Customizing Experiences for Mobile Virtual Reality

Gabriel Agrela
Faculdade de Ciências Exatas e da Engenharia
Universidade da Madeira
Funchal, Portugal
gabrielagrela99@gmail.com

Mónica Cameirão
Faculdade de Ciências Exatas e da Engenharia
Universidade da Madeira
Funchal, Portugal
monica.cameirao@staff.uma.pt

Sergi Bermudez i Badia
Faculdade de Ciências Exatas e da Engenharia
Universidade da Madeira
Funchal, Portugal
sergi.bermudez@staff.uma.pt

***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 issue by proposing the development of packages for customizable content generation, applied to the specific case of mobile Virtual Reality. This work separates the issue in a two-part solution, Procedural Generation of Content and Co-Creation of Content. Additionally, because this work focuses on mobile Virtual Reality, the limitations of hardware related with standalone Virtual Reality 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 implemented for the AViR prototype, a Virtual Reality application for psychological support after pregnancy loss.**

***Index Terms*—Procedural Generation of Content, Co-Creation of Content, Mobile Optimization, Mobile Virtual Reality, Customization of Experiences, Unity3D Packages.**

I. Introduction

A. Thesis outline

The paper will be divided into four main sections:

- Relevant background and exposure of the problem.
- State of The Art discussing previous works in the technologies that will be used for the generation of content, being Co-Creation and Procedural Generation, as well as looking into the major constraints of Mobile Virtual Reality Headsets.
- Methodology reports how the work was done until this point and how it will be done, given the State of The Art research conclusions.
- Conclusion and Future work looks into detail what were the main findings and current progress, as well as explaining the next steps.

B. Background

The creation of video games is a complicated topic that calls for a variety of tools and knowledge. Since there is a lot of industry competition and the majority of games do not break even, it is challenging for developers to generate a profit. This is due to the fact that projects of this nature come at a high cost, including marketing, sales charges, materials, and resources. It is also challenging to plan and budget for these projects because of the industry's high level of unpredictability, which makes it harder to determine if the game will be successful [1].

As one can imagine, a Virtual Reality game will suffer from all the previously discussed constraints, with added difficulties, derived from the newer and more unstable technology, as well as increased costs in hardware.

C. Research Question and Objectives

As seen earlier, game development is a complex and challenging endeavor that is often constrained by limited time and resources. In order to address this issue, this research proposes a solution that utilizes customizing packages for Unity, as well as the cutting-edge technologies of procedural generation and co-creation of content. By implementing these strategies, we aim to release resources that can be redirected toward other areas of the game's development, ultimately leading to a more efficient and successful project.

II. State of The Art

A. Introduction

This chapter includes the literature review of the principal topics of the thesis, painting a picture of the current technologies that play parts in the development of applications for standalone virtual reality headsets, procedural generation of in-game content, and the generation of in-game content by the users (co-creation).

The State of The Art is divided into two different sections:

- **Software**: Where the taxonomy and techniques of procedural generation are discussed and settled, and the usage of co-creation of content, seeing how other games did it and what conclusions can be taken from them;
- **Hardware**: Where the limitations of current stand-alone virtual reality headsets are recognized and the solutions for their shortcomings are researched.

B. Software

1) Procedural In-Game 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 has the ability to significantly improve the gaming experience. [2].

The incorporation of dynamic components in educational serious games can significantly improve students' learning opportunities. This technology can assist maintain 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 in order to succeed in the game, the usage of changing elements can also aid in the development of critical thinking and problem-solving skills. Overall, using this technology in serious games has the potential to transform the way that people learn [3].

a) Taxonomy: A good starting point for this technology is looking at its taxonomy. It is important to note that taxonomy in this area is very scarce and the following should be seen more as a spectrum, rather than a binary comparison [4]:

- **Online versus offline**: Content does not always need to be generated in real-time, some of it might be a good idea to generate it at the developing phase of the application. Let's say I 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. I can instead apply PCG to create it once and use it in the development (offline). In contrast, I might need PCG to create a different weapon every time an NPC 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, you probably don't need to pick it up and continue with your journey, you may just ignore it and find a better one. However, if there's a PCG maze or dungeon that stands between you and the end of the level, it is necessary.
- **Random seed versus parameter vectors**: Content can be generated with a certain degree of control, for example, there's this algorithm that creates a dungeon. At one extreme, you could have a PCG that takes a random seed and generates a dungeon, where you don't 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 if you have a seed, and you 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).
- **Stochastic versus deterministic**: Content could be deterministic, reproducible. This means if you have a seed, and you put it in the algorithm as a parameter, the algorithm's output should always be the same, as long as the seed doesn't 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 the user's current behavior into account (generic), let's say the algorithm just spawns enemies at a rate based on time. An adaptive PCG algorithm would take into account 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, number of enemies slain, or some other player performance metric)[5]. We see a 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 [6].

b) Procedural Techniques: Like it was previously discussed with taxonomy, procedural generation of content techniques are also vaguely, broadly described, and classified/categorized. The following are the most settled techniques in this area. 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 [5]. 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 generating a higher fitness value [4].
- **Constructive Generation Methods**: we may see 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 [7].

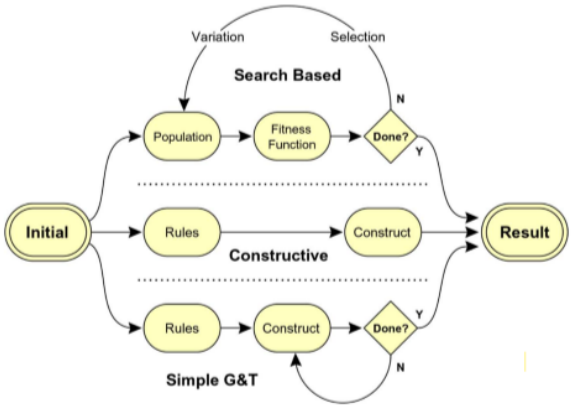

Fig. 1. Three ways of procedural generating content [4]

- **Height-maps or intensity maps**: a constructive method, usually it's closer to the random seed taxonomy spectrum, although most functions used (like perlin noise) are pseudo-random [8], making its output replica-table and therefore deterministic. Using some kind of noise function, 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 [5]. 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.

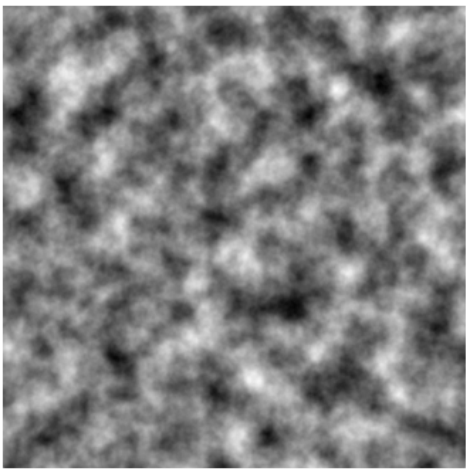

Fig. 2. Perlin noise example

- **Random terrain**: There are different interpretations of a random terrain, however, following an extreme random-seed approach will result in a map with unrelated coordinate values [9]. A way to come around this is by interpolating the values of each coordinate with its adjacent [5]. 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.

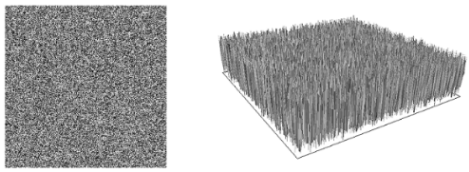

Fig. 3. Example of how random noise, without interpolation, produces a "spiky" and uninteresting terrain

- **Fractal**: Still a constructive method, it can be based on height maps and therefore mostly inherits its taxonomies, however, it's 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 [10] [5].
- **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 [5], agent-based approaches are usually on the end of this spectrum. With this Generate-and-test approach, there are different agents doing different things. In this city creation example, one might create areas, another create roads, and the last one connects the city roads between the areas [11].
- **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, like in the example of 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 patient's emotional state), online (created at loading) and stochastic [6].

2) Co-Creation of Content: Co-creation refers to the method of creating game content in partnership with user engagement. This can take numerous forms, such as allowing players to create their own game or levels, or allowing them to incorporate original artwork or other creative components into the game.

Content co-creation may be a good method for game developers to attract and retain players since it helps them feel involved in and invested in the game. Furthermore, by allowing players to offer new perspectives and methods to content production, it can lower the number of resources utilized by the development team in the design and creation of new material.

Video games may incorporate the co-creation of content in a number of different ways. As an illustration, some games allow players to contribute their own levels or game modes using in-game or mod tools, whilst other games may have community hubs or dedicated forums for this reason. Several games feature formal efforts or agreements with players to co-create content. For instance, some games allow a select group of players to create brand-new events or items for the game.

a) General Definitions [12]:

- **Content**: In general, content refers to a certain type of information that can be delivered in a variety of formats, such as text, photos, audio, or video. It is the content or material that is offered to an audience, and depending on the context, it can take many various forms.
  Content is also a type of creative labor since it frequently includes the expression of ideas or thoughts in a unique and entertaining manner. This can include articles, blog posts, videos, and other types of media that are intended to attract the audience's attention and give them useful information or enjoyment.
  The different parts of the game that comprise the player's experience are referred to as content in the context of gaming. This can include the plot, characters, and locations of the game, as well as any new material generated by the players.
- **User**: To have a user, you need both technology and a human. The technology offers the platform or interface via which the user interacts with the system, whereas the person is the human who uses the technology to execute a certain work or achieve a goal.
  A user cannot exist without technology since there is no way for the individual to interact with the system. Similarly, without a human, the technology would not be employed and hence would be ineffective.
  Technology and the user have a symbiotic connection in which the technology gives the tools for the user to achieve their goals and the user offers the intellect and creativity that propels the technology forward. This connection lies at the center of many of the world's most successful and inventive goods and services, and it is the driving force behind the world's fast technological progress.
- **Virtual World**: Virtual worlds are software systems that allow users to interact with a 3D world in-game. These platforms let users build and personalize their own avatars, explore virtual worlds, and interact in real time with other users.
  With cutting-edge gaming capabilities, appealing visuals, and a range of social elements, virtual worlds are often created to be very immersive and engaging. Users could actually feel as though they are transported into a different universe where they are free to interact and explore in ways that aren't feasible in the real world.
  Platforms for virtual worlds include Second Life, The Sims Online, and World of Warcraft. Many gaming-related websites presently have millions of active visitors. Platforms for exploring virtual worlds and interacting with others that are based in virtual reality often have a significant influence on the game business.

b) Animal Crossing Example [13]: An example of a game where user-generated content is valuable is Animal crossing, as we can see with (Kim, 2014)[13]'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 the player a sense of accomplishment but also allows them to access additional things and features inside the game.
  The museum is a crucial aspect of the game, where players may display the artifacts they have gathered via gameplay. 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.
  While completing pre-programmed tasks are vital in the game, it is the players' imagination and inventiveness that allows 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 player's pleasure in the game but also contributes to the gameplay remaining new and exciting.
- **Game as a labor and vice-versa**: Planning and crafting one's own route within a game can soon become habitual, potentially degrading the overall gaming experience. To overcome this, game designers might incorporate obstacles that push players to think outside the box and experiment with new techniques.
  A game that charges players with keeping a clean city for a specified amount of time, such as two weeks, may be an entertaining and gratifying endeavor. If the player successfully maintains a clean city for the entire two weeks, they may be awarded in-game rewards or other incentives. However, if they fail to keep the city clean, even if only by skipping one day of labor, they may be pushed behind substantially and have to restart the task.
  In contrast to more traditional, heroic games that generally focus around rescue 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 the game developers. Item gathering has long been a common narrative technique in video games.

c) Minecraft Example [14]:

- **Co-creation experiences and values**: The user base of Minecraft (MC) has the ability to influence and alter the game's gameplay, creating distinctive and individualized co-creation experiences and values.
- **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. Survival mode is about collecting items, fighting enemies, and crafting in order to thrive and survive, while adventure mode is about exploring player-created worlds that have certain game-play features limited to prevent disrupting the created world. 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 amount 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. Minecraft 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.
- **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 Minecraft. 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 Minecraft, 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, Minecraft is regarded as a socially networked game that has attracted a large and active community of players.
- **Skill Development**: The third identified Minecraft 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 Minecraft that promotes creativity, collaboration, and problem-solving skills through the game's open-world setting. In order to assist others learn and develop 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 Minecraft players, and it may be a useful tool for others to learn from and hone their abilities.

d) Educational Example [15]:

- 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 in-

volves not just obtaining information but also negotiating and co-creating it with others.

- 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. In regard to narratives, the idea of co-creation often implies that learning entails working together to develop and shape tales and narratives in order to more fully comprehend and make sense of experiences and knowledge.
- 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's and teachers' experiences of creative and fun learning processes.

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

- **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 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 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 must provide the user with tools and techniques to personalize and customize the user's world, and how implementing certain mechanics to keep players engaged is important, like NPC's that explode your house, or garbage that spawns in your city, making you have to interact regularly with the game in order to keep or expand your world. This is often accompanied by a reward system to further exacerbate the sense of accomplishment of engaging with the game mechanics.

C. Hardware

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[16]. However, even though advances have been made in the processing capabilities of SVR headsets, it's still rather important to look into optimization and other techniques to reach 60 fps gameplay. The following tables contain data related to Quest 1 and 2. Documentation on other SVR headsets, like Pico, does not reveal the recommended values as meta does. However, we can expect similar performance in that regard, since Pico, and most new SVR Headsets, use the Qualcomm Snapdragon XR2.

a) Constraints and bottlenecks overview: Given the technicality of the subject, it will be focused on the Meta Quest 2 bottlenecks/architectural constraints and its solutions. The following data is found in the meta developer documentation [17].

- **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 room available on the main and render thread, and graphics API and its usage are all points that have to be looked up close.

TABLE I
RECOMMENDED DRAW CALLS BY SIMULATION AND HARDWARE

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

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 doesn't have many pipeline state changes, like an escape room, or a puzzle game.

- **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's important to monitor and adjust your triangle budgets regularly and stay up-to-date with the latest rendering technology.

TABLE II
RECOMMENDED TRIANGLE COUNT BY HARDWARE

Platform	Triangle Count	Description
Quest 1	350k-500k	Busy simulation
Quest 2	750k-1.0m	Medium simulation

b) 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. [18].

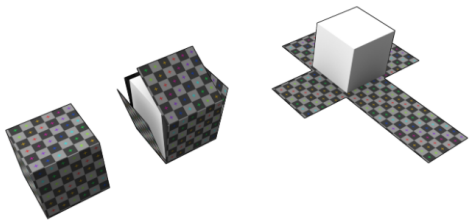

Fig. 4. Visual example of how UV mapping works.

The use of MipMaps can enhance the efficiency of game engine texture rendering by reducing memory and bandwidth requirements. MipMaps are recommended for all textures, except for GUI textures. 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. Overall, the use of MipMaps can occupy around 33% more memory but can greatly improve the performance of texture rendering [19].

- **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, but the CPU and memory requirements are the same. In order 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 [19].
- **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. In order to be drawn on the screen, all of 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 through the use of tools such as Blender's decimate modifier. However, it is more efficient to design low-poly models from the start rather than reducing the polygons in an existing model[18]. 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. When an object is close to the screen 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 save on resources and improve performance by only rendering the amount of detail that is necessary at any given time. Overall, the LOD system is an effective way to optimize game performance and save on system resources [19].

Fig. 5. Visual example of LOD working on a wheel model.

- **Vertices**: By using merge by distance, the number of

vertices may be decreased. By combining vertices that are near to one another, the ”merge by distance” tool enables the simplification of a mesh, much like the decimate modifier does. To prevent issues with overlapping faces, which might affect the application of textures and other features, this can be helpful when designing models. When dealing with duplicate vertices, the ”merge by distance” tool may be used with a distance value of 0, or it can be used with a value higher than 0 to get a result akin to the decimate modifier. The performance of game models may be enhanced and meshes can be simplified with the help of this program.

III. Methodology

A. Current Progress

Customizing Experiences for Mobile Virtual reality is being developed in parallel with AViR, a Virtual Reality application for coping with early pregnancy loss. As such, an agile, iteration-based approach was used as the development methodology. An early plan for AViR was already created, and the following are the description of packages planned and created until this time:

- **Exploring Package (PCG)**: A procedurally generated terrain, including vegetation, with bridges connecting otherwise in-passable bodies of water, with a main path leading to the end, and secondary paths, leading to secondary collectibles (guided by an in-game map).

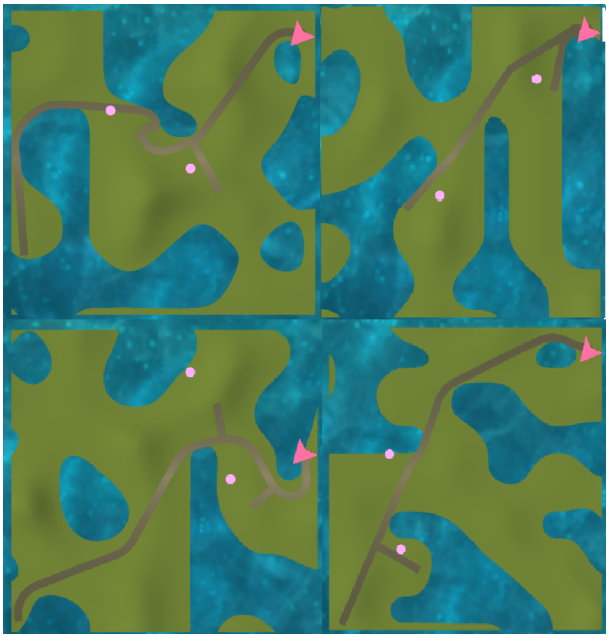

Fig. 6. Examples of the map showing 4 generations with 4 different seeds.

The objective is to collect all tokens/primary collectibles along the main path.

Fig. 7. Token sighting and collection.

Terrain size and texture, vegetation type, size and number, bridge type and collectibles type, message and priority are all customizable settings and parameters the developer can change for its needs.

- **Creation Package (CC)**: A procedurally generated terrain, including vegetation, is created, the user can also place items in the ground. The user can open a menu to choose an item, spawning it, and from there, it can be manipulated in an intuitive style. This means releasing, grabbing, rotating, spawning, removing, and changing the size of the said item.

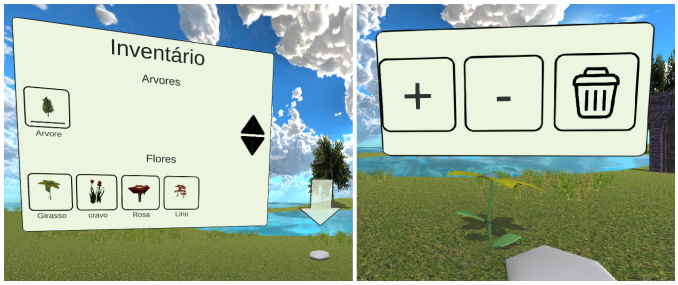

Fig. 8. Garden item menu selection and manipulation.

Any item the developer allowed can be placed any way the user wants, and as many as the developer allowed. Once the user is satisfied, the photograph can be taken and saved as a memory.

The developer can choose which items to unlock, or add new ones, as well as what to do with the picture taken by the user. The procedurally generated setting is customizable as well.

In another scene, there is a room with a chest and a menu. Here the user can record a message through voice, or write a letter, and store it in the chest as a memory. Both the recordings and the letters are interactable and can be played, opened, or deleted.

Fig. 9. Chest with one recording and one message written.

The work was iterated on a weekly basis, where at the end, in the weekly meeting, the current work would be reviewed, where any changes or improvements would be decided, various ideas would be discussed and the details of the next week's iteration would be defined. In parallel, the research on the State of The Art was being done, since later iterations will consist of fixing general issues of the two packages and implementing the methods and techniques discovered in the State of The Artwork. Because it was imperative that the packages would be playable in a Mobile Virtual Reality Headset, the State of the Art research started in the optimization techniques for the mobile development field. As such, early iterations were built to prove the possibility of a stand-alone execution, following iterations, where the packages were being developed, were built with solid foundations of optimization and development techniques that allowed them to be played in the Meta Quest 2 with the recommended, at least 72 frames per second. As some of the early iterations were about generating the terrain, parallel research in the State of The Art was being done about the procedural generation of terrains through height/intensity maps. In this work, the terrain is generated through a height-map technique with a function called Perlin Noise. Other implementations for the bases of the Exploring and the Creation packages are already implemented, and many changes should and will be made based on the techniques, methods, and overall learnings of the State of The Art.

B. Next Steps

The future methodology of this work will continue to consist of the same agile iteration-based approach utilized until now. By the end of each iteration, changes to the previous packages will be implemented based on the findings of the State of The Art. More specifically:

- **Terrain generation and Search-Based Approach** - will be implemented following a Search-Based Approach. There may be scenarios where the terrain is too fragmented by water bodies to allow the creation of a path. This approach lets us choose and grade generations that lead to a favorable creation of paths.
- **Fractals** - Vegetation spawn is very basic right now, the position they spawn is random and the sizes and rotations are static. To allow a more nature-like look, a fractal technique will be applied, this way, there will be forests and meadows (based on the density of grass and/or trees in certain areas). The parameters that set the density and frequency of these agglomerations will also be implemented, following a parameter-vector taxonomy.
- **Agent-Based Approach** - While the path is created following an agent-based approach, there is no mechanism that makes sure the agent can reach its final destination, and therefore, lay a complete path. This approach follows a generate and test taxonomy, although there is no "testing" yet, it will be implemented in the future.
- **Rewards and Narratives** - The reward system (the collection of tokens/collectibles) is not very customizable yet. Future iterations will allow the developer to customize the package easier, this follows the conclusions of the research into the Co-Creation of content, where users enjoy achievements/rewards, especially if these follow or allow them to create a narrative, being through their meaning, and/or their message.
- **Social** - Per our conclusions, the social aspect of a game is essential in the co-creation of content, as such, the resulting scenario in the Creation Package should have more ways to share it, other than the photograph. In case the developers using the package want to make a multiplayer game, and, for example, share their space with other online players, it will be implemented a save functionality that allows the user to save the scenario and its items, allowing the developer to instantiate it upon creating a server to share the scenario with other players.
- **Polishing** - Some general functionalities require more QA testing. Users can walk through water, fall out of the terrain if he approaches the edge and other issues may and will probably arise with testing. Packages lack immersive features at the moment, forests should have nature sounds, rivers should emit a water flow sound, etc. Feedback, in general, is weak, item pick up, button pressing and other interactions should have vibrations, animations, and sounds, letting the user know something happened, in a satisfying way. This is something that will be implemented soon, however, it will be closely monitored in the validation phase of this project and modified, if needed, accordingly.

C. Validation

a) Proof of Concept: As previously stated, this work is being developed parallel to AViR. AViR is a project developing an application about *Adaptive Virtual Reality For Coping With Involuntary Early Pregnancy Loss*. As such, our packages are of use in this project. The exploration package is used in the psycho-education stage, where picking up an object rewards the player with a token and educates him with a relevant

therapeutic message. The creation package is used to create a memorial and safe space, where the user can create his own little garden and photograph it, as well as record memories. Customizing these packages for the various anticipated AViR scenarios successfully indicates that this work is successful.

b) Usability Testing: To test our packages, it is expected that the experiment will gather a total of 15 participants. The ages will be between 18 and 45, although they are not relevant, given the context of this work being a fully customizable base for game developers, the same is defined for gender. It will be a Lab-based testing, users will be brought into the laboratory at arditi in a controlled environment where we will ask them personalized questions if needed, based on our live observations.

For more standardized and quantitative data, we will be using the standard System Usability Scale (SUS) [20]. For more specific data gathering, we will also use a task-based usability questionnaire with the following tasks:

- Pick up all primary tokens
- Pick up all secondary tokens
- Freely create a garden
- Take a picture of your garden
- Record a memory

And the following questions:

- How difficult was it to understand the instructions for this task?
- Were the controls for this task intuitive?
- Did you encounter any problems or frustrations while trying to complete this task? If so, please describe.
- How long did it take you to complete this task?
- Would you say that this task was enjoyable?

REFERENCES

[1] Erik Bethke. *Game Development and Production*. Google-Books-ID: m5exIODbtqkC. Wordware Publishing, Inc., 2003. 436 pp. ISBN: 978-1-55622-951-0.

[2] Tanya Short and Tarn Adams. *Procedural Generation in Game Design*. Google-Books-ID: Rj4PEAAAQBAJ. CRC Press, June 12, 2017. 339 pp. ISBN: 978-1-4987-9920-1.

[3] Jonas Freiknecht and Wolfgang Effelsberg. "A Survey on the Procedural Generation of Virtual Worlds". In: *Multimodal Technologies and Interaction* 1.4 (Dec. 2017). Number: 4 Publisher: Multidisciplinary Digital Publishing Institute, p. 27. ISSN: 2414-4088. DOI: 10.3390/mti1040027. URL: https://www.mdpi.com/2414-4088/1/4/27 (visited on 11/30/2022).

[4] Julian Togelius et al. "Search-Based Procedural Content Generation: A Taxonomy and Survey". In: *IEEE Transactions on Computational Intelligence and AI in Games* 3.3 (Sept. 2011). Conference Name: IEEE Transactions on Computational Intelligence and AI in Games, pp. 172–186. ISSN: 1943-0698. DOI: 10.1109/TCIAIG.2011.2148116.

[5] Noor Shaker, Julian Togelius, and Mark J. Nelson. *Procedural Content Generation in Games*. Computational Synthesis and Creative Systems. Cham: Springer International Publishing, 2016. ISBN: 978-3-319-42714-0 978-3-319-42716-4. DOI: 10.1007/978-3-319-42716-4. URL: http://link.springer.com/10.1007/978-3-319-42716-4 (visited on 11/30/2022).

[6] Sergi Bermúdez i Badia et al. "Toward Emotionally Adaptive Virtual Reality for Mental Health Applications". In: *IEEE Journal of Biomedical and Health Informatics* 23.5 (Sept. 2019). Conference Name: IEEE Journal of Biomedical and Health Informatics, pp. 1877–1887. ISSN: 2168-2208. DOI: 10.1109/JBHI.2018.2878846.

[7] Roland Linden, Ricardo Lopes, and Rafael Bidarra. "Procedural Generation of Dungeons". In: *Computational Intelligence and AI in Games, IEEE Transactions on* 6 (Mar. 1, 2014), pp. 78–89. DOI: 10.1109/TCIAIG.2013.2290371.

[8] Andrei Tatarinov. "Perlin noise in Real-time Computer Graphics". In: (), p. 5.

[9] David S. Ebert et al. *Texturing & Modeling: A Procedural Approach*. Morgan Kaufmann, 2003. 714 pp. ISBN: 978-1-55860-848-1.

[10] F. K. Musgrave, C. E. Kolb, and R. S. Mace. "The synthesis and rendering of eroded fractal terrains". In: *ACM SIGGRAPH Computer Graphics* 23.3 (July 1, 1989), pp. 41–50. ISSN: 0097-8930. DOI: 10.1145/74334.74337. URL: https://doi.org/10.1145/74334.74337 (visited on 12/01/2022).

[11] Thomas Lechner and Uri Wilensky. "Procedural City Modeling". In: (Jan. 1, 2003).

[12] Greg Lastowka. "User-generated content and virtual worlds". In: *Vand. J. Ent. & Tech. L.* 10 (2007), p. 893.

[13] Jin Kim. "Interactivity, user-generated content and video game: an ethnographic study of Animal Crossing: Wild World". In: *Continuum* 28.3 (May 4, 2014). Publisher: Routledge _eprint: https://doi.org/10.1080/10304312.2014.893984, pp. 357–370. ISSN: 1030-4312. DOI: 10.1080/10304312.2014.893984. URL: https://doi.org/10.1080/10304312.2014.893984 (visited on 12/08/2022).

[14] Jamie Lee Jo 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". In: ().

[15] Marjaana Kangas. "Creative and playful learning: Learning through game co-creation and games in a playful learning environment". In: *Thinking Skills and Creativity* 5.1 (Apr. 1, 2010), pp. 1–15. ISSN: 1871-1871. DOI: 10.1016/j.tsc.2009.11.001. URL: https://www.sciencedirect.com/science/article/pii/S1871187109000704 (visited on 01/08/2023).

[16] Nikola Rendevski et al. "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)*. 2022 57th International Scientific Conference on Information, Communication and Energy Systems and Technologies (ICEST). June 2022, pp. 1–4. DOI: 10.1109/ICEST55168.2022.9828656.

[17] *Testing and Performance Analysis — Oculus Developers*. URL: https://developer.oculus.com/documentation/unity/unity-perf/ (visited on 12/08/2022).

[18] Georgios Koulaxidis and Stelios Xinogalos. "Improving Mobile Game Performance with Basic Optimization Techniques in Unity". In: *Modelling* 3.2 (June 2022). Number: 2 Publisher: Multidisciplinary Digital Publishing Institute, pp. 201–223. ISSN: 2673-3951. DOI: 10.3390/modelling3020014. URL: https://www.mdpi.com/2673-3951/3/2/14 (visited on 12/08/2022).

[19] Jiang Jie, Kuang Yang, and Shen 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*. 2011 International Conference on Computational and Information Sciences. Oct. 2011, pp. 875–877. DOI: 10.1109/ICCIS.2011.317.

[20] John Brooke. "SUS: A quick and dirty usability scale". In: *Usability Eval. Ind.* 189 (Nov. 30, 1995).