3D Procedural Modeling and Game Generation of the UJI Library

Lluc Gauxachs Sanz

Final Degree Work
Bachelor's Degree in
Videogame Design and Development
Universitat Jaume I

July 1st, 2021

Supervised by: Michael Gould.

To my parents, whose inspiration brought me here.

ACKNOWLEDGMENTS

First of all, I would like to thank my Final Degree Work supervisor, Michael Gould, for his dedication and support during the whole project.

Secondly, I want to thank Juan Camilo Gómez for the time he spent helping with licenses and other problems. I also would like to thank Danilo Palermo for his help on the initial steps of the project.

Finally, I'd like to thank Sergio Barrachina Mir and José Vte. Martí Avilés for their inspiring LaTeX template for writing the Final Degree Work report, which I have used as a reference writing this report.

ABSTRACT

This project aims to be an explanation of the process, the successes, the failures and all the obstacles that have been faced during the last months in order to create a 3D model of the Universitat Jaume I library in CityEngine, a commercial three-dimensional modeling software which specializes in the generation of 3D urban environments.

The final result includes 3D general exterior and interior visualization of the UJI library as well as the conversion of the model into a game engine format to support a simple horror game inside the UJI library.

CONTENTS

CHAPTER 1

INTRODUCTION

Contents

The fundamental point of this chapter is to state the objectives of the project, the work motivation and the initial state from which it was started.

1.1 Work Motivation

There are many different ways to create digital 3D models: polygonal modeling, sculpting, NURBS modeling, photogrammetry, etc. However, not all of them have been practiced during my learning process. Procedural modeling using CityEngine is a very interesting way to create massive environments in a short time. The possibility to learn how to use such a powerful tool was what drive us to develop this project.

CityEngine is a commercial three-dimensional modeling software application which specializes in the generation of 3D urban environments using a procedural modeling approach. Learning how to use this software was one of the main motivational points as it provides new skills and abilities to 3D artists.

1.2 Objectives

The 3D modeling of the library is part of a larger project already started by the GEOTEC research group at the University Jaume I: the Virtual Smart Campus UJI.

Virtual Smart Campus UJI is a map-based view of the interior and exterior buildings of the university campus that enables employees, students, and visitors to locate an area of interest and review information stored in the human resources and facilities management database.

However, this project will only be focused on the creation of an exterior and interior 3D model of the UJI library in CityEngine, as the main objective is to be able to import the result into Unity to develop a simple first-person horror videogame.

In conclusion, these are the most important goals to achieve during the development of the whole project:

- Become well-versed in CityEngine.
- Learn how to use different types of data to generate 3D models in CityEngine.
- Create a 3D model of the interior and exterior of the UJI library.
- Develop a first-person horror videogame in Unity taking into account all the aspects learnt studying Videogame Design and Development, such as 3D art, conceptual design, software engineering, narrative design, programming, etc.

1.3 Environment and Initial State

After completing the installation of CityEngine, Juan Camilo Gómez provided several AutoCAD files and a project file called 4thVersionUji, where a first start point could be found.

Nevertheless, the 3D model available in that project was spoiled by various errors. The normals were flipped, the planes were out of place, some of them were gone, others were crooked, some textures were missing, etc.

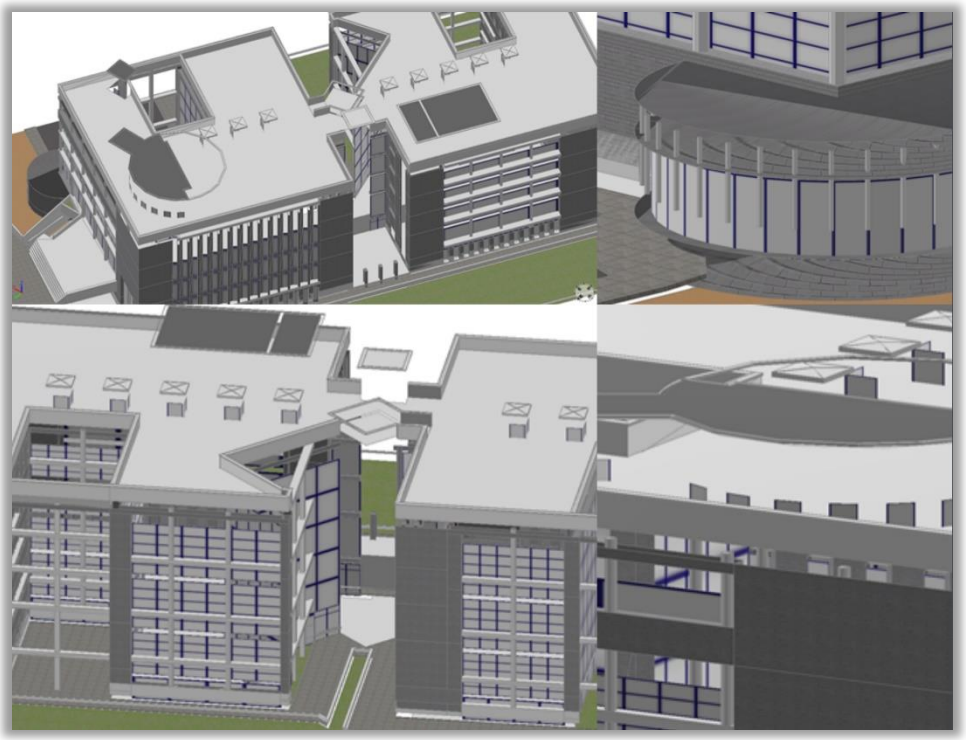

Figure 1: Example of the initial state of the library and its errors

To sum up, the initial state of the project was a rough sketch of the exterior UJI library and quite a few blueprints of all floors in AutoCAD.

During the whole project I had support via email from Juan Camilo and Michael Gould for any question or problem.

CHAPTER 2

PLANNING AND RESOURCES EVALUATION

Contents

This chapter will describe all the information regarding the technical part of the work such as planification, resources evaluation and more.

2.1 Original Planning

Due to different problems during the development of the project, the original organization differs from the final planning. In the next page, the initial planification diagram can be seen.

Even though the main structure of the final one is similar, the periods of time dedicated to each task and the dates were modified as it was decided to turn the project in in July.

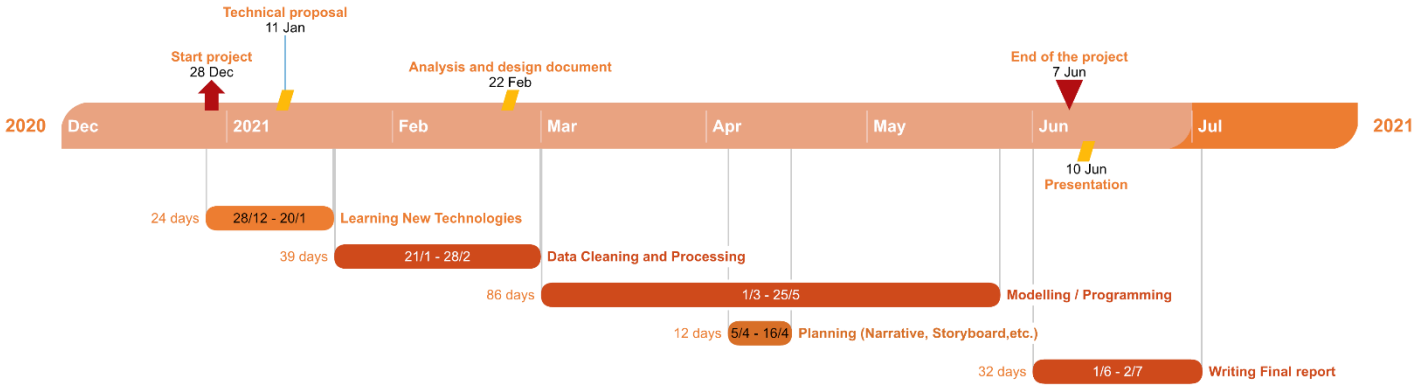

Figure 2: Initial Gantt Diagram

As it can be seen in the diagram, there are five main tasks. Each of them has a certain weight depending on the hours that have to be worked. The next chart pretends to show schematically the weight of each task in the first steps of the project.

TASKS	DAYS	WEIGHT (%)
Learning New Technologies	24	10
Data Cleaning and Processing	39	20
Modelling / Programming	86	60
Planning the Game	12	5
Writing Final Report	163	5

In the next section, the work carried out in each task will be described, as well as the final distribution of weights and planification.

2.2 Final Planning

The planning is a very important task in the project to keep the workflow. In this section, a detailed analysis of how the development of the project has finally been divided is shown.

First of all, let's describe what is going to be done in each of the tasks:

- **Learning New Technologies (45 hours)**: Introduction to CityEngine and ArcGIS Pro. Different types of tutorials were completed to get familiarized with the interface, controls and other parameters of each program. Later on, it was necessary to learn SketchUp as the modelling of the library in CityEngine was not feasible.

- **Data Cleaning and Processing (60 hours)**: The data provided by Juan Camilo Gómez needed to be processed and cleaned. However, the time dedicated to this task was only useful for the exterior visualization of the library because the blueprints of the interior floors could not be processed properly.

- **Planning the Game (30 hours)**: Development of the Game Design Document and conceptual, narrative and artistic design of the game.

- **Writing Final Report (45 hours)**: Writing of the final report of the project and other documents such as the technical proposal.

- **Modelling / Programming (120 hours)**: Creation of the exterior and interior 3D model of the library and programming of the mechanics of the game, the UI and other necessary components.

Below these lines, the final Gantt Diagram of the project can be seen.

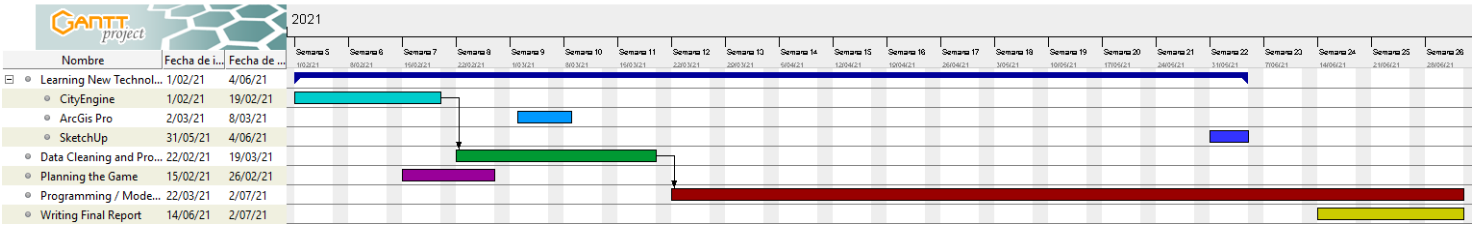

Figure 3: Gantt Diagram

Finally, the distribution of weights has been the following:

TASKS	WEIGHT (%)
Learning New Technologies	15
Data Cleaning and Processing	20
Modelling / Programming	40
Planning the Game	10
Writing Final Report	15

2.3 Resource Evaluation

In this section, an estimated evaluation of the resources needed to develop the project is made. Moreover, the approximated cost in Euros is calculated taking into account the hardware and software needed.

On the one hand, the **hardware** needed to develop this project is a computer with the following specifications:

- **CPU**: AMD Ryzen 7 4800H (4.2GHz)
- **GPU**: NVIDIA GeForce RTX 2060 6GB
- **RAM**: RAM de 2x 8GB SO-DIMM DDR4-3200

Although the project could also have been done in a computer less powerful, it is very recommended to not go lower than the specified requirements to assure a smooth development as CityEngine accustoms to collapse. The estimated cost of these components is 1000€.

On the other hand, the essential software to work in this project is the following:

- **CityEngine** [1]: ArcGIS CityEngine is a commercial three-dimensional modeling software application developed by Esri R&D Center Zurich which specializes in the generation of 3D urban environments. Using a procedural modeling approach, it supports the creation of detailed large-scale 3D city models. There are different types of licenses but the cheapest costs 3300€.

- **ArcGIS Pro** [2]: ArcGIS Pro is another Esri application specialized in the data visualization, advanced analytics, and authoritative data maintenance in 2D, 3D, and 4D. You can access to this software having the CityEngine license.

- **Unity** [3]: cross-platform game engine developed by Unity Technologies. Unity is suitable to develop 3D, 2D, VR and AR experiences for multiple platforms. This game engine was chosen because it is free, has a lot of documentation and tutorials online and also has a great variety of free tools to choose from the asset store.

- **Mixamo** [4]: Mixamo is a free website that allows you to apply animations to your 3D models and export them into Unity or other applications. Very useful in order to add animations to the player and NPCs.

- **Microsoft Word** [5]: Word is a graphical word processing program that users can type with. Its purpose is to allow users to type and save documents. Similar to other word processors, it has helpful tools to make documents. It has been used to create the final report and other documents of the project. Its license costs 120€.

- **Trello** [6]: Trello is a free list-making application used for planning and managing projects. It is very useful to keep track of the tasks to do, the tasks already done and the ones that are in process.

- **GitHub** [7]: GitHub is a free program that provides hosting for software developing. It was used in order to keep a copy of the project and avoid losing progress.

- **SketchUp** [8]: SketchUp is a 3D modeling computer program for a wide range of drawing applications such as architectural, interior design, landscape architecture, civil and mechanical engineering, film and video game design. [] The version used has a cost of 600€ but there is a free version online.

- **Blender** [9]: Blender is a free and open-source 3D computer graphics software toolset used for creating animated films, visual effects, art, 3D printed models, motion graphics, interactive 3D applications, virtual reality, and computer games.

Taking into account the hardware and software used, an approximated amount of 5.020€ would be needed. However, the price could be easily reduced by prescinding of some not-essential software.

CHAPTER 3

SYSTEM ANALYSIS AND DESIGN

Contents

This chapter describes briefly the key elements of the final videogame, such as narrative, artistic and conceptual design. Straightaway, we will dig deeper into the requirement analysis, system design, system architecture and the interface design of the videogame.

3.1 General Description of the Game

Before starting with the requirement analysis, it is necessary to describe what we aim to achieve. In this case, we want to create a first-person horror videogame, but which are the main characteristics of this game?

- **Story Abstract**

The protagonist of the game is Ambrose Cooper, a student from Jaume I University who falls asleep in the library during his intense studies for the final exams. When Ambrose wakes up in the middle of the night, he realizes that he has been locked up alone inside the library.

After that, he decides to find another way out but then some unusual events make him feel that he is not alone anymore.

- **Background**

The videogame pretends to be a representation of a bad dream caused by the anxiety and the stress that final exams cause to lots of students. That is why the tasks/objectives assigned to the player will be related with the exams Ambrose must take if he survives the night.

- **Game Description**

The player job is to unravel the mystery of what is happening. Clues and information will be progressively given through collectibles and monologues as Ambrose talks to himself in order to calm down the nerves.

By all means, the protagonist must avoid the dangers awaiting in the library because he won't be able to defend himself properly as he won't have any kind of weapon. However, he will have the ability to hide in certain places and to distract the monsters using several objects of the environment (e.g., books).

- **Concept Art**

Below these lines, a reference of how the atmosphere of the environment wants to be set in the videogame.

Figure 4: Environment example
Source: https://cutt.ly/HlhQrJK

- **Core Mechanics**

Hiding mechanics: The hiding in *The Library Agony* will be based much around its survival horror counterpart *OutLast*. Like *OutLast* premise, players will have to avoid battles entirely because of the supernatural enemies they will be facing. That is why some locations will be set as "safe areas" where players must return in case they are being chased by an enemy. Those safe areas will correspond to lights that will remain open the whole night.

Other areas will be set as "hiding spots", where players will be able to hide from enemies that are in alert, but still haven't detected them. Therefore, those hiding spots will not be safe if the enemies already detected the player. Some of those hiding spots will be bathrooms or study cubicles.

Noise mechanics: Ambrose must be aware of everything around him and be careful not to make any kind of noise that could draw the enemies attention. In that case, the player will have to hide or go to a safe area to survive.

Players will also be able to distract enemies (if they are in the appropriate state) by throwing objects to the location the player wants them to investigate. This will be useful to get access to certain areas or to solve some puzzles.

Puzzle mechanics: puzzles are going to be very simple. They mainly will focus on the acquisition of an object set in a certain location in order to get access to another area of the game.

- **Game Flow**

The game flow in The Library Agony will be quite linear. Players will be given a portion of the narrative and then be given a reason (objective) in that narrative to explore a certain section of the library. On entering the section, they will encounter 3 things consistently:

- Enemies
- Puzzles
- Narrative elements

Each time a new objective is set to the player, the difficulty will be increased by adding more enemies in the scene. The same scheme will be repeated until the mystery of where those weird creatures come from is solved.

3.2 Requirement Analysis

The last section was a description of the game and, at the same time, a requirement elicitation. In this part, the requirements will be refined and documented in a formal manner to ensure clarity, consistency and completeness.

It is necessary to know that there are two types of requirements: functional and non-functional requirements. The first ones correspond to statements of actions that the game should provide and how the game should react to particular inputs or how the system should behave in particular situations. Nevertheless, the second ones define the quality

attribute of a game, for example, how fast does the game load? Non-functional requirements basically are constraints on the service and functions offered by the system to ensure reliability, availability and in general, a good user experience.

3.2.1 Functional Requirements

Once the difference between functional and non-functional requirements is clear, it is time to start defining the functional requirements:

- **R1**: The player can start the game.
- **R2**: The player can adjust the volume of the game.
- **R3**: The player can quit the game.
- **R4**: The player can move through the level.
- **R5**: The player can pause the game.
- **R6**: The player can pick up items.
- **R7**: The player can pick up books.
- **R8**: The player can throw books to distract enemies.
- **R9**: The player can read the dialogues when necessary.
- **R10**: The enemies have different states and switch between them.

3.2.2 Non-Functional Requirements

The non-functional requirements will be the next ones:

- **R11**: The game will be playable in PC.
- **R12**: The game will be as realistic as possible.
- **R13**: The mechanics will be easy to learn.
- **R14**: The loading time will always be less than 2 seconds.
- **R15**: The UI elements will always be perfectly visible.
- **R16**: The controls will be comfortable.

3.3 System Design

This section presents the logical and operational design of the videogame. To do so, the more suitable diagram is used to describe the different parts of the system design. First of all, let's analyze the functional requirements described above using Use Case charts.

Requirement	R1
Actors	Player
Description	The player starts the game by pressing the *Start game* button
Preconditions	1. The player must be in the *Main Menu*
Normal sequence	1. The player presses the button *Start Game* 2. The camera travels through the environment until it reaches the library main door 3. The system loads the game scene
Alternative sequence	None

Figure 5: Use case chart "Start Game"

Requirement	R2
Actors	Player
Description	The player modifies the game volume
Preconditions	1. The player must be in the *Options Menu*
Normal sequence	1. The player clicks the *Options* button in the *Main Menu* 2. The system takes the player to the *Options Menu* 3. The player modifies the volume using a slider 4. The slider value is applied to the *Audio Mixer* in order to modify the volume of the game
Alternative sequence	None

Figure 6: Use case chart "Volume settings"

Requirement	R3
Actors	Player
Description	The player quits the game by pressing the *Quit* button
Preconditions	1. The player must be in the *Main Menu*
Normal sequence	1. The player presses the button *Quit* 2. The system quits the game
Alternative sequence	None

Figure 7: Use case chart "Quit Game"

Requirement	R4
Actors	Player
Description	The player moves through the level.
Preconditions	1. The player must be in the game scene 2. The player must be outside the *Pause Menu* 3. The player must be outside a hiding spot 4. The character must not be in a dialogue sequence
Normal sequence	1. The player presses the W, A, S, D or arrow keys 2. The player moves in the direction assigned to the pressed button
Alternative sequence	1. The player presses the W, A, S, D or arrow keys 2. The character can not move because it is colliding with another object

Figure 8: Use case chart "Player Movement"

Requirement	R5
Actors	Player
Description	The player pauses the game by pressing the *Esc* key
Preconditions	1. The player must be in the game scene
Normal sequence	1. The player presses the *Esc key* 2. The system pauses the game and displays the *Pause Menu*
Alternative sequence	None

Figure 9: Use case chart "Pause the Game"

Requirement	R6
Actors	Player
Description	The player can pick up items by pressing the E key
Preconditions	1. The player must be in the game scene and outside the *Pause Menu*
Normal sequence	1. The player gets close enough to an interactive object 2. The player presses the interaction key E 3. The object is taken
Alternative sequence	None

Figure 10: Use case chart "Pick up object"

Requirement	R7
Actors	Player
Description	The player can pick up books by pressing the E key
Preconditions	1. The player must be in the game scene and outside the *Pause Menu*
Normal sequence	1. The player gets close enough to a book 2. The player presses the interaction key E 3. The book is taken
Alternative sequence	3.1 The book is not taken if the player already holds a book

Figure 11: Use case chart "Pick Up Book"

Requirement	R8
Actors	Player
Description	The player can throw books by clicking the mouse button
Preconditions	1. The player must be in the game scene and have a book
Normal sequence	1. The player aims to throw the book 2. The player presses the mouse button 3. The object is thrown away
Alternative sequence	None

Figure 12: Use case chart "Throw books"

Requirement	R9
Actors	Player
Description	The player can read the dialogues when necessary
Preconditions	1. The player must be in the game scene and outside the *Pause Menu*
Normal sequence	1. The game “pauses” 2. The dialogue appears 3. The player presses the continue button until it is finished
Alternative sequence	None

Figure 13: Use case chart "Dialogues"

Requirement	R10
Actors	Enemies
Description	The enemies switch their state
Preconditions	1. The player must be in the game and trigger a change of state
Normal sequence	1. The player enters the field of view of the enemy 2. The enemy changes its state
Alternative sequence	1.1 The player makes a sound

Figure 14: Use case chart "Enemies States"

An activity diagram was also designed to have a general idea of how the game woudl work. This diagram can be seen in the next page.

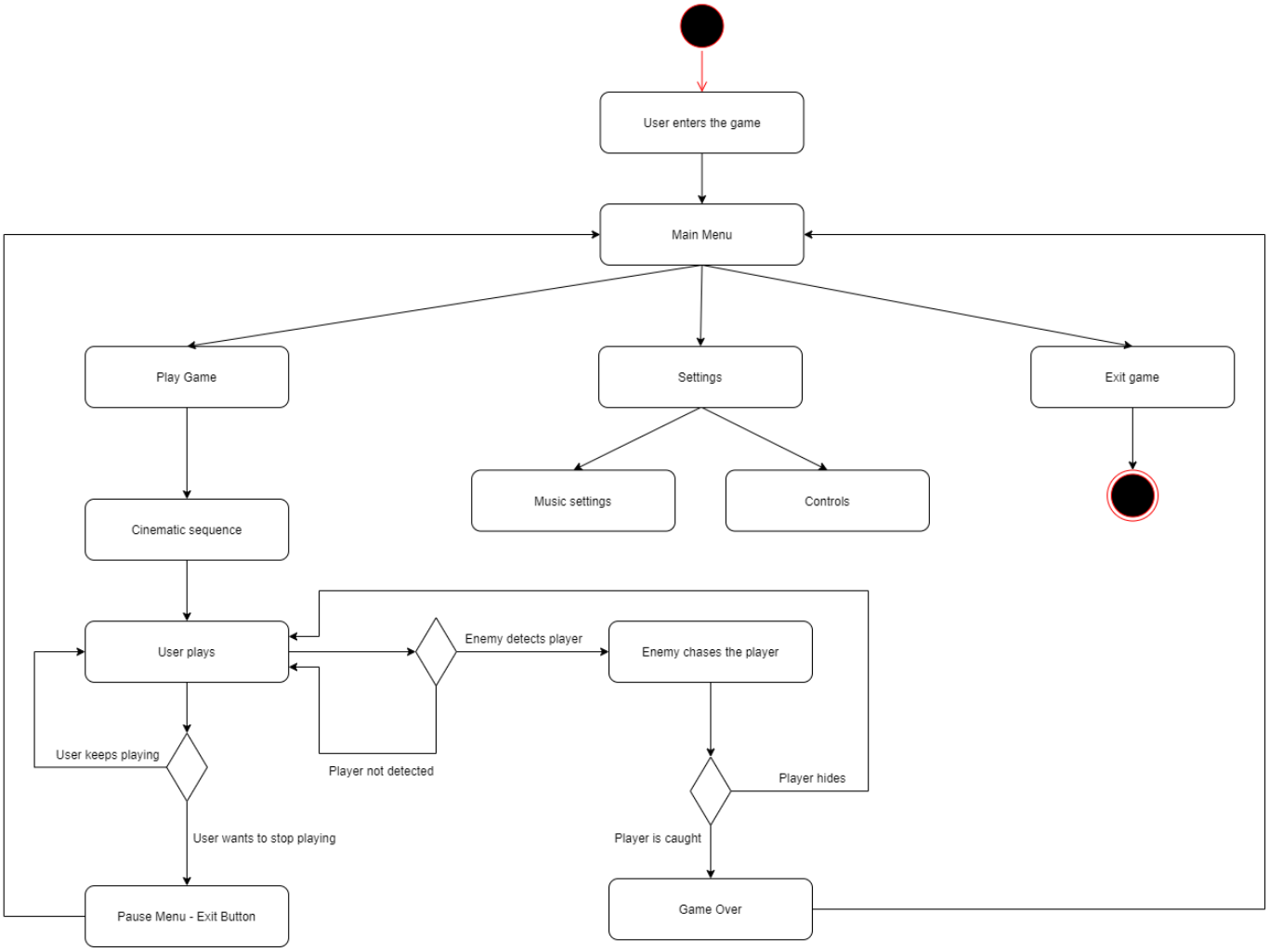

Figure 15: Activity diagram

3.4 System Architecture

The Library Agony was developed in Unity 2020.3.0f1 version, which needs the following minimum system requirements:

- **Operating system**: Windows 7 (SP1+) and Windows 10
- **CPU**: x86, x64 architecture with SSE2 instruction set support
- **GPU**: Graphics card with DX10, DX11, DX12 capabilities.
- **Additional requirements**: Hardware vendor officially supported drivers.

This information was taken from Unity's documentation [10], but it does not offer a guarantee that it will work on all systems that satisfy those requirements.

The game was developed and tested on a computer with these hardware and software features:

- **Operating system**: Windows 10
- **CPU**: AMD Ryzen 7 4800H (4.2GHz)
- **GPU**: NVIDIA GeForce RTX 2060 6GB
- **RAM**: RAM de 2x 8GB SO-DIMM DDR4-3200

If the game is going to be played in a laptop computer, it is recommended to use a mouse for users' comfort.

3.5 Interface Design

The GUI changes depending on the circumstances of the player. First of all, let's take a look at the Main Menu.

It was decided to keep the UI as simple as possible. This is why the buttons are simply created of a round box with transparency and the adequate text above them.

Figure 16: Main Menu UI Design

Once the player starts the game the only elements that appear in the UI are:

- The objectives of the player
- Dialogues when necessary

An example of how objectives will be displayed in the game can be seen in Figure 18. The dialogues are also going to be very simple and will be compound of a box with transparency and the appropriated text to be displayed.

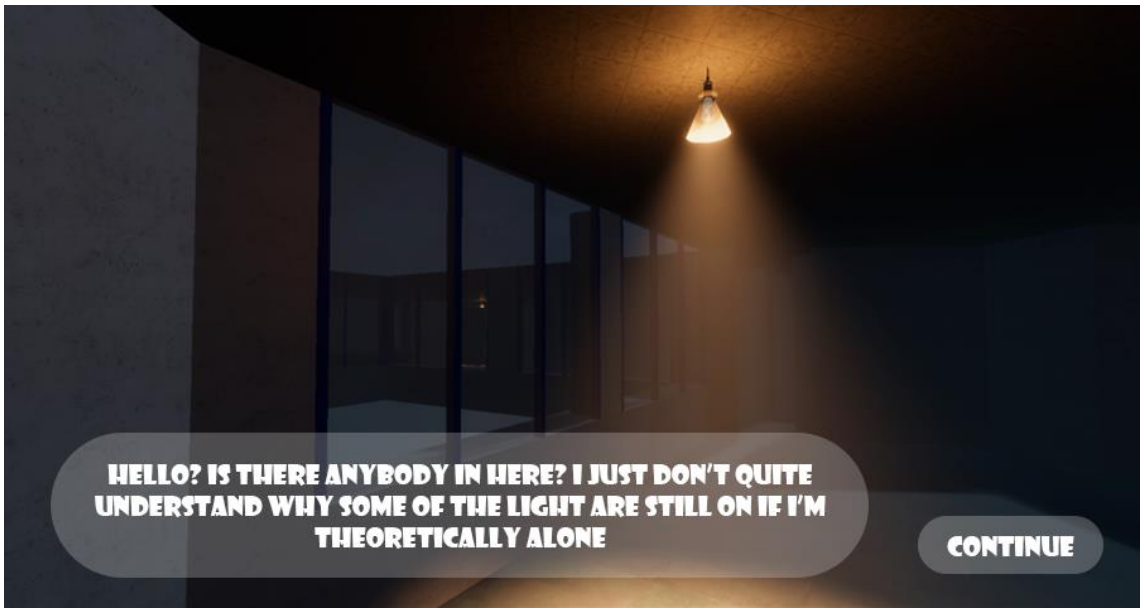

Figure 17: UI Dialogues Design

However, other elements of the game have been implemented in the UI in a more natural way. For example, when the player picks up a book, it appears in the screen as if it was held by the player.

Figure 18: Example of a UI element implemented naturally (the book)

For the purposes of our game, no more GUI elements will be needed.

CHAPTER 4

WORK DEVELOPMENT AND RESULTS

Contents

This chapter is an explanation of how the project has been developed from the start to the end. It also includes an evaluation of the results and how some original ideas were modified during the process due to inconveniences or recommendations of the supervisor.

4.1 Work Development

The work development is going to be explained in chronological order as it is the most organized way to understand the workflow, the problems faced and the solutions taken.

4.1.1 Starting Up in CityEngine

One of the main objectives of this project was to be able to create a procedural model using CityEngine. This is why the first step was to complete all the tutorials available in the ArcGis documentation webpage [11].

After completing those tutorials, the basics about CityEngine were learnt. This includes several ways to import different types of resources into the software, basic shape grammar, polygonal modelling, publication of scenes in the web, a first look to procedural generation of streets and 3D models using CGA rules and more.

Once the essential skills of the program were acquired, the importing of the files provided by Juan Camilo Gómez of the University Jaume I started.

First of all, let's list all the files provided at the beginning of the project:

- **4thVersionUJI**: a CityEngine project with some 3D models of the UJI buildings and textures.
- **AutoCAD Blueprints**: blueprints of all the floors of the library in .dwg format.

The 4thVersionUJI, contained 4 different scenes but only one of them with useful information: ViscaUJI1.cej. In that scene, there were some buildings of the university and one of them was the library. However, it was full of errors (empty windows, missing textures, flipped normal, displaced planes) as it can be seen in Figure 1.

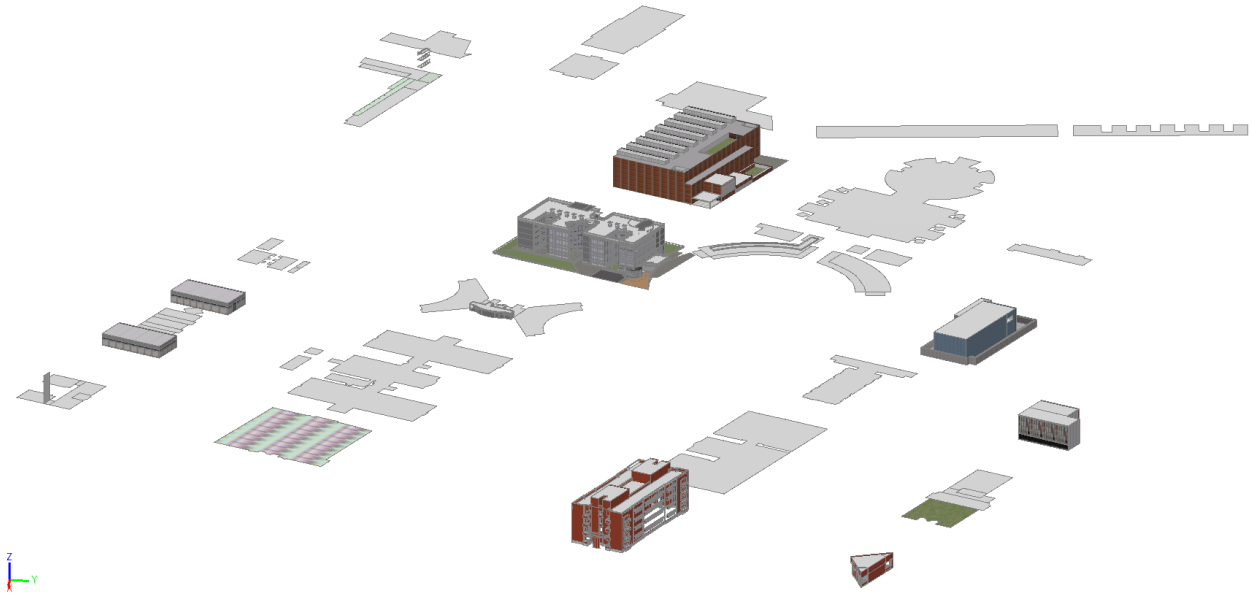

Figure 19: Buildings and shapes of the ViscaUJI1 scene in 4thVersionUJI

Even though an exterior 3D model was already available there, the main idea was to create from the start the whole library. Moreover, fixing all those errors might have taken more time than starting from the initial blueprints.

Then, the AutoCAD blueprints were converted into .dxf format since it is the only similar accepted format for import in CityEngine. However, the result obtained (Figure 21) was very different from the original blueprints (Figure 20).

Figure 20: Blueprint of the floor 0 in .dwg format

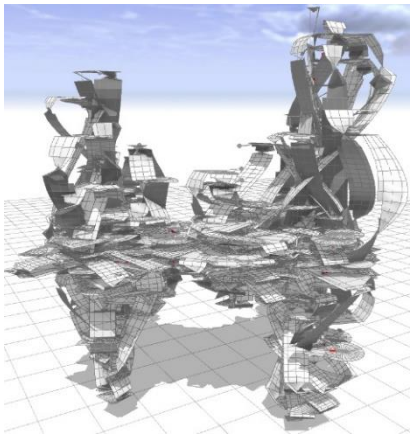

Figure 21: Imported blueprints into CityEngine

Although the figure obtained was quite interesting, it was of no use for us. For a while, different importing options were tried out but the results were always the same or worse. This was the first time the project was stuck.

Nevertheless, the project continued progressing in other areas, for example the narrative of the game and other elements were defined.

4.1.2 Refining the Narrative

In the *Library Agony*, only three characters are relevant to the story. The first one is Ambrose, the protagonist. The second one is Anna, the girl who is connected to the mystery of the weird monsters that appear in the library at night, and the third ones are the monsters (enemies).

- **AMBROSE**

Ambrose is a very diligent student who sometimes exceeds himself more than he can take, what usually causes him a lot of anxiety and stress in some stages. This is exactly what happened to him in the library, when he fell asleep while studying after more than 10 hours in a row.

However, he is also a guy who knows how to enjoy himself, with a great sense of humor and a really sarcastic tone. He really likes to talk to himself as if he was the main character of a movie.

- **ANNA**

Anna Moreno is the antagonist of game. However, that is not known until late in the game, as she is the main pilar of the mystery around those weird creatures.

Anna was also a really studious girl who, ten years ago, got locked up in the library by her "friends" on purpose. She was really preoccupied about an exam and her "friends" were really tired of hearing her concerns at all time. That's why they decided to lock her in one of the study cubicles. That way, "she would be able to study the whole night" they said laughingly. Despite of her pleas not to let her alone there, they did.

What they didn't know was that she was claustrophobic and asthmatic, a literally lethal combination for her. A few minutes after her friends left, she had an asthma attack which led to a slow and painful death by asphyxia.

The next day, the body of the young woman was found by one of the library workers who, instead of reporting the body, decided to use it to blackmail the rector of the university, who wanted to avoid, at all cost, the bad repercussion that would have had the appearance of a dead lady inside one of the university buildings.

That is why the body of Anna Moreno was never found, and of course, the reason why she is still missing. Her friends never knew what happened to her, but they didn't really want to know as they felt guilty of her disappearance.

After all these events, the spirit of Anna remained in the library. During the nights, it shows up trying to collect souls of other students as a way of vengeance.

- **MONSTERS**

The monsters that appear in the library are a corporal representation of those students who Anna has taken with her to the world of the dead, becoming a Wendigo/zombie who serves her.

4.1.3 ArcGIS Pro

After talking to Michael Gould and Juan Camilo Gómez about the problems importing the provided data, they recommended to try it out in ArcGIS Pro, another Esri software specialized in the data visualization, advanced analytics, and authoritative data maintenance in 2D, 3D, and 4D.

During a few weeks, a bunch of ArcGIS Pro tutorials were made in order to start learning this new software [12].

Danilo Palermo, a GIS Analyst and Developer specialized in the processing of cartographic data, tools and 2D-3D solutions and platform apps, arranged an appointment with us to help out in the initial steps of the project. In the online meeting he demonstrated how to geo-localize data from AutoCAD files. However, the files he used where previously prepared to facilitate the process.

Although Danilo Palermo's help was very much appreciated, it was not very useful as most of the process he followed could not be applied to our data.

4.1.4 Trying New Data

As soon as Juan Camilo Gómez and Michael Gould knew that no progress could have been done in the 3D model yet, they provided new data to try out:

- **SmartCampus.gdb**: A geodatabase containing all the Smart Campus data.
- **DatosInteriores.gdb**: A geodatabase containing information about the interior floors of the library.
- **Separated .dwg files of the floors**

We started with the SmartCampus.gdb data. The problem with this file was that, probably because of the huge amount of information it contained, CityEngine was collapsing every time it was tried to be imported. No matter how many times we tried, it was always the same result.

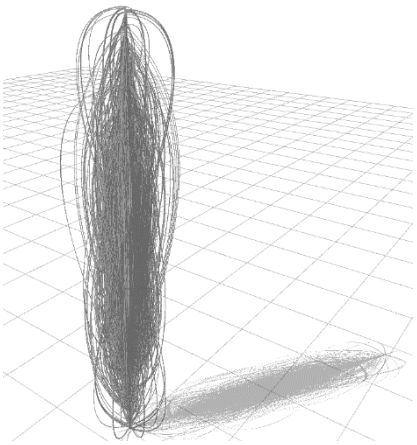

Figure 22: DatosInteriores.gdb data in CityEngine

Next, DatosInteriores.gdb could be imported without any problem. However, the results were, again, pretty far away from what it was expected. The same happened with de separated .dwg files of the different floors. Some of them were converted to .dxf and imported into CityEngine obtaining similar weird results as well.

In the end, the only shapes we could obtain were located in the first AutoCAD file that was shared (Figure 21). If we hid the some of the layers, we could see these shapes:

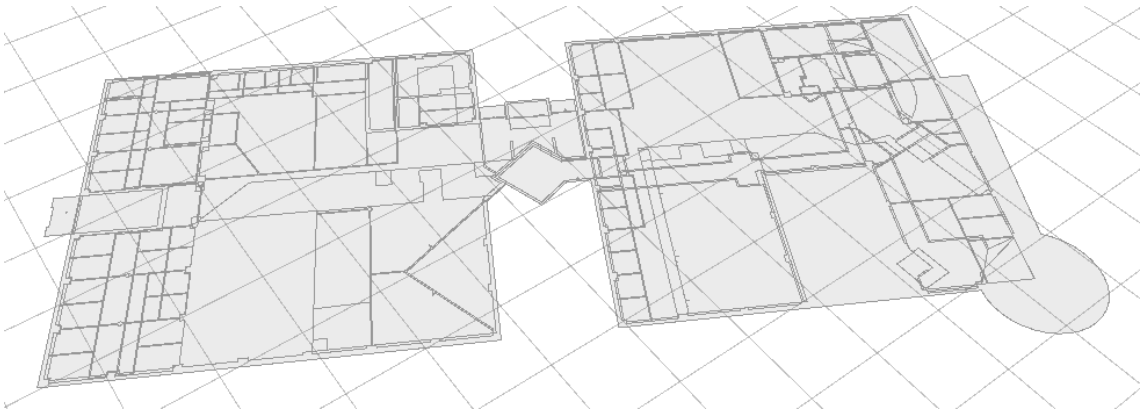

Figure 23: Shapes from the first AutoCAD files

Nevertheless, these shapes corresponded to different floors and were all mixed. It was also useless because it was very difficult to identify which shape corresponded to which floor and in conclusion, very costly to separate them manually.

4.1.5 The Best Solution

At this point, the time was running out so the strategy to face the project needed to be changed. After a lot of thinking it was decided to recover the first 3D library model and fix it to get the exterior part.

About the interior model, the best solution was to create it using a specialized software in architectural and interior design called SketchUp. The downside was that it was another new software to learn.

But what about the procedural modelling? From the beginning of the project, one of the main objectives was to make a procedural model of the UJI library. As the library could not be done procedurally, it was decided to create a city in CityEngine using a procedural approach (see Section 4.1.8).

4.1.6 Fixing the Library

The errors that could be found in the first version of the exterior of the library were mainly seven:

a) Huge amount of displaced planes
b) Some overlapping planes
c) Textures applied incorrectly
d) Missing textures
e) Missing geometry
f) Broken geometry
g) Flipped normals

In order to solve those problems different options were considered. First of all, there is a button in CityEngine called Cleanup Shapes which is supposed to get rid of double vertices, correct geometry errors and more. Although this tool was able to get rid of some of the problems, it was also generating new ones, such as moving vertices to different positions and creating really weird shapes. This is why it was decided to correct the problems one by one using the following solutions:

a) Use the translation tools to move the wrong planes into their correct position
b) Delete the overlapping visible planes.
c) Re-apply textures using the Texture shapes tool.
d) Search for suitable textures and apply them using the Texture shapes tool.
e) Use the polygonal modelling tools to create the missing geometry.
f) Delete the broken geometry and re-model it.
g) Import the model into Unity to see which planes were flipped because an option to visualize normals in CityEngine could not be found.

At the end of the process, a pretty decent exterior 3D model of the library was obtained. Some pictures can be seen in the next pages.

Figure 24: Final 3D model of the library

Next, a few pictures of comparison between the original 3d model and the final one.

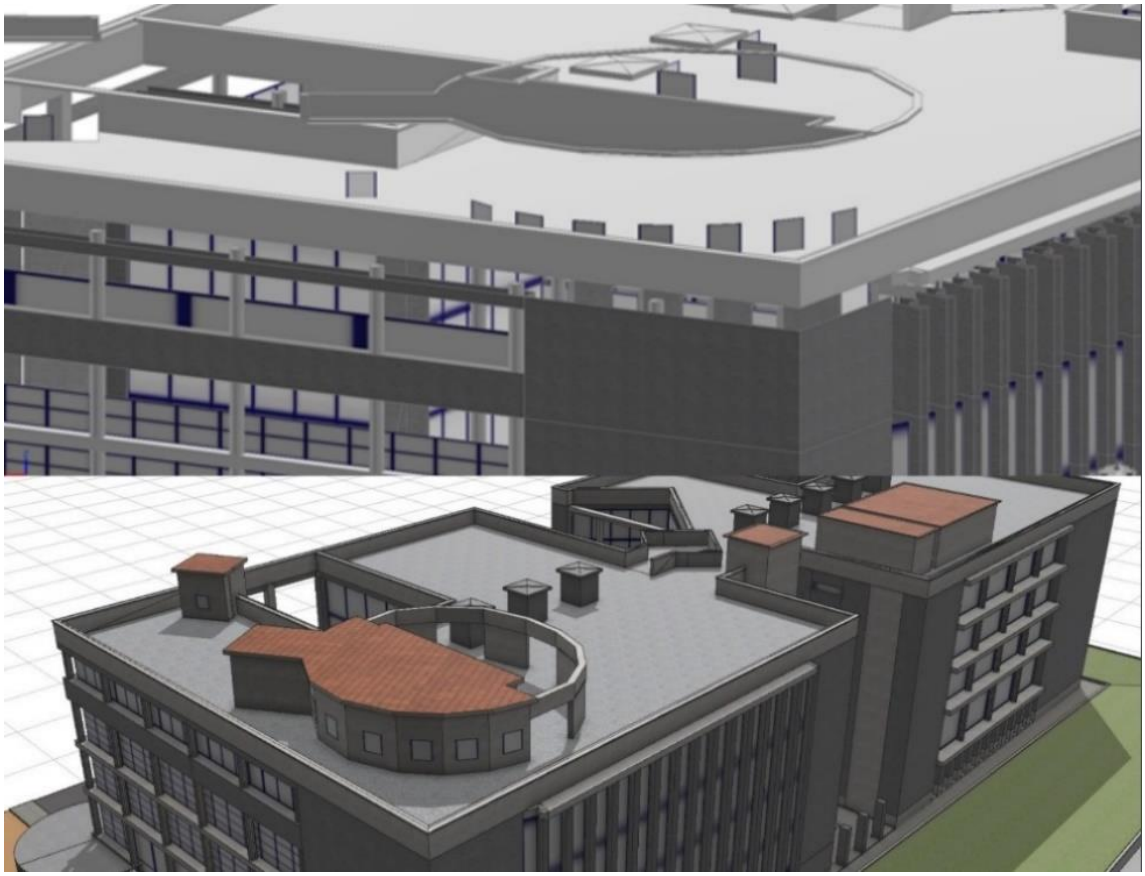

Figure 25: Library comparison 1

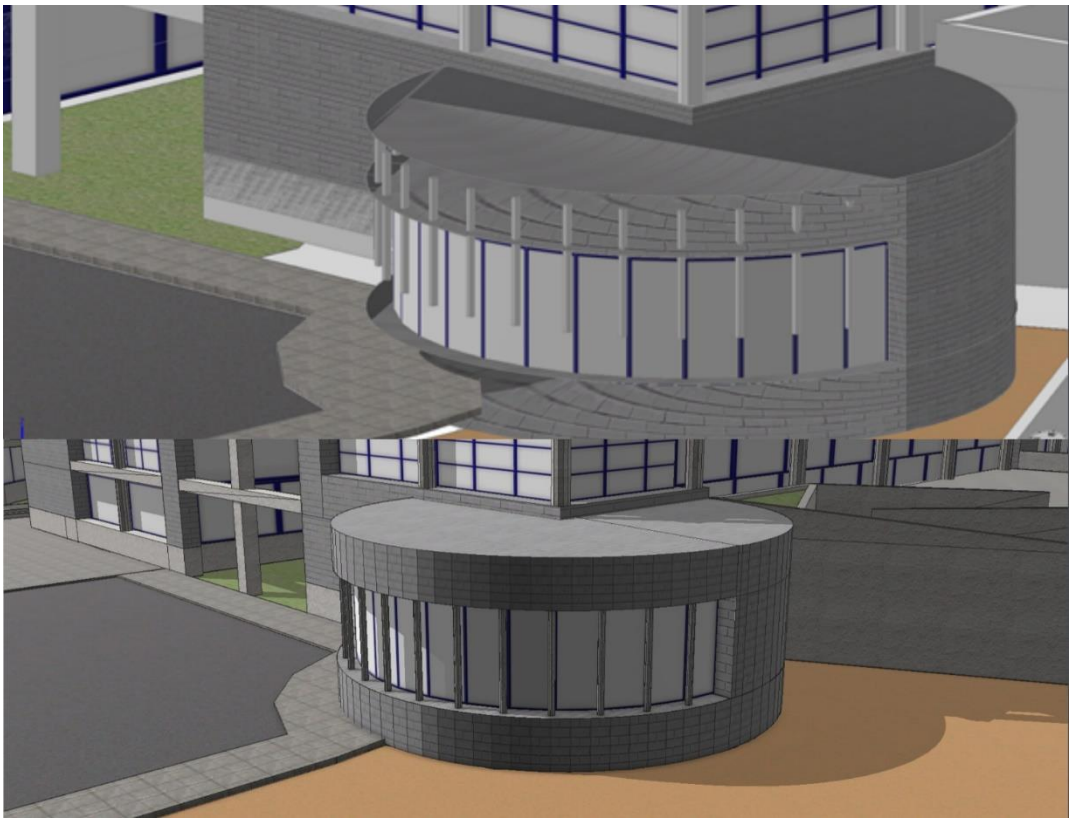

Figure 26: Library comparison 2

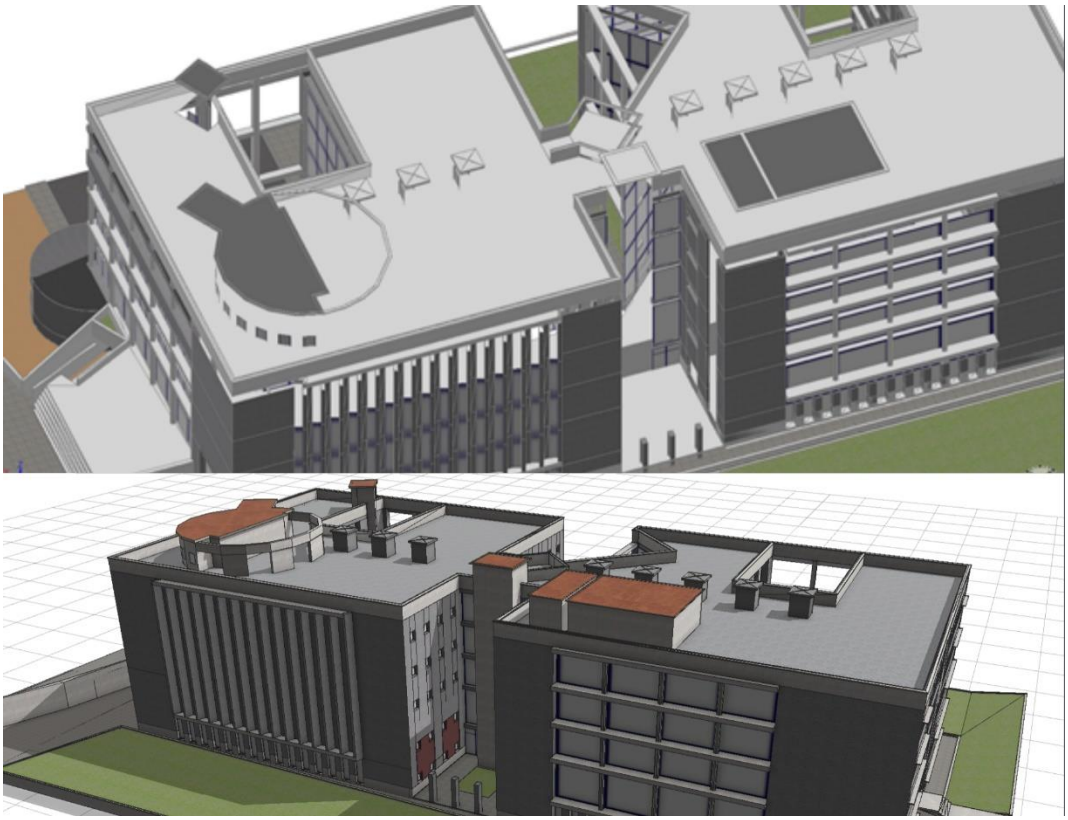

Figure 27: Library comparison 3

4.1.7 Modeling the Interior of the Library

After a few tutorials in SketchUp [13], the basics about this software were learnt and we were ready to start modeling the interior of the library. As we had limited time, it was decided to only create one floor of the library because it was all we needed to create the videogame.

Polygonal modelling in SketchUp works by extruding AutoCAD shapes to create the desired 3D models. From the tutorials it was learnt that it is really important to follow some rules when modelling architectural structures from AutoCAD files in SketchUp:

- Creation of Groups
- Creation of Components
- Face Orientation

The Creation of Groups is very important to avoid the shapes to be joined when extruded. This is useful because if the shape needs to be moved or modified for any reason, it can be done without affecting the adjacent shapes. For example, without the creation of Groups, the table in the picture couldn't be moved because it would have been joined to the floor.

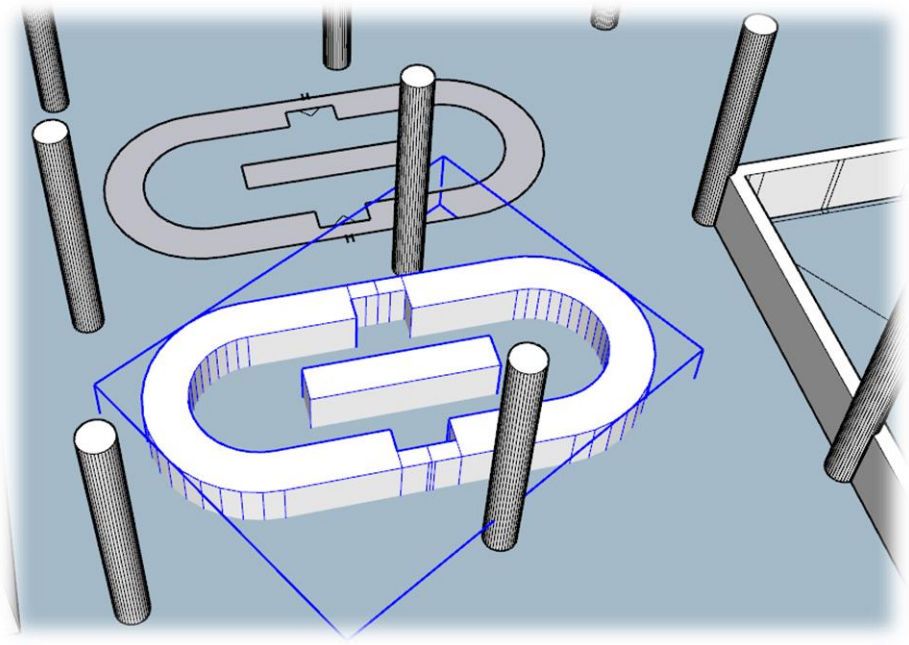

Figure 28: Example of a Group in SketchUp

The creation of Components is very similar to the creation of Groups but with a small difference. Using Components, we apparently obtain the same result as using Groups, however, Components are objects that are repeated in the building. By creating a Component, you are able to modify all instances of that object by just changing one of them. For example, we used Components to create the individual windows and columns.

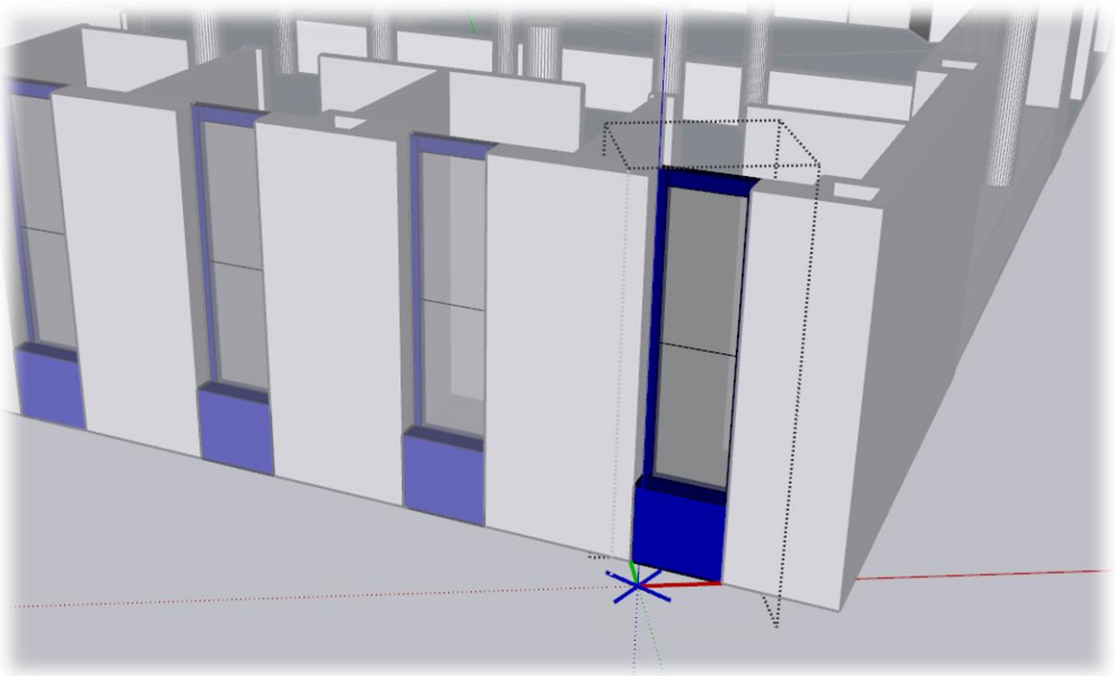

Figure 29: Applying materials to just one window actually applies it to all of them

Finally, sometimes when extruding shapes, the normals appear reversed. To check the model's normals, it is necessary to look at it styled with SketchUp's default texture. For this, we just had to choose View > Face Style > Monochrome to hide any colors or textures added to the model and see the default material. In the example below, we can see how reversed normals appear in a different shaded grayish color.

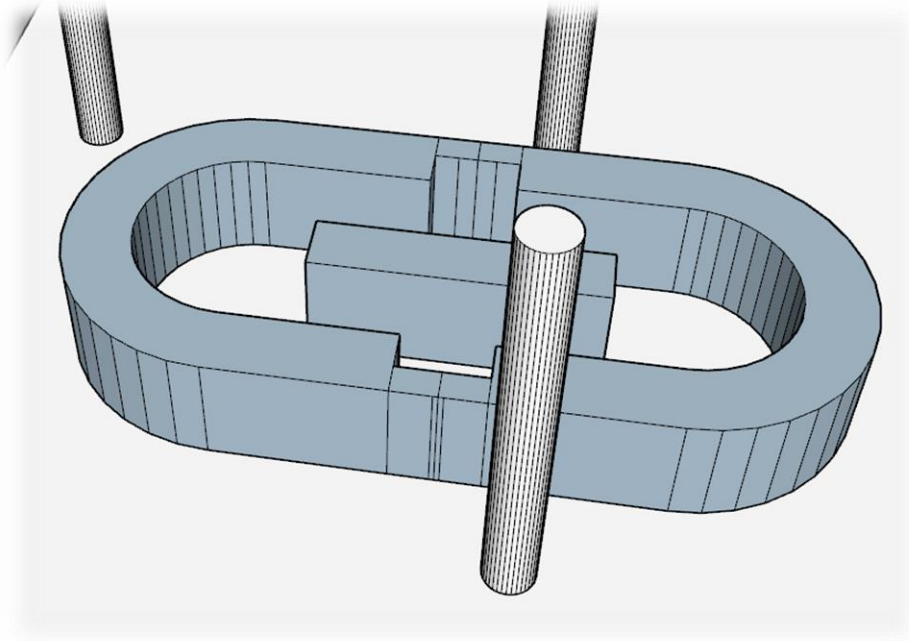

Figure 30: Example of reversed Normals

Next, a few pictures of the process and the final result of the 3D model.

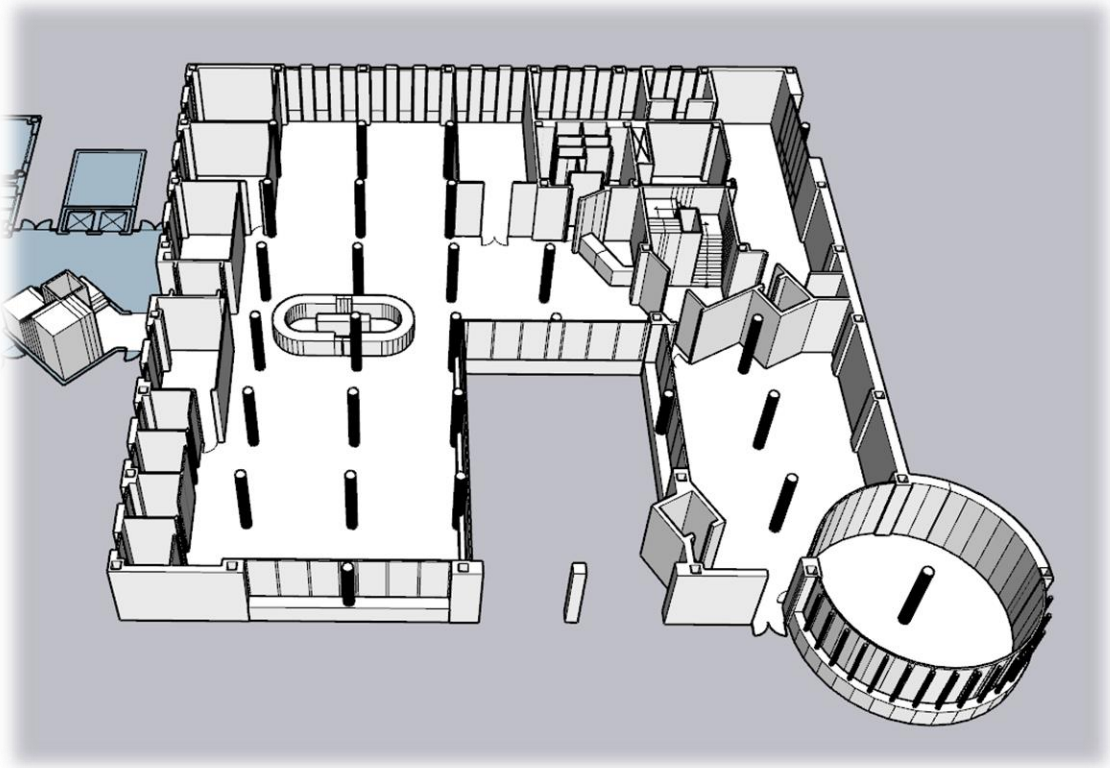

Figure 31: First half of the library floor

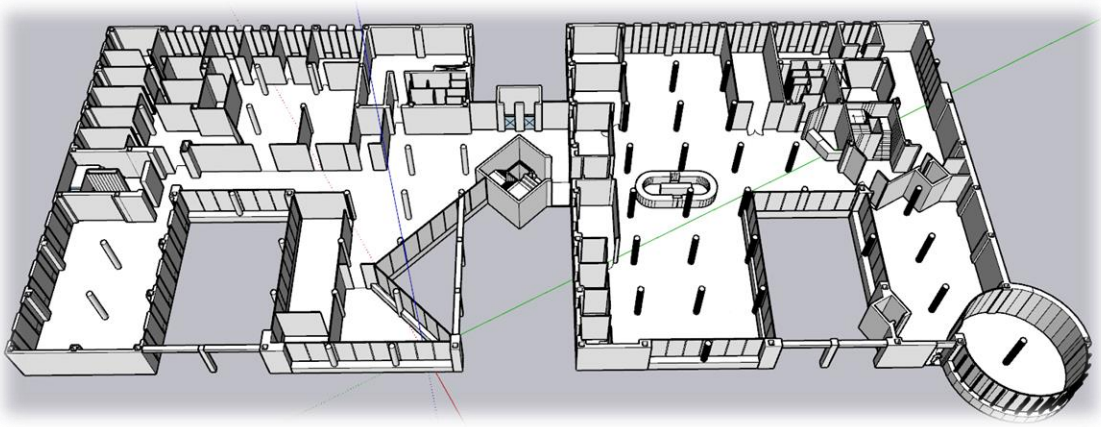

Figure 32: The whole 3D model of the library

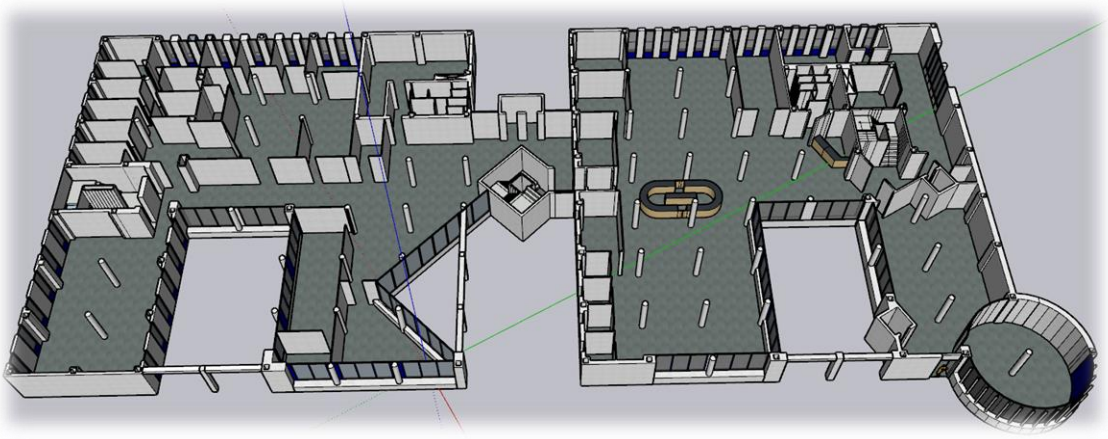

Figure 33: The whole 3D model with textures and materials

4.1.8 Procedural Modelling of a City

Detailed 3D urban environments require a lot of work. It would take too much effort to create all the city elements individually, that's why we are using the procedural modeling software CityEngine to generate them. The term procedural simply means that rules are used to iteratively create 3D content using GIS features and attributes.

Before start, let's understand which are the elements that compose a 3D city model in CityEngine [14].

The built environment takes into account buildings, street furniture such as benches or trees, and other 3d objects.

The legal environment considers 3D zoning and 3D land ownership.

Finally, the natural environment, which is based in land cover, atmosphere and geology.

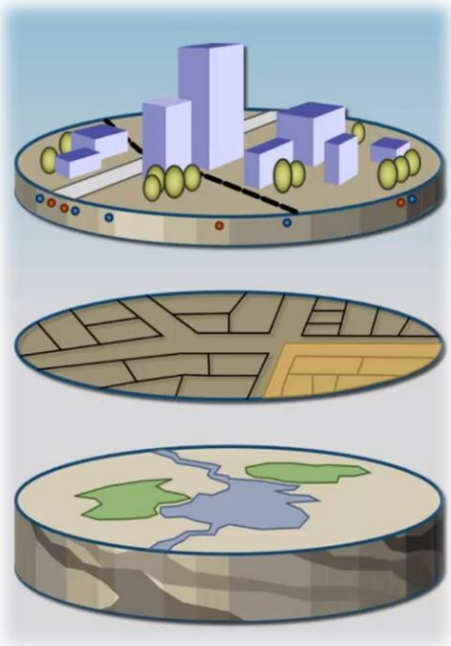

Figure 34: Representation of the 3 environments

It is clear that our city will mainly focus on the built environment.

Before explaining in depth the process followed to create the city, let's describe briefly the steps taken. First of all, GIS Data was obtained from a geodatabase. Next, it was imported into CityEngine. The geometry and attributes within the GIS data were the inputs for the model generation. After that, the basic elements of the city were created with the minimum detail. Finally, details such as texturing or facade generation were added to the 3d models.

The previous step to import the geodatabase was to create a terrain layer within the scene using an aerial image and a height map. The software uses gray scale height maps in order to generate them. This means that with only two textures, it is very easy to build a terrain with height variations.

At this point, the scene was ready to import the different layers from the geodatabase file. Each layer represents different 3D elements: building footprints, rapid transit lines and stations, untextured multi-patch buildings and parks. It is important to notice that the coordinate system for all the layers matches the one of the aerial image to get the desired results.

Figure 35: First layers from the geodatabase imported

Nevertheless, some of the imported layers used 2D features to generate the information displayed instead of 3D features. This caused that the Parks layer was not aligned correctly to the terrain surface. These 2D layers can be easily aligned using the height values from the terrain surface. In GIS, this process is known as draping.

For the next step static models were imported. These are highly detailed 3D models for more important areas such as Downtown. However, they also needed to be aligned correctly to the terrain surface. After all the process, we got a pretty complete and detailed 3D version of Honolulu.

Figure 36: Honolulu at half of the process

Once all the GIS data was imported, it was time to dig into the procedural rules used to create 3D models. Procedural rule files (.cga) are used to drive mass modeling. When a rule is applied to a shape, the rule will access GIS values stored in that shape such as building height or roof type to generate the model. Optionally, a rule can use facade textures and assets like trees or street furniture.

Using rules from ESRI.lib the buildings left to be built from 2D shapes were generated. The rules were also used to apply textures and vegetation and the final result can be seen under these lines.

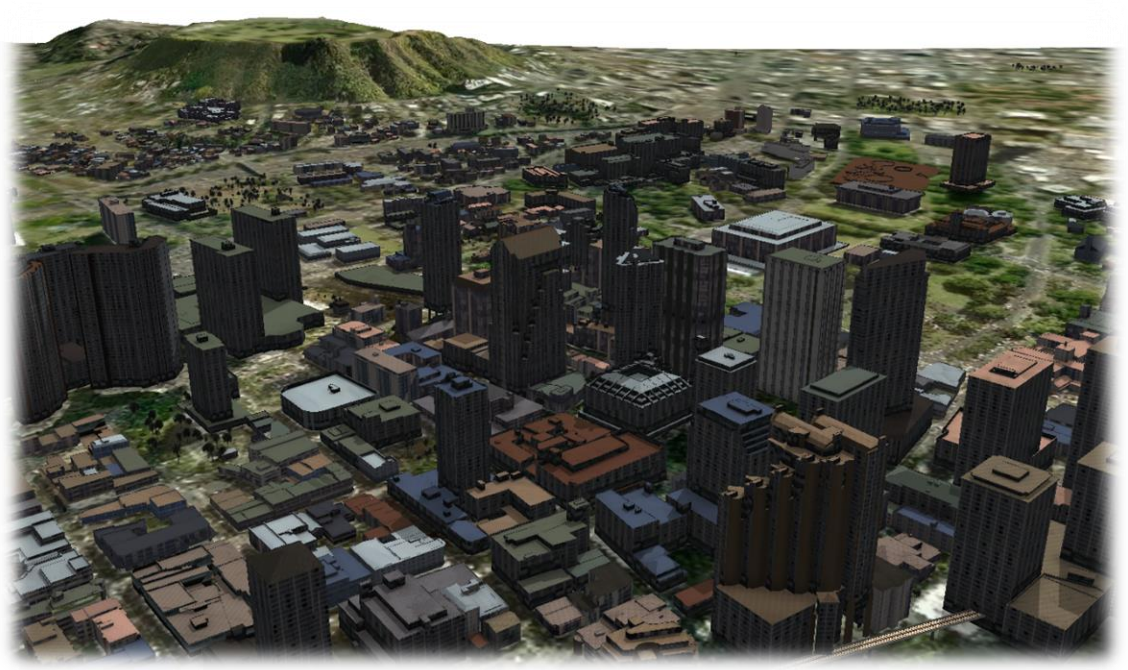

Figure 37: Final result of the city

Finally, some bookmarks/points of interest were located on each station and the project was exported into a CityEngine WebScene format (.3ws). The final result can be seen in the next link. You can use the tools in the scene to navigate freely or press the Play button to see an animation going over all bookmarks.

https://uji.maps.arcgis.com/apps/CEWebViewer/viewer.html?3dWebScene=4bdbebbc1ce2473e89adda3e31be5f5d

4.1.9 Setting up Unity's environment

Once we had the 3D model of the library and the city, it was time to start setting up the game project in Unity. In order to get a more realistic result, it was decided to use the High-Definition Render Pipeline (HDRP) [18]. For this, a new empty project called The Library Agony was created. Then, the High-Definition RP was installed from the Package Manager. Common errors were fixed and the materials in scene were upgraded so as to they could be visible.

The first step to start setting up the environment was to create a Volume in the scene to control the visual environment settings (Night Environment). In this object, we add a new Visual Environment volume override, which basically tells the HDRP which kind of sky and fog we want to see through any camera within the volume.

Subsequent to setting the environment type to HDRI Sky, an HDRI Sky volume override is added too, and the desired cube-map is selected. In our case, a moonlight skybox was downloaded from the Asset Store [15]. It is important to set the intensity mode to multiplier to let the HDRP know that we want the lighting intensity to be set from a standard value, in our case 1. Next, we want to set the Ambient Mode to Dynamic to let the scene receive ambient influence from our skybox. This way, we can get very realistic looking environments. Finally, the values of intensity and temperature of the directional light were changed in order to get a more similar look to the moon.

Figure 38: The scene after configuring the Volume settings

After completing the HDRP volume settings, we were ready to start giving life to the city. Some lights were placed in several buildings and streets. At the same time, some 3D models such as benches or streetlights were located in the streets to achieve more realism [16, 17].

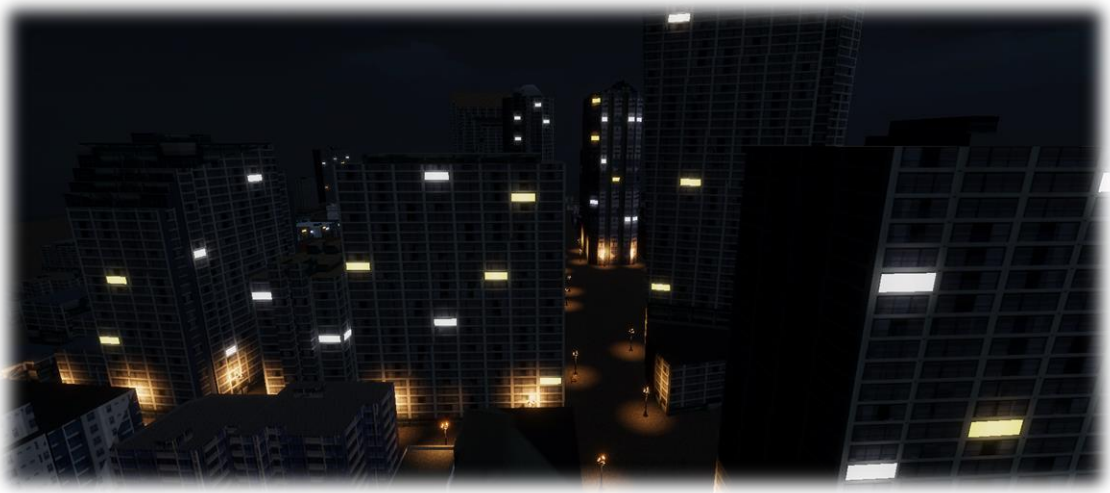

Figure 39: The city after adding some basic lighting

This is a good starting point for the night lighting but more features of the HDRP can be used to really add some atmosphere to the scene. Adding a Bloom post-process override to the volume helps simulate depth by blurring the light sources and making them like it is dispersing through moisture in the air. For this, it is only necessary to change the Bloom Intensity value to the desired value, in our case 0,3.

However, even more depth can be created by simulating atmospheric scattering. For this, a fog override has to be added. In there, we have to enable the Volumetric fog and change the Fog Attenuation Distance to the one that satisfies our needs. This fog will only be visible when the camera is close to ground.

Figure 40: Bloom and volumetric fog

The same settings with some variations were used to set the environment from inside the library.

4.1.10 Improving Game Performance

In order to improve the game performance Level of Detail (LOD) was implemented in the benches and other objects among the scene. Different LODs were created in Blender using the Decimate modifier. Once the object is imported to Unity, it automatically creates the different LODs. In the images on the right, it can be appreciated the difference between de 3 levels: LOD0 (128.000 tris), LOD1 (12.000 tris) and LOD2 (1.280 tris).

Occlusion culling was also Baked to prevent Unity from performing rendering calculations for GameObjects that are completely hidden from view (occluded) by other GameObjects.

Figure 41: 3 LODs

4.1.11 Making the videogame

The videogame takes place inside the library in a new Unity scene. The scenario is slightly different from the one modeled in SketchUp because most of the columns were replaced with bookshelves or tables [19, 20].

In order to safe some time, most of the Assets used in the videogame were taken from the internet or the Unity Asset Store. As our player movement doesn't require complex movements, a simple FPS Controller was imported ready to be used [21]. However, the required modifications were applied, such as the ability to pick up books or other objects of interest by pressing the E key.

The 3D models of the enemies and its animations were taken from Mixamo [22]. Once we had all the necessary assets in the project, the main scene was built.

After that, we started programming the enemy's behavior. The way they work is pretty simple. Using Unity NavMesh they are capable of calculate the most effective path to a desired position. The enemies start patrolling between different Waypoints until the player makes a sound or enters the enemy's field of view [23], which triggers a change of state.

In the enemy's script Update, different functions that change the target position are executed depending on their current state, which is checked using a switch-case.

Figure 42: Enemy chasing the player

After that, the dialogue system was implemented and programmed to pop up exactly when it needs to. Using a coroutine we are able to make the letters appear one by one in a really cool effect.

The last elements to be programmed were the books, also taken from the Asset Store [24]. These will be distributed in different places on the floor. The player will be able to pick them up by pressing the E key and then he will be able to throw them by pressing the click button. To throw the book we simply apply a force to it.

Finally, music and sound effects were added to create a better atmosphere and to make the player be more uncomfortable/terrified while playing the game. The music and sound effects were taken from different webpages [25].

4.2 Results

Based on the objectives cited in point 1.2, we can say that some of them have been achieved in a better way than others.

Let's go over the objectives that were completely achieved. First of all, we have been able to become well-versed in CityEngine, learning about all the areas that this software offers: polygonal modelling, procedural modelling and generation of 3D models from Geographical Information System data.

Secondly, a totally functional and realistic simple videogame has been developed taking into account all the elements that compound it: 3D art, conceptual design, software engineering, narrative design, programming, etc.

Although the exterior 3D model of the library could be finished in CityEngine, the whole 3D model of the interior could not be completed due to the numerous problems faced when importing the library files into CityEngine. These problems finally forced us to look for another solution and we ended up learning about another professional architecture 3D modelling software called SketchUp. Even though this was not one of our original objectives, it is a well-welcomed knowledge.

To sum up, all the objectives have been satisfactorily achieved except modelling the interior of the library in CityEngine using the provided files.

CHAPTER 5

CONCLUSIONS AND FUTURE WORK

Contents

In this chapter, the conclusions of the work, as well as its future extensions are shown.

5.1 Conclusions

The final degree project is a great opportunity to prove yourself of what you are capable of. The main motivation to make this project was to learn about a new way of modelling: procedural modelling in CityEngine. However, we ended up learning a lot of things.

Firstly, the problems faced along the way made us realize that maybe procedural modelling in CityEngine was not the ideal approach to model the UJI library, a single building that needs to be a true copy of reality.

Nonetheless, CityEngine, as its name well indicates, is a great tool to create quickly massive environments such as cities.

SketchUp is the perfect software to create the 3D model of the interior of the library from the data provided (AutoCAD files) because it is a program specialized in architectural modelling.

Working on the creation of a videogame we also have been able to better understand and experience how it feels to develop all the aspects of a videogame on your own, which is harder than it could seem. Moreover, thanks to the HDRP we also learnt how to create more realistic environments in Unity.

5.2 Future Work

This project could be farther developed in numerous ways, from keeping modelling the other floors of the library, to add new functionalities to the game. If we have had more time, these are some of the elements that would have been incorporated:

- First-person arms animations
- More interacting objects
- More narrative elements
- Other types of enemies
- Dubbing

The project could have also taken another path. After creating the floors in SketchUp, it could have been reimported to CityEngine to complete the whole model and publish it in the web the same way we did with the Honolulu city, so it is something that could also be done in the future. Nevertheless, the idea of creating a videogame was way more attractive for this project.

BIBLIOGRAPHY

[1] En.wikipedia.org. 2021. CityEngine - Wikipedia. [online] Available at: https://en.wikipedia.org/wiki/CityEngine [Accessed 1 June 2021].

[2] Esri.com. 2021. ArcGIS Pro | Buy ArcGIS Software Online. [online] Available at: https://www.esri.com/es-es/store/arcgis-pro [Accessed 1 June 2021].

[3] Technologies, U., 2021. Unity. [online]. Available at: https://unity.com [Accessed 1 June 2021].

[4] En.wikipedia.org. 2021. Mixamo - Wikipedia. [online] Available at: https://en.wikipedia.org/wiki/Mixamo [Accessed 1 June 2021].

[5] Microsoft.com. 2021. Microsoft Word - Word Processing Software | Microsoft 365. [online] Available at: https://www.microsoft.com/en-us/microsoft-365/word [Accessed 1 June 2021].

[6] Trello.com. 2021. Trello. [online] Available at: https://trello.com/es?&aceid=&adposition=&adgroup=120671214657&campaign=13266939482&creative=525267309308&device=c&keyword=trello&matchtype=e&network=g&placement=&ds_kids=p64120629721&ds_e=GOOGLE&ds_eid=700000001550057&ds_e1=GOOGLE&gclid=CjwKCAjwz_WGBhA1EiwAUAxIcRcspYqYJGN9eTIAydWv7K39MFD825dhokBZfFW5gdfEU9-6qCNbQhoCMqgQAvD_BwE&gclsrc=aw.ds [Accessed 1 June 2021].

[7] En.wikipedia.org. 2021. GitHub - Wikipedia. [online] Available at: https://en.wikipedia.org/wiki/GitHub [Accessed 1 June 2021].

[8] Ca.wikipedia.org. 2021. SketchUp - Viquipèdia, l'enciclopèdia lliure. [online] Available at: https://ca.wikipedia.org/wiki/SketchUp [Accessed 1 June 2021].

[9] Es.wikipedia.org. 2021. Blender - Wikipedia, la enciclopedia libre. [online] Available at: https://es.wikipedia.org/wiki/Blender [Accessed 1 June 2021].

[10] Technologies, U., 2021. Unity - Manual: System requirements for Unity 2020 LTS. [online] Docs.unity3d.com. Available at: https://docs.unity3d.com/Manual/system-requirements.html [Accessed 20 June 2021].

[11] Doc.arcgis.com. 2021. Tutorials overview — ArcGIS CityEngine Resources | Documentation. [online] Available at: https://doc.arcgis.com/en/cityengine/2019.1/tutorials/introduction-to-the-cityengine-tutorials.htm [Accessed 21 June 2021].

[12] Pro.arcgis.com. 2021. ArcGIS Pro quick-start tutorials—ArcGIS Pro | Documentation. [online] Available at: https://pro.arcgis.com/en/pro-app/2.6/get-started/pro-quickstart-tutorials.htm [Accessed 10 June 2021].

[13] Help.sketchup.com. 2021. SketchUp best practices and applied principles | SketchUp Help. [online] Available at: https://help.sketchup.com/en/sketchup-best-practices-and-applied-principles [Accessed 13 June 2021].

[14] Esri.com. 2021. Esri Training. [online] Available at: https://www.esri.com/training/catalog/search/ [Accessed 7 March 2021].

[15] Unity Asset Store. 2021. AllSky Free - 10 Sky / Skybox Set. [online] Available at: https://assetstore.unity.com/packages/2d/textures-materials/sky/allsky-free-10-sky-skybox-set-146014 [Accessed 19 June 2021].

[16] Turbosquid.com. 2021. 3D Blender bench. [online] Available at: https://www.turbosquid.com/FullPreview/Index.cfm/ID/1030323 [Accessed 19 June 2021].

[17] Turbosquid.com. 2021. 3D street light - TurboSquid 1445090. [online] Available at: https://www.turbosquid.com/3d-models/3d-street-light-1445090 [Accessed 19 June 2021].

[18] Youtube.com. 2021. Setting up Environment Lighting in Unity HDRP. [online] Available at: https://www.youtube.com/watch?v=OXCB-LKCK_Y [Accessed 20 June 2021].

[19] Turbosquid.com. 2021. Bookcase free. [online] Available at: https://www.turbosquid.com/3d-models/bookcase-ase-books-free/688367 [Accessed 27 June 2021].

[20] Turbosquid.com. 2021. Free 3D model adjustable working table - TurboSquid. [online] Available at: https://www.turbosquid.com/3d-models/3d-model-adjustable-working-table-1411898 [Accessed 28 June 2021].

[21] Unity Asset Store. 2021. Standard Assets (for Unity 2018.4). [online] Available at: https://assetstore.unity.com/packages/essentials/asset-packs/standard-assets-for-unity-2018-4-32351 [Accessed 28 June 2021].

[22] Mixamo.com. 2021. Mixamo. [online] Available at: https://www.mixamo.com/#/?page=1&query=running&type=Motion%2CMotionPack [Accessed 29 June 2021].

[23] Youtube.com. 2021. How to Add a Field of View for Your Enemies [Unity Tutorial]. [online] Available at: https://www.youtube.com/watch?v=j1-OyLo77ss [Accessed 30 June 2021].

[24] Unity Asset Store. 2021. Books. [online] Available at: https://assetstore.unity.com/packages/3d/props/interior/books-3356 [Accessed 29 June 2021].

[25] ZapSplat - Download free sound effects. 2021. Fantasy Archives. [online] Available at: https://www.zapsplat.com/sound-effect-category/fantasy/page/4/ [Accessed 1 July 2021].