsets represent the frontier of immersive gaming experiences. Their self-contained plug-and-play nature offers a high degree of mobility and freedom. Despite current limitations in image

quality and refresh rates, advancements in processor technology suggest a promising future for SVR. However, as VR continues to penetrate the mainstream, considerations around user comfort, safety, and accessibility become paramount.

Considering the insights gleaned from this state-of-the-art, future research and development efforts should consider these trends and their implications. The convergence of these technologies offers a promising pathway towards creating more engaging, immersive, and personalized gaming experiences. However, careful attention should be paid to the challenges and considerations that each trend presents, ensuring a balanced and thoughtful approach to game design and development.

3 Design, Development, and Validation Techniques

3.1 Introduction

This project was conducted concurrently with the development of AViR (FCT funded project, EXPL/CCI-INF/0298/2021), a VR application designed to assist women who have experienced early pregnancy loss. The research and development processes were interwoven, with findings from state-of-the-art research directly influencing the design and development of the project. A particular focus was placed on performance due to the constraints inherent to mobile VR headsets. The iterative design process, punctuated by regular prototyping and testing phases, allowed for the gradual refinement of the project. The insights gained from the state-of-the-art research were integrated into the implementation of customization packages in the AViR project. Once these packages were developed and refined, they were isolated for independent use and further optimized through consultative sessions with developers in the NeuroRehabLab (https://neurorehablab.arditi.pt/).

Validation of the project was conducted via four distinct means: focus group with developers to validate the implementation of the developed packages in different projects; usability studies that involved users testing the implementation examples of the packages; comparative technical analysis throughout the project development; and sporadic sanity tests executed during the development process. These various testing and validation methods ensured that the developed packages were robust, user friendly and fit for purpose.

3.2 Iterative Development Process

An iterative development approach is at the core of the methodology of the project. This practice, a standard at NeuroRehabLab, is characterized by cycles of planning, development, testing, and evaluation, facilitating constant refinement of the VR experiences being developed. Central to the iterative development process were the weekly meetings, providing an organized rhythm to the project. Progress was consistently evaluated, and tasks related to testing, implementing, or fixing aspects of the project were assigned based on the discussions and the work presented. The team participating in these meetings comprised project supervisors, a psychologist, and a PhD in informatics engineering student. Their diverse expertise allowed comprehensive evaluation and feedback addressing multiple facets of the project. Criticisms and advice received were incorporated into tasks for the following week, facilitating the project's continual evolution and refinement. This method allowed a swift response to emerging challenges and an effective incorporation of new findings, thereby enhancing the overall project quality.

3.3 Toolkits and Software

This project utilized a comprehensive set of development tools and software, each contributing uniquely to different stages of production. Here is an overview of these tools and their specific roles:

1. **Unity3D (https://unity.com/):** We chose Unity3D as the primary game engine due to its wide-ranging capabilities in building complex 3D environments and compatibility with numerous VR platforms. The rich set of resources provided by Unity's active community, alongside its intuitive user interface, made it an ideal choice for implementing PCG and co-creation tools.

2. **OpenXR for Unity:** To guarantee a wide spectrum of hardware compatibility, we used the OpenXR platform for Unity. OpenXR provides a standard API that works across multiple VR hardware platforms, ensuring our project's accessibility and future scalability. The integration of OpenXR with Unity streamlined our development process and ensured smooth performance across diverse VR devices.

3. **VSCode (https://code.visualstudio.com/):** As our Integrated Development Environment (IDE), we used Visual Studio Code (VSCode). It offers features like IntelliSense for efficient code completion, an advanced set of debugging tools, and seamless Git integration. Its support for a multitude of programming languages and user-friendly interface significantly enhanced our coding efficiency.

4. **PlasticSCM to GitHub:** Initially, PlasticSCM was our choice for version control, given its robust branching and merging features. However, as the project advanced, we transitioned to GitHub for its comprehensive issue tracking, for being the standard at NeuroRehabLab, and more intuitive user interface. GitHub's widespread adoption in the developer community also provides better opportunities for future collaborations.

5. **Photoshop (https://www.adobe.com/products/photoshop.html) and DaVinci Resolve (https://www.blackmagicdesign.com/pt/products/davinciresolve):** Adobe Photoshop was utilized for editing UI elements, creating icons, and crafting textures when necessary. DaVinci Resolve was our tool of choice for editing and composing video demonstrations, providing an effective means to showcase our project's features.

6. **Hardware:** Various computer configurations were used to test our project to ensure broad accessibility and performance. These configurations included a range of CPU and GPU setups: Ryzen 3600 with RX 580 8GB, Ryzen 3600 with GTX 1070 Ti, and i7-1120 with RTX 3060M. For VR hardware, we chose the Meta Quest 2 due to its high resolution, positional tracking capabilities, and its widespread usage in the VR community.

7. **Unity Assets:** We used assets from various sources in the Unity Asset Store to expedite the development process. These assets were selected based on their quality, versatility, and compatibility with our project. Their usage helped in enhancing the visual and interactive elements of our VR environments.

The technical aspects of this project played a pivotal role in shaping the overall VR experience. The careful selection of development tools, software, and hardware configurations catered to the complexities of VR development, ensured broad accessibility, and facilitated the seamless execution of our design principles. Utilizing Unity3D as our game engine, OpenXR for cross-platform compatibility, VSCode for efficient coding, and a range of hardware configurations, we believe we succeeded in creating robust and versatile packages.

The true measure of the project's success lies in its effectiveness and impact, which we assessed through a series of rigorous evaluations. These evaluations not only validated our methodologies and technical choices but also helped us in refining our packages to meet the needs and expectations of both the end-users and developers. The next section delves into the specific metrics and evaluation methods we used to assess the effectiveness of the developed packages and define our project's success criteria.

3.4 Validation Approaches

3.4.1 Technical Analysis

This method allowed us to directly compare two different scenarios related to various elements of the project. Our primary use of technical analysis was to assess the efficiency of different optimization methods in the context of VR development. This was especially true for the "Exploration" package, which was the most demanding in terms of performance. We closely monitored performance metrics such as FPS, draw calls, and triangle counts across multiple loads and optimization techniques. This analysis was conducted using a Meta Quest 2 (Table 3) and a system equipped with a Ryzen 5 3600, GTX 1070 ti, and 16GB RAM (Table 20). This systematic approach helped us determine the limits of the Meta Quest 2 (an generally other SVR HMDs), and which techniques were most effective, guiding our decisions for their implementation in the project. After identifying the best optimization techniques, we were able to achieve the desired FPS for the mobile build by adjusting the LOD settings. We also conducted other tests in a less structured manner, comparing aspects like standard vs. mobile shaders, texture resolutions, compression methods, and lighting techniques. The results of these tests were more subjective, often depending on visual quality and the perceived value of the performance increase.

3.4.2 Sanity Testing

Sanity testing, a basic form of testing conducted to ensure that functionalities of a system or component are working as expected, was utilized sporadically throughout the project. It served as a quick checkpoint to catch any severe, high-level problems, especially for the procedurally generated content in the Exploration package, which was prone to bugs. Data collected through these sanity tests primarily consisted of written user feedback, including observations, usability comments, and bug reports. This helped in identifying unnoticed usability issues and bugs, promoting the system's overall stability and user experience.

3.4.3 Usability Testing

A crucial component of the evaluation methodology was usability testing, conducted at various stages of development for all three packages: Exploration, NPC, and Garden. Usability testing was performed using different strategies, such as direct observation, the think-aloud method, questionnaires, and interviews, depending on the development stage and the package being tested. For example, during the later stages of development, the Exploration and Garden packages were tested with a group of participants inexperienced in gaming (see 5.2.1.2 and 5.3.1.2: Preliminary Usability Testing). Their interaction with the system was observed, and their thoughts were recorded using the think-aloud method, alongside post-test interviews. This approach provided rich in-depth data to further refine the packages.

In the final validation phase, a more systematic and formal approach was taken. Thirteen participants were recruited to test implementation examples of the three packages (see section 6.3). The purpose of this phase was to answer the research questions RQ2 and RQ3: "Can Unity packages for custom content generation engage users in gameplay?" and "Can users noticeably differentiate between the personalities of AI agents in a Unity package during gameplay?" (specific to the NPC package see section 6.4). For each package, participants were given a task-based questionnaire to complete during the session. Upon session completion, they were then asked to fill out additional questionnaires designed to evaluate different aspects of the package.

3.4.4 Focus Group

While the packages were implemented in two projects, AViR and EmotionalVRSystem, providing an opportunity to evaluate the performance of the packages in real-world scenarios, we believed it was necessary to gather specific insights from developers. For that purpose, we conducted a focus group with developers towards the latter part of the project (see section 6.2). During this session, real-time demonstrations of each package implementation were showcased and feedback was collected on their utility and effectiveness. This interactive and open discussion format allowed for constructive feedback and collective brainstorming, enabling the identification of potential improvements to enhance the effectiveness of the packages in real-world scenarios.

3.5 Conclusion

The intertwined relationship between research and development, anchored by the iterative development process, has ensured that the packages are both innovative and grounded in practical needs. The range of validation approaches, from technical analysis to usability studies, has been instrumental in refining the packages, ensuring their reliability and effectiveness in real-world scenarios. The tools and software selection further ensured the project's versatility and adaptability across different platforms and hardware configurations. As the project moves forward, these foundational principles and methodologies will continue to guide its evolution, ensuring its continued relevance and utility in the rapidly changing landscape of gaming and VR.

4 Setup

4.1 Introduction

The setup process for our packages is designed to be straightforward and efficient, allowing developers to easily integrate our solutions into their projects. The developer can start by downloading the desired package either from our repository on GitHub (https://github.com/NeuroRehabilitation) or from the Unity store (https://assetstore.unity.com/publishers/90965). In GitHub, the developer needs to download the package in the "releases" section of the repository. Furthermore, given that the packages are open source, the developer can choose to download the source code and manually import it to his/her project or start the project as that repository. If the package is obtained from the Unity store, it can be directly imported into the Unity project using the Unity interface. Furthermore, the packages are accompanied by detailed instructions and documentation. This guidance can help developers understand the functionalities and customization options of the packages, and how to best integrate and use them within their projects. Once the packages have been imported into the developer's project, they will have access to a range of customizable parameters. These parameters differ between packages due to their unique functionalities and purposes. These parameters can be easily adjusted according to the developer's needs, allowing for significant flexibility and customization. We aimed to keep the setup process simple to ensure our packages are accessible and easy to use, helping developers create personalized experiences with ease.

4.2 Infrastructure Setup

4.2.1 Hardware Used

This project employed the Meta Quest 2 (formerly known as Oculus Quest 2) as the exclusive VR headset for all stages of development and testing. The headset was selected for its standalone capability, wireless freedom and advanced hardware specifications that made it suitable for our experimental scenarios. However, due to the use of OpenXR, virtually any headset could be used for development. Development work was primarily conducted on a high-performance computing machine equipped with a Ryzen 3600 CPU, a GTX 1070Ti graphics card, and 32GB of RAM. This robust system allowed us to manage resource-intensive tasks such as 3D modeling, environment rendering and other computational activities required in the development of our packages. Additionally, an Elgato Stream Deck (https://www.elgato.com/us/en/p/stream-deck-mk2-black) was employed during the testing of the NPC package. This device was used in combination with the Wizard of Oz technique to simulate intelligent behavior in NPCs during early stages of testing. The Stream Deck provided a convenient interface to manually control the NPC actions in real-time, helping us to rapidly prototype and test different interaction scenarios.

4.2.2 Software Configuration and Setup

The project was developed using a Windows 10 operating system, providing the necessary compatibility and support for our chosen tools and software. Among these, the OpenXR SDK was selected due to its robust capabilities and compatibility with various VR headsets. This allowed us to ensure that the functionality of the packages would not be compromised, even if developers decided to use different VR-specific SDKs. The development heavily relied on three APIs for the NPC package: Microsoft's Speech-ToText (STT), Text-To-Speech (TTS), and OpenAI's GPT-3.5. The choice of GPT-3.5 over GPT-4 was influenced by its faster response times, although developers

have the flexibility to adjust this parameter based on their specific needs. To manage and install software packages required for the project, which were not available or outdated in the Unity Store, we employed NuGet. This was particularly instrumental for integrating Microsoft Speech Services required for the implementation of TTS and STT features.

4.3 Development Environment

The development environment was set up around Visual Studio Code (VSCode). Despite the prevalent use of Visual Studio in similar projects, we chose VSCode due to its lightweight nature, ease of use, and an extensive marketplace of extensions. This choice proved to be beneficial as it streamlined our development process significantly. Two key extensions utilized within VSCode were Unity and C#. The former provided a comprehensive toolset for developing in Unity, while the latter offered useful tools and features that enhanced productivity while working with C#, the chosen programming language for this project.

The transition from PlasticSCM to GitHub for version control marked an important shift in our development process. We initially faced usability issues with PlasticSCM and found that this sentiment was shared among many users [32–34]. In addition, in GitHub, the built-in Git feature in VSCode was our primary tool for version control, with GitHub Desktop being employed to resolve any arising issues.

4.4 Package Setup

As mentioned before, the setup procedure of our packages requires the developer to import the packages from either the GitHub repository or the Unity Asset Store. Alternatively, users may elect to initiate the project with the source code directly obtained from GitHub.

4.4.1 Exploration Package

The setup process for the Exploration Package requires the user to initiate the scene where the capabilities of the packages are showcased. For greater flexibility, the prefab can be imported directly into an existing scene.

4.4.1.1 Mesh Generator

The Mesh Generator component (Fig. 12) includes several configurable parameters to ensure fine-grained control over the generated terrain:

- *Terrain Coloration*: Assign a distinct color and an associated timing to determine the coloration of specific segments of the mesh, based on a normalized height. It should be noted that this coloration method is additive rather than absolute.
- *Terrain Generation Customization*: This component allows for considerable customization, including setting granularity (where higher values yield a more granular, or 'bumpy', terrain), determining the average height of the terrain, setting the number of 'octaves' or layers of Perlin noise, adjusting persistence (which impacts how the terrain is affected by octaves), and defining lacunarity (which influences the frequency of each octave). It also facilitates customization of the seed utilized for terrain generation, including an option to randomize the seed with each generation.

– *Path Customization*: Path Customization provides the opportunity to define parameters such as path gradients (manually or automatically, enabling smoother color transitions between the path and terrain triangles), the number of waypoints, lateral deviation values, the direct distance parameter (which influences path generation), and the number of minipaths that introduce additional paths once the primary path is established.

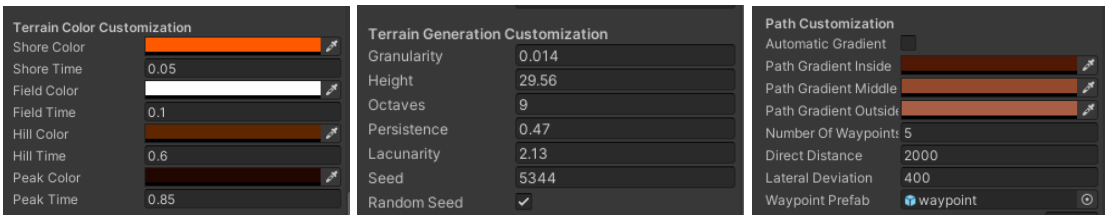


Fig. 12. Preview of the Mesh Generator component in the inspector.

4.4.1.2 Spawn Vegetation

Within this component (Fig. 13), users are empowered to introduce new vegetation into the scene:

– *Prefab*: The prefab to be spawned, which should incorporate the same components as provided examples.

– *Spawn*: A toggle to determine whether the object is to be spawned.

– *Num Objects*: The density of the spawned objects can be controlled through this parameter.

– *Octaves*: This specifies the number of Perlin noise layers.

– *Persistence*: This higher the value, the greater the impact of octaves on the terrain.

– *Pocket Threshold*: This value determines the likelihood of the object spawning within a pocket.

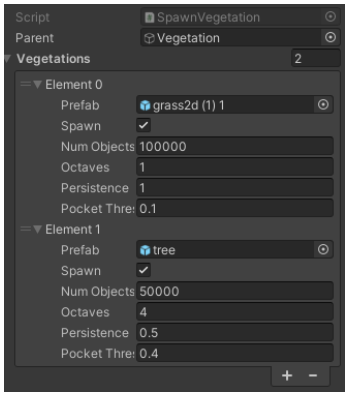


Fig. 13. Preview of the Spawn Vegetation component in the inspector.

4.4.1.3 Path Creator

The Path Creator (Fig. 14) allows for the specification of each item to be dropped by choosing the prefab and its drop rate to be spawned along the path.

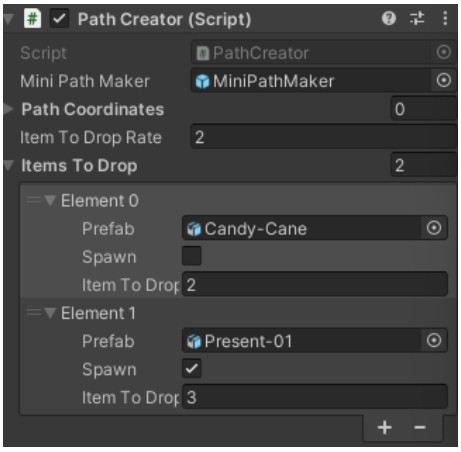


Fig. 14. Preview of the Path Creator component in the inspector.

4.4.2 Non-Playable-Character Package

The NPC package provides a comprehensive set of features to construct interactive NPC's in various scenarios. It demonstrates extensive flexibility and modularity, making it applicable across diverse contexts that demand an intelligent or creative NPC. While a pre-configured conversational agent is included in the package, the possibilities extend well beyond that specific application. Notably, due to the wider scope and diversity of potential applications, this package needs more extensive programming compared to the Exploration package, particularly when defining the inputs and outputs of the NPC's "brain".

4.4.2.1 Voice and Text Processing Services

This section delves into the various scripts and services implemented to handle voice capture, speech-to-text conversion, conversational AI integration, and text-to-speech synthesis. These services (Fig. 15) collectively enable a comprehensive voice interaction system, allowing users to engage with the virtual environment in a more immersive manner.

- **Voice Capture:** This feature facilitates continuous capturing of the user's voice in chosen time slices. The extracted audio can then be relayed to other scripts as necessary.
- **Speech to Text Service:** Leveraging Microsoft's speech API, this service converts the captured voice data into text. Developers are required to provide their API key and can choose the server region and processing language from a dropdown menu. There's flexibility to adapt the script for new languages or if standard language codes change.
- **OpenAIChat Integration:** Acting as the heart of the conversational system, the OpenAIChat script powers the generation of dialogues and actions for the agent. Developers have the liberty to customize various parameters related to the ChatGPT API, enhancing the conversational experience.
- **Text to Speech Service:** For those seeking a more auditory experience, this service synthesizes text into spoken voice. Again, it's built on Microsoft's infrastructure, requiring developers to input their API key. The service offers extensive customization options, from language and voice to gender and style. The chosen style's intensity can be tweaked, depending on the voice model used, ensuring a tailored auditory experience.

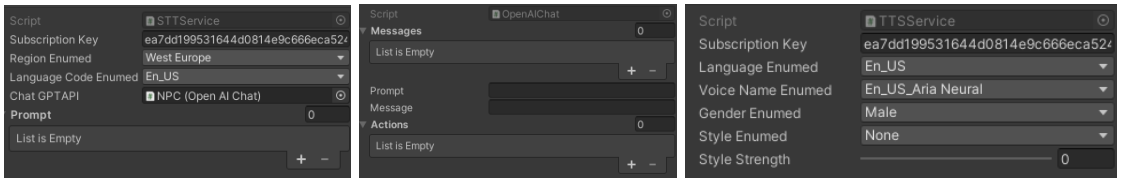


Fig. 15. Preview of the Voice and Text Processing Services in the inspector.

4.4.2.2 Agent Behaviour

Beyond merely producing text responses, the NPC delivers a set of actions in response to the interaction of the user. Developers can define a list of executable actions in the form of strings. The NPC will then select and execute the most fitting actions from this list in response to user interactions. Developers can also specify the character's memory capacity (i.e., how many responses and questions the character retains). Furthermore, the script allows developers to define the knowledge base of the character. This could include information such as the location of specific items or the purpose of certain objects. Lastly, the personality of the character is determined within this script. Developers can specify how the NPC should behave (Fig. 16), whether that be as an easily irritated wizard who dislikes interruptions, an enthusiastic mad scientist eager to share his thoughts, or any other conceivable persona.

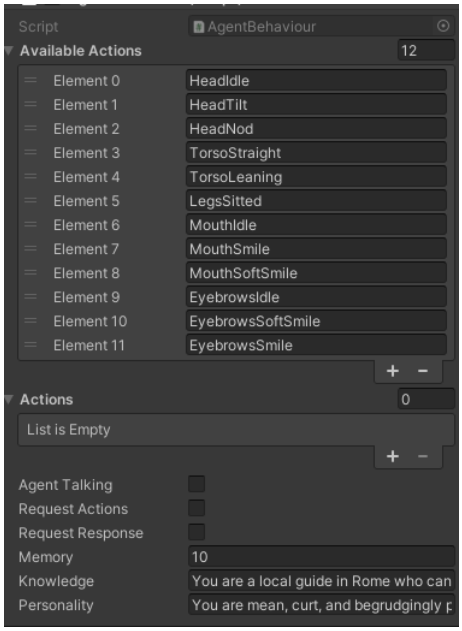


Fig. 16. Preview of the Agent Behaviour component in the inspector.

4.4.3 Garden Package

The Garden package allows the user to create its own environment. This is done by allowing the developer to introduce different interactable, spawnable items, and allowing the user spawn and interact with them in the virtual world. The Garden Package is designed to be effortlessly incorporated into a project. Developers only need to import the prefabs they wish to use (Memory Chest, Spawn Menu, Gallery).

In the Spawn Menu, instead of manually creating objects, setting their hierarchy, padding, margin, etc., developers can simply utilize the MenuContent.cs script to specify the objects to spawn (Fig. 17). This efficiency is achieved through a structured list of classes. Each category, such as 'benches', holds different items, and each item contains an icon, a prefab, and a title. As such, developers can conveniently organize and identify various items within their 'garden'. Moreover, developers have the option to save their garden setup - specifically, the names of items and their positions - to a JSON file. This allows the stored items to be automatically instantiated whenever necessary, providing a mechanism for preserving and reproducing specific garden configurations.

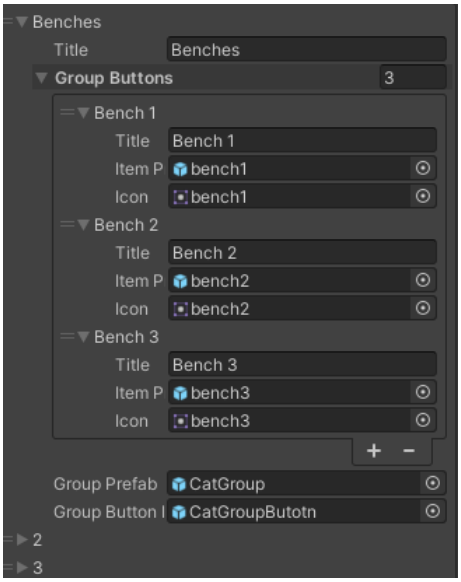


Fig. 17. Preview of the Menu Content component in the inspector.

4.5 Dependencies

4.5.1 Non-Playable-Character Package

Among all the packages, the NPC Package stands out as having significant dependencies. This package relies on the Microsoft Speech API package, which needs to be installed via NuGet for Unity. The unofficial Microsoft Speech API package available in the Unity store does not function properly, hence the necessity for the NuGet installation. Although developers who are importing the packages will not need to perform this setup, it is worth noting for debugging purposes. Knowledge of this dependency could prove essential in troubleshooting and resolving potential issues that may arise during development.

4.6 Conclusion

The project setup process has been meticulously designed to be both straightforward and effective. The choice of high-performance computing system, software configuration, development environment, and VR equipment has proven to be reliable and efficient. The three packages—Exploration, NPC, and Garden—provide unique features that facilitate the creation of a rich, immersive VR environment.

The transition to GitHub for version control brought improvements in traceability and efficiency, demonstrating the importance of selecting the right tools for project management. Moreover, the dependencies for the NPC package and the need for its installation via NuGet for Unity underscore the importance of carefully managing and documenting dependencies in complex projects.

Looking forward, the modularity of the project setup allows for scalability and adaptability. The highly customizable parameters in each package make it possible to extend or modify the functionalities to suit specific development needs. This project setup has not only proven effective for the current VR projects but also provides a strong foundation for future VR and non-VR projects, potentially accommodating even more complex scenarios and functionalities.

5 Development

5.1 Introduction

5.1.1 Overview of Development

In our state-of-the-art, we discerned two primary methodologies by which these packages could economize development resources and facilitate automated content generation: PCG and Co-Creation of Content. Accordingly, our initial objective was to design two distinct packages, each aligning with one of these methodologies. However, the concurrent rapid advancement of LLMs presented an intriguing opportunity to expand our scope. Recognizing the potential of leveraging LLMs to generate content imbued with intelligence, we opted, albeit at an advanced stage in the project's development, to initiate the construction of an additional procedural generation package—referred to as the NPC (Non Playable Character) package. Complementing this, we developed the Exploration package within the Procedural Generation section, designed to synthesize a natural environment replete with paths and strategically placed items, thereby creating a more immersive experience. In contrast to the procedural generation approach, the Co-Creation Package, embodied by the Garden package, empowers users to shape their own worlds through co-creation, offering the capability to spawn personalized objects.

A defining characteristic of our development process was the commitment to ensuring compatibility with mobile VR systems on the Android platform, such as the Meta Quest 2. Particular attention was paid to the optimization of game objects, models, and shaders to ensure fluid performance within the constraints of mobile VR. Nonetheless, the flexibility of our design permits and expects deployment across various platforms supported by Unity, offering developers the liberty to customize game objects to enhance quality as platform capabilities permit. Thus, our packages serve as a versatile toolset, primed to support a broad array of development needs in the continually evolving landscape of digital content creation.

5.2 Procedural Generation

5.2.1 Exploration Package

5.2.1.1 Development Journey

The genesis of the Exploration Package was rooted in a proof-of-concept endeavor targeting terrain generation with vegetation that is compatible with mobile VR (since this was the package were performance mattered the most). The initial phase confronted various challenges, primarily owing to the hardware limitations inherent to mobile builds, which often manifested as inscrutable issues. In response to these complexities, we embarked on a systematic exploration of performance impact through A/B testing, creating a benchmark scene. Our analysis culminated in the identification of optimal graphics settings, detailed in Table 3. Experiment number 7 (Fig. 18) emerged as the most efficacious configuration, although with some noted limitations such as unoptimized models and shaders for mobile.

Table 3. Comparison of Graphics Settings, Observed in Meta Quest 2.

n	Spaced Trees	Tree Count	Resolution	FPS	MSAA	Anisotropic Textures	Shadows
1	yes	0k	1x	72	no	no	yes
2	yes	10k	1x	72	no	no	yes
3	yes	20k	1x	<40 (very low fps loading)	no	no	yes
4	yes	30k	1x	<40 (even lower fps loading)	no	no	yes
5	no	10k	1x	72	no	no	yes
6	no	10k	2x	10	no	no	yes
7	no	10k	1.3x	72	no	no	yes
8	no	10k	1.3x	70	2x	no	yes
9	no	10k	1.3x	67	4x	no	yes
10	no	10k	1.3x	50	8x	no	yes
11	no	10k	1.3x	65	4x	yes	no
12	no	10k	1.3x	63	4x	yes	yes

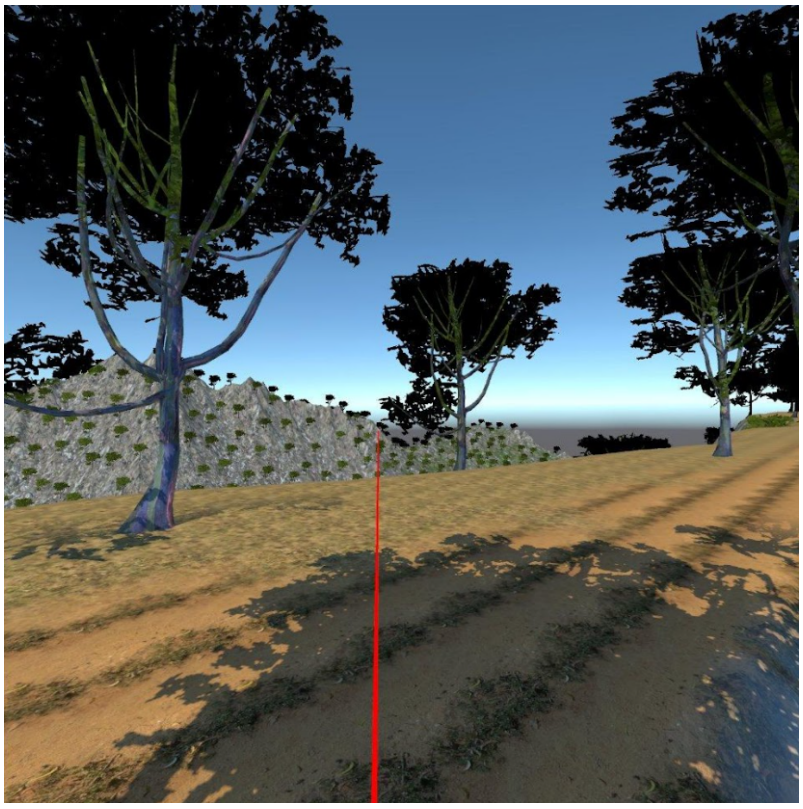


Fig. 18. Demo of experiment number 7.

There are some limitations that should be acknowledged. Foremost, there is the absence of optimized models and shaders specifically tailored for mobile platforms. Additionally, the terrain is not generated procedurally. All these are factors that may influence the fidelity of the FPS values

presented in the associated table. It is critical to recognize that these FPS values may fluctuate in varying scenarios, given the influence of external variables that are not represented in the table, such as the type of lighting, draw calls, and triangle count, among others. Nevertheless, the data gathered serves as a robust guideline, contributing valuable insights into the system's behavior. Subsequent research was meticulously conducted, culminating in the data that was previously elucidated in the state-of-the-art section. From this comprehensive investigation, several salient conclusions were derived as follows.

Mobile VR platforms exhibit reduced performance with triangle counts exceeding one million. Draw Calls more than 600 are similarly detrimental to performance. Certain shaders, particularly those associated with tree branches and leaves (Fig. 19), have been found to halve performance and baked lighting is favored over real-time lighting.

Fig. 19. Early test scenario. Left is optimized with mobile shaders and baked lighting, resulting in playable FPS; right is unoptimized with realtime light, resulting in unplayable FPS.

In pursuit of an optimal aesthetic that would seamlessly align with the performance constraints inherent to mobile VR platforms, we explored various visual styles. Recognizing the necessity for both visual appeal and low computational overhead, we conducted experimentation with multiple designs (Fig. 20). A substantial portion of these assets was sourced from the Unity Store, leveraging freely available resources to facilitate our iterative design process. This approach allowed us to assess the interplay between aesthetics and performance, guiding our selection of a style that met the unique requirements of the mobile VR environment.

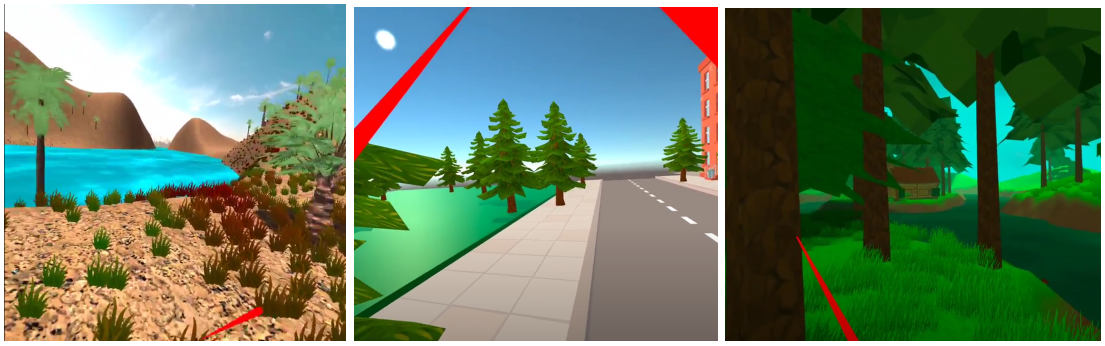


Fig. 20. Examples of each of the three styles (savannah, urban nature and nature, respectively) we explored.

The initial exploration of a suitable environment led us to gravitate towards a nature-inspired scenario, aligning with our foundational concept of a customizable terrain integrated with a path. This baseline, while inherently adaptable and open to extensive customization by developers, serves as a starting point or exemplary model. The chosen scenario of an open world, adorned with a path and vegetation, possesses a generic appeal, ensuring its applicability across diverse projects.

With the aesthetic style preliminarily selected, the development process entered an iterative phase, focusing on the optimization and refinement of the chosen game objects. Key challenges and solutions were encountered in the following areas:

1. **Grass**: The rendering of grass represents a substantial challenge in game development—a challenge that is often underestimated by those outside the field, but which can dramatically impact performance. The complexity of rendering grass is such that even slight variations in approach can lead to significant variations in FPS. Our exploration of this issue (see Fig. 21 for the evolution) encompassed various techniques:

   – *3D Grass*: While initially considered, 3D grass was quickly deemed infeasible due to its resource-intensive nature and unsatisfactory visual appeal.

   – *2D Grass*: Recognized as a standard even in AAA games, 2D grass presented a more viable option. We investigated several 2D assets with a billboarding effect but found them to be inefficient for our requirements, given the awkwardness of the rotating body due to the billboarding effect.

   – *Two-Sprite Method*: By crossing two sprites of grass, we created an illusion of three dimensionality. This technique provided satisfactory results for a time yet necessitated additional optimization to enhance the appearance of abundance. Clumping grass in bundles and leveraging LOD Grouping allowed for strategic rendering based on distance.

   Dissatisfied with existing solutions, we eventually embarked on the development of a custom shader in GLSL, specifically optimized for mobile VR. This innovation significantly improved quality, incorporating settings to customize grass deformation (simulating wind) and billboarding distance. The transition from the two-sprite method was not without consideration, as the close billboard effect might appear unnatural depending on user locomotion, so this is something the developers might want to keep in mind. The noise function-based spawning of the

grass and developer-controlled LOD Grouping further refined the rendering process, contributing to a solution that balanced both aesthetic appeal and performance efficiency.

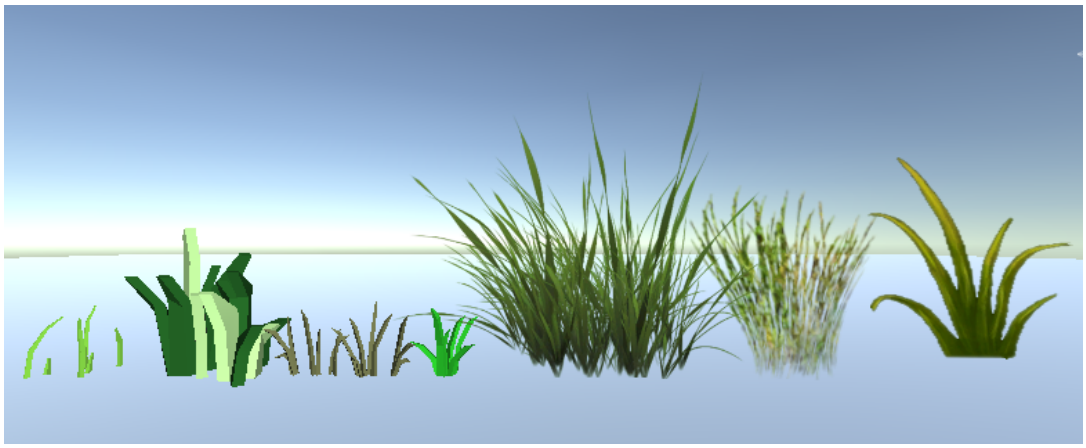


Fig. 21. Grass evolution throughout the development of the project. Left is oldest version, right is the newest, and being used in the package.

2. **Trees**: The rendering of trees presented challenges analogous to those encountered with grass, particularly regarding the leaves, which often employ a similar sprite-per-branch approach. An extensive examination (see Fig. 22 for the evolution) of shaders not specifically tailored for mobile platforms revealed performance issues when applied to trees within a mobile environment. Specifically, any tree asset incorporating transparency in the leaves led to a marked reduction in FPS on mobile devices, even with a single tree in view. This phenomenon appeared to be linked to overdraw issues associated with shader-based transparency handling on mobile platforms.

Recognizing the necessity for a mobile-specific solution, we identified and implemented a shader available in the Unity store, named *Trees/LeavesFast* [35], designed explicitly for mobile applications. This choice effectively addressed the transparency challenge, yielding satisfactory performance without compromising visual quality. The adoption of this shader underscores the importance of mobile-specific optimizations in rendering complex natural elements, demonstrating that generic solutions may lead to unforeseen inefficiencies within the unique constraints of the mobile VR environment.

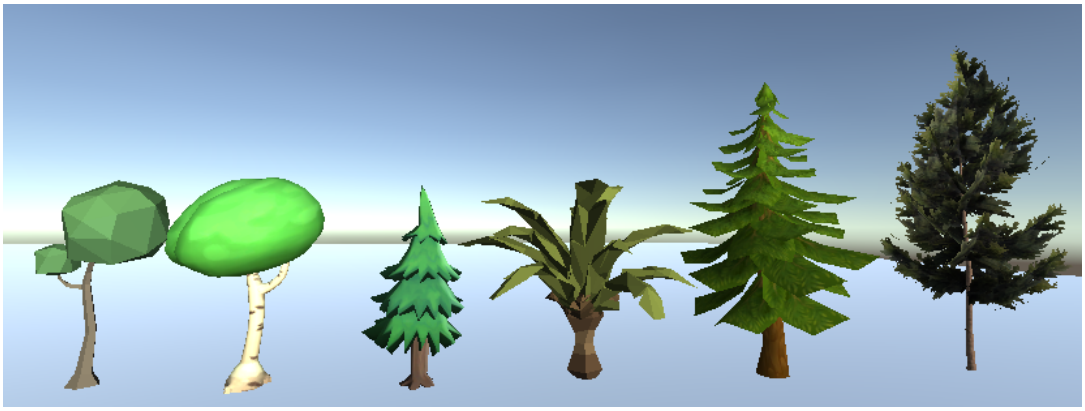


Fig. 22. Tree evolution throughout the development of the project.

Populating a terrain with trees, contrary to intuitive assumptions, proves to be a less demanding task than rendering grass. While the juxtaposition may seem incongruous, it can be explained by the differing density requirements between these two natural elements. To create a convincing forest, a relatively small number of trees is often sufficient, whereas a meadow demands a substantial density of grass to achieve a similar level of realism. In our development process, we nevertheless recognized the importance of optimizing tree rendering to align with the performance constraints specific to mobile virtual reality. To this end, we implemented a LOD Grouping strategy (Fig. 23) for the trees, allowing for dynamic adjustment of detail based on distance. As the trees recede from the viewer's perspective, they are rendered with progressively reduced detail, thereby conserving computational resources without sacrificing the overall visual experience. Furthermore, we applied a customizable noise function much to the likeness of the grass.

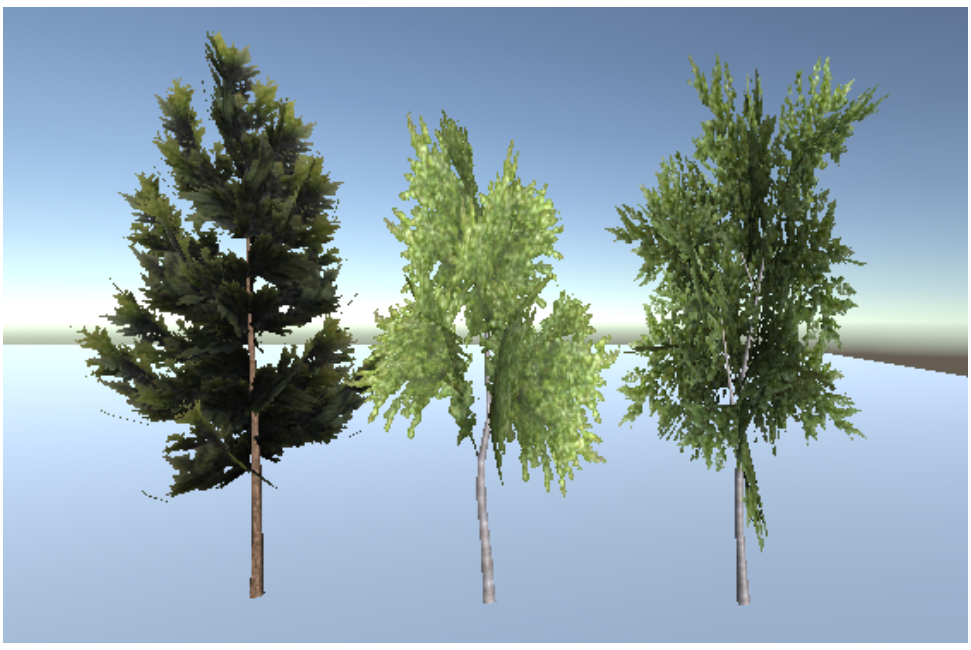


Fig. 23. Tree LOD, tris count: 400, 100, 70.

3. **Mesh Generation**: The initial approach to terrain generation leveraged the well-known Perlin noise algorithm, applied through a single octave layer to the heights of individual vertices on a mesh. A detailed examination of the specific algorithm used for mesh generation is presented later in this thesis.

4. **Path Creator**: The initial iteration of the path creation agent relied on manually placed placemarks, with the distance between them transversed and rendered using a line renderer. This approach, however, posed two significant challenges:

   (a) *Sharp, Unnatural Line*: The initial method created a stark and unrealistic boundary between the path and the terrain. A solution was found by introducing a transparency gradient to the edges of the line renderer, softening the issue. Later, we decided to occlude this by adding vegetation specifically at the edge of the path (Fig. 24).

Fig. 24. Evolution of the design of the path (path iterations: sharp edges, transparency gradient edges, and occluded edges by vegetation, respectively).

   (b) *Roll Values and Terrain Incline*: The line renderer's limitation on varying roll values between points led to a noticeable cut-off on one side of the path when the terrain was situated on an incline (Fig. 25). Addressing this issue proved more complex. While further iterations on this method were ultimately unfruitful, a provisional solution for the prototyping stage was implemented by mirroring the terrain's incline to the line renderer. Though not without limitations, this approach provided a temporary resolution to the alignment challenge.

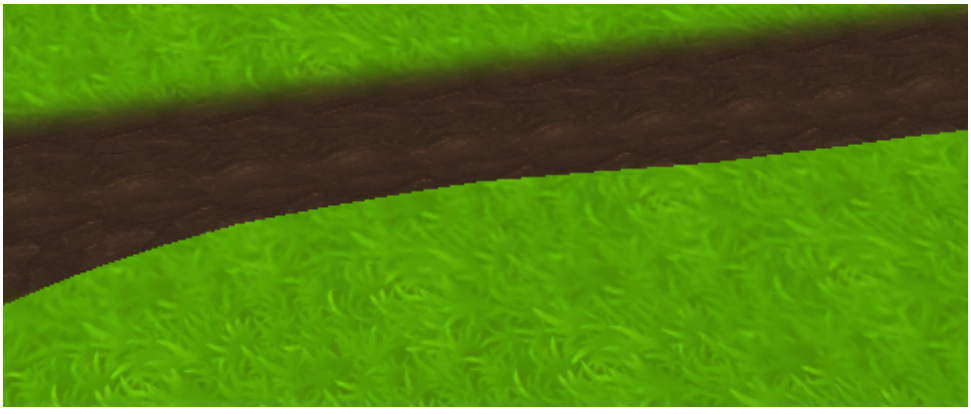


Fig. 25. Cut-off issue due to terrain incline.

5. **Water**: Water movement in video games is commonly simulated using vertex displacements within shaders. This approach, though effective, is computationally intensive, particularly for a mobile VR's GPU unit. Consequently, most assets from the Unity store were found to be incompatible with the Android build. One strategy to overcome this was to replace the vertex displacement in the shader with height displacement in the vertices of the water mesh. Unfortunately, this solution did not yield satisfactory results. It remained performance-heavy and, despite its potential acceptability if it were visually appealing, it looked unattractive. An alternative solution involved displacing the x and z axes of the texture, creating the illusion of waves in a 2D mesh. This approach was more successful and could be likened to a calm lake or body of water (see Fig. 26 for the comparison). It had a minimal impact on performance and was visually enhanced by adding reflections. The latter technique proved more acceptable in terms of both aesthetics and computational efficiency.

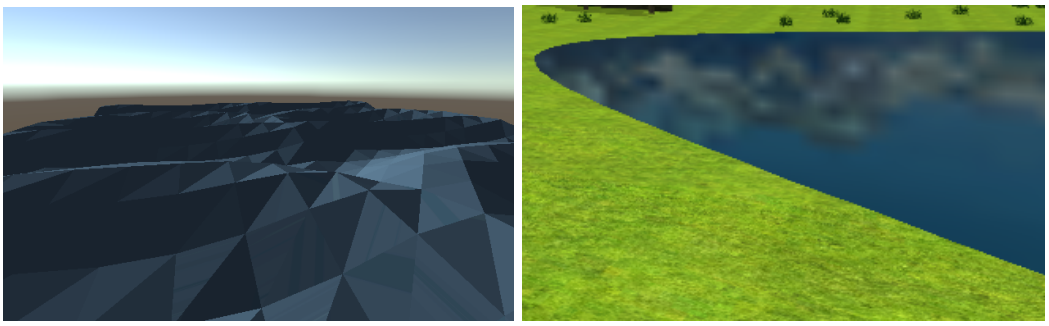


Fig. 26. Water design evolution, left is the mesh vertex displacement, right is the texture displacement technique.

This (Fig. 27) is what the prototype looked like at this point.

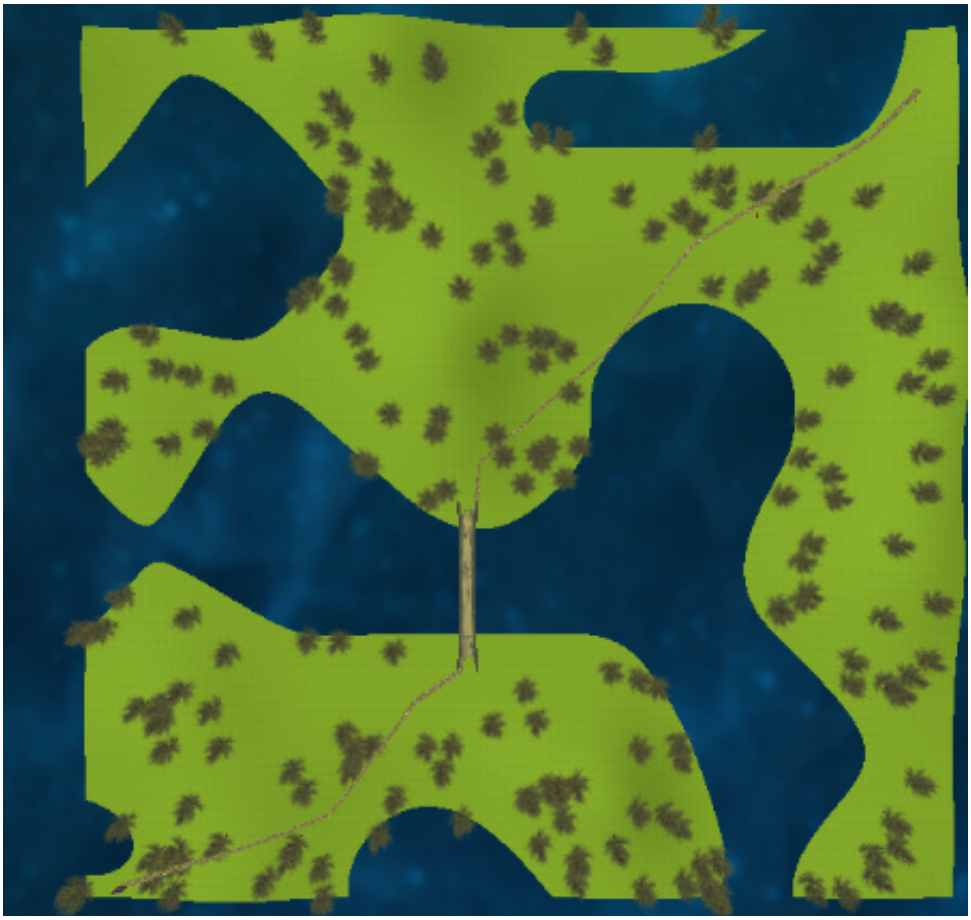


Fig. 27. Bird View of the generated terrain.

5.2.1.2 Test and Refinement

1. *Sanity Test 1*: The AViR team conducted a series of sanity checks to assess the game's functionality and robustness. A team member, Computer Engineering PhD student, with no prior exposure to the game undertook these evaluations using the build for the Meta Quest 2. The tester explored multiple terrain generations and executed a wide array of plausible in-game actions. The outcomes of this testing phase were twofold:

   (a) **Bug Identification**: Several bugs were detected during this phase. While these bugs were duly addressed, their specific details fall beyond the scope of this work. What is pertinent, however, are the underlying design flaws that gave rise to many of these bugs.

   (b) **Design Issues with the Path Line Renderer**: A notable design challenge emerged with the Path Line Renderer. Two specific methods, implemented to rectify the issue of the path line renderer being partially obstructed by inclined terrain, presented challenges:

   - **Alignment of Line Renderer**: Our strategy to align the entire line renderer's roll rotation with the incline angle at the player's standing position proved problematic. Given that the player remained always levelled, our approach involved an invisible cube near the player that determined its rotation and, consequently, the inclination angle. This method was susceptible to errors. Any discrepancies with the cube, such

as unintended rotations or positioning challenges, led to significant inaccuracies in the path's rotation. This not only failed to rectify the initial issue but, in some instances, exacerbated it.

- **Vegetation at Path Edges**: To mask the sharp transition between the path texture and the terrain texture, we spawned vegetation specifically at the path's edges. This method, while conceptually sound, had its challenges. Accurate placement of vegetation required calculations based on the path maker agent's position and rotation. Any premature termination of the path maker agent or discrepancies in its position and rotation often resulted in misplaced vegetation.

Following the evaluation from the sanity check, our attention was focused on addressing the prominent design flaw associated with the path line renderer. In our efforts to rectify this issue, we deliberated over two potential solutions:

(a) **Custom Shader for Additive Painting**: This solution proposed the development of a custom shader that permitted the additive painting of individual triangles on the mesh. As the path maker agent traversed the terrain, it would effectively "paint" the trail onto the mesh, ensuring a seamless integration of the path into the terrain.

(b) **Texture Blending Post Path Creation**: This alternative approach considered storing the path created by the agent. Upon the path's completion, a texture would be generated, showcasing the dirt path and bordered by a white margin. This texture would then be blended onto the mesh, providing a visual representation of the path.

After careful consideration, we opted for the first solution (Fig. 28). This decision was influenced by several factors:

- **Simplicity**: The custom shader approach was conceptually more straightforward, requiring fewer steps and less complexity in its implementation.

- **Performance Efficiency**: Our terrain mesh employs texture mirroring, a technique chosen for its efficiency in terms of performance and memory conservation. Adopting the second solution would necessitate the introduction of an additional high-resolution texture to cover the entirety of the terrain without mirroring. This would inadvertently increase the performance demand and memory usage, negating the benefits of our current approach.

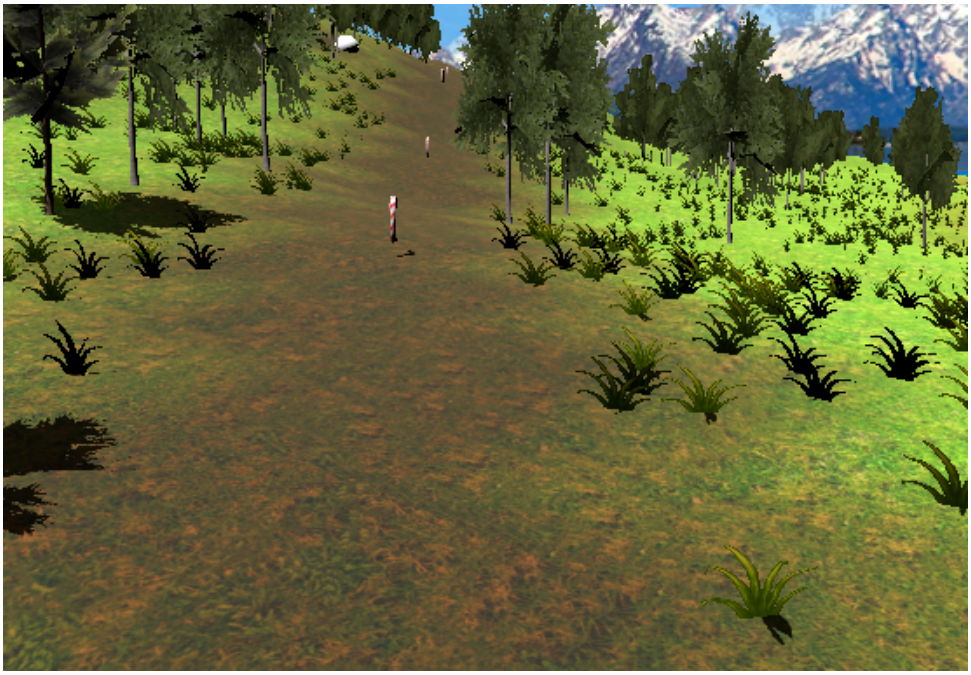


Fig. 28. Example of the new patch utilizing the mesh painting method.

2. *Sanity Test 2*: A second round of sanity checks further assessed the game's robustness and performance. The evaluations were carried out by the same team member who had previously conducted the initial sanity check. Using the build for the Meta Quest 2, the tester delved into multiple terrain generations and undertook a gamut of plausible in-game actions. In stark contrast to the previous evaluation, this round primarily revealed minor bugs, with no significant design flaws identified. However, a noteworthy observation was the abrupt transition between the terrain and bodies of water, resulting in a visually jarring effect. To address this concern, we developed a custom water shader with enhanced capabilities (see Fig. 29 for the comparison):

   – **Aesthetic Enhancement**: The new shader elevated the visual appeal of the water, rendering it more lifelike.

   – **2D Water Movement**: This feature was incorporated into the shader, instead of running in a script.

   – **Foam Border**: Most crucially, the shader introduced a customizable foam-like border at the water's edge. This addition significantly smoothed the transition between the terrain and water, providing a more natural and cohesive look.

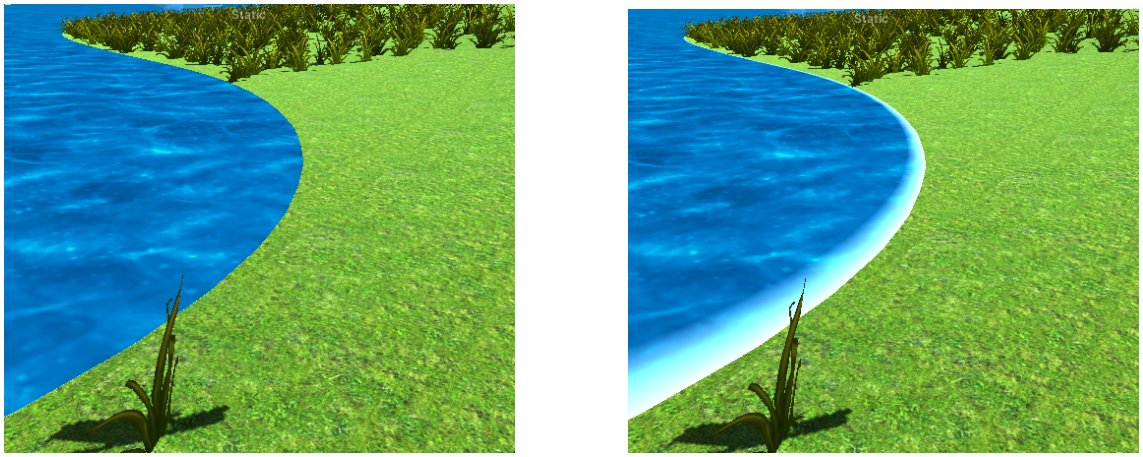


Fig. 29. Water without (left) and with (right) foam.

3. *Preliminary Usability Test:* In a pursuit to ensure this Package's accessibility and appeal to a broader audience, before starting the main studies, an informal usability test was conducted with two women aged between 40-50, who had little to no prior experience in gaming. Initially, challenges were encountered with navigation and controller handling. The unfamiliarity with gaming controls posed a learning curve for the participants. Remarkably, the adaptability of the design facilitated a rapid acclimation process. Within a matter of minutes, the participants became accustomed to the controls and were able to navigate and interact with ease. Notably, no significant usability issues were identified, and no frustration was reported by the participants, attesting to the success of previous refinements.

4. *Pilot Test*: Having exhausted the insights obtainable from our internal evaluation methods, a fortuitous opportunity presented itself. A member of the NeuroRehabLab expressed interest in leveraging our Exploration Package to expedite the commencement of their project. This pilot test offered invaluable insights. Testing our package in a real-world application provided feedback and observations that were not discernible in controlled settings, such as our previous sanity checks. It is also important to note the distinction in this test, since it is from the developers perspective.

   The developer's objective was to create a serious game using Unity. This game would guide participants through a journey across four distinct environments, each meticulously designed to evoke a specific emotion. Subsequently, emotional responses would be captured and analyzed using an Electroencephalogram (EEG). Over the span of three non-consecutive months, this collaboration yielded a wealth of feedback, including:

   (a) *Item Drop Angle*: A straightforward feedback from the developer pertained to the orientation of items dropped by the agent. It was suggested that, by default, items should align with the direction of the path to ensure consistency and enhance the user experience.

   (b) *Terrain Border Adjustments*: A noticeable concern with the terrain mesh was the abrupt truncation at its borders. To address this, we introduced a modification to the mesh generation algorithm. Specifically, as a vertex approached the edge of the terrain, its height was progressively decreased (see Fig. 30 for comparison). This adjustment ensured that the terrain edges gradually descended, forming natural looking shores that seamlessly submerged

beneath the water plane. This enhancement not only rectified the cut-off issue but also added a more organic transition between the terrain and the water, enhancing the overall aesthetic appeal.

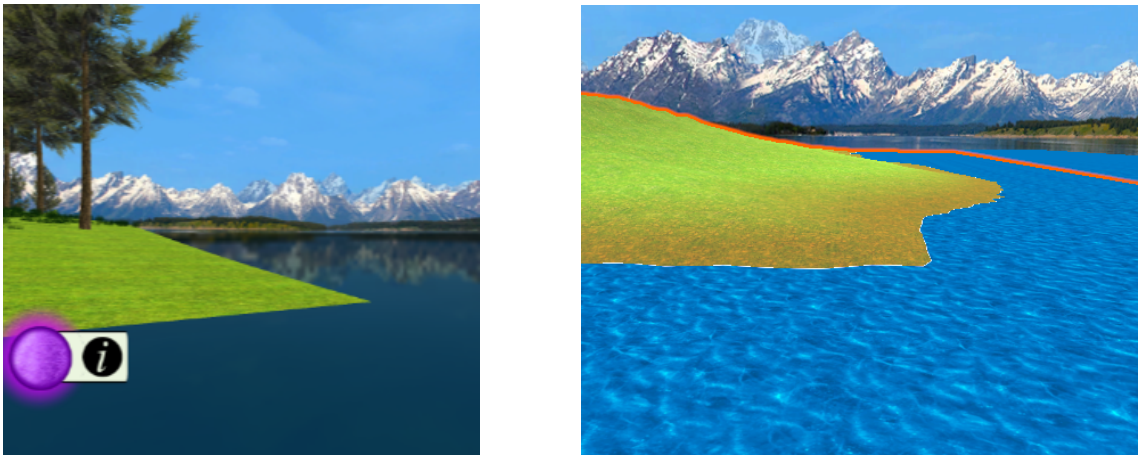


Fig. 30. Previous mesh generation technique with sharp, squared border (left) vs new technique with border blending at the shore with water (right).

(c) *Path Enhancement*: Feedback from the developer underscored the need to enhance the intrigue of the path, which was perceived as too linear and monotonous. Addressing this critique proved to be a complex undertaking. Our agent employs Unity's default NavMeshAgent technique. In this approach, the agent begins on a mesh, computes the optimal route to the destination, and proceeds accordingly. We explored multiple strategies to introduce variability and interest into the path:

i. **Obstacle Integration**: Introducing obstacles seemed a viable method to disrupt the linearity of the path. However, this often resulted in the agent failing to identify a feasible route to the destination, rendering the strategy ineffective.

ii. **Noise-Induced Diversions**: We experimented with the addition of noise as a force influencing the agent's movement. The intention was to generate more curves and deviations in the path. However, this approach presented challenges. The function introducing the noise was not equipped to discern obstacles such as walls or water bodies. Consequently, the agent frequently encountered impediments, resulting in temporary stalling. Additionally, the resultant paths were not substantially more engaging.

iii. **Waypoint Deviation**: Our final strategy entailed the introduction of waypoints interspersed along the direct route. These waypoints exhibited perpendicular deviations from the straight path, effectively creating bends and turns. This approach was markedly more successful, yielding procedural paths that were both engaging and customizable.

(d) *Multiple Octave Layers for Terrain Generation*: Feedback from the developer during the pilot study emphasized the potential benefits of utilizing multiple octaves to generate the terrain, aiming for a more organic and realistic landscape. While we had previously contemplated the application of fractal-based methods (in this context, pseudo-fractal) for various

aspects of content generation, the feedback prompted us to integrate it at this juncture (see Fig. 31 for comparison). By embracing a multi-layered noise approach, we observed a marked improvement in the terrain's appearance. This method produced terrains that were not only more intricate but also closely mirrored the natural undulations and topographical variations found in real-world landscapes.

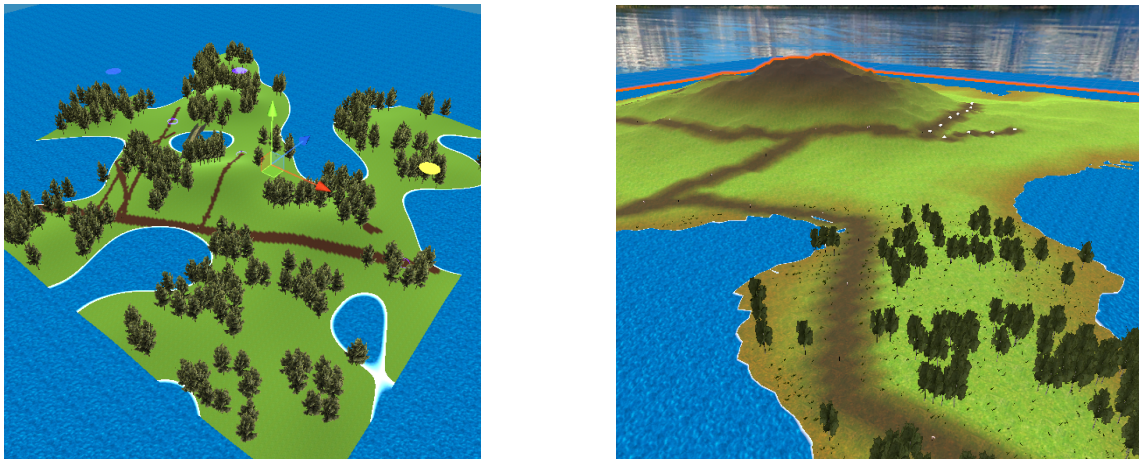


Fig. 31. Previous terrain and path generation methods (left) vs new (right).

(e) *Implementing a Generate-and-Test Approach to Terrain Generation*: While not a requirement for the developer's project, we identified this phase as an opportune moment to incorporate a generate-and-test (GnT) strategy for terrain creation. Our initial inclination was towards the Search-Based Approach. However, upon further analysis, the feasibility of implementing selection and variation functions for mesh generation appeared limited. Given that seed values for terrain generation are essentially random, distinguishing "better" seeds seemed impractical. Consequently, our adopted GnT method operates as follows.: It initiates with a seed value and then generates a terrain. If the resultant terrain does not accommodate a viable path (for example, if there is an impassable terrain feature, like a steep hill or a river), the algorithm increments the seed value slightly and regenerates the terrain. This iterative process continues until a suitable terrain, which supports a feasible path, is generated.

(f) *Terrain Generation Preservation*: Recognizing the significance of replicability, particularly in the context of serious games used for research, we designed this module with a high degree of determinism. This ensures that, given the same parameters, the terrain generation process will produce consistent results. Nonetheless, there remains a degree of variability, particularly in path generation, which can exhibit slight differences across iterations. To accommodate scenarios where developers require complete determinism, we introduced a feature that allows for the preservation of a specific terrain generation. Through this functionality, a developer can save a particular terrain configuration as a prefab, thereby ensuring its consistent replication in subsequent uses.

5. *Study RQ1 Evaluation and Feedback*: We organized a focus group to assess the project's alignment and effectiveness in addressing Research Question 1 (RQ1). Detailed insights from this

evaluation are elaborated upon in the "Studies" section. Beyond the study findings, the focus group provided valuable feedback that informed subsequent refinements:

- **Documentation Enhancements**: Participants emphasized the importance of comprehensive documentation. Consequently, we enriched the existing documentation with additional tooltips and ensured the inclusion of a detailed readme to guide users more effectively.
- **Path Gradient Automation**: Feedback indicated a preference for automatic gradient transparency for paths rather than manual settings. In response, we designed an algorithm that seamlessly generates gradient transparency for the paths.
- **Versatile Item Drops**: A unanimous suggestion from the group was the incorporation of functionality allowing for the dropping of multiple item types. To cater to this, we expanded the module to support a list of droppable objects, thereby offering developers greater flexibility in item interactions.

5.2.1.3 Package Architecture/Workflow

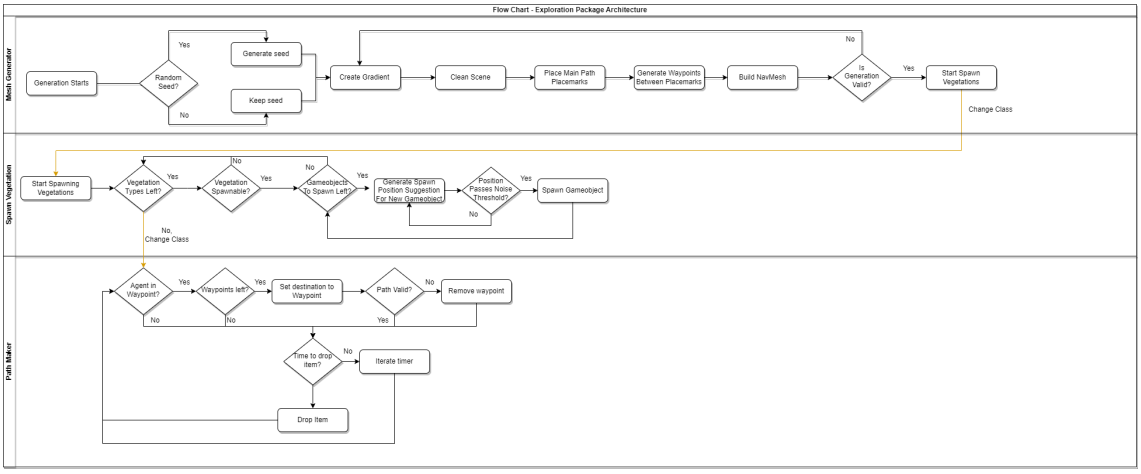


Fig. 32. Flowchart/Swimlane portraying the architecture of the Exploration Package.

The architecture of the Exploration Package (Fig. 32) is grounded in a series of sequential and parallel operations aimed at creating a rich and varied terrain. Here, we delineate the primary stages of the process:

- **Mesh and Structure Generation Algorithm**: This algorithm orchestrates the creation of both the mesh and essential "structures" for the terrain. It systematically processes various objects in a sequential manner, ensuring the meticulous generation of integral components such as waypoints and placemarks. This structured approach ensures the consistent assembly of the terrain's foundational elements, laying the groundwork for subsequent features and interactions.
  - **Gradient Mapping**: A gradient mapping is devised to facilitate the nuanced rendering of geographical features such as shores, plains, and mountains. This can be done early in the algorithm since uses normalized heights.
  - **Seed Generation**: A seed is generated for the terrain creation or adopted from one supplied by the developer, depending on its availability.

- **Mesh Generation**:

  The mesh generation algorithm is designed to create a terrain mesh using Perlin noise combined with multiple octaves to achieve a more natural look. The process can be broken down into the following stages:

  1. **Vertex Creation**: The vertices array is initialized to represent the mesh's vertices. A nested loop iterates over the dimensions x and z. For each combination of x and z, the algorithm calculates:

     * The distance to the closest edge to ensure that the terrain smoothly tapers off towards the edges.

     * The noise height using multiple octaves of Perlin noise: Perlin noise is a gradient noise function used extensively in procedural texture and terrain generation. The concept of using multiple octaves comes from the idea of layering the noise functions at different frequencies and amplitudes. By combining multiple "layers" of noise, one can achieve more complex and organic-looking results.

       · **Amplitude and Frequency**: Each octave has its own amplitude and frequency. The amplitude determines the height of the peaks and valleys in the noise, while the frequency determines how often they occur.

       · **Octave Iteration**: For each octave, the Perlin noise value is computed for the current x and z coordinates, but with the coordinates scaled by the current frequency. This noise value is then multiplied by the current amplitude and added to the total noise height.

       · **Persistence and Lacunarity**: Two key parameters, persistence and lacunarity, control the change in amplitude and frequency across octaves. Persistence is a value between 0 and 1 and determines how much the amplitude decreases for each subsequent octave. A high persistence will result in subsequent octaves having a strong influence on the final shape, making the terrain rougher. Lacunarity is a value greater than 1 and dictates how much the frequency increases for each subsequent octave. A lacunarity of 2, for example, would double the frequency with each octave.

       · **Final Noise Height**: By iterating over multiple octaves and accumulating the noise values, the final noise height represents a combination of large-scale features from the lower octaves and finer details from the higher octaves. This layered approach ensures that terrains have both broad, sweeping features and intricate, smaller details, mimicking the complexity of natural landscapes.

     * The actual height (y) of the vertex, which is influenced by both the noise height and the edge factor.

  2. **Height Normalization**: By iterating over the entire set of vertices, the algorithm identifies the minimum and maximum heights. This range is subsequently used to normalize the height values.

3. **Color Assignment**: Each vertex is assigned an additive color based on its normalized height. This allows for the creation of terrains with varying colors, such as snowy peaks or sandy valleys, depending on the gradient provided.

4. **Triangle Generation**: For rendering purposes, the terrain mesh is represented as a collection of triangles. The triangles array is populated by iterating over the dimensions x and z. For each pair, six triangle vertices are defined to create two triangles that together form a quad.

5. **UV Mapping**: UV coordinates are generated for each vertex, allowing for the application of textures to the mesh. The UVs are defined based on the position of the vertex within the x and z dimensions, ensuring that textures are uniformly mapped across the terrain.

6. **Finalization**: The generated colors array is assigned to the mesh, preparing it for rendering with the desired visual characteristics.

- **Scene Cleaning**: To ensure a pristine environment for the new generation, the scene undergoes a cleaning process to eliminate remnants from previous generations, like place marks and way points.

- **Placemark Positioning**: The algorithm strategically places the placemarks for the path creator. This positioning includes the initial placemark, where the path starts, and the destination placemark, where the path ends.

  * **Initial Placemark**: The algorithm attempts to position the initial placemark by repeatedly selecting a random location on the terrain. It uses a 'Scout' game object to check if the selected location is suitable for placement, ensuring the location is not in water and has land within a specified distance. If a suitable location is found, the 'PathMaker' is placed at that location, and the process ends. Otherwise, the algorithm continues searching for a suitable location.

  * **Destination Placemark**: The algorithm similarly tries to position the destination placemark. Using the 'Scout' game object again, it picks random locations and checks if they are suitable for placement. Once a suitable location is found, the 'finishPosition' variable is set to this location, indicating the end point of the path.

  * **Surrounding Area Check**: This helper function, 'CheckSurroundingArea', assesses the area surrounding a given position. It ensures that there's land within 10 units in all four cardinal directions (right, left, forward, and backward) and water within 20 units. This check ensures that the placemarks are placed in locations that are close to both land and water, providing a more interesting and diverse path.

  * **General Flow**: The main function, 'PlacePathMarks', orchestrates the placement of the placemarks. It first disables the navigation agent on the 'PathMaker' to prevent it from moving. It then attempts to place the initial and destination placemarks using the aforementioned methods. The function also ensures that the direct distance between the start and finish positions is within an acceptable range. If not, it decrements the 'directDistance' and tries again until a suitable path is established.

- **Navigation Mesh Validation**: A navigation mesh is constructed and assessed for validity. An invalid result triggers a new iteration with a different seed, whereas a valid result initiates the path creation and vegetation spawning.

– **Path Creation**: Operating through three parallel tasks, the path creator evaluates the timing for item spawning, detects collision with the end waypoint, and monitors proximity to existing waypoints to determine the transition to subsequent waypoints.

– **Vegetation Spawning**: This class is responsible for the procedural placement of various vegetation elements on the generated terrain. The vegetation is distributed based on Perlin noise values, which help to ensure a natural-looking distribution:

- **Perlin Noise with Octaves**: This function generates a noise value based on the Perlin noise algorithm, enhanced with multiple octaves.
- **SpawnObject Function**: This function manages the spawning of vegetation on the terrain.
  * **Vegetation Loop**: The function iterates through each type of vegetation defined in the ‘vegetations‘ list. If the ‘spawn‘ flag for a vegetation type is set to false, it skips to the next vegetation.
  * **Random Offsets**: To avoid repetitive patterns, random offsets (‘offsetX‘ and ‘offsetZ‘) are generated for each vegetation type. These offsets are used in the Perlin noise calculations. However, the developer might choose to have them customizable in the inpector, making this module even more deterministic.
  * **Positioning Loop**: For each vegetation type, the function attempts to spawn the predefined number of objects (‘numObjectsToSpawn‘). It picks random ‘x‘ and ‘z‘ positions within the bounds of the generated terrain.
  * **Noise Calculation**: A Perlin noise value is calculated for the chosen spawn position. This value determines the likelihood of vegetation spawning at that location.
  * **Raycasting and Instantiation**: If the random value is below the calculated noise value minus the ‘pocketThreshold‘ (ensuring vegetation spawns in clusters or "pockets"), a raycast is performed downwards from a predefined height (‘SpawnHeight‘). If the ray hits a terrain surface (and not water), the vegetation prefab is instantiated at the hit position with a random rotation around the y-axis, giving a more natural look.

5.2.2 NPC Package

5.2.2.1 Development Journey of the NPC Package

The genesis of the NPC Package can be traced back to the conclusion of the agent study of the parallel project, AViR, which focused on an investigating the impact of certain physical attributes of conversational agents on participants' comfort levels. This study served as an invaluable foundation, shedding light on aspects that could enhance the user's engagement with virtual conversational interfaces. During this period, state-of-the-art LLMs such as chatGPT were gaining prominence. This technological advancement prompted a pivotal shift in our development strategy, leading to the conceptualization of a new module: the NPC Package. Initially envisioned as a prototype, the objective was to create a conversational agent that could be tailored to individual developer preferences. The main characteristics of the prototype included:

- *Utilization of Physical Attributes:* The package incorporates the physical attributes identified by the AViR study as influential in enhancing the participant's comfort with the conversational agent.

- *Voice Recording:* It is equipped with the capability to record the user's voice, facilitating more authentic interaction.

- *Speech-to-Text Conversion:* The user's voice is converted into text, paving the way for nuanced dialog processing.

- *Integration with chatGPT API:* The text input is sent to the chatGPT API with a specific prompt, soliciting an appropriate dialog response.

- *Text-to-Speech Functionality:* The received text response is transformed into speech, allowing the agent to verbally communicate with the user.

- *Animated Response:* The agent is endowed with the ability to produce animations while responding, adding a layer of realism to the interaction.

- *Complete Agent*: This agent (Fig. 33) is designed to serve as the primary character or avatar for users navigating the virtual environment. The agent's foundation is derived from a Unity Store asset, which we further enhanced to offer more versatility and customization. Here's an overview of the agent's components and capabilities:

  - **Base Model**: The foundational character model is procured from the Unity Store [36]. This pre-designed model ensures a visually appealing and professional starting point, reducing the initial development time.

  - **Customization Capabilities**: Recognizing the need for individuality and flexibility in serious games, we have integrated features that allow developers to personalize the agent's appearance. This ensures that the agent can be tailored to better fit different game themes or user preferences.

  - **Animation Suite**: To provide a realistic and engaging user experience, we have equipped the agent with a comprehensive set of animations. These animations cover a broad spectrum of movements and actions, ensuring the agent can interact naturally with the environment and other entities.

  - **Layered Animations**: We further refined the animation system by dividing animations into layers. This layered approach permits developers to seamlessly integrate custom animations or blend multiple animations. For instance, the upper body can perform one action (like waving) while the lower body executes a different action (like walking). This ensures fluidity in movements and enhances the character's dynamism.

  - **Mobile Optimized**: Shaders and textures are optimized for mobile VR.

Fig. 33. Conversational Agent Prototype.

5.2.2.2 Test and Refinement

1. *Sanity Test 1* This phase of testing was conducted by two members of the AViR Team (Computer Engineering PhD student and a Master student in Cognitive Neuroscience and Neuropsychology) who had no prior interaction with the prototype. The primary objective was to evaluate the feasibility of engaging in an authentic conversation with the avatar. This informal study was instrumental in highlighting the real-world conversational capabilities of the system. Although minimal bugs were detected, two significant design challenges emerged:

   – **ChatGPT Reliability:** During this period, the reliability of chatGPT was suboptimal, manifesting in inconsistent server availability and occasional delays in response times. These issues intermittently hampered the fluidity of the conversation.

   – **ChatGPT Response Freedom:** A more nuanced issue pertained to chatGPT's occasional deviation from the provided prompt. This inconsistency ranged from mildly incongruent dialog responses to severe errors, such as generating improper JSON responses, leading to the breakdown of the communication thread. This highlighted the need for enhanced control over the dialog generation process to maintain the integrity of the interaction.

   Upon identifying the aforementioned challenges, we meticulously evaluated potential solutions to overcome the observed limitations in the NPC Package. Two primary alternatives were considered:

(a) **Utilizing an Alternative LLM Service or Deploying an Open Source LLM:** One option was to explore other LLM services or to implement our own server using an open-source LLM, such as LLama, which had just been introduced around 1 or 2 months prior to the study. However, this approach was quickly dismissed. At that time, the reliability of open-source LLMs seemed inadequate for our specific requirements, and the creation and maintenance of a dedicated server were deemed excessive and misaligned with our development objectives.

(b) **Restricting the Freedom of chatGPT:** The second solution involved constraining chatGPT's autonomy in dialog generation. By limiting chatGPT to evaluating the user's dialog response and subsequently selecting from a set of predefined responses, we could exert greater control over the interaction. While this approach introduced the risk of generating more formulaic, "cookie-cutter" responses, the trade-off was considered acceptable. Though the responses might be perceived as somewhat shallow, this constraint ensured the stability and integrity of the dialog without causing significant disruption to the overall user experience.

After careful consideration, the second solution was selected for implementation at this stage of development. The choice was guided by a pragmatic assessment of our goals and constraints, recognizing that the potential drawbacks were outweighed by the benefits of ensuring a more reliable and controlled conversational flow.

2. *Sanity Test 2* The second phase of testing was performed with the same participants from the initial sanity check, focusing on an informal assessment of the conversational agent's response quality. This phase was less exploratory in nature, as the objective was primarily to validate the improvements and ascertain whether the conversational agent's interaction met the desired standards. Despite the limited scope of this evaluation, it yielded valuable insights. Notably, no significant bugs were encountered, and the interaction with the conversational agent was deemed successful. However, a critical observation emerged: the customization capabilities of the package were perceived as too shallow. While the ability to tailor the AI's responses and animations was present, it fell short of offering a truly immersive and personalized experience. Consequently, a strategic decision was made to expand the scope of the conversational agent into a more comprehensive NPC package, constituting the final version. This decision was driven by the following considerations:

   – **Inclusion of a Fully Autonomous Conversational Agent:** Rather than merely providing a conversational agent, the package would include it as an example of a fully autonomous conversational agent, incorporating all the developed features, but the package would not be exclusively about it, but about the modules/components.

   – **Dual Functionality:** This enhanced scene would serve a dual purpose, both as an extra asset for developers who might require the conversational agent as an NPC in their game and as a demonstration of the package's capabilities, showcasing the utilization of every submodule.

   The final version of the components included in the package are as follows:

   – **Voice Capture:** This feature automatically captures the user's voice in snippets of 5-second durations, allowing for an authentic and interactive dialogue.

- **Speech-to-Text (STT):** The voice capture is processed to generate textual output. Developers have the flexibility to customize parameters such as language and region, catering to diverse user preferences.

- **Integration with chatGPT:** Leveraging the advancements in OpenAI's offerings, the package was able to fully utilize chatGPT once again. The release of "function calling" by OpenAI ensured near-perfect compliance with the required output structure, significantly enhancing reliability.

- **Text-to-Speech (TTS):** This component takes the chatGPT output and converts it into voiced speech. Customization options abound, allowing developers to tailor language, voice type, gender, style, and other auditory characteristics.

- **Agent Behavior:** The behavior of the conversational agent is defined through this module. Developers can customize various aspects, including the range of possible actions, memory size, knowledge base, and personality attributes.

3. *Studies on Research Question 1 (RQ1)* The evaluation of the NPC package under RQ1 revealed promising results. Participants exhibited a positive engagement with the package. A significant highlight was the section of the study that allowed participants to create a simulation involving two characters. Not only was this simulation swiftly constructed, but participants also demonstrated comprehension of the process, expressing enjoyment in both the development and the opportunity to shape the personalities of the virtual characters. This observation underscores the package's potential in fostering creativity and providing an intuitive and rewarding user experience.

4. *Studies on Research Questions 2 and 3 (RQ2 and RQ3)* The exploration of RQ2 and RQ3 focused on the quality of interactions and the identification of potential errors within the NPC package. Overall, the interactions were characterized by fluidity and minimal errors. The few glitches that were encountered were predominantly attributed to connection issues, such as weak Wi-Fi signals, leading to occasional prompts from the agent requesting users to repeat their inputs. Detailed insights and a comprehensive analysis of these interactions are elaborated in the studies section of this thesis.

5.2.2.3 Package Architecture/Workflow

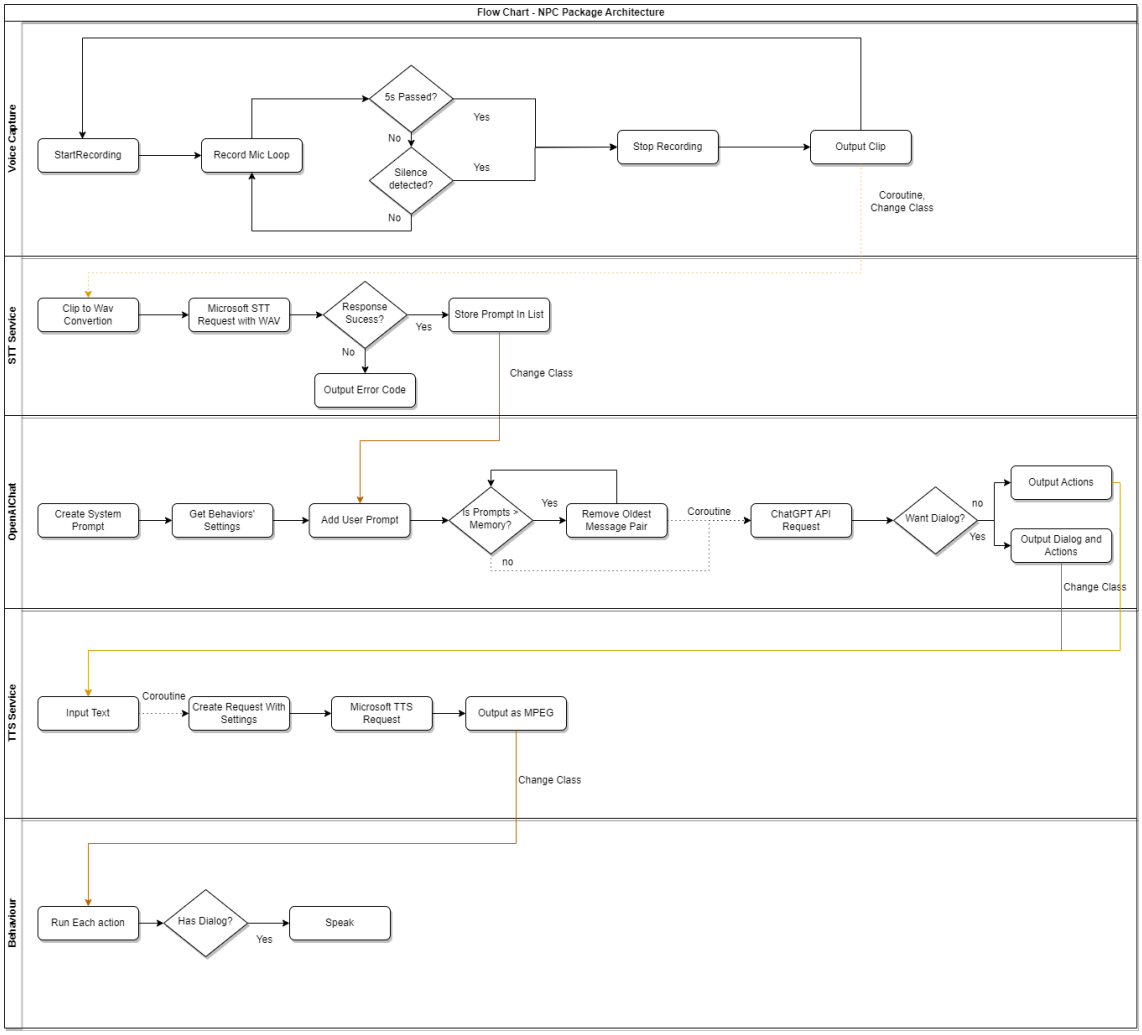


Fig. 34. Flowchart/Swimlane portraying the architecture of the NPC Package.

The architecture of the NPC package is an intricate composition of five interrelated modules, designed to function sequentially yet retaining flexibility for customization. The diagram illustrating the architecture is presented above (Fig 34). The key elements of the architecture and their functionalities are as follows:

1. **Voice Capture:** This module employs a dual-threaded mechanism, one monitoring a 5-second interval, and the other detecting silence by calculating the average sound peaks in the microphone. Two scenarios trigger the response computation: a 5-second lapse or detected silence. This script offers flexibility for developers to modify as required.

2. **Speech-to-Text (STT) Service:** Utilizing Microsoft's STT service, this module translates the recorded audio clip into a WAV file and requests a corresponding text string from the API. Developers can determine the subsequent actions based on the type of trigger (silence or 5-second interval).

3. **OpenAIChat:** This module orchestrates the internal structure needed for the API request, incorporating messages, such as the system definition that dictates the role of the chatGPT instance, and assimilating the knowledge and personality as defined in the behavior module. The conversation is built from inputs, often derived from the STT service. The request structure employs a more complex creation method through function calling, allowing precise control over the received response, ensuring strict adherence to the required JSON structure. When the `RunConversation()` function is triggered, the prompt is processed, and a request to the chatGPT API is made. The received response is stored in the historic messages, forming the memory of the NPC. The developer can then decide the subsequent actions or dialogues.

4. **Text-to-Speech (TTS) Service:** Activated when a dialogue is required, this module accepts input from the OpenAIChat module and requests a sound file based on the settings in the inspector and the data provided in the response. Developers have the flexibility to determine the handling of the sound file.

5. **Behavior:** This module houses settings for potential actions to be considered by chatGPT, along with the personality, knowledge, and memory of the agent. When the `Act()` function is triggered, the NPC executes actions from OpenAIChat and speaks the sound file if applicable. The code includes provisions for running animations, such as hand movements when speaking, but customization by the developer is expected. Specific actions, for example "OpenDoor", or any other action the developer might need, require developer-implemented logic to define the behavior when selected by the OpenAIChat module.

Crucially, the architecture is designed with modularity and adaptability in mind. Modules such as the OpenAIChat, STT, and TTS can be replaced with alternative services or altogether omitted, depending on the developer's requirements.

5.3 Co-Creation

5.3.1 Garden Package

5.3.1.1 Development Journey

The inception of the Garden Package was motivated by the aspiration to provide users with a virtual environment wherein they could realistically and creatively design their environment. This initiative led to the development of a prototype encompassing several distinct mechanics that collectively contributed to an immersive co-creation experience:

- **Spawnable Menu:** This feature (Fig. 35) is activated by a button press on the controller, revealing a menu that allows users to select the desired item to spawn within the garden environment.

- **Item Spawn:** Upon selection from the spawn menu, the chosen item materializes adjacent to the menu, guided by a directional arrow. This intuitive design ensures a seamless transition from selection to placement.

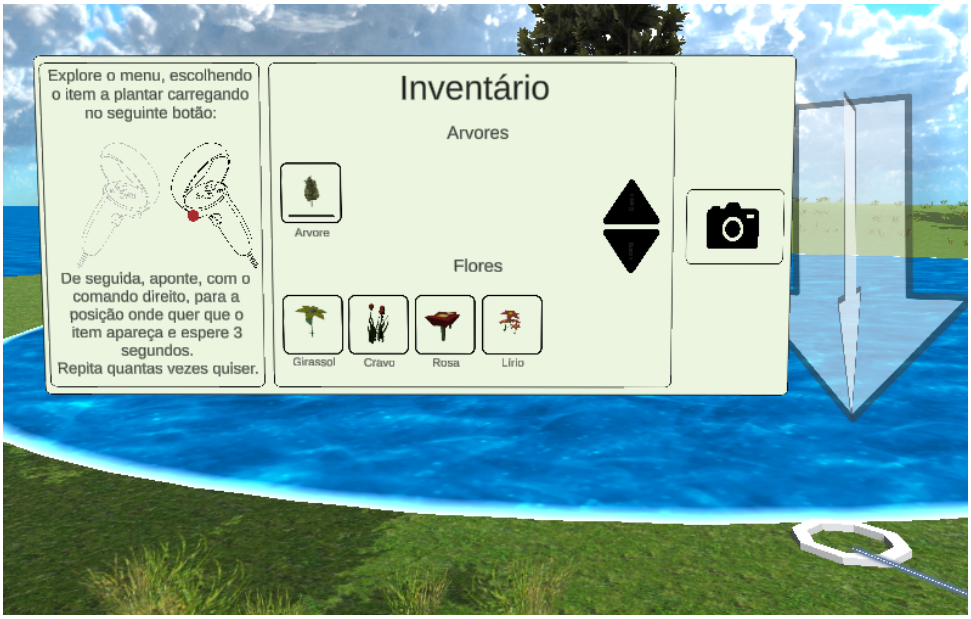


Fig. 35. Item Spawn Menu.

– **Item Manipulation:** The user can approach the spawned item and engage with it (Fig. 36) through the grip button of the hand controller. This interaction allows for picking up, relocating, and dropping the item within the garden space. Additional controls facilitate rotation, resizing, and deletion, offering comprehensive customization.

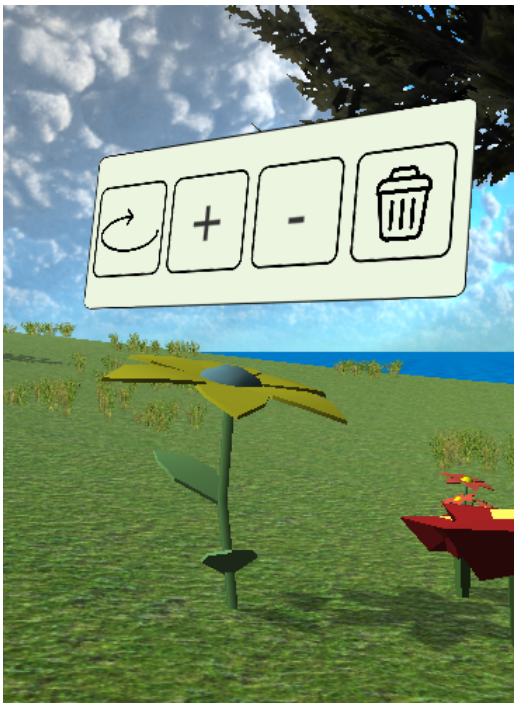


Fig. 36. Item example with Menu.

– **Record Memories:** A special chest, equipped with a menu interface (Fig. 37), enables users to encapsulate voice or text messages. Voice messages can be recorded onto a cassette and stored in the chest, while text messages can be inscribed onto paper and sealed in an envelope within the chest. This novel feature adds a layer of personalization and emotional resonance to the virtual garden.

Fig. 37. Memories Chest with Menu.

– **Photograph:** Enhancing the immersive experience, a photograph functionality is integrated into the package. By pressing a designated button on the controller, users can capture a photograph of their virtual garden, rendered in a nostalgic Polaroid style (Fig. 38). This feature encapsulates moments within the virtual space, allowing users to retain visual memories of their creative endeavors.

Fig. 38. Photograph Polaroid.

5.3.1.2 Test and Refinement

1. *Sanity Test 1:* Conducted with a member (Computer Engineering PhD student) of the AViR team who had not tested the scenario previously, the primary focus was on identifying bugs and usability issues. While certain bugs were promptly addressed, key usability challenges emerged:
   - The initiation process for garden creation was found to be non-intuitive, with the button-press mechanism for spawning the menu perceived as awkward.
   - The lack of a mechanism to remove the spawned menu, other than relocating it, was identified as a limitation.
   - Persistent UI elements (e.g., resize and delete controls) above items were found to detract from the aesthetics of the photographs.

   To rectify these issues, strategic modifications were implemented:
   - The menu was anchored to a fixed spot on the map (determined by the developer) to enhance accessibility, while retaining the option to spawn an extra menu at the user's discretion.
   - The persistent UI issue was resolved by concealing these elements during photograph capture, preserving the visual integrity of the image.
2. *Sanity Test 2:* This phase was conducted with the same AViR team member involved in the previous sanity check. While some minor bugs were identified and promptly addressed, the primary focus remained on usability issues. The refinements made after the first sanity check appeared to have mitigated most of the usability concerns, as evidenced by the absence of significant issues in this session.
3. *Preliminary Usability Testing:* Recognizing the interactive nature of the package and the importance of ensuring accessibility to a diverse user base, the decision was made to extend the testing phase. An additional extensive usability testing phase was conducted with women aged

40-50, who had virtually no previous gaming experience, aside from testing the exploration package. The objective was to identify potential usability barriers for users unaccustomed to gaming or VR. Several challenges were observed:

- **Button Confusion:** Frequent pressing of incorrect controller buttons led to unintended actions such as wrongful menu spawning or accidental photograph capturing, adding to the confusion.
- **Spawn Items:** The participants struggled with shaky hands when attempting to press the menu buttons, hindering item selection.
- **Item Manipulation:** This proved to be the most complex aspect for the participants. They found it difficult to comprehend the grip button's function for grabbing items and struggled with resizing, moving, and deletion controls.

Addressing these challenges necessitated significant refinements:

- The spawning of the menu was entirely removed, and the photo button was relocated from the controller to the menu to minimize confusion.
- The threshold for detecting the drag interaction was increased to accommodate shaky hands, ensuring accurate detection of button clicks.
- A complete overhaul of the item manipulation mechanic was undertaken. The ability to grab items was removed, replaced with a 3-second timer indicating the item's spawn location based on the reticle's pointing direction. Post-spawning, users could only resize or delete the item, simplifying the interaction process.

Beyond addressing usability challenges, this phase also spurred the integration of new functionalities to enhance both user and developer experiences:

- **Automated Menu Population:** To streamline the process of menu customization, an automated system was implemented. Developers can now easily configure the menu through the Unity inspector, specifying details such as item images, prefabs to spawn, item names, and categories. The menu automatically updates to reflect these settings, eliminating the need for manual UI adjustments within the Unity hierarchy.
- **Save File Functionality:** A new saving feature was introduced, allowing users to preserve their virtual garden's state. By clicking a dedicated save button, the garden's configuration, including item positions and other physical characteristics, is stored in a JSON file. This functionality not only enhances the user's continuity of experience but also provides developers with a flexible tool. It can be leveraged for more manual approaches to multiplayer gaming or other creative applications.
- **Rotate Functionality:** The ability to rotate virtual items within the garden adds a layer of customization and control, enabling users to align objects according to their creative vision. This simple yet impactful feature amplifies the user's agency in shaping the virtual space, facilitating a more nuanced and individualized expression.

4. *Study RQ1:* This study was designed to explore specific aspects related to Research Question 1 (RQ1). While a more comprehensive analysis can be found in the studies section of this thesis, a crucial insight emerged from the developer's perspective: the desire for a functionality that allows for the creation of a gallery (Fig. 39), comprising pictures taken by the user within the

virtual garden, was strongly expressed. This feature would enable users to revisit and reflect on their creative journey, capturing the evolution of their virtual garden. In response to this insight, the gallery functionality was implemented. When loading the garden, the system also retrieves a gallery containing previous photos taken by the user. This feature enhances the user's connection to their virtual space, allowing for a more personal and reflective experience.

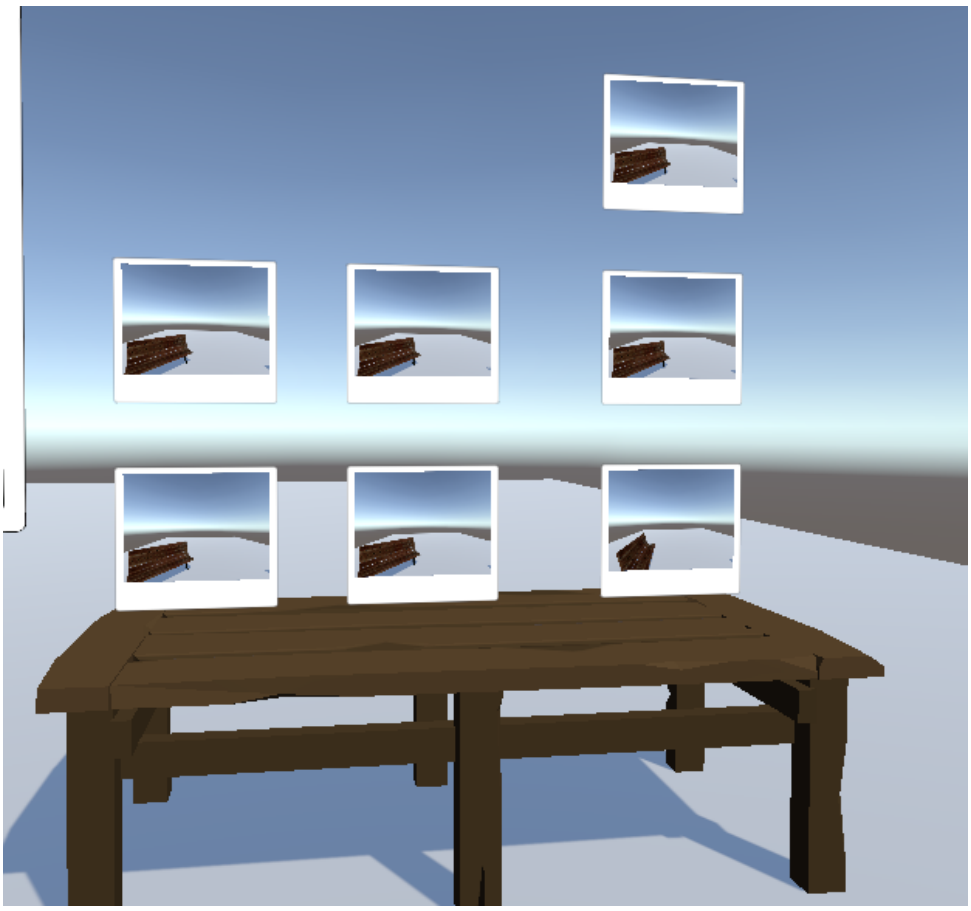


Fig. 39. New gallery. In this example, it is being implemented as a table with photos.

5. *Study RQ2:* Key findings from this study include:

   – **Enhanced Engagement:** Participants exhibited slightly increased engagement and enjoyment with the Garden Package compared to other packages. The creative freedom and interactive features seemed to resonate with users, fostering a more immersive and satisfying experience.

   – **Usability Concerns:** Despite the positive reception, the study also revealed some usability issues. These were relatively minor in nature and did not significantly detract from the overall experience. Both observational data and quantitative metrics affirmed the low impact of these issues.

5.3.1.3 Package Architecture/Workflow

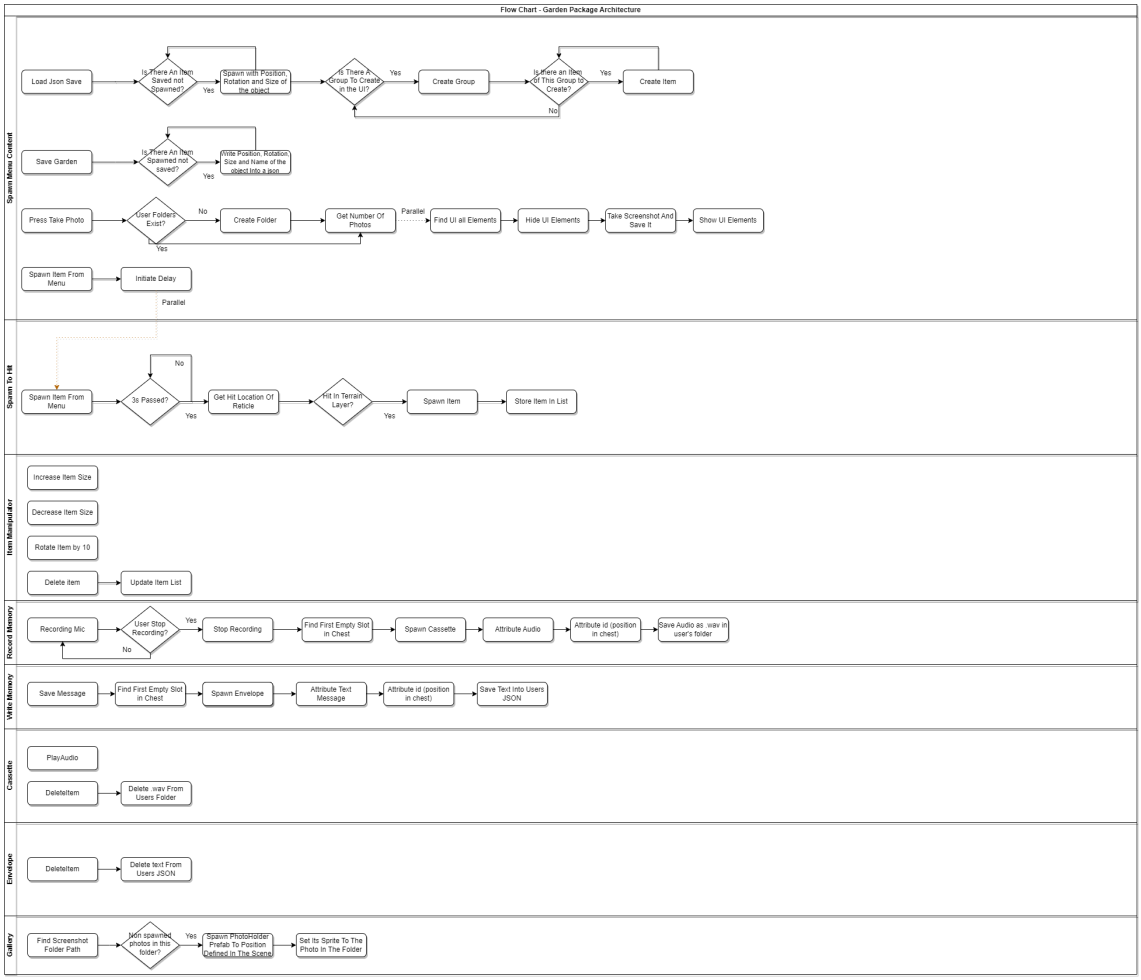


Fig. 40. Flowchart/Swimlane portraying the architecture of the Garden Package.

Despite numerous iterations, mainly to address interaction and usability challenges, the Garden Package's architecture is characterized by its relative simplicity compared to other packages. The architectural components are as follows:

– **Spawn Menu Content:** Responsible for various menu functionalities.

- *Initialization:* The garden's initialization commences by examining the user's JSON file, spawning each item based on the data saved within. The position, scale, and rotation of each item are matched accordingly. Subsequently, the menu is populated using customized settings within the Unity inspector, defining each group of items, icons, buttons, prefabs, and names.
- *Spawn:* Incorporates a button listener to invoke the spawn function of the `SpawnToHit` Class.
- *SaveGarden:* Facilitates the saving process, whereby the spawned items' attributes are stored in a JSON file, mirroring the initialization process in reverse.

- **Spawn To Hit:** Assigned to the right controller, this class governs the item's spawning at the reticle's hit position.
  - *SpawnItem:* Manages the item's spawning, updating the UI with the time remaining and validating the hit layer before proceeding with the spawn and save operations.
- **Item Manipulator:** This class handles each spawned item's menu, offering options to enlarge, shrink, rotate incrementally, or delete the item. A billboard effect is also integrated.
- **Record Memory:** This class initiates an audio recording function via the device's microphone. The recording process continues until the user activates a designated termination control. Upon cessation, the system algorithmically identifies the subsequent available slot within the designated storage chest. The recorded audio, along with its unique identifier (corresponding to its position within the chest), is then allocated to this slot and saved as .wav into the user's folder.
- **Write Memory:** When the user opts to save a written message, the algorithm systematically identifies the subsequent unoccupied slot within the designated storage chest. An envelope representation is instantiated, to which the composed text message and its unique identifier (derived from its positional placement within the chest) are attributed. Subsequently, the textual content is committed to the user's JSON data file for persistent storage.
- **Cassette:** This class plays the audio of the cassette when the user presses the button and deletes it when the delete button is pressed, removing also the .wav in the user's folder.
- **Envelope:** Item and the JSON entry with that item's text are created or deleted when the user presses the respective button.
- **Gallery:** Gallery is populated with the photos taken by the user by finding them in his folder and instantiating them in the gallery positions.

5.4 Conclusion

Through a meticulous exploration of PCG and Co-Creation methodologies, this work has successfully designed and implemented innovative packages that serve the multifaceted needs of customizable content creation. The Exploration and NPC packages, built on advanced procedural algorithms, have set new standards in terrain generation and conversational agent integration, achieving realism and interactivity. The strategic decision to expand the NPC package, embracing advanced AI models, has enriched the conversational experience, while the user-focused design of the Garden package has brought the concept of co-creation to life. These achievements reflect the core values of versatility, customization, and user empowerment that guided this project.

A defining feature of our development process has been the unwavering commitment to mobile VR compatibility, optimization, and cross-platform flexibility. We believe the resulting packages offer fluid performance and adapt to diverse platform capabilities, affirming our dedication to quality and adaptability.

6 Studies

6.1 Introduction

Given the multifaceted nature of this project, a rigorous validation effort was imperative to evaluate its effectiveness and address the three proposed RQs (section 1.2). This validation encompassed various methods, ranging from the previously discussed informal testing during the development phase to a more formal validation phase wherein three specific studies were conducted, as described in the next sections. These studies were structured to provide comprehensive insights into the usability and applicability of the developed packages. By focusing on these key areas, the work aims to contribute to the field by demonstrating how procedural generation and co-creation can enhance both the development process and the player's experience in-game.

6.2 Study on RQ1. Can the use of customizing packages for Unity that generate content through co-creation and procedural generation improve the efficiency of game development projects?

6.2.1 Objective and Rationale

The primary objective of this study is to ascertain the efficacy of the designed packages in aiding developers during the game development process. The validation phase involved running a focus group comprising experienced developers to meticulously analyze and evaluate the packages. Through structured discussions and analytical dissection of the packages, invaluable feedback was elicited, contributing to a nuanced understanding of the utility, usability, and possible enhancements for the toolset.

6.2.2 Methodology

The validation exercise employed a structured focus group method, enriched by a tailored questionnaire to extract both qualitative and quantitative insights. The focus group, spanning 90 minutes, convened four Unity-experienced researchers from the NeuroRehabLab. Post a brief introduction, participants provided their written informed consent and granted permission for session recording.

The evaluation unfolded through the following steps:

1. **Exploration Package:** With a goal to create an island replete with a trail of candy, participants first familiarized themselves with the package's import process. Following this, they engaged in collaborative terrain generation, making decisions on terrain characteristics, path specifications, vegetation, and other pertinent attributes. Their active participation culminated in a feedback session.
2. **Garden Package:** The task here centered around customizing a menu for bench spawning. Unlike the previous package, the Garden Package was integrated into an existing project using Unity's native functionalities. Participants were subsequently guided to create an object, fine-tuning its spawn settings and attributes. Feedback was also solicited upon completion.
3. **NPC Package:** An initial demonstration showcased a conversational agent, encapsulating the comprehensive features of the package. This was followed by an exercise wherein participants collaboratively developed a short simulation featuring two interacting characters, each with distinct personalities. Feedback was sought at the end of this segment.

The session culminated in an open forum discussion, enabling participants to reflect on and discuss the overall project. To conclude, participants were presented with a customized questionnaire (Annex, Figs. 54 and 55), designed to capture in-depth qualitative (through open ended questions, regarding any feedback the participant might have forgotten to give) and quantitative (through Likert scale ranging from 1 to 7 questions regarding ease of use, functionality, customizability and compability) data on each package.

6.2.3 Findings

The results are presented in two parts according to the type of data, namely, qualitative and quantitative. The qualitative data were collected through the focus group feedback and the subsequent custom questionnaire, the quantitative data, however, were gathered exclusively by the custom questionnaire.

6.2.3.1 Qualitative Feedback

The positive/constructive qualitative feedback is represented through the following Table 4,5 and 6, each corresponding to different aspects of the exploration, NPC, and garden packages:

Table 4. Comprehensive Feedback for Exploration Package.

Type	**Usability**	**Customization**	**New Feature Suggestions**	**General Impressions**
Positive		Participants liked being able to customize the slope tolerance accepted by the pathmaker		Mini paths make sense to be uninteresting
Positive		Customization was clear except for some Perlin noise parameters		Discussion on how the waypoints of the path are created
Constructive	Not enough tooltips in the variables	Path gradient should be algorithmical instead of manual	It would be good to have a deformation of the terrain mesh in the path	
Constructive	Should Document the performance significance of each setting in the documentation	Should change the name of time (terrain color customization variable)	When path drops items, more options should be available	
Constructive	Clear unused in the hierarchy	Should change the name of generations to maps or similar		

Table 5. Comprehensive Feedback for NPC Package

Type	Usability	Customization	New Feature Suggestions	General Impressions
Positive	It's cool to be modular			Interesting to create a simulation between 2 AI's in 10 minutes
Positive				It's cool to provide Open source as it gives access to the package and source code
Constructive	Agent behavior hard to understand	Provide different but equivalent animations to avoid repetition		
Constructive	Inspector for knowledge and personality should have a bigger input box			

Table 6. Comprehensive Feedback for Garden Package

Type	Usability	Customization	New Feature Suggestions	General Impressions
Positive	Seems user-friendly	Easy to customize photo background		
Constructive			Should have an option for a gallery after taking pictures	

6.2.3.2 Quantitative Analysis

Following the collection of qualitative data, the study proceeded with quantitative analysis. It should be noted that the data gathered from only four participants does not permit definitive conclusions in isolation but serves as valuable supplementary information in conjunction with the qualitative findings, seen in Table 7 and portrayed in Fig. 41.

Table 7. Quantitative Data Across Packages

Package	ID	Ease of Use	Functionality	Customizability	Compatibility
Exploration	1	6	5	6	5
	2	6	5	6	6
	3	4	5	5	5
	4	3	5	6	4
	Mean ± SD	4.75 ± 1.5	5 ± 0	5.75 ± 0.5	5 ± 0.82
Garden	1	6	7	6	7
	2	7	6	5	4
	3	5	5	4	3
	4	6	5	4	4
	Mean ± SD	6 ± 0.82	5.75 ± 0.96	4.75 ± 0.96	4.5 ± 1.73
NPC	1	7	6	6	6
	2	6	7	5	6
	3	5	6	4	5
	4	3	5	4	4
	Mean ± SD	5.25 ± 1.7	6 ± 0.82	4.75 ± 0.96	5.25 ± 0.96

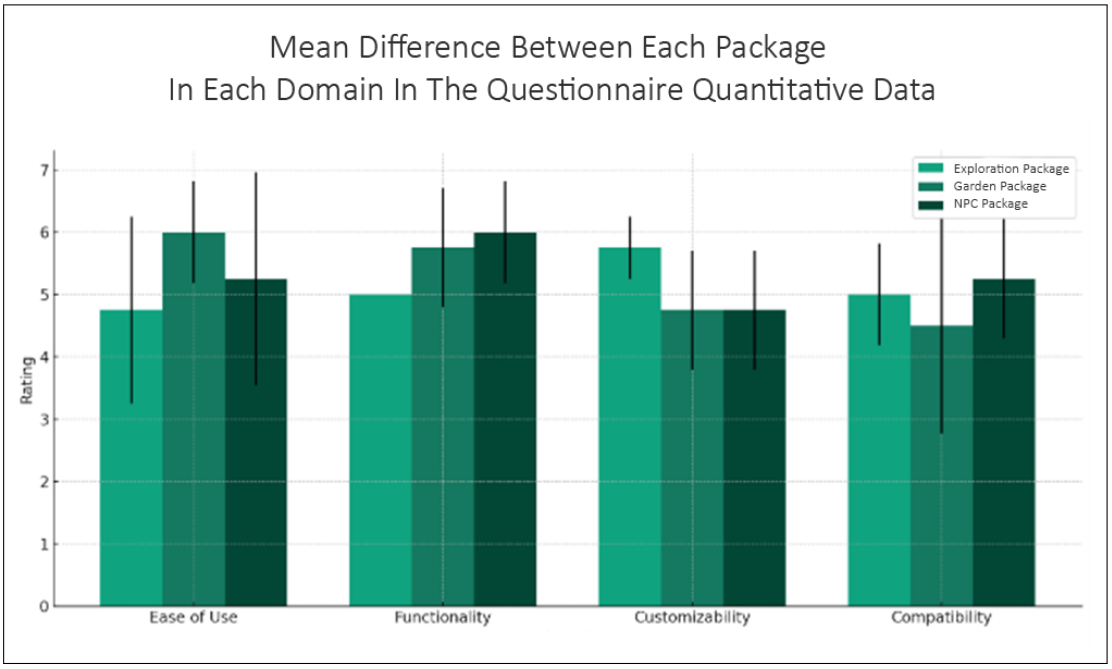


Fig. 41. Quantitative Feedback Comparison Between Packages

6.2.4 Discussion

The findings of this study lead to a generally positive response to the RQ1. While certain actions were identified to enhance the overall quality of the packages, the development and testing phases produced an optimistic perception of the project among the developers in the focus group, corroborated by the observations and quantitative data. The successful implementation of these packages, whether in part or in full, in the aforementioned projects, also signifies the positive impact and potential of these tools. We also identified what actions to take based on the constructive feedback (Table 8).

Table 8. Actions to take for all Packages.

Package	**Usability**	**Customization**	**New Feature Suggestions**	**General Impressions**
Exploration	-Implement more tooltips -Document section about performance -Clean project	-Implement automatic gradient in path -Refactor variables -Refactor folders	-Implement multiple item drop	
Garden			-Implement Gallery	
NPC	-Document well -Text box instead of input box	-Implement more animations that are equivalent but random		

6.3 Study on RQ2. Can Unity packages for custom content generation engage users in gameplay?

6.3.1 Objective and Rationale

The scope of this investigation goes beyond the analysis of the ease and efficiency with which developers can utilize the designated packages. We also address the engagement and quality of gameplay experienced by end-users. This engagement must be commensurate with that found in projects of similar caliber that have opted not to utilize these packages. In other words, this research aims to determine whether the selected packages contribute to, or potentially hinder, the creation of captivating and satisfying gameplay, thereby aligning with the overarching goals and standards of the industry.

6.3.2 Methodology

The sample of this study consisted of a convenience sample of 13 participants (4 females, 9 males), ranging in age from 18 to 48 years. All participants were students or staff members from a university setting. Before data collection all participants were given information about the purpose of the study and provided their written informed consent. Regarding the specific tasks to be done by the participants, representative examples were crafted for each package to illustrate their function. For the exploration and garden packages, similar implementations to the ones found in AViR were used, modeling the psychoeducation exploration and garden memorial creation modules. Conversely, the NPC package was implemented as a conversational agent, but with greater creative freedom in its responses and animations. The study design included three sessions for each participant, lasting approximately 15-20 minutes. Each session encompassed a task-based questionnaire, tailored to the specific module and constructed to permit participants a degree of freedom in their engagement. The sequence of sessions was randomized. The tasks in each session were the following:

- Exploration Package: Participants were required to read instructions, undertake a loading tutorial, utilize a map, capture all primary and secondary items, locate the exit, and seek the reward.

- Garden Package: Participants were directed to the garden center, tasked with creating a garden containing a minimum of five items, photographing it, returning to the house, recording and writing messages, exiting, and finally, looking for messages and searching for the photo.

- NPC Package: The session began with participants reading the narrative designated for that session, followed by engaging in conversation with the agent about the narrative for four minutes.

For each package, we evaluated specific interaction domains through the following assessment instruments: Usability, evaluated through a sickness/dizziness questionnaire (S/D) and the SUS [37]; Presence, through the Slater-Usoh-Steed questionnaire [38]; Motivation, assessed via the Gaming Experience Questionnaire (GEQ) [39] and Intrinsic Motivation Inventory (IMI) [40]; and Ecological Validity (Annex, Figs. 57-62, respectively). In addition to the questionnaires, participants were instructed to use the think-aloud method during the sessions, articulating their thoughts, feelings, and reasoning as they navigated through the tasks. This technique facilitated an understanding of the participant's reasoning processes, providing further insights into their interaction with the packages. Concurrently, observations were recorded, noting particular behaviors, errors, and reactions, enriching the qualitative dimension of the study.

6.3.3 Findings

In the following sections, we present the results separately for each package. It should be noted that the methodological approach remained consistent across packages, employing the same questionnaires and data collection techniques.

6.3.3.1 Exploration Package

We start by presenting the results of the quantitative instruments. Beginning with the usability questionnaires, for the Sickness and Dizziness questionnaire, rated on a 1-7 Likert scale, we obtained a mean score of 2.72 ± 0.73 for Sickness and a mean score of 1.72 ± 0.81 for Dizziness. For the SUS, we obtained a mean score of 74.0 ± 9.0, which corresponds to a 'B - Good' classification. Regarding presence, the Slater-Usoh-Steed scale revealed a mean score of $40.5 \pm 12.3\%$ (Table 9).

Table 9. Summary of Usability and Presence Metrics for the Exploration Package.

ID	Usability			Presence
	S/D		SUS	Slater-Usoh-Steed
	Sickness	Dizziness	Score	%
1	3.33	2.00	61.7	40.00
2	2.67	1.00	70.0	53.33
3	2.33	1.00	81.7	53.33
4	1.67	1.00	73.3	46.67
5	3.67	2.33	65.0	26.67
6	2.67	3.00	66.7	16.67
7	1.67	1.00	86.7	40.00
8	3.33	2.67	86.7	36.67
9	3.00	2.00	83.3	26.67
10	2.00	1.00	61.7	56.67
11	2.33	1.00	71.7	33.33
12	4.00	3.00	73.3	53.33
13	2.67	1.33	80.0	43.33
Mean	2.72 ± 0.73	1.72 ± 0.81	74.0 ± 9.0	40.51 ± 12.31

Concerning the motivational aspects of the package, we start by showing the results for the GEQ. This in-game module evaluates a multifaceted range of domains, specifically, Immersion, Flow, Competence, Negative Affects, Positive Affect, Tension, and Challenge, rated in 0-4. We highlight the results for the Immersion at 2.8±0.5 and Flow 2.2±0.9 domains, as they better relate to RQ2. In alignment with these findings, for the results in the IMI we highlight a score of 4.77±0.71 in the Interest/Enjoyment domain (Table 10).

Table 10. Summary of GEQ In-Game and IMI Questionnaire data, Exploration Package.

	GEQ							IMI
ID	Immersion	Flow	Competence	Neg_Affect	Pos_Affect	Tension	Challenge	I/E
1	2.7	2.0	2.0	1.3	1.7	0.7	2.3	3.57 ± 0.73
2	2.7	2.0	1.7	1.0	1.0	0.3	2.3	5.29 ± 1.04
3	2.7	1.3	1.7	0.0	3.3	0.0	3.3	6.14 ± 1.18
4	1.7	0.7	3.3	0.0	3.7	0.3	1.7	6.36 ± 1.18
5	3.0	3.0	2.3	0.7	3.0	0.0	3.0	4.00 ± 0.87
6	2.3	0.7	3.0	1.0	2.7	0.0	2.0	5.93 ± 1.13
7	3.3	3.0	3.0	0.0	2.7	0.0	3.0	2.29 ± 1.04
8	3.0	2.7	2.7	0.3	2.7	0.0	3.0	5.07 ± 1.13
9	3.3	3.7	3.3	0.3	3.3	0.0	3.7	5.07 ± 1.13
10	2.7	2.0	2.3	1.0	2.7	0.3	2.3	3.14 ± 1.18
11	2.7	2.3	2.0	1.7	1.0	1.0	2.0	6.36 ± 1.70
12	3.3	2.0	2.3	1.0	2.7	0.3	2.3	4.21 ± 1.04
13	3.0	2.7	2.3	1.0	3.0	0.3	3.0	4.64 ± 1.18
Mean	2.8 ± 0.5	2.2 ± 0.9	2.5 ± 0.6	0.7 ± 0.5	2.6 ± 0.8	0.3 ± 0.3	2.6 ± 0.6	4.77 ± 0.71

Lastly for the quantitative data, the assessment of ecological validity rated on a 1-7 Likert showed a mean score of 4.5±0.5, indicating an overall positive user response.

Now, we shift our focus to the qualitative aspects of our findings, specifically targeting errors and "soft errors." These elements are essential in assessing the user experience with the exploration

package. These insights were drawn from careful observation and the application of the "think aloud" method. The data were subsequently categorized to highlight particular areas of interest or concern (Table 11). The consideration of both errors and "soft errors" allows us to comprehend more subtle issues that might influence usability or enjoyment.

Table 11. Summary of Errors and Soft Errors for the Exploration Package.

	Soft Errors				Errors	
ID	User did not read messages until the end	User did not follow the path	User overshoots reward pick-up hitbox	Recovered?	User did not find the door to leave	User did not use the map
1	x		once	x		
2	x		twice	x		
3						
4		x				
5						
6			twice	x		
7			twice	x		
8			once	x		
9			once	x		x
10						
11			once	x		
12			once	x		
13						

6.3.3.2 Garden Package

Our analysis continues with the quantitative data analysis of the Garden Package.

Beginning again with the usability questionnaires (Table 12), we consider the Sickness and Dizziness metrics. These metrics also yielded a relatively low mean score at 2.50±0.81 and 1.62±0.76 respectively. The System Usability Scale (SUS) further substantiates these findings, with the Garden Package also achieving a favorable 'B - Good' classification at 73.6±8.4. Finally the Slater-Usoh-Steed scale revealed a decent mean score closer to 50%:

Table 12. Summary of Usability and Presence Metrics for the Garden Package.

ID	Usability			Presence
	S/D		SUS	Slater-Usoh-Steed
	Sickness	Dizziness	Score	%
1	3.33	2.00	56.7	43.33
2	2.67	1.00	63.3	60.00
3	2.33	1.00	73.3	60.00
4	1.67	1.00	61.7	33.33
5	3.67	2.33	76.7	46.67
6	2.67	3.00	78.3	70.00
7	1.67	1.00	78.3	46.67
8	3.33	2.67	80.0	46.67
9	3.00	2.00	75.0	23.33
10	2.00	1.00	70.0	60.00
11	2.33	1.33	80.0	76.67
12	4.00	3.00	85.0	20.00
13	2.67	1.33	78.3	30.00
Mean	2.50 ± 0.81	1.62 ± 0.76	73.6 ± 8.4	47.44 ± 17.54

Now unto the motivational aspects of the package, we again consider the gaming experience questionnaire with Immersion at 2.7±0.6 Flow at 2.2±1.1 and the Intrinsic Motivation Inventory again highlights a promising score in the Interest/Enjoyment Domain of 5.48±0.59 (Table 13).

Table 13. Summary of GEQ In-Game and IMI Questionnaire data, Garden Package.

ID	GEQ							IMI
ID	Immersion	Flow	Competence	Neg_Affect	Pos_Affect	Tension	Challenge	I/E
1	2.3	2.7	2.7	1.3	2.7	1.3	2.7	3.79 ± 0.57
2	2.0	3.0	2.3	1.0	3.3	0.0	2.7	5.71 ± 0.57
3	3.0	2.0	1.3	0.0	3.3	0.0	2.7	5.71 ± 1.60
4	1.7	0.3	2.0	2.0	3.0	0.7	1.0	5.29 ± 0.57
5	2.3	0.7	4.0	1.0	2.7	0.0	3.0	4.43 ± 2.07
6	3.0	3.0	3.3	1.0	2.7	0.0	2.7	6.79 ± 0.57
7	3.3	1.7	1.7	0.7	2.7	0.0	2.3	5.93 ± 1.67
8	3.0	2.7	3.3	0.0	4.0	0.0	3.3	5.71 ± 2.36
9	3.3	3.3	3.3	1.7	3.3	0.3	3.3	5.71 ± 0.57
10	3.0	2.3	2.3	1.3	3.0	1.0	3.0	5.29 ± 1.04
11	3.7	4.0	3.7	0.0	4.0	0.0	3.7	5.93 ± 1.88
12	2.7	1.7	2.7	1.0	2.3	0.3	3.0	5.29 ± 1.04
13	2.0	0.7	2.0	2.0	2.7	1.0	1.7	5.71 ± 1.35
Mean	2.7 ± 0.6	2.2 ± 1.1	2.7 ± 0.8	1.0 ± 0.7	3.1 ± 0.5	0.4 ± 0.5	2.7 ± 0.7	5.48 ± 0.59

Lastly, the assessment of ecological validity also indicates a good a score at 4.9±0.5.

Now, visualizing unto the more qualitative data (Table 14), observations categorized into errors:

Table 14. Summary of Errors for the Garden Package.

ID	User Did not read/understand instructions	User failed to point to a spawnable area
1		
2	x	x
3	x	
4		
5	x	
6	x	
7	x	
8		
9	x	x
10		
11		
12	x	
13		

6.3.3.3 NPC Package

Our analysis continues with the quantitative data analysis of the NPC Package.

Beginning with the usability questionnaires (Table 15), we consider the Sickness and Dizziness metrics. These metrics Also yielded a relatively low mean score of 2.51±0.81 and 1.54±0.67. The SUS further substantiates these findings with the NPC package having an even higher grade 'A - Excellent' at almost 83. Finally, the Slater-Usoh-Steed scale revealed a similar mean score around 46 percent:

Table 15. Summary of Usability and Presence Metrics for the NPC Package.

ID	Usability			Presence
	S/D		SUS	Slater-Usoh-Steed
	Sickness	Dizziness	Score	%
1	2.00	2.00	70.0	50.00
2	1.00	1.00	90.0	86.67
3	2.00	1.00	73.3	56.67
4	2.33	1.00	75.0	46.67
5	3.33	1.67	73.3	36.67
6	4.00	3.00	85.0	63.33
7	3.00	2.00	81.7	36.67
8	3.00	2.00	81.7	40.00
9	3.33	1.00	96.7	10.00
10	2.67	1.00	86.7	46.67
11	2.33	2.33	83.3	43.33
12	1.67	1.00	78.3	53.33
13	2.00	1.00	100.0	30.00
Mean	2.51 ± 0.81	1.54 ± 0.67	82.7 ± 9.1	46.15 ± 18.05

Turning again our attention to the motivational aspects of the package, we will first consider the gaming experience questionnaire at 3.1±0.5 for Immersion and 2.7±1.0 for Flow. The Intrin-

sic Motivation Inventory also highlights a promising score in the Interest/Enjoyment Domain of 5.81±0.85 (Table 16).

Table 16. Summary of GEQ In-Game and IMI Questionnaire data, NPC Package.

ID	GEQ							IMI
ID	Immersion	Flow	Competence	Neg_Affect	Pos_Affect	Tension	Challenge	I/E
1	2.7	3.0	2.7	1.0	3.3	1.7	3.0	5.50 ± 0.00
2	3.0	3.3	2.0	1.0	2.0	0.0	2.7	6.36 ± 1.18
3	2.3	1.0	1.7	0.0	3.7	0.0	3.3	5.50 ± 1.22
4	3.0	2.7	2.7	0.3	3.0	0.3	3.0	5.50 ± 1.22
5	2.7	2.0	2.7	1.0	3.0	0.0	2.7	5.29 ± 2.02
6	3.0	0.7	3.3	1.0	2.7	0.3	2.0	6.57 ± 0.73
7	3.3	3.0	3.0	0.3	2.7	0.0	2.7	5.29 ± 1.04
8	3.3	3.3	3.3	1.0	3.3	0.3	3.3	6.57 ± 0.73
9	3.3	3.0	2.3	0.0	3.3	0.0	3.0	6.36 ± 1.18
10	2.7	2.3	3.0	0.3	3.0	1.3	1.7	5.71 ± 2.20
11	4.0	4.0	3.7	0.0	4.0	0.0	2.7	6.14 ± 0.80
12	3.0	2.7	3.0	0.3	3.0	0.3	1.7	6.36 ± 0.80
13	4.0	4.0	3.0	0.3	4.0	0.3	2.7	4.43 ± 1.67
Mean	3.1 ± 0.5	2.7 ± 1.0	2.8 ± 0.6	0.5 ± 0.4	3.2 ± 0.6	0.4 ± 0.5	2.6 ± 0.6	5.81 ± 0.85

Lastly, the assessment of ecological validity with a high mean, at 5.2±0.5.

Now finished with the fully quantitative data, we move into the errors:

Table 17. Errors.

ID	NPC misunderstood the participant	NPC responded poorly (or erroneously)	Other NPC error
1			
2			
3	once		
4			lost connection
5	once		
6			
7			
8			
9			
10			
11			
12			
13			

6.3.4 Discussion

6.3.4.1 Usability and Presence

– **Sickness and dizziness questionnaire:** The assessment of dizziness and sickness across the different packages yielded scores that objectively reflect a minimal occurrence of symptoms such as nausea or related discomforts. This outcome was further validated by comparison with

analogous projects, such as "A Virtual Reality Bus Ride as an Ecologically Valid Assessment of Balance: A Feasibility Study" [41] where participants engaged with a CAVE-like virtual reality system instead of a HMD.

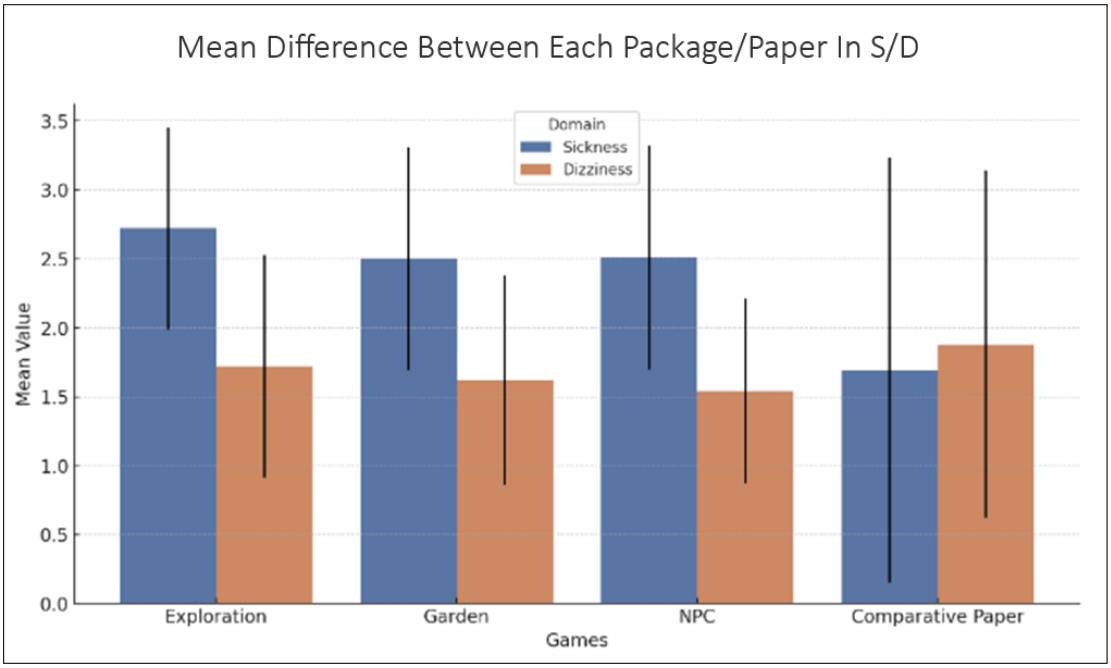


Fig. 42. Sickness and dizziness questionnaire mean and standard deviation for each package in comparison with "A Virtual Reality Bus Ride as an Ecologically Valid Assessment of Balance: A Feasibility Study" [41].

The slightly elevated scores (Fig. 42) for sickness across our packages can be reasonably ascribed to the use of an HMD, which may be more susceptible to issues such as discomfort and eye strain compared to a CAVE system. Dizziness scores were largely commensurate with the comparative paper, with the exploration package exhibiting the highest propensity for dizziness—likely due to the increased head movement and travel required. These low scores may be atrributed to our implementation of good practices for comfort in VR, such as maintaining a high FPS count, teleporting instead of continuous movement, etc. Conversely, the NPC package, which required minimal movement, yielded the lowest score, similarly with the Garden package. Importantly, both domains in our study registered scores well below the mid-point threshold of the scale at 3.5. It is worth noting that the comparative paper appears to have utilized a more simplistic sickness/dizziness questionnaire, comprising only one question for each domain. In contrast, our questionnaire included six questions, split evenly across the two domains. This difference in methodology introduces an element of subjectivity and suggests that direct comparisons may not be wholly equivalent. Consequently, while these comparisons provide useful context, the divergence in assessment tools should temper any overreliance on direct numerical equivalences. The observed trends nonetheless offer valuable insights into the relative performance of the packages and contribute to a broader understanding of user comfort and well-being in virtual reality environments.

– **System Usability Scale:** The evaluation of system usability yielded ratings that predominantly fall within the categories of Good and Excellent (Fig. 43). A close examination reveals the Exploration and Garden packages sharing analogous scores—a result that aligns with expectations given their similar mechanical foundations (such as walking and button-clicking functionalities).

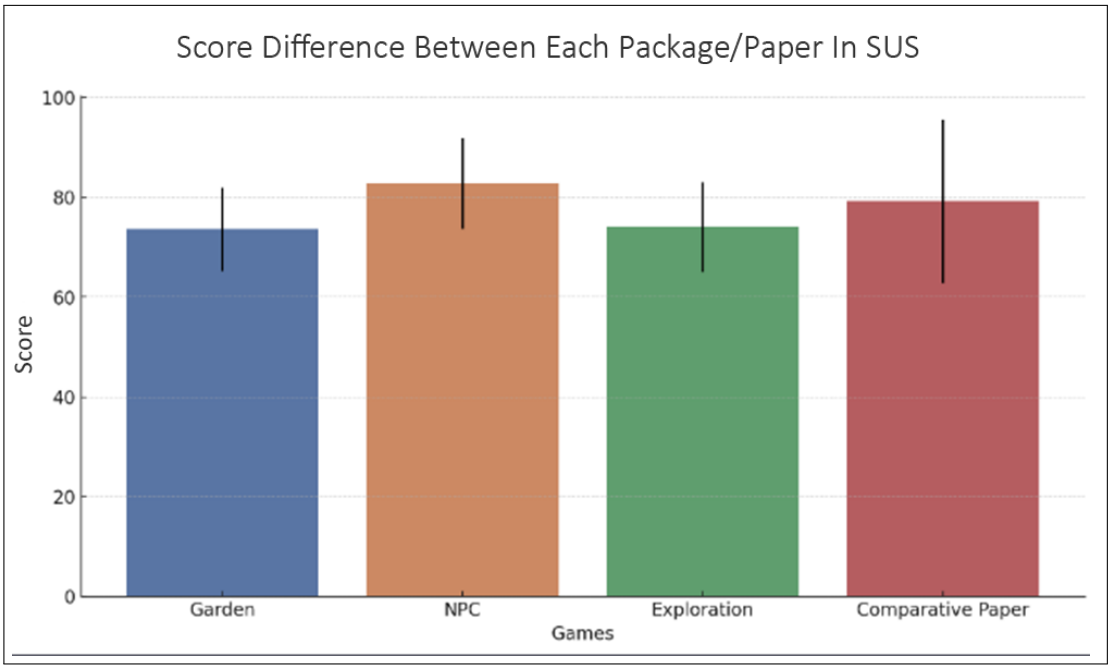


Fig. 43. System Usability Scale questionnaire mean and standard deviation comparison with "Efficacy of Augmented Reality-Based Virtual Hiking in Cardiorespiratory Endurance: A Pilot Study" [42].

A minor decrement in the usability score for the Garden package could be discerned, which might be attributable to a higher incidence of observed errors in the Garden scenarios. These errors, although relatively minor, may have engendered a degree of frustration among users. However, it must be noted that this slight difference is within the margin of error or standard deviation, rendering it inconclusive.

The NPC package once again distinguished itself with the highest usability rating, a result that can be readily ascribed to the intuitive nature of its mechanics (consisting mainly of speaking, listening, and observing the NPC). The absence of controller requirements further contributed to this favorable assessment.

The outcomes of this investigation seem to align with existent literature, notably with the study titled "Efficacy of Augmented Reality-Based Virtual Hiking in Cardiorespiratory Endurance: A Pilot Study" [42]. This study employs VR within a cave-like infrastructure.

– **Slater-Usoh-Steed:** Within the context of the Slater-Usoh-Steed evaluation (Fig. 44), the exploration package yielded a mean score of 40.5%, constituting the lowest among the three packages. Nevertheless, this score aligns with anticipated outcomes and parallels those of similar VR experiences, as exemplified by "A Virtual Reality Bus Ride as an Ecologically Valid Assessment of Balance: A Feasibility Study" [41].

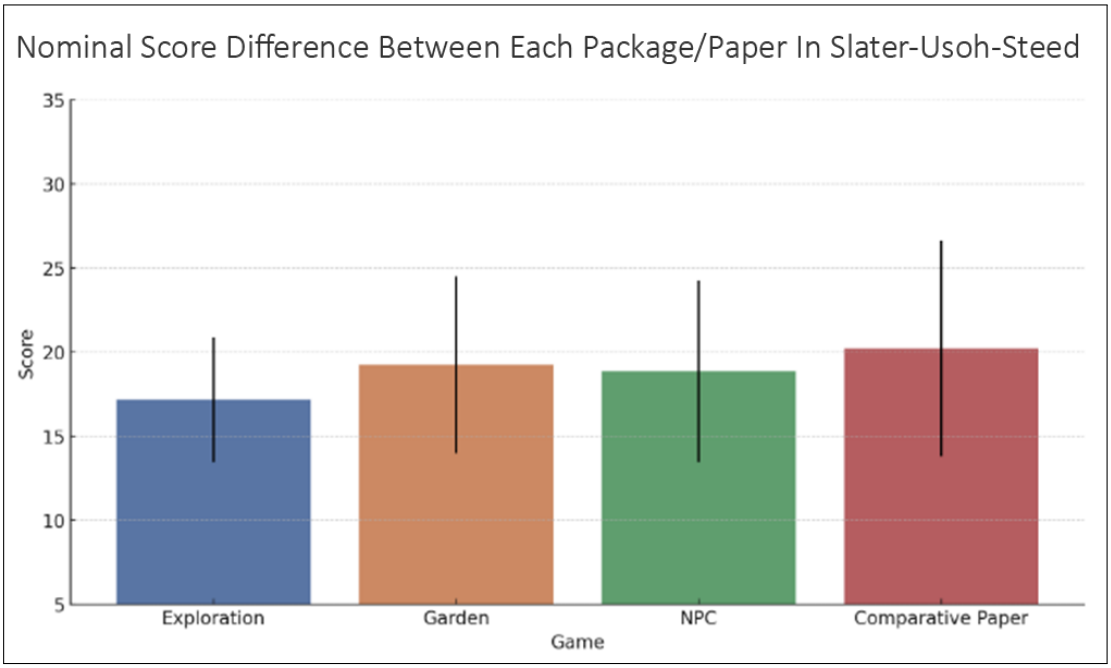


Fig. 44. Slater-Usoh-Steed questionnaire mean and standard deviation comparison "A Virtual Reality Bus Ride as an Ecologically Valid Assessment of Balance: A Feasibility Study" [41].

This minor variance might be ascribed to factors such as movement dynamics and visual quality. Preliminary sanity tests revealed pronounced nausea in participants unaccustomed to virtual reality, prompting the integration of a teleportation system. Though effective in minimizing discomfort, this modification may have marginally undermined presence. Additionally, the need to accommodate the constraints of less powerful hardware, such as the Meta Quest 2, necessitated a reduction in graphical fidelity. This was particularly evident in an open environment replete with vegetation, where the sacrifices in visual quality were more pronounced.

In contrast, the Garden package achieved the most favorable score. This can be attributed to the diminished emphasis on the aforementioned hindrances within its mechanics. While the exploration package accentuates movement and nature observation—leading to a heightened sense of artificiality—the Garden package's restricted focus on item placement and message creation might mitigate this effect.

The NPC package exhibited a higher score relative to the Exploration Package in the evaluation of virtual presence. The controlled environment in which NPC conversations took place, characterized by an enclosed space, allowed for an enhancement in graphical quality and detail. Coupled with the participant's lack of movement (as they were seated), this contributed to the observed outcome. The nuanced quality of the conversations, substantiated by the data from this study and the insights from research on RQ3, seemed to cultivate a deeper sense of presence with the virtual agent. This finding delineates the intricate interplay between environmental constraints, user interaction, and the presence attributes within virtual reality, shedding light on valuable considerations for future VR design, especially in the context of mobile VR platforms. Nevertheless, the score remained slightly below that of the comparative paper and the Garden Package. Since the questionnaire was focused on evaluating presence, this discrepancy may be attributable to the use of a non-ultra-realistic NPC model, as well as imperfections in

animations and speech synthesis. Despite these limitations, the results remain commensurate with relevant works in the field. Moreover, the potential for enhancing presence scores through tailored choices in animations, models, or synthesizers is inherent in the design philosophy of this project. Such customization aligns with the project's objectives and affirms the flexibility and adaptability of the package to suit varying developmental needs and preferences.

6.3.4.2 Engagement

– **Gaming Experience Questionnaire:** The GEQ (Fig. 45), comprised of six distinct domains, elucidates nuanced insights into the user experience across the different packages. The NPC package consistently emerged as the superior performer across most domains. A comparative analysis with other game related studies, such as "The Use of Game Modes to Promote Engagement and Social Involvement in Multi-User Serious Games: A Within-Person Randomized Trial with Stroke Survivors" [43] reveals parallelism in the positive domains. However, it is worth noting that negative domains, such as tension and negative affect, were marginally elevated in our packages. This can be related to the greater propensity for errors and resultant frustrations inherent to VR applications. The negative affect in the garden package, which exhibited the highest error frequency, exemplifies this trend. Nevertheless, these negative indicators remain modest, with tension ranging from 0.3 to 0.4 and negative affect from 0.5 to 1.0, both well below the midpoint of the scale at 2.0.

Fig. 45. Gaming Experience Questionnaire - In-Game questionnaire mean and standard deviation comparison between domains and games with "The Use of Game Modes to Promote Engagement and Social Involvement in Multi-User Serious Games: A Within-Person Randomized Trial with Stroke Survivors" [43].

A particular point of interest lies in the flow score, where the exploration package, at 2.2, is half a point lower than the other packages, which themselves are 0.4 points beneath the scores in the comparative studies. To unpack this further, in the GEQ - In-Game Module, Flow is calculated as the mean of responses to the statements "I forgot everything around me" and "I felt completely absorbed." These metrics bear resemblance to the presence indicators in the Slater-Usoh-Steed questionnaire, thereby mirroring similar trends: exploration marginally lags behind other packages, which in turn are slightly below the comparative game study. In certain aspects, such as immersion and positive affect, the packages outperformed the comparative study, affirming the efficacy of the design choices. However, a holistic examination of the data reveals a pattern that generally aligns with the comparative paper.

– **Intrinsic Motivation Inventory:** The IMI was employed to gauge the Interest/Enjoyment domain within the study (Fig. 46), revealing encouraging outcomes. The NPC package secured the highest mean score at 5.81, reflecting a robust engagement level. Following closely, the Garden package registered a mean score of 5.5, while the Exploration package, though lower mean score at 4.8, still demonstrated a commendable level of interest and enjoyment. This latter score still aligns, even slightly higher, with findings from comparable game studies, such as "Adaptive Control of Cardio-Respiratory Training in a Virtual Reality Hiking Simulation: A Feasibility Study" [44] which reported an IMI Interest/Enjoyment score of 4.6.

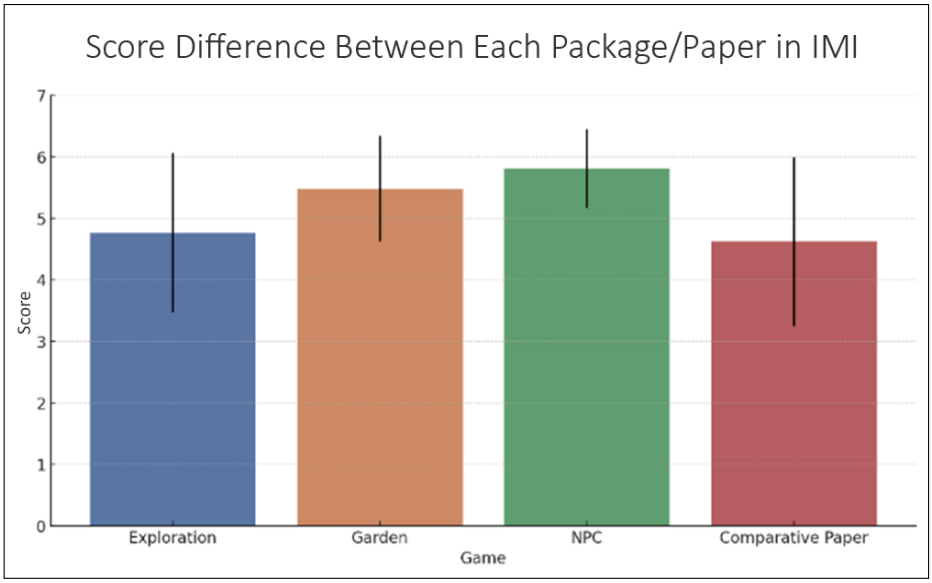


Fig. 46. Intrinsic Motivation Inventory questionnaire mean and standard deviation comparison in the Interest/Enjoyment domain with "Adaptive Control of Cardio-Respiratory Training in a Virtual Reality Hiking Simulation: A Feasibility Study" [44].

Uncategorized observations further underscored instances of heightened amusement and enjoyment within the Garden and NPC packages. These findings can be logically attributed to the inherent mechanics of the tasks. Specifically, constructing a garden represents an engaging and interactive endeavor, innately more stimulating than the relatively mundane activity of capturing objects along a trail. Similarly, conversing with a conversational avatar, particularly one

imbued with witty and sarcastic traits, introduces an element of novelty that likely contributed to enhanced enjoyment.

6.3.4.3 Ecological Validity

Ecological Validity (Fig. 47), being a custom questionnaire, does not lend itself to direct comparison with other games or studies. Nevertheless, it serves as a valuable metric for understanding participants' perceptions of the virtual worlds' validity. The mean scores are generally positive, reflecting similar trends in immersion and presence data. Intriguingly, participants perceived the NPC scenario as more ecologically valid than others, despite the NPC being "cartoonish" rather than "ultra-realistic" design. Initially, concerns were raised regarding a potential validity break owing to the uncanny valley effect, possibly exacerbated by awkward or erroneous actions, animations, or speeches by the NPC. However, this did not materialize in the results. The observed validity could plausibly be attributed to the genuineness of the conversation with the agent, coupled with a sufficiently detailed virtual environment to render a convincing level of realism.

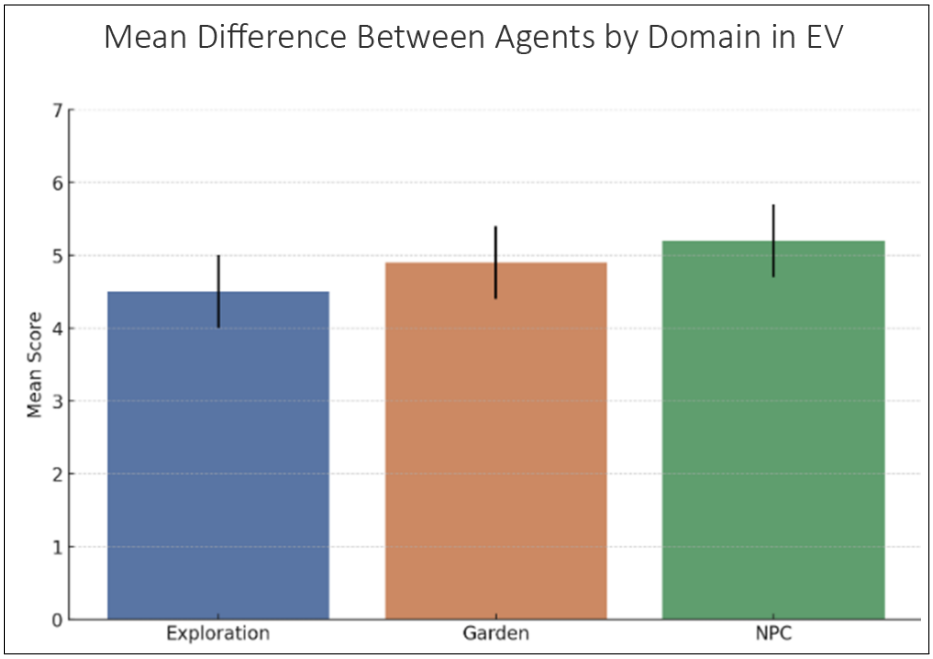


Fig. 47. Ecological Validity questionnaire mean and standard deviation comparison.

6.3.4.4 Observations

The salient observations recorded during this study pertain largely to errors and "soft-errors"—minor mistakes or issues that may not be critical to this research but are worthy of mention for prospective developers.

In the exploration package, most errors were of minor consequence, often involving the user teleporting past an object's hitbox and failing to capture it. This usually occurred once per session, and participants consistently recognized their error, rectifying it by teleporting to the correct location to obtain the object.

The garden package, despite its more restrained movement, posed some usability challenges related to object placement. An initial, informal usability study with three women aged 40-50 ex-

posed difficulties in manipulating virtual objects as if they were tangible. This interaction paradigm proved non-intuitive for this particular demographic. Consequently, a revised solution was implemented, allowing users to select, position, and modify objects through menu controls. While this adjustment yielded improvements, it was not without its challenges. A significant number of participants initially struggled with this mechanic, possibly due to neglecting to read the brief tutorial or being distracted or nervous (though this remains speculative). With minimal guidance, however, all participants were able to master the system.

The agent package's performance was generally smooth, with notable observations confined to occasional connection problems (affecting the NPC's ability to access the API) and minor transcription errors in the NPC's Speech-to-Text API. Participants quickly resolved these issues through corrective speech.

6.3.4.5 Conclusion

In this study, we have drawn upon serious games as a comparative benchmark, recognizing their closer alignment with the specific implementations of AVIR presented in our research. The choice of serious games, as opposed to entertainment-oriented ones, facilitates a more nuanced understanding of the attributes and potentials of the developed packages.

The exploration package, though showing marginally lower results compared to the other packages, holds its ground when juxtaposed with similar serious games. The observed disparity in results may stem from the inherent nature of the exploration implementation, which was less interactive than the others. However, this should not be misconstrued as a deficiency in the package itself. Its success, or otherwise, is contingent on the developer's implementation and the underlying objectives. In our example, the emphasis was not placed predominantly on factors such as enjoyment and immersion but rather on the psycho-educational aspect of gathering and learning from the captured objects.

The Garden and NPC packages, distinguished by higher engagement and enjoyment levels, showcased the potential of more interactive and immersive experiences. Ecological validity further affirmed the realism and applicability of the virtual environments across different design aesthetics.

In summary, this study contributes valuable insights into the design and evaluation of virtual reality packages within the context of mobile VR. It underscores the importance of aligning design choices with specific objectives and user needs, emphasizing that success is not solely determined by conventional metrics but also by how well the implementation aligns with intended learning or experiential outcomes. The findings offer both a validation of the proposed approaches and a roadmap for future developers and researchers seeking to leverage VR for serious gaming, education, and beyond. The versatility and adaptability of these packages open exciting avenues for exploration, underscoring the potential of VR as a transformative medium in diverse applications.

6.4 Study on RQ3. Can users noticeably differentiate between the personalities of AI agents in a Unity package during gameplay?

6.4.1 Objective and Rationale

The emergent use of LLMs in conversational and general NPC intelligence presents a novel and intriguing facet of game development. Recognizing the novelty and potential impact of this area, the study focuses on this customization aspect of the project, particularly how and if differences

in character personality can be perceived by users. The study thus aimed to investigate users' ability to differentiate between two contrasting personalities of the same NPC character, thereby addressing the pertinent research question.

6.4.2 Methodology

The study selected a convenience sample of 13 participants (4 females, 9 males) whose ages ranged from 18 to 48 years. All the participants were linked to a university setting, either as students or as staff members. Prior to the data collection, participants were informed about the study's objectives and furnished their written consent. Specific scenarios were tailored to portray the character personalities and environments. The study design entailed participants experiencing two distinct scenarios, with each interaction lasting around 4 minutes. The structure of the sessions was as follows:

- Preliminary Orientation: A succinct tutorial was provided at the onset to familiarize participants with the virtual agent's interaction mechanisms. This involved participants verbally engaging with the agent and awaiting a response after a brief 3 to 5-second pause.
- Character Personalities and Environments: The study encapsulated two distinct personalities, namely happy and angry, and two contrasting settings: a library environment showcasing a librarian character, and a Roman setup with a guide persona. These elements were systematically interwoven to craft varied yet analogous scenarios.
- Scenario Implementation: Participants were randomized to one of the created scenarios. A narrative script was presented to help guide the discourse with the NPC, ensuring a smooth and natural flow of conversation.
- Assessment and Feedback Collection: Upon completing each interaction, participants filled out a Self-Assessment Manikin [45] (SAM, Annex, Fig.63) questionnaire and a customized questionnaire (Annex, Fig. 64), specifically tailored to grasp the intricacies of this study's context. These tools aimed to comprehensively gauge participants' perspectives and feelings about the NPC's demeanor.
- Repetition and Comparison: Post-assessment, participants proceeded to the next available scenario and character personality. This repetitive structure was crucial to ensure participants were exposed to all distinct characters and environments, fostering an extensive comparative analysis of the NPCs' personalities.

Evaluation was rooted in specific interaction metrics. To capture the participants' experiences, qualitative methodologies, like observations and think-aloud methods, were incorporated. During the interaction sessions, participants were encouraged to verbalize their thoughts, sentiments, and motives.

6.4.3 Findings and Discussion

6.4.3.1 SAM

Each participant was administered a standard SAM questionnaire corresponding to each scenario. The observed variations were systematically analyzed, as illustrated in Figure 48. Subsequently, the difference of the agents was statistically evaluated using t-tests, the results of which are detailed in Table 18.

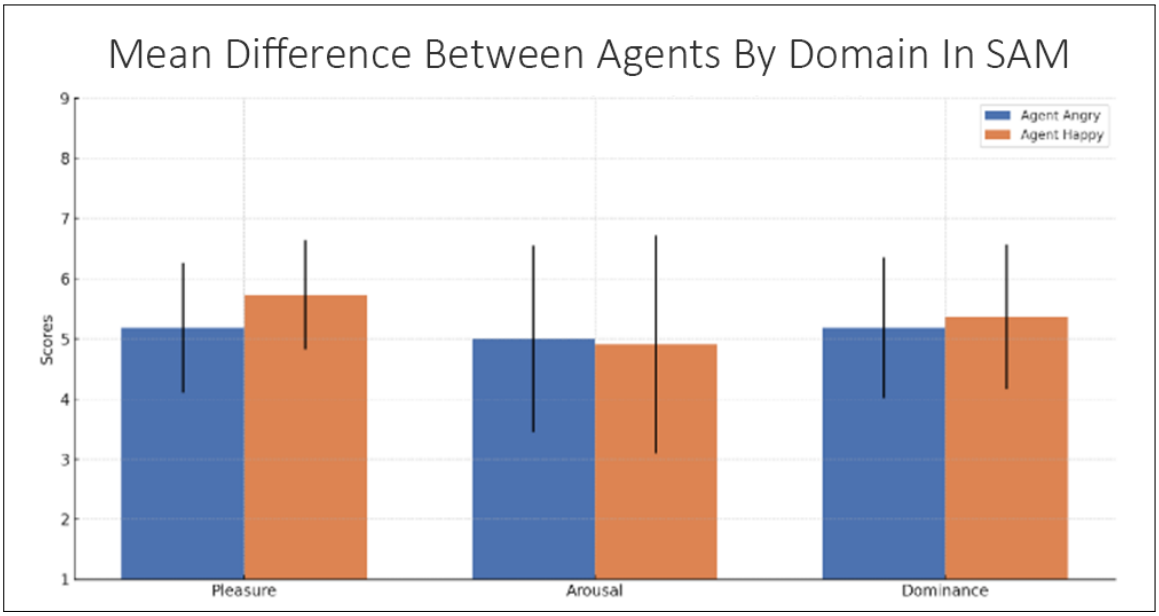


Fig. 48. Barplot of the difference between the two agents, question per question, on SAM.

Table 18. Affective Ratings for the Agent.

Measure	Pleasure	Arousal	Dominance
Mean ± SD	0.55 ± 1.37	-0.09 ± 1.58	0.18 ± 1.72
t	1.32	−0.19	0.35
p	0.22	0.85	0.73

6.4.3.2 Custom Agent Assesment Questionnaire

A comparative analysis was performed on the quantitative data extracted from the questionnaires related to two NPCs, designated as Happy and Angry. This comparison was executed by evaluating the means and employing a paired t-test to either refute or confirm the presence of a statistically significant perceived difference between these NPCs (Table 19 and Fig. 49).

Table 19. Difference between Angry agent and Happy Agent (Happy - Angry) - Custom Questionnaire.

Metric	Worried	Excited	Scared	Calm	Pleasant	Genuine	Feel	Presence	Like	Interaction	Engaging
Mean	-1.3077	0.7692	-1.1538	0.4615	1.9231	-0.2308	-0.539	-0.3846	0.9231	-1.4615	-1.462
SD	1.6013	1.6408	1.8640	1.2659	2.7827	0.9268	1.3301	1.1209	2.2532	1.984	1.898
t	0.012	0.117	-2.23	1.31	2.49	-0.90	-1.46	-1.24	1.48	-2.66	-2.78
p	0.012	0.117	0.045	0.213	0.028	0.387	0.170	0.240	0.165	0.021	0.017

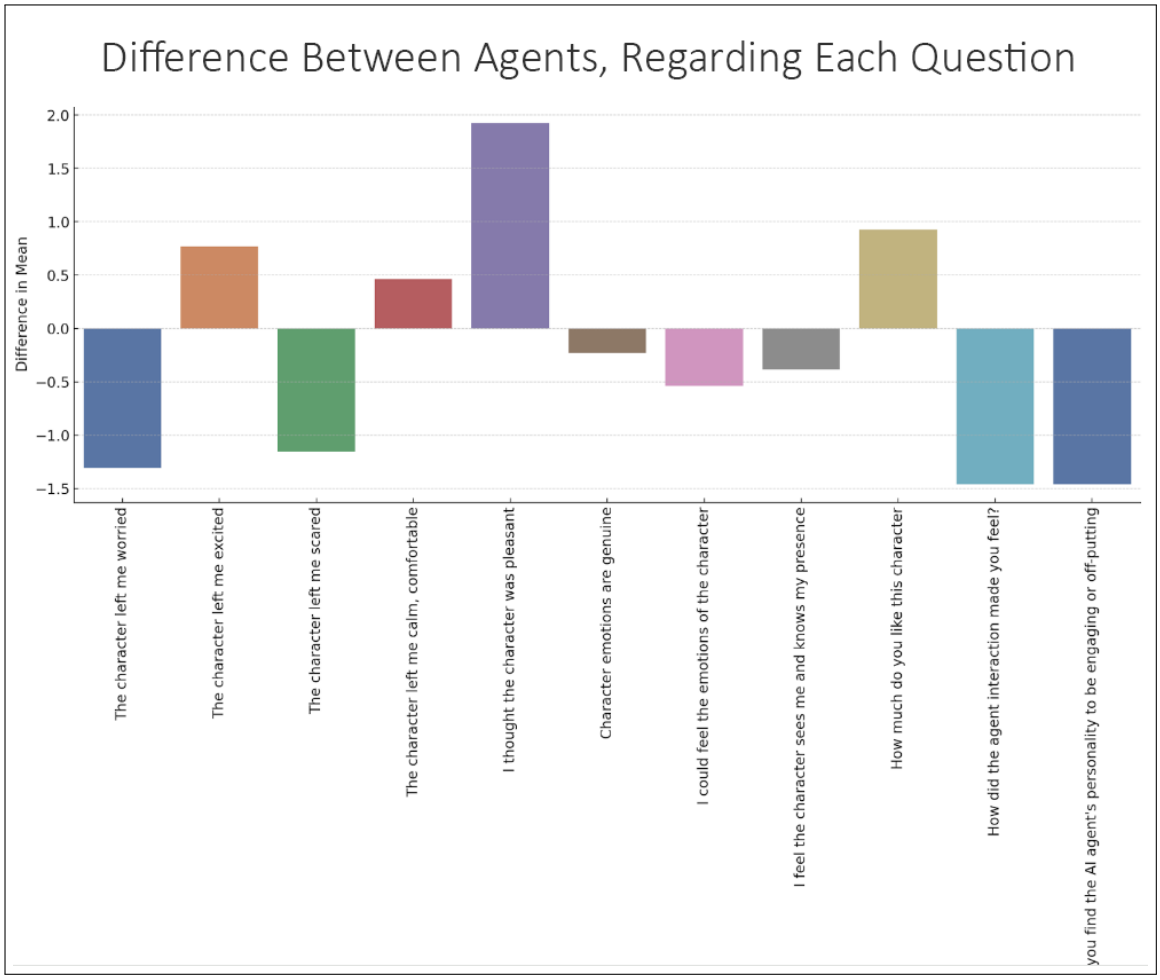


Fig. 49. Barplot of the difference between the two agents, question per question, on the custom questionnaire (Happy - Angry).

Qualitative data often presents challenges in objective visualization, and as such, we have employed word clouds (Figs. 50 and 51) to elucidate the observations made by participants concerning the two NPCs.

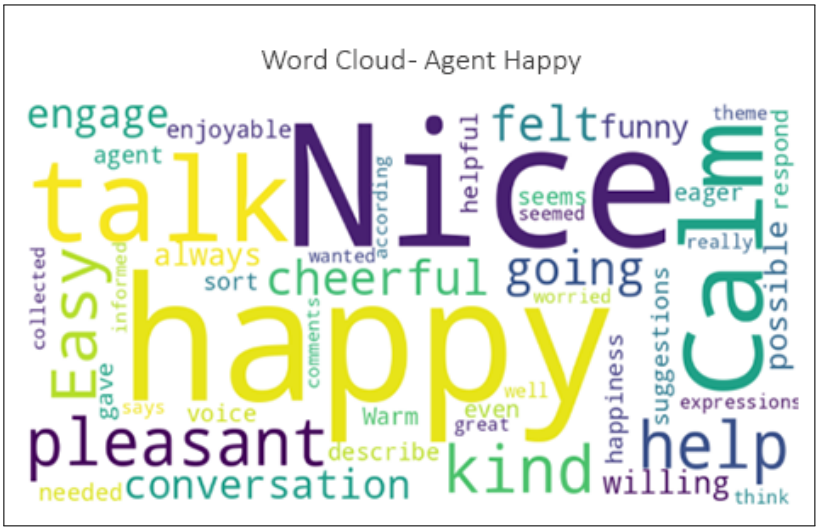


Fig. 50. Happy NPC Cloud W.ord

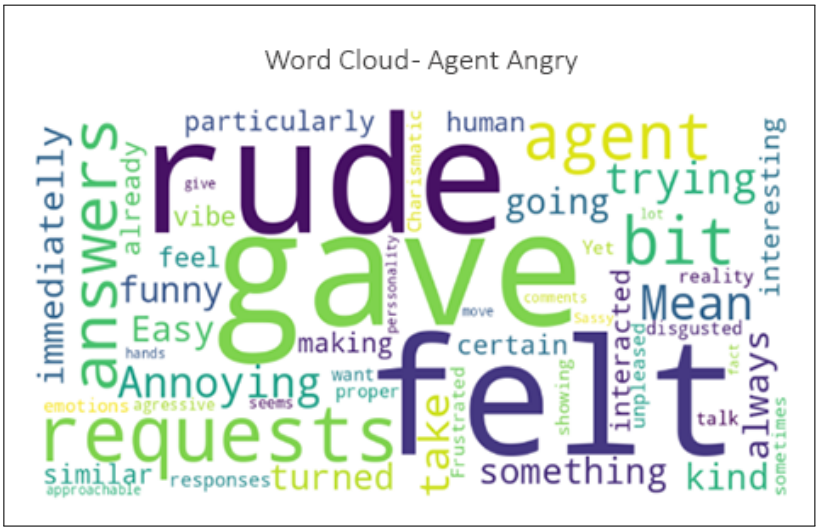


Fig. 51. Angry NPC Cloud Word.

Additionally, the TextBlob Python library was utilized for a sentiment/polarity analysis of participants' observations (Fig. 52), further enhancing our insights.

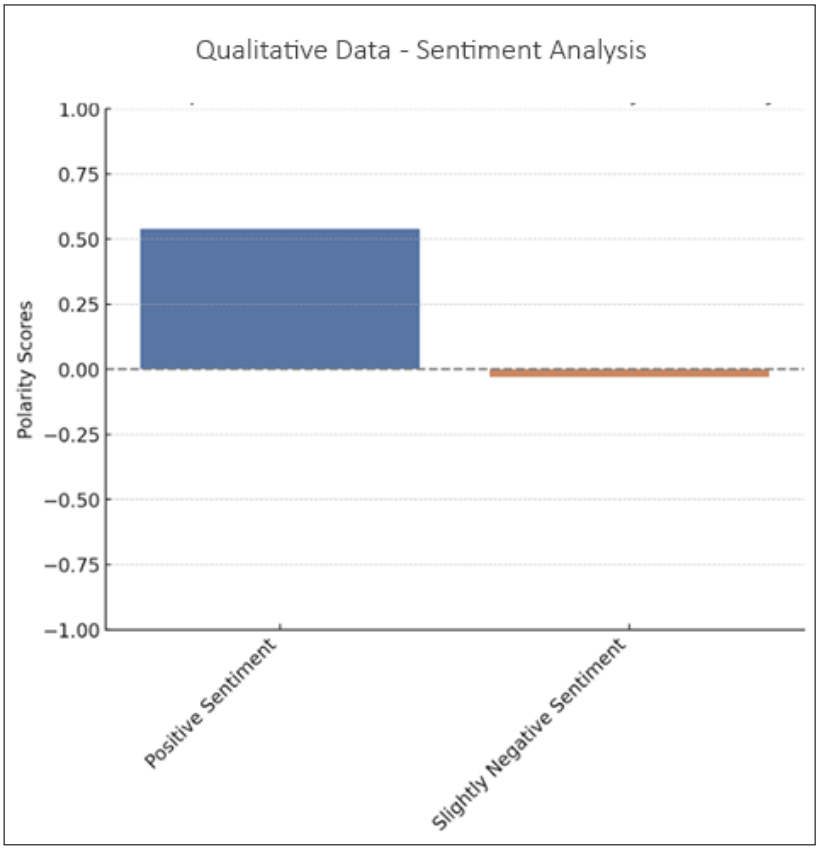


Fig. 52. Text Polarity analysis of the qualitative data from the custom questionnaire.

6.4.4 Conclusion

We did not go into much more detail on the SAM data since it proved to be inconclusive. The SAM data was not able to show a significant difference between the two agents. From this data we can simply conclude that the domains arousal, dominance and, to a lesser extent, pleasure, as defined by the SAM, seem to have no significant difference between our agents.

However, the custom questionnaire data was able to show a significant difference between the two agents. The findings of this study unambiguously illustrate perceived differences in the personalities of the agents, which closely align with the personality variations of the NPCs. While some metrics do not manifest a statistical disparity (such as "Are character emotions genuine" and "I could feel the emotions of the character"), more critical measures that directly relate to the research question (e.g., "I thought the character was pleasant," "The character left me worried") reveal significant variations in mean and, consequently, in the p-values of the paired t-tests.

The qualitative data, although contingent upon reader interpretation, also appear to underline the stark contrasts between the happy and angry agents, as discerned in the word clouds where the primary expressions range from 'happy' and 'nice' to 'rude'. The text polarity analysis further corroborates these distinctions, albeit the negativity associated with the angry agent is minimal. Based on the collected observations, this could be attributed to the participants' overall positive response to the experiment.

6.5 Cross-Study Reflection

Across the studies, several overarching themes and insights emerged, reflecting the multifaceted nature of the research questions explored.

In the study addressing RQ1, the focus was on the efficacy of the designed packages in aiding game development. Insights into functionalities, potential improvements, and feedback from experienced developers emphasized the usability of the packages and their potential to enhance the development process.

For RQ2, the study delved into the engagement and quality of gameplay experienced by end-users. The inquiry aimed to determine whether specific packages contribute to or potentially hinder captivating gameplay. Challenges related to usability, hardware constraints, and virtual reality considerations were explored, offering nuanced insights into the interplay between design, technology, and user experience.

In the investigation of RQ3, the study provided insights into users' ability to differentiate between NPC personalities, reflecting the novel aspect of character customization in game development.

Together, these studies underscore the potential benefits of the Unity packages in various dimensions of game development, from aiding developers to enhancing user engagement and experience.

6.5.1 Conclusion

The validation process, as illustrated by the conducted studies, offers a rich tapestry of insights into the potential and applicability of the developed Unity packages. From streamlining the game development process to crafting engaging gameplay and distinct NPC personalities, the contributions of this research are significant and multifaceted. The recognition of limitations and challenges serves as a roadmap for future research, paving the way for continuous improvement and innovation in the field. This iterative process of validation, reflection, and growth stands as a testament to the robust and dynamic nature of this research endeavor.

7 Conclusion

7.1 Summary of Research Objectives and Main Findings

The overarching objective of this research endeavor was to design and implement a trinity of software packages, each serving unique functions, tailored to streamline the developmental process of game developing, especially within the field of mobile VR. In pursuit of this goal, we believe the work herein presented has satisfactorily achieved multiple key milestones.

First, the procedural generation package has proven itself to be a robust tool for automating the synthesis of dynamic environmental elements, such as terrains, pathways, and vegetation. The efficacy of this package was empirically corroborated through both informal evaluations and formal analyses, addressing RQ1 and RQ2. Second, the co-creation package was developed with a focus on end-user customization, enabling functionalities ranging from personalized messaging storage to the crafting of virtual gardens. The package not only enhances user engagement but also fosters a sense of ownership over the virtual environment. And last, the NPC package offers an adaptable scaffold for designing NPC behavior. This package permits a myriad of customizations, thereby providing developers with considerable latitude in tailoring NPC interactions to suit specific project requirements.

Further to the aforementioned achievements, the packages were subjected to real-world application tests, revealing that they collectively serve to elevate user engagement and enhance the overall interactive experience. Notably, the utility of these packages extends beyond mere entertainment focused applications. Preliminary indicators suggest that the developed software packages are particularly apt for applications in which gameplay is ancillary to the main objective. This has been especially observable within the context of 'serious games,' a genre often leveraged for educational or therapeutic aims.

7.2 Significance and Contributions

The research presented herein elucidates the invaluable utility of the developed software packages, particularly within the context of academic research and serious games. Specifically, empirical observations at NeuroRehabLab corroborate the packages' applicability in ongoing projects such as EmotionalVRSystem and AViR. Notably, the need for organic terrains, conversational agents—especially those that are easily steerable—and user-customizable environments is pervasive in the serious games landscape.

Crucially, the significance of these packages lies in their accessibility and ease-of-integration, features that are particularly beneficial for professionals operating at the intersection of serious games, therapy, and education. Given that developers in these sectors often lack extensive experience in game development—a consequence of their primary focus being on therapeutic or educational outcomes—the proposed packages serve to mitigate the developmental load. By streamlining the implementation of frequently recurring elements, the packages allow these professionals to reallocate their resources and intellectual capital towards their core objectives, whether they be therapeutic intervention or pedagogical engagement.

It is reasonable, moreover, to extend this rationale to the broader realm of game development. The ease-of-use and modular nature of these packages make them ideal tools for prototyping and iterative design processes, thereby expediting the development cycle. Given the ubiquity of elements

such as natural terrains, conversational agents, and user-customizable environments in contemporary video games, the packages could serve as vital assets for developers aiming to either rapidly prototype novel concepts or to enrich existing game worlds. This is why it was very important to us to make this work very accessible by making it compatible and very optimized for mobile VR, allowing it to be used in a wide range of projects.

7.3 Limitations and Future Work

The limitations that we present not only elucidate the constraints regarding the studies but also provide avenues for future exploration. The following delineates the specific limitations encountered in the studies related to RQ1, RQ2, and RQ3, and suggests directions for future work.

7.3.1 RQ1

The primary constraint in the study addressing RQ1 pertains to the informality of package testing. In an ideal research scenario, a formal investigation would have encompassed 4 to 8 projects, with at least one developer from each project utilizing at least one of the three packages under study. Coupled with a focus group, this approach would have fostered a more robust validation. However, the practicality of assembling such a diverse array of developers, particularly within the confines of a master's thesis, presents logistical challenges that precluded this comprehensive evaluation.

7.3.2 RQ2

The study related to RQ2 was relatively unencumbered by limitations. The participant pool size aligned with expectations, and the instruments employed yielded rich insights into gameplay quality. Comparisons with other games or studies further substantiated the findings. A potential area for refinement could include the application of these packages in non-serious games. While the study's focus on serious games may be viewed as a minor limitation, the inclusion of diverse implementation examples could enhance the robustness of the findings.

7.3.3 RQ3

A nuanced limitation in the study on RQ3 was the occasional reticence of a minority of participants to provide candid questionnaire responses. Instances were observed wherein the characterization of an NPC did not align with the participant's apparent reaction. While these inconsistencies did not detract from the overarching conclusion that participants were able to discern personality differences, they did introduce a subtle ambiguity in the sentiment analysis. Future studies may consider employing methodologies that mitigate this limitation, such as in-depth interviews, to garner more accurate and reflective participant feedback.

7.3.4 Future Work

The journey of academic inquiry is inherently iterative, with each study paving the way for subsequent exploration. Building on the limitations identified in the studies related to RQ1, RQ2, and RQ3, there are several avenues for future research. For RQ1, there exists a potential for formalized testing with a larger and more diverse array of projects, delving deeper into the multifaceted applications of the packages. RQ2's findings could be bolstered by examining the applicability of packages in a broader spectrum of games, encompassing both serious and non-serious genres. Furthermore, the sentiments and perceptions of participants in RQ3 suggest an opportunity for more in-depth qualitative research methodologies, such as immersive interviews.

7.4 Final Remarks

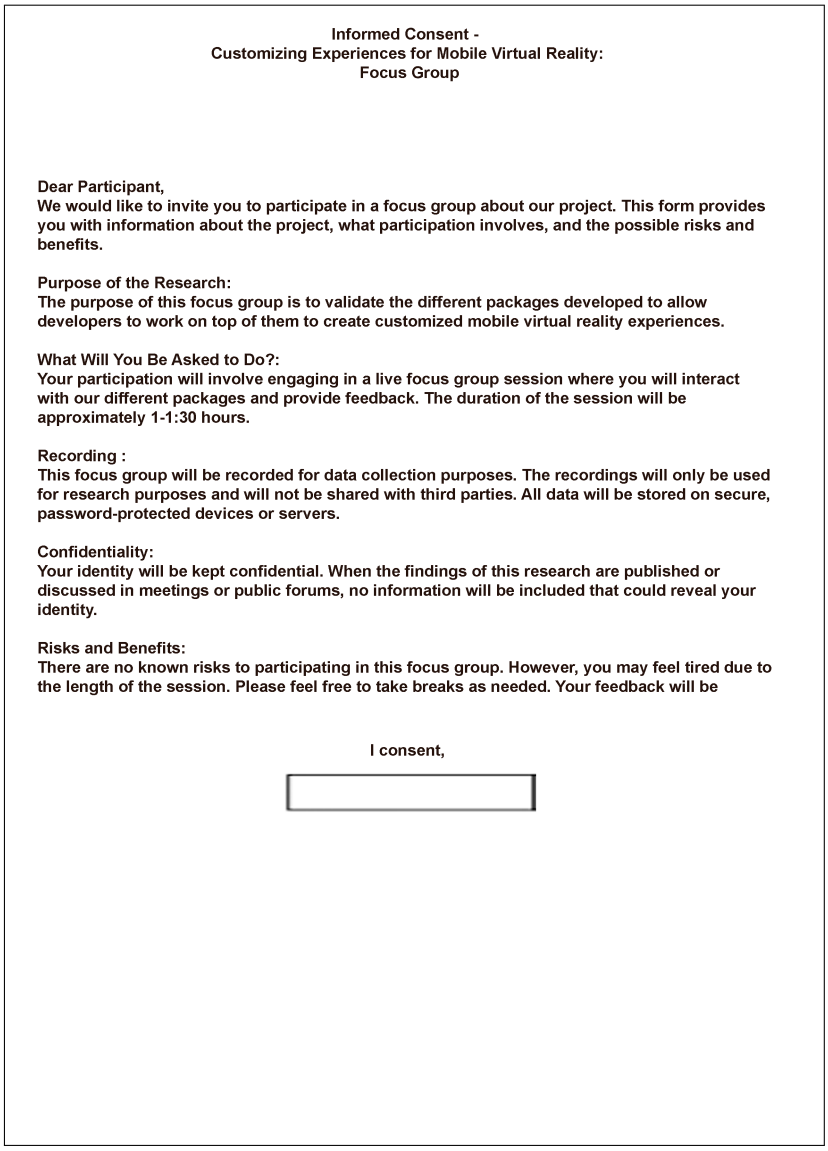
The field of mobile VR gaming is one of constant change and untapped potential. This thesis, in offering a suite of developer-friendly software packages, aims to make navigating this complex landscape a bit more manageable. We've worked to lower the entry barrier, especially for those in sectors like serious gaming, therapy, and education, who may be experts in their own right but not necessarily in the intricacies of game development.

Our work carries significance on several levels. Academically speaking, it opens new doors for subsequent research. From a practical standpoint, it delivers ready-to-use, versatile tools for today's developers. While the packages were carefully designed to be as plug-and-play as possible, their mobile VR compatibility significantly broadens their utility beyond this research project.

We should not overlook the growing importance of cross-disciplinary applications. Our work isn't just for game developers; it stands to benefit any field that could use gaming for anything from medical training to educational programs. The modularity of our packages lends itself well to such varied uses, providing a strong foundation for future interdisciplinary endeavors.

This thesis marks both an ending and a beginning—an ending to a phase of intensive development and testing, and the beginning of broader applications, adjustments, and hopefully, innovations in the dynamic field of mobile VR and gaming in general. As researchers, developers, or simply as people who are passionate about technology's ability to shape the world, we look forward to seeing how these contributions will be adopted, adapted, and built upon in the years to come.

8 Annex



Informed Consent -
Customizing Experiences for Mobile Virtual Reality:
Focus Group

Dear Participant,
We would like to invite you to participate in a focus group about our project. This form provides you with information about the project, what participation involves, and the possible risks and benefits.

Purpose of the Research:
The purpose of this focus group is to validate the different packages developed to allow developers to work on top of them to create customized mobile virtual reality experiences.

What Will You Be Asked to Do?:
Your participation will involve engaging in a live focus group session where you will interact with our different packages and provide feedback. The duration of the session will be approximately 1-1:30 hours.

Recording :
This focus group will be recorded for data collection purposes. The recordings will only be used for research purposes and will not be shared with third parties. All data will be stored on secure, password-protected devices or servers.

Confidentiality:
Your identity will be kept confidential. When the findings of this research are published or discussed in meetings or public forums, no information will be included that could reveal your identity.

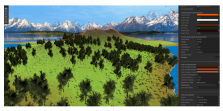
Risks and Benefits:
There are no known risks to participating in this focus group. However, you may feel tired due to the length of the session. Please feel free to take breaks as needed. Your feedback will be

I consent,

Fig. 53. Informed concent for study on RQ1

Customizing experiences for mobile virtual reality - Focus Group

Exploration Package



Very poor very easy

Ease of Use: How easy did you find this package to use?

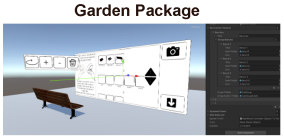
Customizability: How would you rate the customizability of this package?

Compatibility: How well would this package work in other projects?

Functionality: How well did the package perform as expected?

Open-ended question: What improvements, if any, would you suggest for this package?

Garden Package



Ease of Use: How easy did you find this package to use?

Customizability: How would you rate the customizability of this package?

Compatibility: How well would this package work in other projects?

Functionality: How well did the package perform as expected?

Open-ended question: What improvements, if any, would you suggest for this package?

Fig. 54. Custom questionnaire for the focus group part 1

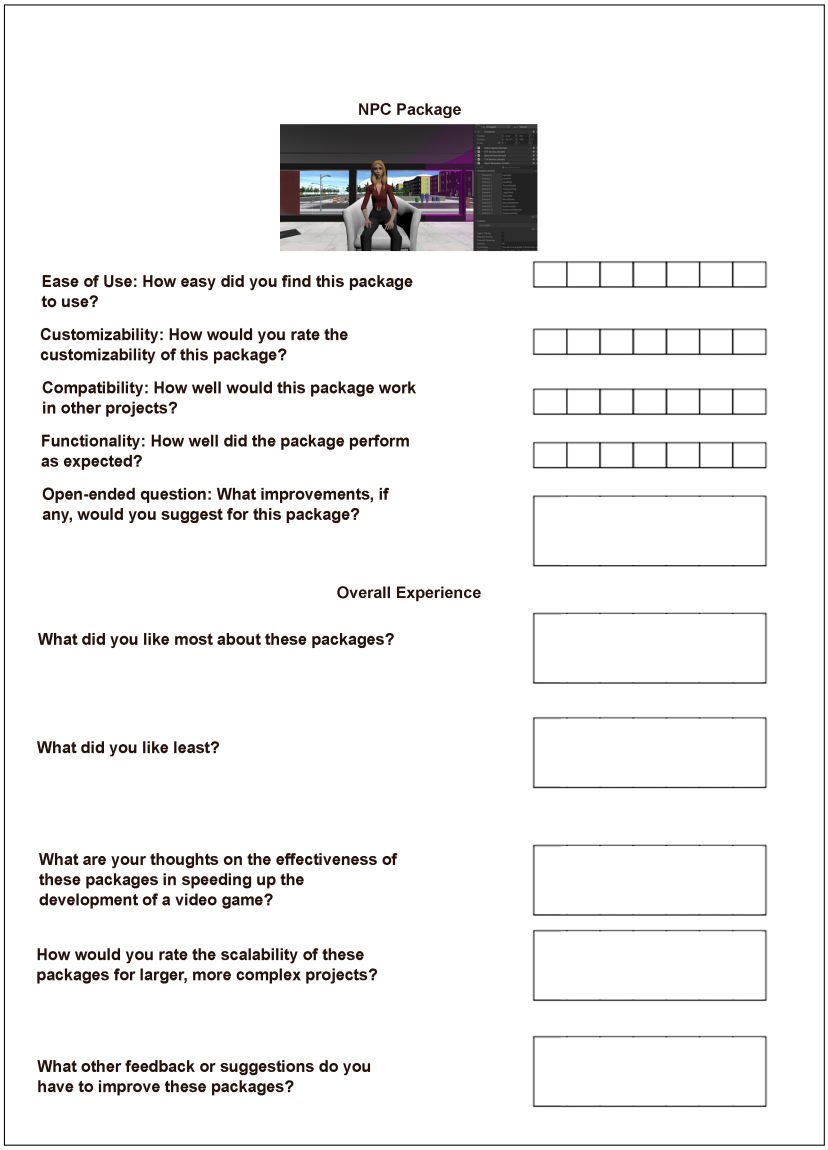
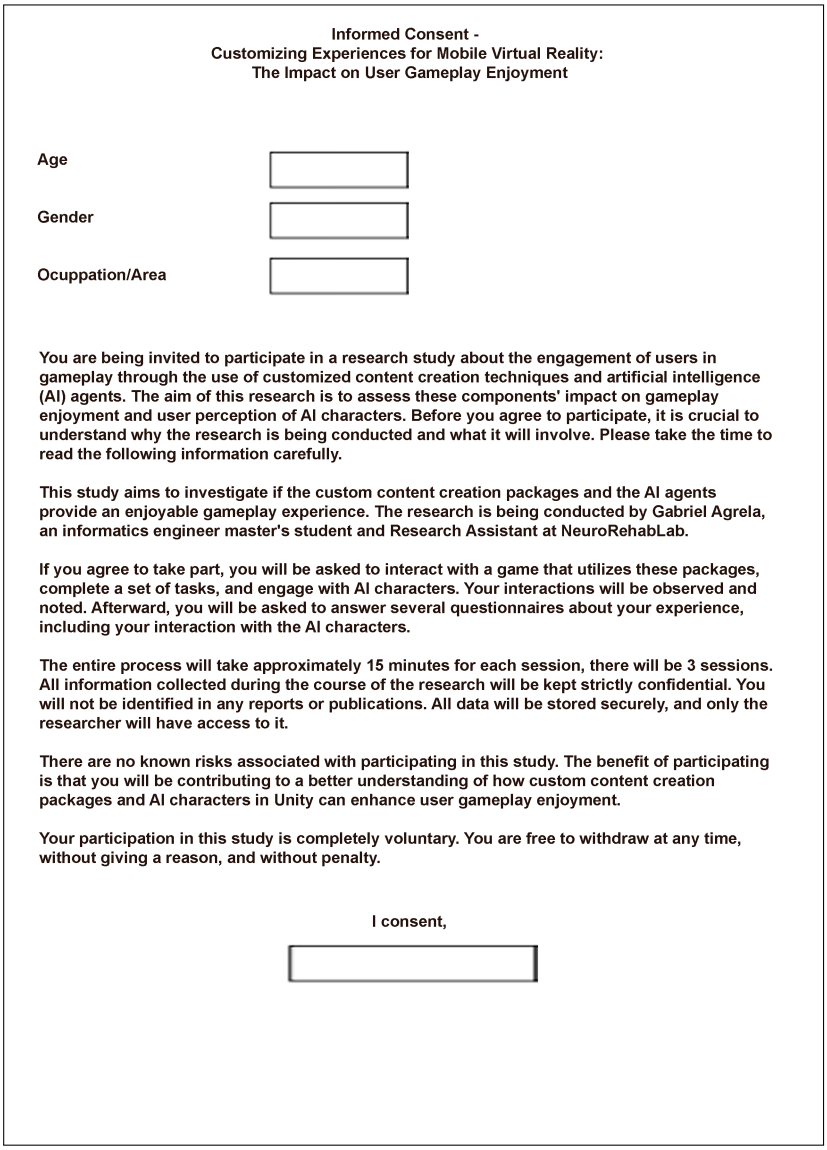


Fig. 55. Custom questionnaire for the focus group part 2



Informed Consent -
Customizing Experiences for Mobile Virtual Reality:
The Impact on User Gameplay Enjoyment

Age

Gender

Ocuppation/Area

You are being invited to participate in a research study about the engagement of users in gameplay through the use of customized content creation techniques and artificial intelligence (AI) agents. The aim of this research is to assess these components' impact on gameplay enjoyment and user perception of AI characters. Before you agree to participate, it is crucial to understand why the research is being conducted and what it will involve. Please take the time to read the following information carefully.

This study aims to investigate if the custom content creation packages and the AI agents provide an enjoyable gameplay experience. The research is being conducted by Gabriel Agrela, an informatics engineer master's student and Research Assistant at NeuroRehabLab.

If you agree to take part, you will be asked to interact with a game that utilizes these packages, complete a set of tasks, and engage with AI characters. Your interactions will be observed and noted. Afterward, you will be asked to answer several questionnaires about your experience, including your interaction with the AI characters.

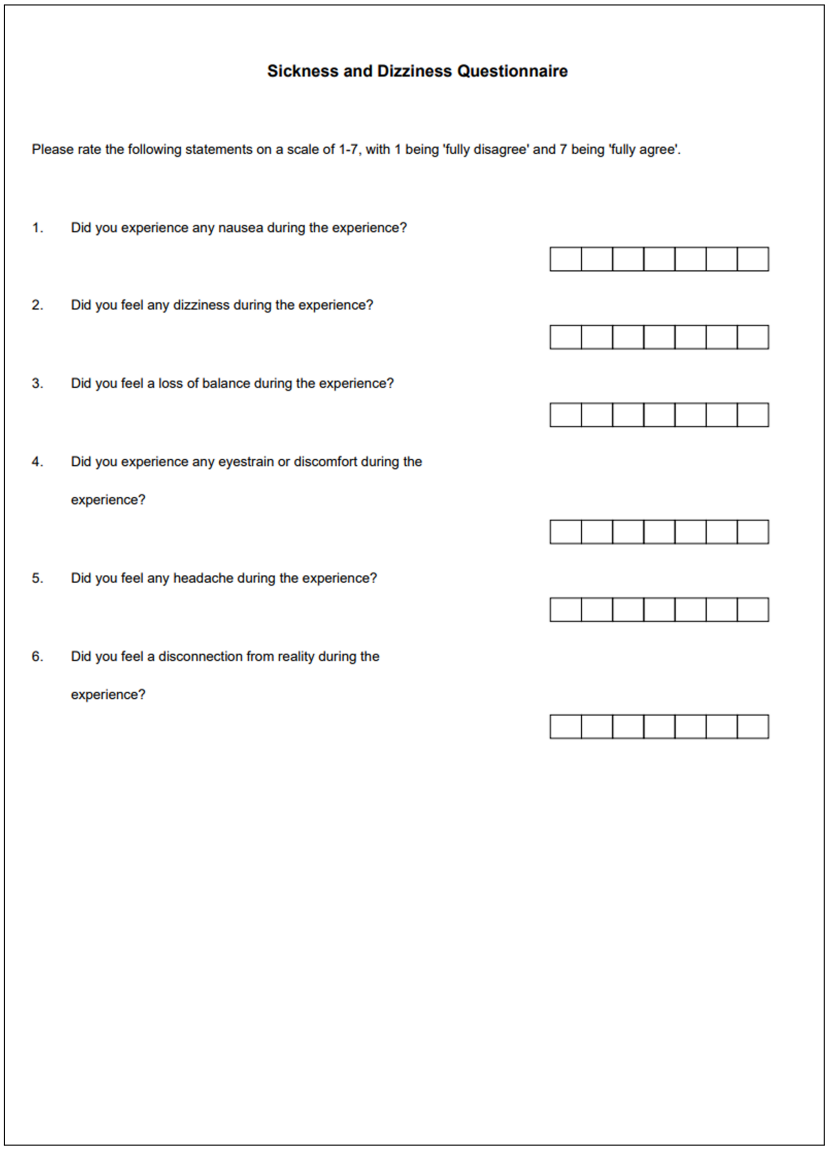
The entire process will take approximately 15 minutes for each session, there will be 3 sessions. All information collected during the course of the research will be kept strictly confidential. You will not be identified in any reports or publications. All data will be stored securely, and only the researcher will have access to it.

There are no known risks associated with participating in this study. The benefit of participating is that you will be contributing to a better understanding of how custom content creation packages and AI characters in Unity can enhance user gameplay enjoyment.

Your participation in this study is completely voluntary. You are free to withdraw at any time, without giving a reason, and without penalty.

I consent,

Fig. 56. Informed consent for study on RQ2 and RQ3



Sickness and Dizziness Questionnaire

Please rate the following statements on a scale of 1-7, with 1 being 'fully disagree' and 7 being 'fully agree'.

1. Did you experience any nausea during the experience?

2. Did you feel any dizziness during the experience?

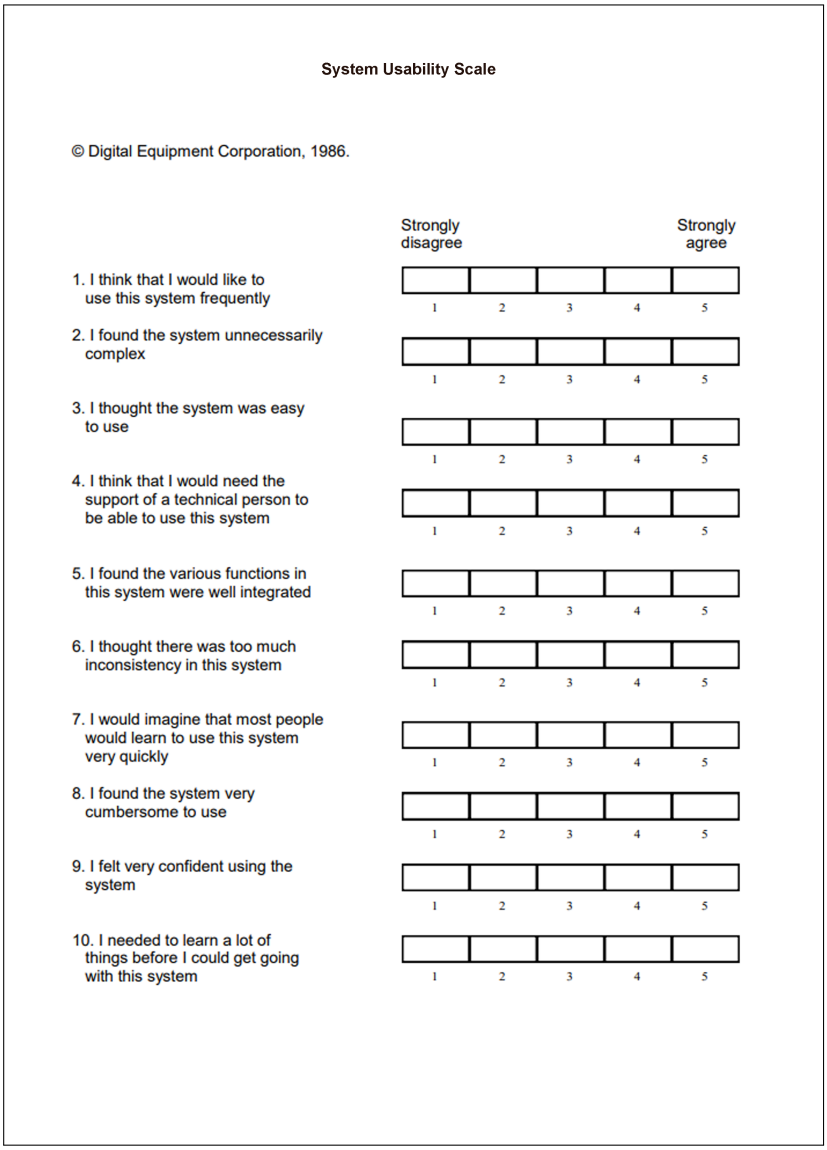
3. Did you feel a loss of balance during the experience?

4. Did you experience any eyestrain or discomfort during the experience?

5. Did you feel any headache during the experience?

6. Did you feel a disconnection from reality during the experience?

Fig. 57. Sickness and Dizziness questionnaire

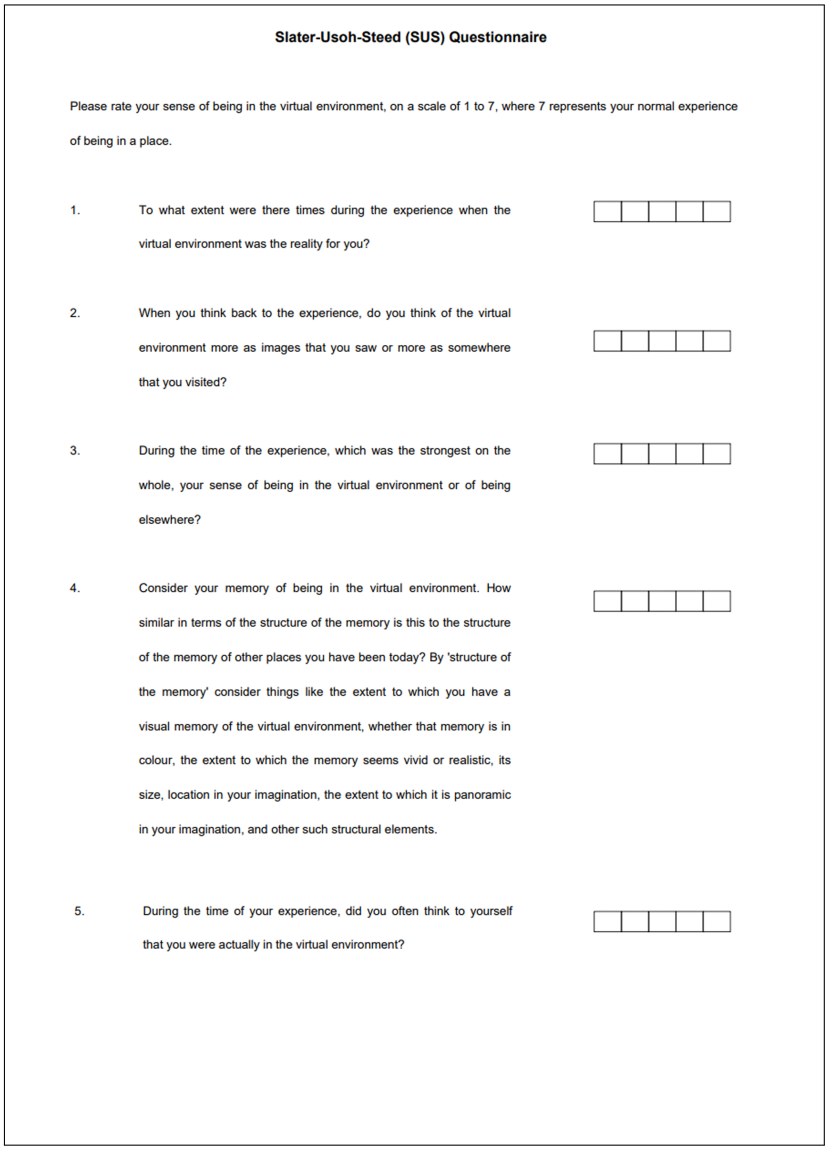


System Usability Scale

© Digital Equipment Corporation, 1986.

	Strongly disagree				Strongly agree
1. I think that I would like to use this system frequently	1	2	3	4	5
2. I found the system unnecessarily complex	1	2	3	4	5
3. I thought the system was easy to use	1	2	3	4	5
4. I think that I would need the support of a technical person to be able to use this system	1	2	3	4	5
5. I found the various functions in this system were well integrated	1	2	3	4	5
6. I thought there was too much inconsistency in this system	1	2	3	4	5
7. I would imagine that most people would learn to use this system very quickly	1	2	3	4	5
8. I found the system very cumbersome to use	1	2	3	4	5
9. I felt very confident using the system	1	2	3	4	5
10. I needed to learn a lot of things before I could get going with this system	1	2	3	4	5

Fig. 58. System Usability Scale questionnaire



Slater-Usoh-Steed (SUS) Questionnaire

Please rate your sense of being in the virtual environment, on a scale of 1 to 7, where 7 represents your normal experience of being in a place.

1. To what extent were there times during the experience when the virtual environment was the reality for you? ☐☐☐☐☐

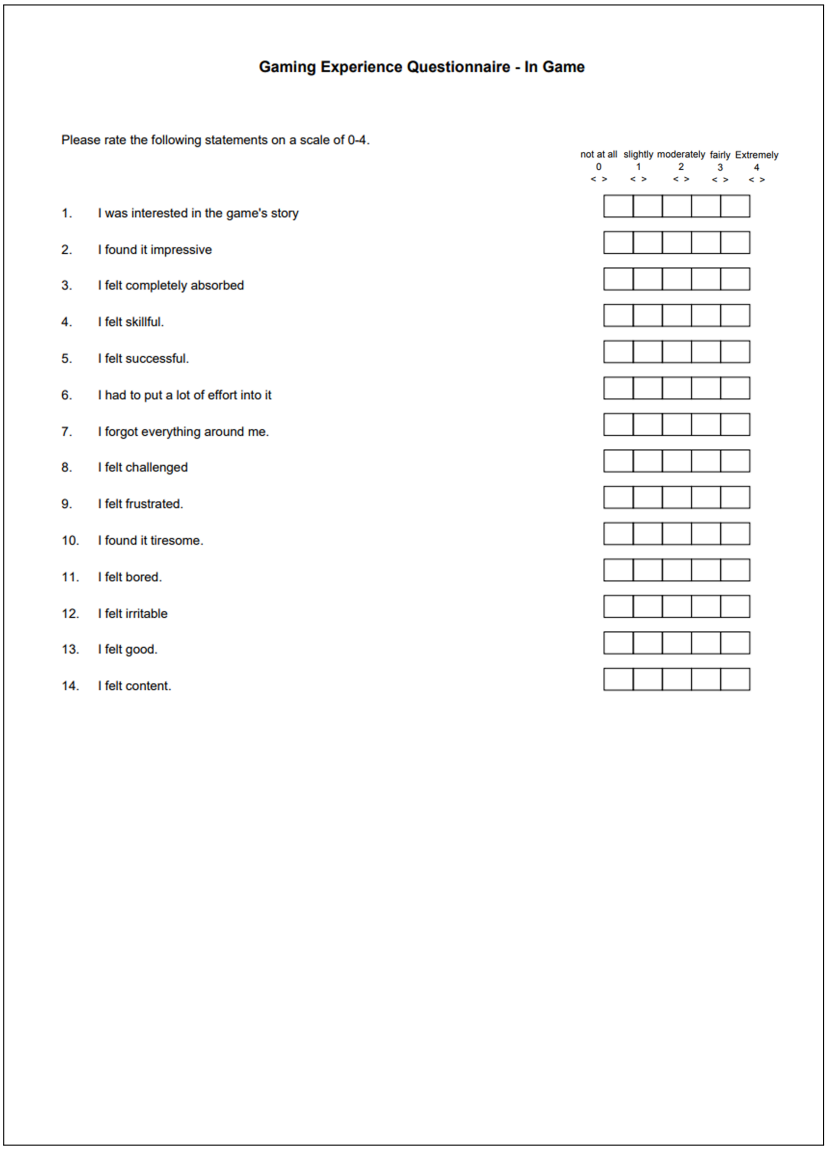
2. When you think back to the experience, do you think of the virtual environment more as images that you saw or more as somewhere that you visited? ☐☐☐☐☐

3. During the time of the experience, which was the strongest on the whole, your sense of being in the virtual environment or of being elsewhere? ☐☐☐☐☐

4. Consider your memory of being in the virtual environment. How similar in terms of the structure of the memory is this to the structure of the memory of other places you have been today? By 'structure of the memory' consider things like the extent to which you have a visual memory of the virtual environment, whether that memory is in colour, the extent to which the memory seems vivid or realistic, its size, location in your imagination, the extent to which it is panoramic in your imagination, and other such structural elements. ☐☐☐☐☐

5. During the time of your experience, did you often think to yourself that you were actually in the virtual environment? ☐☐☐☐☐

Fig. 59. Slater-Usoh-Steed Presence Questionnaire

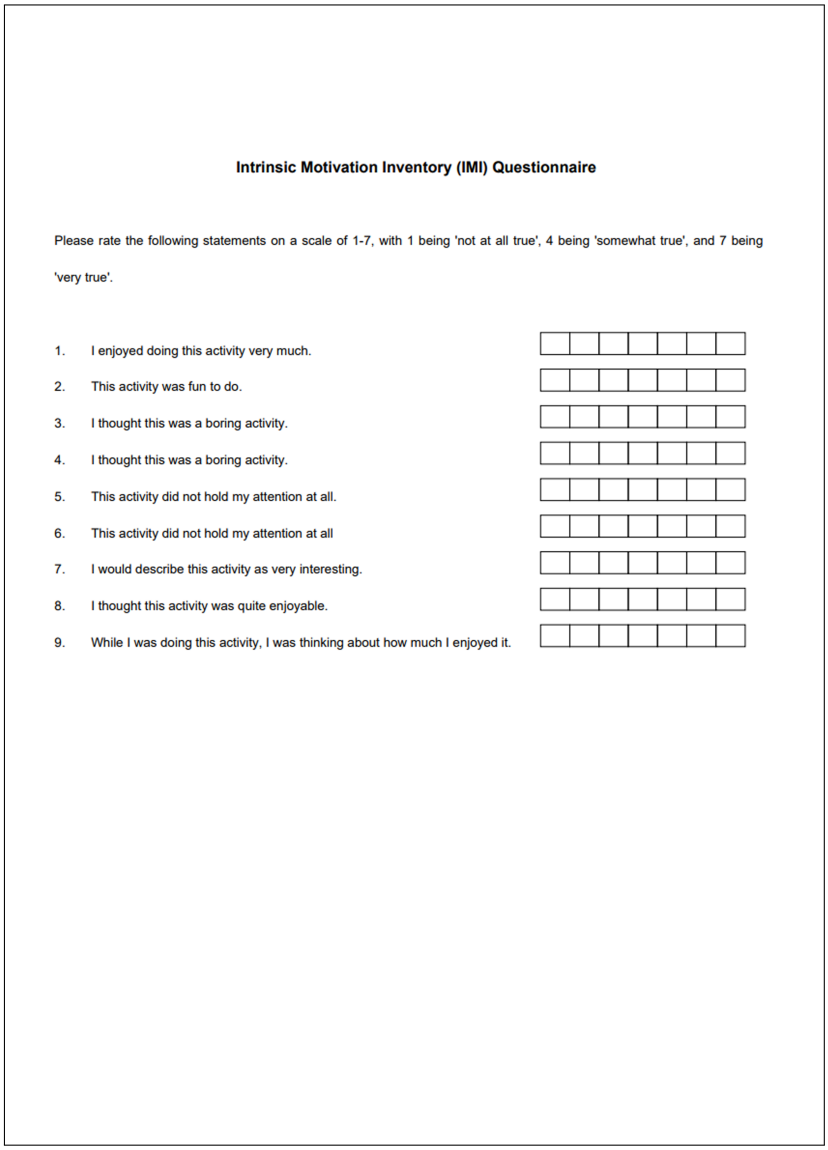


Gaming Experience Questionnaire - In Game

Please rate the following statements on a scale of 0-4.

		not at all 0	slightly 1	moderately 2	fairly 3	Extremely 4
1.	I was interested in the game's story	☐	☐	☐	☐	☐
2.	I found it impressive	☐	☐	☐	☐	☐
3.	I felt completely absorbed	☐	☐	☐	☐	☐
4.	I felt skillful.	☐	☐	☐	☐	☐
5.	I felt successful.	☐	☐	☐	☐	☐
6.	I had to put a lot of effort into it	☐	☐	☐	☐	☐
7.	I forgot everything around me.	☐	☐	☐	☐	☐
8.	I felt challenged	☐	☐	☐	☐	☐
9.	I felt frustrated.	☐	☐	☐	☐	☐
10.	I found it tiresome.	☐	☐	☐	☐	☐
11.	I felt bored.	☐	☐	☐	☐	☐
12.	I felt irritable	☐	☐	☐	☐	☐
13.	I felt good.	☐	☐	☐	☐	☐
14.	I felt content.	☐	☐	☐	☐	☐

Fig. 60. Game Experience Questionnaire - In Game

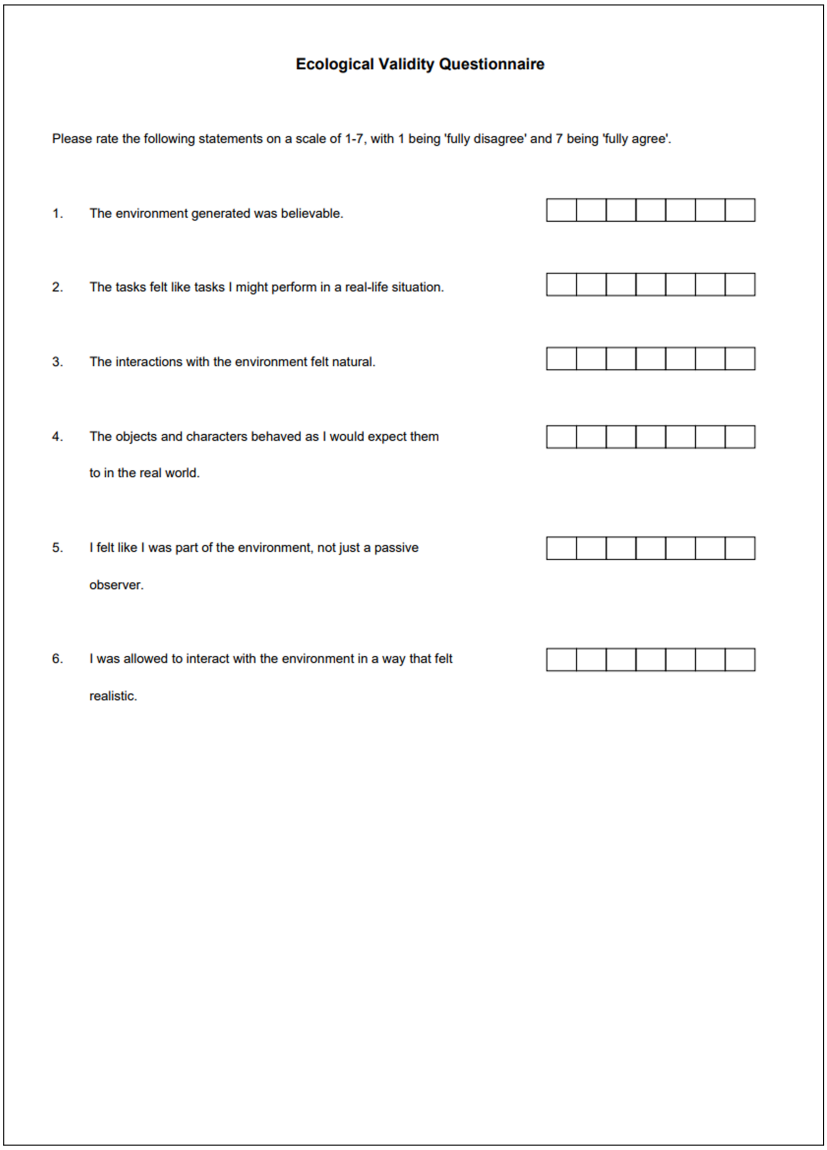


Intrinsic Motivation Inventory (IMI) Questionnaire

Please rate the following statements on a scale of 1-7, with 1 being 'not at all true', 4 being 'somewhat true', and 7 being 'very true'.

1. I enjoyed doing this activity very much.
2. This activity was fun to do.
3. I thought this was a boring activity.
4. I thought this was a boring activity.
5. This activity did not hold my attention at all.
6. This activity did not hold my attention at all
7. I would describe this activity as very interesting.
8. I thought this activity was quite enjoyable.
9. While I was doing this activity, I was thinking about how much I enjoyed it.

Fig. 61. Intrinsic Motivation Inventory Questionnaire.



Ecological Validity Questionnaire

Please rate the following statements on a scale of 1-7, with 1 being 'fully disagree' and 7 being 'fully agree'.

1. The environment generated was believable. ☐☐☐☐☐☐☐
2. The tasks felt like tasks I might perform in a real-life situation. ☐☐☐☐☐☐☐
3. The interactions with the environment felt natural. ☐☐☐☐☐☐☐
4. The objects and characters behaved as I would expect them to in the real world. ☐☐☐☐☐☐☐
5. I felt like I was part of the environment, not just a passive observer. ☐☐☐☐☐☐☐
6. I was allowed to interact with the environment in a way that felt realistic. ☐☐☐☐☐☐☐

Fig. 62. Ecological Validity Questionnaire

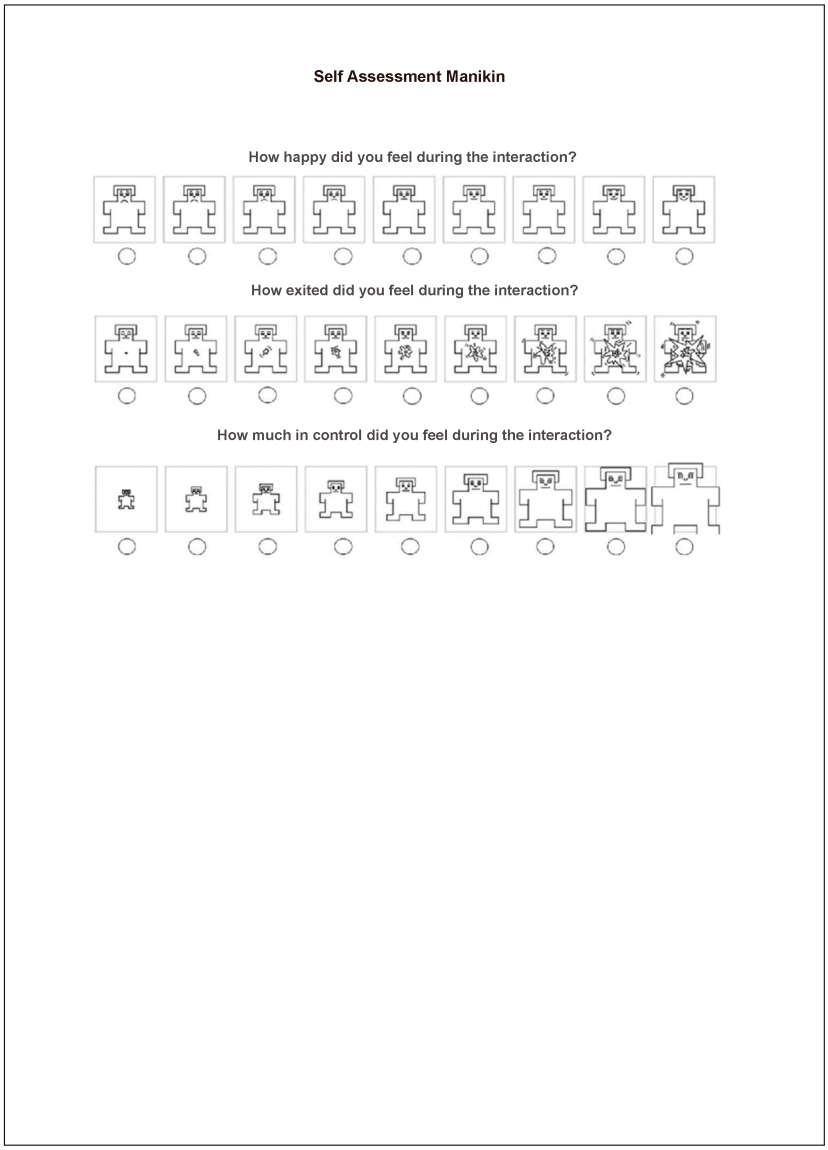


Fig. 63. Self-Assessment Manikin Questionnaire

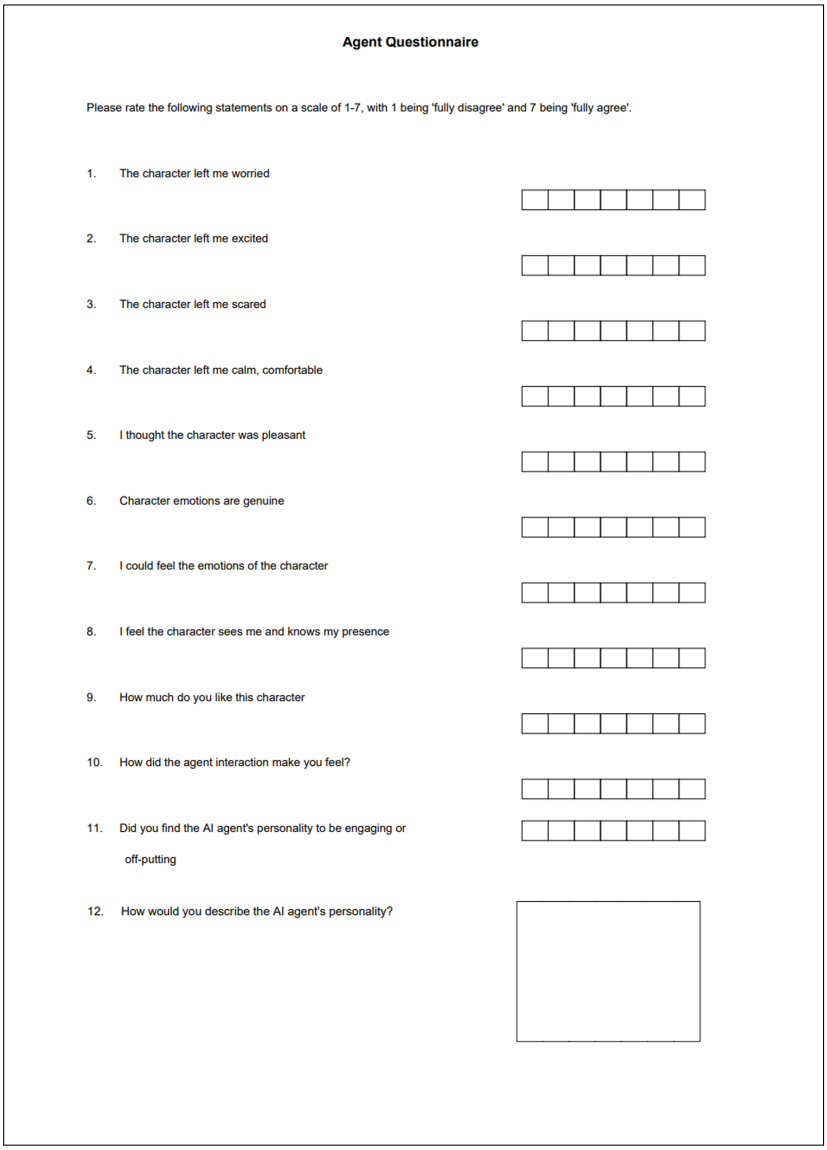


Fig. 64. Custom Agent Questionnaire

Assets	Techniques				Metrics				
	Unoptimized Default	Static Batching Techniques	Culling	LOD Technique	FPS	Batches	Tris Count	Asset Count	FPS Increase
Grass	x	-	-	-	30	110	1.2m	100k	0.0%
	-	x	-	-	33	85	961k	100k	10.0%
	-	x	x	-	150	25	163k	100k	400.0%
	-	x	x	x	155	24	135k	100k	416.7%
Tree	x	-	-	-	180	3126	1.5m	50k	0.0%
	-	x	-	-	190	74	1.5m	50k	5.6%
	-	x	x	-	280	60	415k	50k	55.6%
	-	x	x	x	285	56	208k	50k	58.3%

Table 20. 3-shot average technical analysis of optimization techniques on the vegetation assets

References

[1] E. Bethke, *Game Development and Production*. Wordware Publishing, Inc., google-Books-ID: m5exIODbtqkC.

[2] G. Lastowka, "User-generated content and virtual worlds," *Vand. J. Ent. & Tech. L.*, vol. 10, p. 893, 2007.

[3] T. Short and T. Adams, *Procedural Generation in Game Design*. CRC Press, google-Books-ID: Rj4PEAAAQBAJ.

[4] J. Freiknecht and W. Effelsberg, "A survey on the procedural generation of virtual worlds," vol. 1, no. 4, p. 27, number: 4 Publisher: Multidisciplinary Digital Publishing Institute. [Online]. Available: https://www.mdpi.com/2414-4088/1/4/27

[5] J. Togelius, G. N. Yannakakis, K. O. Stanley, and C. Browne, "Search-based procedural content generation: A taxonomy and survey," vol. 3, no. 3, pp. 172–186, conference Name: IEEE Transactions on Computational Intelligence and AI in Games.

[6] SXSW: Creating 87 bazillion guns for borderlands. [Online]. Available: https://www.engadget.com/2010-03-16-sxsw-creating-87-bazillion-guns-for-borderlands.html

[7] The director. [Online]. Available: https://left4dead.fandom.com/wiki/The_Director

[8] N. Shaker, J. Togelius, and M. J. Nelson, *Procedural Content Generation in Games*, ser. Computational Synthesis and Creative Systems. Springer International Publishing. [Online]. Available: http://link.springer.com/10.1007/978-3-319-42716-4

[9] S. Bermúdez i Badia, L. V. Quintero, M. S. Cameirão, A. Chirico, S. Triberti, P. Cipresso, and A. Gaggioli, "Toward emotionally adaptive virtual reality for mental health applications," vol. 23, no. 5, pp. 1877–1887, conference Name: IEEE Journal of Biomedical and Health Informatics.

[10] R. Linden, R. Lopes, and R. Bidarra, "Procedural generation of dungeons," vol. 6, pp. 78–89.

[11] A. Tatarinov, "Perlin noise in real-time computer graphics," p. 5.

[12] libnoise: What is coherent noise? [Online]. Available: https://libnoise.sourceforge.net/coherentnoise/index.html

[13] D. S. Ebert, F. K. Musgrave, D. Peachey, K. Perlin, and S. Worley, *Texturing & Modeling: A Procedural Approach*. Morgan Kaufmann.

[14] F. K. Musgrave, C. E. Kolb, and R. S. Mace, "The synthesis and rendering of eroded fractal terrains," vol. 23, no. 3, pp. 41–50. [Online]. Available: https://doi.org/10.1145/74334.74337

[15] T. Lechner and U. Wilensky, "Procedural city modeling."

[16] A. Summerville, S. Snodgrass, M. Guzdial, C. Holmgård, A. K. Hoover, A. Isaksen, A. Nealen, and J. Togelius, "Procedural content generation via machine learning (pcgml)," *IEEE Transactions on Games*, vol. 10, no. 3, pp. 257–270, 2018.

[17] I. H. Sarker, "Deep learning: a comprehensive overview on techniques, taxonomy, applications and research directions," *SN Computer Science*, vol. 2, no. 6, p. 420, 2021.

[18] S. Goldfarb-Tarrant, T. Chakrabarty, R. Weischedel, and N. Peng, "Content planning for neural story generation with aristotelian rescoring," *arXiv preprint arXiv:2009.09870*, 2020.

[19] S. Sarsa, P. Denny, A. Hellas, and J. Leinonen, "Automatic generation of programming exercises and code explanations using large language models," in *Proceedings of the 2022 ACM Conference on International Computing Education Research-Volume 1*, 2022, pp. 27–43.

[20] J. Kim, "Interactivity, user-generated content and video game: an ethnographic study of animal crossing: Wild world," vol. 28, no. 3, pp. 357–370, publisher: Routledge _eprint: https://doi.org/10.1080/10304312.2014.893984. [Online]. Available: https://doi.org/10.1080/10304312.2014.893984

[21] J. L. J. Grohn, "Exploring co-creation experience and value in the video game industry: how gamers create value through a rule changing online game that has no rules."

[22] M. Kangas, "Creative and playful learning: Learning through game co-creation and games in a playful learning environment," vol. 5, no. 1, pp. 1–15. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1871187109000704

[23] N. Rendevski, D. Trajcevska, M. Dimovski, K. Veljanovski, A. Popov, N. Emini, and D. Veljanovski, "PC VR vs standalone VR fully-immersive applications: History, technical aspects and performance," in *2022 57th International Scientific Conference on Information, Communication and Energy Systems and Technologies (ICEST)*, pp. 1–4.

[24] G. J. C. W. December 10 and 2021. Omdia research reveals 12.5 million consumer VR headsets sold in 2021 with content spend exceeding $2bn. Section: blogs. [Online]. Available: https://www.gamedeveloper.com/blogs/omdia-research-reveals-12-5-million-consumer-vr-headsets-sold-in-2021-with-content-spend-exceeding-2bn

[25] Steam hardware & software survey. [Online]. Available: http://web.archive.org/web/20200813051538/https://store.steampowered.com/hwsurvey

[26] Steam hardware & software survey. [Online]. Available: http://web.archive.org/web/20210813051444/https://store.steampowered.com/hwsurvey

[27] Steam hardware & software survey. [Online]. Available: https://store.steampowered.com/hwsurvey/Steam-Hardware-Software-Survey-Welcome-to-Steam

[28] G. Koulaxidis and S. Xinogalos, "Improving mobile game performance with basic optimization techniques in unity," vol. 3, no. 2, pp. 201–223, number: 2 Publisher: Multidisciplinary Digital Publishing Institute. [Online]. Available: https://www.mdpi.com/2673-3951/3/2/14

[29] J. Jie, K. Yang, and S. Haihui, "Research on the 3d game scene optimization of mobile phone based on the unity 3d engine," in *2011 International Conference on Computational and Information Sciences*, pp. 875–877.

[30] U. Technologies. Unity - manual: Optimizing shader runtime performance. [Online]. Available: https://docs.unity3d.com/Manual/SL-ShaderPerformance.html

[31] J. French. Billboards in unity (and how to make your own). [Online]. Available: https://gamedevbeginner.com/billboards-in-unity-and-how-to-make-your-own/

[32] Disable/remove plastic scm. [Online]. Available: https://forum.unity.com/threads/disable-remove-plastic-scm.1277663/

[33] I have to rant about plastic scm. [Online]. Available: https://old.reddit.com/r/Unity3D/comments/srmv61/i_have_to_rant_about_plastic_scm/

[34] Plastic scm really complicated. [Online]. Available: https://forum.unity.com/threads/plastic-scm-really-complicated.1245055/

[35] Mobile tree package. [Online]. Available: https://assetstore.unity.com/packages/3d/vegetation/trees/mobile-tree-package-18866

[36] Detective alice - 3d stylized humanoid character. [Online]. Available: https://assetstore.unity.com/packages/3d/characters/humanoids/humans/detective-alice-3d-stylized-humanoid-character-195953

[37] J. Brooke, "Sus: a "quick and dirty'usability," *Usability evaluation in industry*, vol. 189, no. 3, pp. 189–194, 1996.

[38] M. Slater, M. Usoh, and A. Steed, "Depth of presence in virtual environments," *Presence: Teleoperators & Virtual Environments*, vol. 3, no. 2, pp. 130–144, 1994.

[39] W. A. Ijsselsteijn, Y. A. de Kort, K. Poels, A. Jurgelionis, and F. Bellotti, "Characterising and measuring user experiences in digital games," in *conference; ACE 2007; 2007-06-13; 2007-06-15*, 2007, pp. 1–4.

[40] R. M. Ryan, "Control and information in the intrapersonal sphere: An extension of cognitive evaluation theory." *Journal of personality and social psychology*, vol. 43, no. 3, p. 450, 1982.

[41] A. Gonçalves, M. Montoya, R. Llorens, and S. Bermúdez i Badia, "A virtual reality bus ride as an ecologically valid assessment of balance: a feasibility study," *Virtual Reality*, vol. 27, no. 1, pp. 109–117, 2023.

[42] M. A. Ahmad, H. Sousa, É. R. Quintal, and S. Bermúdez i Badia, "Efficacy of augmented reality-based virtual hiking in cardiorespiratory endurance: a pilot study," in *Proceedings of the 14th International Joint Conference on Biomedical Engineering Systems and Technologies*, 2021, pp. 575–582.

[43] F. Pereira, S. Bermúdez i Badia, C. Jorge, and M. S. Cameirão, "The use of game modes to promote engagement and social involvement in multi-user serious games: a within-person randomized trial with stroke survivors," *Journal of NeuroEngineering and Rehabilitation*, vol. 18, pp. 1–15, 2021.

[44] R. Lima, M. A. Ahmad, H. Sousa, and S. B. i Badia, "Adaptive control of cardio-respiratory training in a virtual reality hiking simulation: A feasibility study." in *BIOSIGNALS*, 2022, pp. 91–99.

[45] M. M. Bradley and P. J. Lang, "Measuring emotion: the self-assessment manikin and the semantic differential," *Journal of behavior therapy and experimental psychiatry*, vol. 25, no. 1, pp. 49–59, 1994.

[46] Unity asset store - the best assets for game making. [Online]. Available: https://assetstore.unity.com/