Performance Analysis and Optimization Techniques in Unity 3D and Blender of a large-scale serious game simulation

Student name: Nikoleta Koleva,

Student number: 22040611

May 3, 2023

Abstract – This report aims to provide a methodology for the optimization of 3D models in Blender and Unity for the serious game "Pykrete bergs to Antarctica" - a large scale simulation game providing a solution to global warming. The game involves a large number of assets displayed at the same time. The purpose of the report is to compare optimization techniques from existent research by performing analysis on Unity 3D tools and Blender optimization approaches. The report's hypothesis is that by choosing a game engine and graphics program suitable to the developer's experience and the development of a serious game, combined with optimization techniques, the serious game would achieve good overall performance. The project's methodology consists of the tasks of describing the optimization approaches taken for the project, analyzing their efficiency, applying the optimization methods suitable for the development of the serious game and giving suggestions prior to future work. As a hardware requirement, the game will be implemented for a Windows or MAC device. After the implementation of the techniques identified in Unity 3D, limitations were found, such as the heavy reliability of the prototype on the memory of the device. However, the prototype performs well on most devices and the client was satisfied as it works well on a desktop device, which was the primary target. The research conducted has a significant importance as the prototype was built in an effective way, based on the techniques in the literature review, and limitations were found in the development process.

Index Terms – Blender, Unity 3D, optimization, techniques, analysis, performance

1 Introduction

1.1 Problem Overview

Climate change is transforming the underlying economic, political, social and environmental assumptions on which the nations have operated since the advent of the industrial revolution in the 19th century. Unless deep reductions in greenhouse gas emissions occur in the coming decades, global warming of 1.5°C and 2°C above pre-industrial levels will be exceeded during the 21st century (Urry, 2015) resulting in the transformation of the human, animal and plant life.

The solution proposed in this report is in the form of a serious game - a popular gamified approach used to prevent climate change and promote pro-environmental behaviors (Douglas, 2021). The scenario suggested in the game is transporting pykrete bergs to Antarctica's coastline, resulting in the carbon extracted from the atmosphere and the global warming frozen.

1.2 Research Topic

Creating serious games can be a very complex process, as it requires considering various hardware and software limitations. Since the game discussed in this paper is large in scale and involves many assets rendered on screen, it is a critical requirement to prioritize performance. Accordingly, there are optimization techniques that can be applied in different stages of the development of the game.

1.3 Existing Research

Popular optimization approaches in Unity 3D for game performance optimization are draw call batching and using a specific level of detail on 3D models based on their distance from the camera. Other techniques discussed in the existing research are optimizing meshes by reducing their polygon count, removing doubles, cutting draw calls by combining meshes and applying multiple materials on a single 2D texture.

1.4 Research Objectives

The objectives of this paper are to research existent optimization techniques in a suitable game engine and graphics program and apply them effectively to produce a serious game with high quality 3D models while ensuring good performance and player experience satisfaction.

1.5 Terms

- **Model:** A 3D asset in a video game that is created by adding textures and other features to a mesh.
- **Baking:** A method of preprocessing performed on game assets and data to ensure they load and perform well in real-time and do not slow down gameplay due to requiring a lot of processor or GPU capacity.
- **Texture:** A visual wrapping placed around GameObjects, such as the skin on a character.
- **Texture mapping:** The process of applying textures

to game objects.

- **Materials:** Editor objects that store the properties of surfaces, such as texture, shader, and color tint settings.
- **Draw calls** - Draw call is a call to graphics API to render objects. And each draw call contains all information about the objects that are going to be drawn.
- **Polygon:** A computer-programmed series of lines that form a three-dimensional (3D) object.
- **Mesh:** A collection of vertices, edges, and faces that act as the foundation of a model in a video game.
- **Batching:** Combining meshes into the same draw call is called Batching.
- **Static batching:** A technique that groups mesh that stay static in the runtime to reduce the draw calls.
- **Dynamic batching:** A method that group meshes that are static or moving in the runtime to reduce the draw calls.

1.6 Overview of paper's structure

- **Literature review** – A brief introduction is given to serious games. Key literature is discussed and evaluated, with its advantages and limitations.
- **Methodology** – Shows how research was conducted, the methods chosen and how they were implemented into the game.
- **Results** – The outcomes of the research/technical implementations are discussed. Results are shown and the evaluation approaches taken are described.
- **Conclusion** - Shows the value of the research and gives a summary of the report.

2 LITERATURE REVIEW

2.1 Introduction

Nowadays serious games are an established field of study. Most would attribute the rise of serious games to Clark C Abt who first created the term in 1970 and Ben Sawyer's who popularized it in 2002 (Wilkinson, 2016). However, considering the rich history of non-digital games, preceded by discussions of games that are traceable to the work of Plato, it can be stated that serious games are a contemporary manifestation of centuries old theories and practices.

To provide a good experience for the players, advanced knowledge of the game engine and the graphics program is required. Even if each serious game requires a unique approach for optimization, there are still key techniques that can be adopted in most situations. The aim of this literature review is to fill this gap, by evaluating the best techniques while performing appropriate tests and analyses with Unity 3D tools and Blender optimization approaches.

2.2 Game Engine Choice

The underpinning of developing this game is the game engine. Popular game engines are Unity 3D, Unreal Engine, CryENGINE, Shiva and Game maker (Singh et al., 2022). Vohera et al.'s paper concludes that in terms of platform deployment criteria, Unity and Unreal are the best, however, when it comes to visuals and animation, Unreal and CryEngine come out on top. Since the development team is more experienced with Unity and Unreal engine, they were further discussed.

Unity engine was chosen as it provides friendly tools, such as scriptable objects, with C# being a powerful language allowing for the use of object-oriented programming and applying postprocessing techniques (Masopust, 2021). Unity also has access to the .NetFramework providing additional new C# features such as dynamics (Jitendra, 2021).

2.3 Choice of 3D modelling program

To create and optimize the 3D models designed for the game, a suitable graphics program would need to be selected. There are many programs in this area, commercial and non-commercial, which have a greater or lesser number of tools for creating graphics models. For this project, the most sophisticated program of the non-commercial sector would be chosen - Blender (Pokorný, 2010), allowing the developer a huge number of modelling tools for the analytic and polygonal representation. Other programs considered for the project include CAD, which allows more advanced functions such as storing 3D data on models (Nzetchou, 2019), but due to the software price, it was decided to use Blender as it fully covers the current requirements for the serious game at this stage of development.

2.4 Problem Background

On execution of an application (serious game), the central processing unit (CPU) of the device executes its instructions. When executing a 3D computer graphics application, a frame requires the execution of millions of CPU commands, which have to be finished within a certain time to sustain a good frame rate (Singh et al., 2022). If this is not managed by the CPU in a timely manner, as a result, the game may be slower or freeze. There are multiple reasons why bottlenecks may appear, and numerous optimization approaches that could be undertaken to mitigate their appearance and optimize the game.

2.5 Optimization Techniques

Commonly used approaches in multiple works (Turpeinen 2020, Narkilahti,2021, Singh et al., 2022, Lehtola, 2018) for game performance optimization are draw call batching and using a specific level of detail on 3D models based on their distance from the camera. While these methods are a more popular approach, there is a gap in the research as other authors (Koulaxidis, 2022, Dere et al.'s, 2010). also explore optimization methods in Blender in the modelling stages. In their research, they have involved optimizing

meshes by reducing their polygon count, removing doubles, cutting draw calls by combining meshes and applying multiple materials on a single 2D texture.

After reading more literature on the topic, it was decided that reduced level of detail won't be applied to the low poly models as Dere et al.'s work states that it is more efficient to design models with reduced polygons/triangles from the beginning than reducing them to an existing one. This is a result of the decimate modifier in Blender being used on the model, where a small ratio value could corrupt the final object.

2.6 Conclusion

In conclusion, there are currently gaps in the existing research as more can be applied to optimize 3D models and ensure that the game provides smooth and responsive experience. Starting from the 3D modelling software, it can be assured that assets are optimized to a certain methodology. As they will be manually designed, their files would be always accessible and additional changes can be made at a later date if the client desires or if the developers see a need for further improvement of the game's performance. Additionally, efficiency in the game can be improved by applying the optimization approaches in Unity 3D.

3 METHODOLOGY

3.1 Research Questions

- What techniques can be applied to the 3D models in Blender to allow for optimal game performance and are they effective for the optimization of the serious game?
- What optimization methods can be implemented for the development of the serious game in Unity 3D and do they ensure satisfactory player experience?

3.2 Hypothesis/Subject

Choosing a game engine and graphics program suitable to the developer's experience and the development of a serious game, combined with optimization techniques would allow for good overall performance of the game.

3.3 Research gathering methods/Approaches

The research conducted for this paper comprises of journals, conference papers and books related to the optimization of a serious game. The methods chosen for the development were based on literature, with the intent of finding a suitable methodology for creating a serious game with high quality 3D models, optimized both in the graphics program and in Unity 3D.

3.4 System Architecture

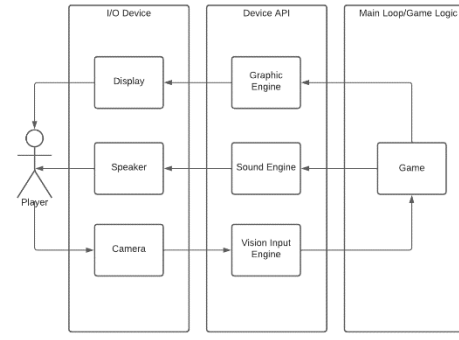

Figure 1: System Architecture Diagram

Components

- Graphics Engine - the software which in association with an application program helps to draw graphics on the player's computer's display device. In this case, Unity engine.
- Sound/Audio Engine - the component that consists of algorithms for dealing with sound and inbuilt programs embedded in the game. For the game, FMOD is used to implement the sounds.
- Rendering & Vision-Input Engine - Unity supports DirectX 11 (DX11) and OpenGL Core graphics APIs. For this project, DirectX 11 will be used as its enabled on default.
- I/O Devices - mouse, keyboard, speaker, monitor etc.)

3.5 Game design

In the game the player plays as an intergovernmental committee formed by the General Assembly of the United Nations to stop global warming by storing carbon extracted from the atmosphere in pykrete bergs and transporting them to Antarctica's coastline.

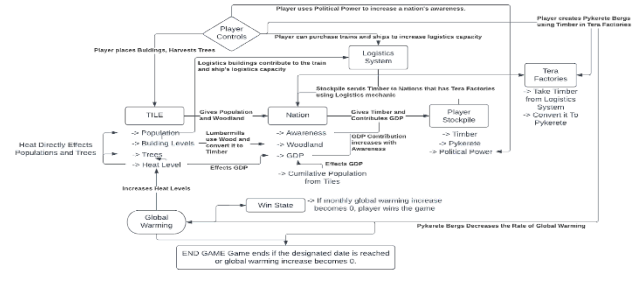

Figure 2: Game Flow

There are 3 layers of simulation in the game:

- Global Warming simulation
- Population, GDP, Politics and GDP Contribution

simulation
- Pykrete production and transportation simulation

Players use tools to produce pykrete production - they grow different species of trees on different geographic locations and transport them via logistics to Antarctica.

The game world is the real world divided into different tiles, which are the nations. Each nation represents a collection of countries or in some cases bigger nations like Russia, China, and USA.

3.6 Tasks

- Describe the optimization approaches taken for the project and analyze their efficiency.
- Apply the optimization methods suitable for the development of the serious game.
- Give suggestions prior to future work

3.7 Technical Requirements

Delivery platform

The game was initially designed to be built for a 1080x1920 display resolution. However, it is planned to be suitable for any resolution, to better accommodate the client's requirements.

Hardware requirements

To run the game, a Windows or MAC device will be required.

Software requirements

The project will be built in Unity 2022.1.16f1 version as the computers on the university campus support it. This would assist in testing the game and showing progress to the client.

3.8 Technical Implementation/Procedure

Asset Creation and showcase

Weber and Penn (1995) introduced a system to model trees using a parametric approach. Their method is implemented in Blender as a famous plugin named Sapling Tree Gen, which was used for the initial creation of the trees for this project. However, the polygon (faces) count of the trees was very high (Figures 3, 4, 5).

Figure 3: Redwood tree

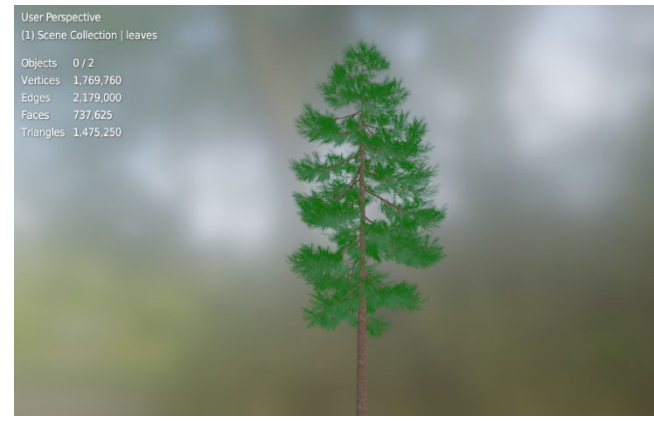

Figure 4: Pine tree

Figure 5: Willow tree

As this would have caused lag due to the project's nature of showing many assets on the screen at the same time, it was chosen to recreate the high polygon trees with low polygon ones, starting from scratch and do the same for the train station, pykrete factory and port models. Polygon count can be seen in Figure 6, 7, 8, 9, 10, 11, 12.

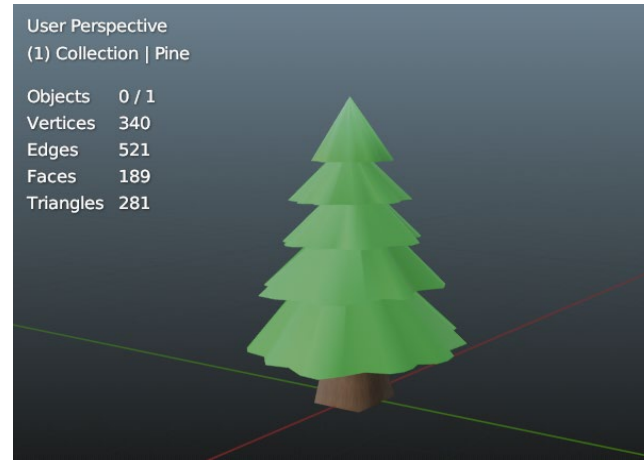

Figure 6: Pine tree

Redwood tree was modelled similarly to the pine tree (Figure 6).

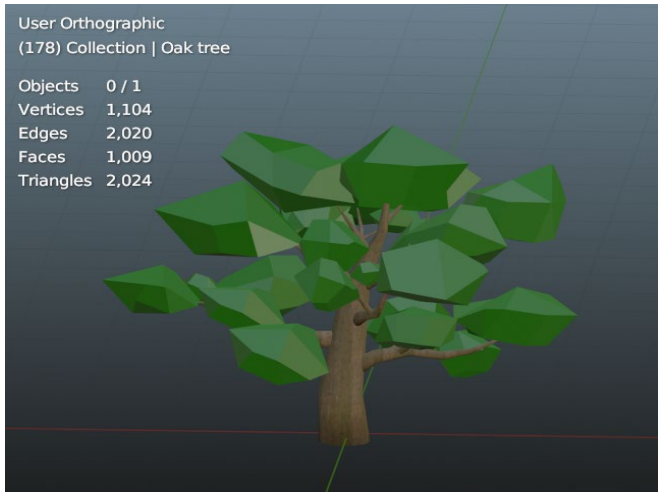

Figure 7: Oak tree

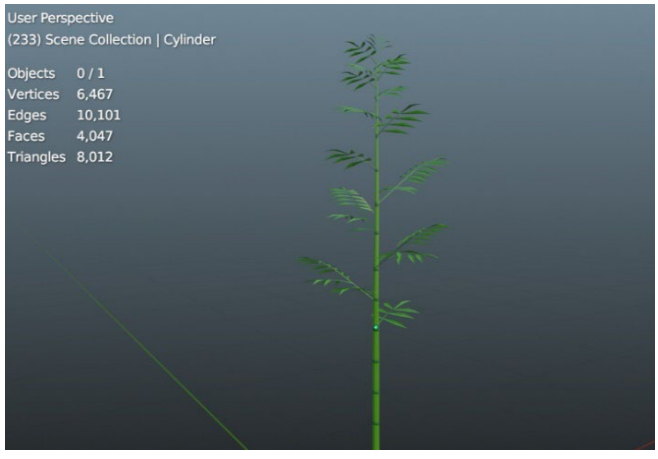

Figure 8: Bamboo model

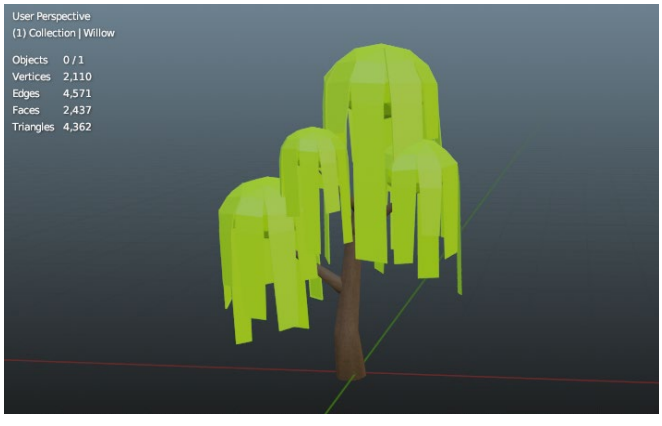

Figure 9: Willow model

For each model, the same modeling methodology was used. The models creation and optimization consisted of three stages - Pre-modeling, Modeling and Post modeling.

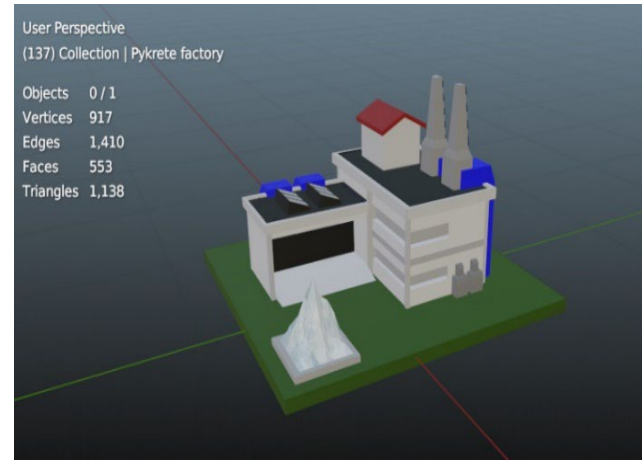

Figure 10: Pykrete factory model

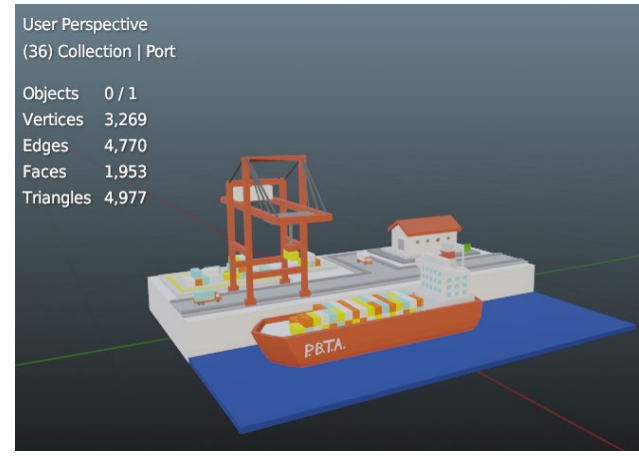

Figure 11: Port model

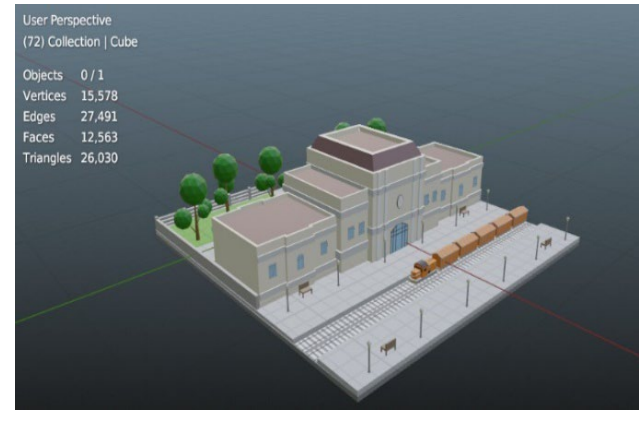

Figure 12: Train station model

Figure 13: World map model

The level design of the game comprises a world map created in Blender (Figure 13) using a high poly SVG map containing data on each country. The map was later reduced in poly count by the team and countries (separate meshes) were grouped into nations to ease player interaction.

Additionally, 2D UI sprites were created for the user interface. Train and ship models were separated from the models and were added in the game as logistics assets. In the next sections the methodology for creation and optimization of the assets used in the game is described, for Blender and Unity 3D.

Pre modeling stage

Before the modeling stage, it is required to have sufficient data to create a model. In this case, images were used, which were examined from various angles (like top, front and side). Once the image was approved, it was added in Blender and the modelling process was started.

Modeling

In this stage the actual modeling was done, along with checking the basic functionality and optimization of the mesh. The modeling is implemented with basic shapes like square, circle, rectangle to keep the mesh to a low polygon count, while using the image as reference. Comparison is made from different angles to examine if the model matches the image.

The next step was functionality check - to check if the capability of the object serves the purpose it is designed for. In this step it is checked whether the model can perform its function properly. In creation of objects where the mesh starts/ends, there could be multiple end/start vertices. This can be rectified by removing the duplicate vertices and reducing the number. Furthermore, more/undesired number of vertices would lead to increase in the file size, render time and could show display issues with the faces in the game engine. Consequently, if there was an issue with double vertices, redundant vertices were removed and the correct objects were sent ahead to the next step of post modeling.

Post Modeling

At the end of the production stage, a functional and optimized model is created. To reduce draw calls, it is recommended to use a single texture for all materials used in the 3D model (Koulaxidis, 2022). Figure 14 shows textures baked in Blender using the Diffuse method to map each part of the mesh to a 2D texture using UV mapping.

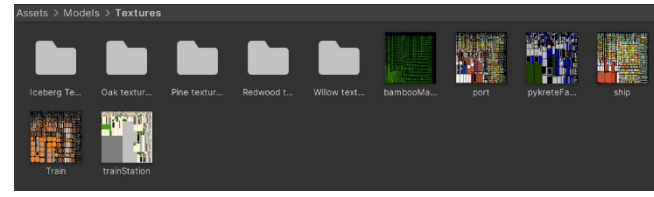

Figure 14: 2D Textures baked in Blender (Screenshot made in the Unity 3D project folder)

The texture maps used for the markings of the texture were created with maximum details and take up as much space as required to avoid errors with the color mapping. Meshes or objects from the same mesh were combined into a single object, for example, the trunk and the leaves were a separate object initially in the model (Figure 15), but later they were combined as one object in Blender to reduce the number of draw calls (Figure 16).

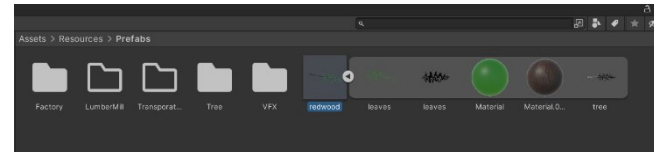

Figure 15: Low performance high polygon model containing separate objects

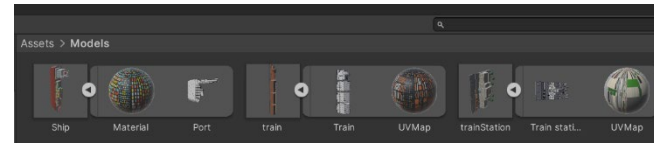

Figure 16: Examples of optimized low polygon assets

The final functionality check involves ensuring the model is shown correctly in the game and is rendered in a compatible format by packing all the necessary components along with it. The model is also checked for the optimization of the mesh, vertices, textures etc. before the last stage of packing.

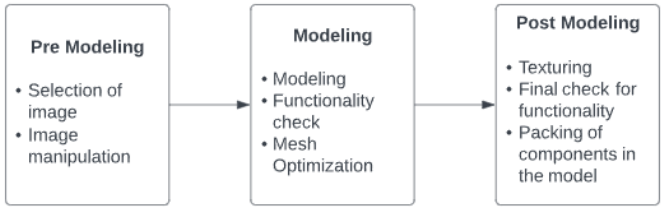

Figure 17: Details of the Modeling process

3.9 Batching

For this project, static batching was applied on the nations, the tiles and the auxiliary materials of the map. By doing this, batches of drawing requests for static objects were combined, resulting in the reduction of the total number of drawing request batches. The objects that this approach was applied to share the same material and are immobile (Figure 18).

Figure 18: Example of static batching application

3.10 Stakeholder decisions and engagement

The client's decisions have affected the overall design of the project. At the beginning of the project's development, there were two game design ideas:

- Smaller scale, resource-management, and transportation game for pykrete, that doesn't include a world map.
- A global simulation/game, which includes a global warming simulation, grand scale pykrete production management and global transportation systems.

The client decided that the second option would be more suitable, and afterwards the project implementation began, followed by regular meetings in which progress was presented, feedback was received and new features were discussed. Information on the technical data and videos of project progress were exchanged by email every two weeks.

3.11 Future work

Due to the short amount of time this project was developed for and the large number of project tasks, a limited number of 3D optimization techniques were considered. In the future, to further enhance player experience, it is recommended to apply dynamic batching to moving 3D objects such as boats, ships, trains in the game.

The key techniques required for the optimization of the application were adopted, and it is recommended for the rest of the assets used in the game to be created according to the client's requirements and to be optimized, in order to achieve optimal performance.

4. Results

4.1 Technical implementation/Research questions outcome

The results showed that the research questions were partially answered - the optimization techniques used both in Blender and in Unity 3D were effective for the development and optimization of the project. Low polygon 3D models made in Blender using the methodology outlined showed significant reduction in file size (Figure 20), compared to the high polygon ones (Figure 19). This resulted in faster rendering time in the game. Additionally, the number of draw calls was reduced by the texture optimization methods applied (Figure 16). This reduced the workload on the GPU and made the game playable on devices with lower GPU specifications. However, the recommended device for the game would have a dedicated graphics card, as the game utilizes Unity's universal render pipeline (URP) for increased graphical fidelity.

3D Model	FBX File size	Polygon Count
Pine	97.6mb	1,072,679
Redwood	59.4mb	751,015
Willow	41.2mb	550,263

Figure 19: High Poly Models Statistics

3D Model	FBX File size	Polygon Count
Pine	32kb	189
Redwood	27kb	154
Willow	137kb	2,437

Figure 20: Low Poly Models Statistics

While playing the game on a personal device, it is noticeable that the frames per second are steady, mostly around 60 FPS, and on moving around the map/spawning objects can drop to 42 FPS. There is a high number of batches as seen in the statistics in the Unity editor (Figure 21). It must be noted that the batching technique uses more memory than other techniques for combining batches of drawing requests, so it may not always be the best option. For example, on devices with limited memory, it is not a good idea to use static batching technology if there is a high number of assets.

Figure 21: Game statistics displayed in Unity 3D

4.2 Evaluation approaches

One of the approaches used for the evaluation of the prototype is the functional testing conducted by the stakeholders. The feedback received included improvements on the user interface, the loading screen and the game design, which were taken into account and worked on for the final version of the prototype. There was overall satisfaction with the game's performance and the 3D modelling assets.

For the final build, the prototype was tested by the entire team. The user interface was made more user-friendly by applying borders to the panels and buttons with custom made fonts. The game's performance and visuals were evaluated on a variety of devices with different resolutions, and if issues arose, they were assigned to the relevant team member responsible for the task.

5. Conclusion

The research conducted has a significant importance as the prototype was built in an effective way, based on various optimization techniques which were studied by performing analysis on Unity 3D tools and Blender optimization approaches. After the implementation of the techniques identified in Unity 3D such as static batching, limitations were found, such as the heavy reliability of the prototype on the memory of the device. However, the prototype performs well on most devices and the client was satisfied as it works well on a desktop device, which was the primary target.

The overall aim of the report was satisfied, and the project was completed successfully. By researching 3D modeling optimization techniques and applying them effectively from the literature review, the serious game "Pykrete bergs to Antarctica" achieved a good overall performance based on the client's feedback and received positive response on the designed game assets.

In conclusion, this project has fulfilled its main aim of preventing climate change and promoting pro-environmental behaviors through a serious game and has provided satisfying user experience as per feedback from the client. The research questions in the Introduction section were answered, the hypothesis was challenged, but satisfied in comparison to the client's requirements.

References

Arranged in chronological order.

[1] Weber, J., & Penn, J. (1995). Creation and rendering of realistic trees. In Proceedings of the 22nd annual conference on Computer graphics and interactive techniques, pp. 119-128. Available from: https://dl.acm.org/doi/pdf/10.1145/218380.218427 [Accessed 19 May 2023].

[2] Pokorný, P. (2010). The creating and preparing 3d graphics models to use them in the real-time application. In Annals of DAAAM and Proceedings of the International DAAAM Symposium. Danube Adria Association for Automation and Manufacturing, DAAAM. Available from: https://www.daaam.info/Downloads/Pdfs/proceedings/proceedings_2010/22091_Annals_1_head.pdf [Accessed 19 May 2023].

[3] Dere, S., Sahasrabudhe, S., & Iyer, S. (2010). Creating open source repository of 3D models of laboratory equipments using Blender. In 2010 International Conference on Technology for education pp. 149-156. Available from: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5550044 [Accessed 19 May 2023].

[4] Urry, J. (2015). Climate change and society, Palgrave Macmillan UK, pp. 45-59 Available from: https://link.springer.com/chapter/10.1057/9781137269928_4 [Accessed 19 May 2023].

[5] Wilkinson, P. (2016). A brief history of serious games. In Entertainment Computing and Serious Games: International GI-Dagstuhl Seminar 15283, Dagstuhl Castle, Germany, July 5-10, 2015, Revised Selected Papers. Springer International Publishing, pp. 17-41, Available from: https://link.springer.com/chapter/10.1007/978-3-319-46152-6_2 [Accessed 19 May 2023].

[6] Lehtola, A. (2018). Optimizing Unity Projects. Available from: https://www.theseus.fi/bitstream/handle/10024/150040/Lehtola_Aleksi.pdf?sequence= [Accessed 19 May 2023].

[7] Nzetchou, S., Durupt, A., Remy, S., & Eynard, B. (2019). Review of CAD visualization standards in PLM. In Product Lifecycle Management in the Digital Twin Era: 16th IFIP WG 5.1 International Conference, PLM 2019, Moscow, Russia, July 8–12, 2019. Springer International Publishing, pp. 34-43, Available from: https://link.springer.com/chapter/10.1007/978-3-030-42250-9_4 [Accessed 19 May 2023].

[8] Turpeinen, M. (2020). A Performance Comparison for 3D Crowd Rendering using an Object-Oriented system and Unity DOTS with GPU Instancing on Mobile Devices. Available from: https://www.diva-portal.org/smash/get/diva2:1467022/FULLTEXT01.pdf [Accessed 19 May 2023].

[9] Douglas, B. D., & Brauer, M. (2021). Gamification to prevent climate change: A review of games and apps for sustainability. Current opinion in psychology, 42, pp. 89-94. Available from: https://www.sciencedirect.com/science/article/abs/pii/S2352250X21000555?casa_token=yhK-skYhXq4IAAAAA:wcJkY83kaCPPNxQUp-cSj6y3pkGjGorFf722QLYaz-

blVM6Jy2AucV5cnq1EnfiRVpvIKrivKDA [Accessed 19 May 2023].
[10] Narkilahti, A. (2021). Unity-optimointi. Available from: https://www.theseus.fi/bitstream/handle/10024/493440/UNITY-OPTIMOINTI.pdf?sequence=2 [Accessed 19 May 2023].
[11] Jitendra, M. S., Srinivas, A. S., Surendra, T., Rao, R. V., & Chowdary, P. R. (2021, October). A study on game development using unity engine. In AIP Conference Proceedings. AIP Publishing LLC. Available from: https://pubs.aip.org/aip/acp/article/2375/1/040001/949991/A-study-on-game-development-using-unity-engine [Accessed 19 May 2023].
[12] C. Vohera, H. Chheda, D. Chouhan, A. Desai and V. Jain, "Game Engine Architecture and Comparative Study of Different Game Engines," (2021) 12th International Conference on Computing Communication and Networking Technologies (ICCCNT), Kharagpur, India, 2021, pp. 1-6, Available from: https://ieeexplore.ieee.org/abstract/document/9579618 [Accessed 19 May 2023].
[13] Masopust, L., Pasewaldt, S., Döllner, J., & Trapp, M. (2021). Integration of Image Processing Techniques into the Unity Game Engine. In IMPROVE, pp. 137-144, Available from: https://www.researchgate.net/profile/Matthias-Trapp-2/publication/351282512_Integration_of_Image_Processing_Techniques_into_the_Unity_Game_Engine/links/608fba7b299bf1ad8d72acad/Integration-of-Image-Processing-Techniques-into-the-Unity-Game-Engine.pdf [Accessed 19 May 2023].
[14] Koulaxidis, G., & Xinogalos, S. (2022). Improving Mobile Game Performance with Basic Optimization Techniques in Unity. Modelling, 3(2), pp. 201-223, Available from: https://www.mdpi.com/2673-3951/3/2/14 [Accessed 19 May 2023].
[15] N. P. Singh, B. Sharma and A. Sharma, "Performance Analysis and Optimization Techniques in Unity 3D," (2022) 3rd International Conference on Smart Electronics and Communication (ICOSEC), Trichy, India, 2022, pp. 245-252, Available from: https://ieeexplore.ieee.org/abstract/document/9952025?casa_token=QU8S4otzUPQAAAAA:A0yq5GkYoVAeA23ui0y2SGM45McY7l9mZfh5UzjJwr0XosrGfOwk319bmjaiO-rxNrOjIBgMTQ [Accessed 19 May 2023].