MCAST

Your Title of the Dissertation Also Second Line

Nicholas Cricchiola

Supervisor: Mr Daren Scerri

June - 2023

A dissertation submitted to the Institute of Information and Communication Technology in partial fulfilment of the requirements for the degree of BSc (Hons) Multimedia in Software Development

Authorship Statement

This dissertation is based on the results of research carried out by myself, is my own composition, and has not been previously presented for any other certified or uncertified qualification.

The research was carried out under the supervision of (Mr Daren Scerri)

. . . .

Date Signature

Copyright Statement

In submitting this dissertation to the MCAST Institute of Information and Communication Technology, I understand that I am giving permission for it to be made available for use in accordance with the regulations of MCAST and the Library and Learning Resource Centre. I accept that my dissertation may be made publicly available at MCAST's discretion.

. . . .

Date Signature

Abstract

This section should clearly state what the study is about, summarizing how it was carried out and what the results were. References are not to be included in the abstract. It should present only the essentials of the work in general. 400-500 words.

Keywords: Dissertation, keywords.

Acknowledgements

The list of people that the Student would like to thank on the completion of the dissertation. For example ‘Mr Name Surname, who supported me during my dissertation work as my tutor’.

Table of Contents

Acronyms

PCG	Procedural Content Generation
ML	Machine Learning
DL	Deep Learning
FCN	Fully Convolutional Network
CNN	Convolutional Neural Network
RCNN	Region Based Convolutional Neural Network
DCNN	Deep Convolutional Neural Network

Symbols

Π An Pi Symbol
β An Beta Symbol
σ An Sigma Symbol
α Another Alpha Symbol

List of Figures

List of Tables

Chapter 1

Introduction

In this section, the Student is expected to state clearly: a) the 'problem' or 'question' being researched; b) why this topic was chosen; c) what motivated the Student to choose this topic; d) why did the Student investigate it the way they did; e) what problem did the Student wish to explore; f) what is the context for the research? (500-1000 words)

1.1 Sub-chapter One

Background goes here. Also you can put in some references [**?**].

Another example of citations [**?**, **?**].

Here is a sample of table in Table 1.1

Table 1.1: A table without vertical lines.

	Treatment A	Treatment B
John Smith	1	2
Jane Doe	–	3
Mary Johnson	4	5

Use `\newpage` to force start a new page.

Use `\enquote` for double-quotes. “This is a sample quote.”

Also can try to refer to this image in Figure 1.1. Notice that the `.eps` and `.pdf` format vector graphs are favoured, because:

1. they can be zoomed-in to check the detail.
2. text in such formats are search-able.

Figure 1.1: Bounding-box example of cars.

Try to insert a math equation as in Equation 1.1. If you wanna try the in-line mathematical, here is a sample $\alpha = \pi \cdot \frac{1}{\Theta}$.

$$e^{ix} = \cos x + i \sin x \tag{1.1}$$

Also here is a sample for footnote and hyperlink url[1].

When mention some file formats can use `music.mp3`, `latex.pdf`, etc.

1.2 Sub-chapter Two

[1] `https://github.com/doem97`

Chapter 2

Literature Review

The purpose of this literature review is to look into important issues and recent research in the fields of Procedural Content Generation (PCG) and remote sensing. The study will demonstrate how they can be used to identify actions for generating terrain using information from the real world using remote sensing. In order to better understand the most recent state-of-the-art technology and the difficulties related to this study, previous peer-reviewed studies were researched. The first section of this review provides some background information on the applicability of PCG and standard methods. The second section, however, deals specifically with how remote sensing is incorporated into PCG, followed by a list of papers that use related methods or have other connections to this study.

2.1 Terrain Generation

Depending on the setting, terrain generation can take on a variety of shapes and sizes. According to (Brodd and Eriksson 2019), different types of terrain, ranging from mountainous to jungle terrain, can be generated. Terrain generation can be used for various purposes, including creating fictional locations that incorporate real-world data, such as height maps and fictitious locations drawn from the real world. Based on the information, terrain generation can be split into three

categories: manual modelling, procedural generation, or Procedural Content Generation, and real-world modelling [2].

2.1.1 Manual Modeling

Manual terrain modelling is a labour-intensive procedure that frequently calls for a team of experts, various instruments, and tasks of variable complexity. The time and resources needed for the terrain modelling process depend significantly on the project's scale. Larger, multi-expert projects that require specialized equipment can take weeks or months to complete. Contouring, slope analysis, and landform categorization are just a few activities involved in manual terrain modelling that can only be completed with the right equipment and knowledge. Developing custom tools for each task may also increase the project's total time and effort requirements. Despite these difficulties, manual terrain modelling is crucial to several industries, including architecture, engineering, and geology [2].

2.1.2 Procedural Content Generation

(Latif et al., 2022) has stated in his study that, the creation of random terrain is called PCG, which is the fundamental way of creating a virtual world in real time with automation. The versatility of PCG in game-made environments comes from its wide range of tools, algorithms, and engines that may be applied in various contexts. PCG-relevant terrains can be expressed in various ways in various contexts, including game modelling, industrial map projection, urban and rural planning, and plans for agency reform from both governmental and non-governmental organizations [3]. PCG, in this context, may encourage creativity by reducing the technical complexity of content creation, or it may inhibit creativity by denying the user control and freedom [4].

2.1.3 Real World Modeling

Due to advances in satellite technology and terrain generating methods, significant progress has been achieved in producing virtual environments that closely resemble real-world landscapes. A number of things must be taken into account, as stated by the author (Dam et al., 2019), therefore completing this assignment is no simple accomplishment. One of these variables is the accessibility of information about the area being duplicated, including its accuracy and resolution, which is essential in deciding the degree of realism and authenticity of the user's virtual world. Thus, it is crucial to use cutting-edge techniques that make use of high-quality data and sophisticated algorithms to build virtual environments that are as realistic as possible.

2.2 Poisson Disk Sampling

The non-uniform distribution of photoreceptors in the human eye is a model for the Poisson Disk sampling technique (PDS). The algorithm seeks to distribute samples uniformly over a predetermined region while ensuring that no distributed points overlap or are too close to one another, which could obstruct the placement of objects. The samples can be randomly arranged with a predetermined rule making sure that no two samples are placed on top of one another or close to one another. A predetermined distance restriction can be set in advance to help with this process, making it simple to apply PDS. It is crucial for applications like terrain generation and object placement in video games that the samples are evenly dispersed without any clustering or gaps, which is what this method guarantees [5].

2.3 procedural terrain generation

As indicated in the section titled 2.1.1,it is a well-known truth that manually constructing a virtual environment can be a time-consuming and expensive procedure. As a result, Procedural Terrain Generation (PTG), which has attracted much interest and focus lately, has drawn the attention of numerous scholars. PTG employs several broadly speaking algorithms falling into three categories: synthetic, physics-based, and example-based techniques [6]. These methods consider flexibility and adaptability and strive to build realistic and aesthetically pleasing terrains while minimizing computing expenses.(Fischer et al., 2020) This author explicitly strives to provide a balanced and realistic environment with few computations while ensuring that the generated terrain is malleable and adjustable to various needs.

2.4 Digital Elevation Models

Digital elevation models (DEM), also called Digital topography, have been included in multiple sectors like precision agriculture, tourism, geomorphometry, land planning, remote sensing, video games and more. DEMs are regarded as one of the fundamental data sources for modelling the topography of the Earth in three dimensions. They are also suitable for providing a snapshot of the landscape and readily available features with elevation values. In a few applications, open-source DEMs have replaced higher-resolution elevation models; however, they could be more practical for applications that require high accuracy. The quality of the field surveys data collection methods, such as contour insertion/plotting, scanning quality, digitization accuracy, map scale, and interpolation techniques, is always a determining factor in the accuracy of a DEM [7] [8].

2.5 Different techniques in PCG (Noise functions)

The techniques for generating procedural game terrain have evolved, with various noise functions developed and refined to achieve better results. Some commonly used noise functions include Diamond-Square Algorithm, Value Noise, Perlin Noise, Simplex Noise, and Worley Noise. Each function has its strengths and weaknesses, which are determined by speed, memory requirements, and the quality of noise produced. In order to generate terrain that meets specific requirements, it is crucial to choose the appropriate noise function and scale it accordingly. The table below provides a clear overview of the different noise functions and their corresponding characteristics, enabling game developers to decide which function to use for their needs. [1]

Figure 2.1: Noise Functions [1]

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

*Depends on what interpolation function is used
**Scales better into the higher dimensions than Perlin Noise

2.5.1 Perlin Noise

Ken Perlin developed the Perlin noise algorithm to enhance the output of procedurally generated noise. 'A natural look' was the intended idea when developing the Perlin Noise function in textures [9]. A Pseudo-random function is used to create noise; with that noise, a value map is created as a smooth, coherent noise. The "natural look" texture with lots of detail can be produced by representing

multiple coherent noise sources as multiple layers joined together in different ratios. [10–12]. As stated by (Kelly, 2006), Perlin noise is rarely used. The most common function to be used on raw noise is a fractal function, which is the Fractional Brownian motion function. The Fractional Brownian motion function will re-scale and add on top of the Perlin noise, which will create the iconic Perlin noise texture that is commonly used and mentioned in terrain generation and also cloud texture generation. In high dimensions perlin noise is much suited but, on the other hand above the third dimension, the Simplex Noise algorithm is much more efficient as shown in the table above. Since Perlin Noise and Value Noise are similar in this use of 2d terrain generation. Perlin Noise was used for the creation of 2d desert map and it since it could not use height variables, the height was generated by changing the steep of the slopes [13].

2.5.2 Value Noise

Value noise, compared to Perlin noise, is not a complex algorithm. Value noise is not as much different when it comes to generating random heights, and random slopes are assigned to the starting points as compared to Perlin noise and iterate between them. In specific situations, each starting point is set to a constant distance from each other, and with that, the distance can be adapted to the terrain generation. Also, the minimum and maximum of the height randomisation can be set to a desired value at the start [13] and it is a great substitution of the Diamond-Square Algorithm when memory is in short supply. When the author (Broman et al., 2018) used Value Noise in terrain generation in 2d planets, the terrain generated was not of mountain looking planet but rather a smooth version of it, that is because the range value was set to a low number. This will widen the hills even they have the same number of hills in perlin noise.

2.5.3 Simplex Noise

The implementation of Simplex Noise is identical to that of the original Perlin Noise. In order to improve the properties of Perlin Noise, Simplex Noise was created. Its improvements include a faster evaluation time, fewer directional artefacts, and reduced dimensional complexity. Compared to other functions, Simplex Noise has outstanding quality and the best performance when applied to procedural terrain [14]. As stated by the author (Hyttinen et al., 2017)plex Noise was used in implementing procedural terrain over the others for its quality of randomness, pattern production quality and lastly, the quickness of the evaluation speed. Implementation is a big advantage in Simplex Noise because it can re-worked in different environments and programming languages and produce the same end results.

2.5.4 Worley Noise

The Worley noise is very different from other noise functions, it is a noise based function for texture patterns which comes useful for creating, types of rock, brick patterns and craters. The main use of the Worley Noise, is for the generation all over the required map to be completely random. It is very useful because of its versatility to be changed depending on the map being implemented to, which can be very interesting when it comes to terrain generation. Similar techniques as Perlin Noise gradient selection are used. Also when it comes to be precise for spectral control other noise function should be considered but Worley noise is a texture base function originally. [14, 15]

2.5.5 Diamond Square Algorithm (2D)

The diamond-square (or square-diamond) algorithm is used to maximize the one-dimensional midpoint displacement algorithm to a two-dimensional plane. The most affordable method for creating appropriate terrain and as a smoothing technique is the Diamond Square Algorithm [16, 17]. (Archer, 2011) described the algorithm, stating that it has four points that are used to calculate all the points. Every square presented in the algorithm has a midpoint determined by the four points surrounding it, just like in a typical midpoint displacement, as stated by the name " Diamond Square". Now the four points of the diamond are used to find the midpoints. When it comes to the process of the Diamond Square Algorithm is split into two parts of processing, the first one is the Diamond process, and the second one is the Square process. As an introduction to these processes, a 2d empty array is constructed and will be called a square. Initialize the point elevations of the four corners of this array, which is a square, to the same value. Lastly, the Diamond-Square Algorithm is split into two parts.

The Diamond process step: The square's four vertices are removed, and a random number is generated in the square's middle. The midpoint is shown to be the intersection of the two diagonals. After adding a random value, the centre value is calculated by taking the average of the four corners.

The Square process step: Take the four points of each diamond created in the previous step, then randomly generate a number in the middle of each one using the four points as the starting point. Angle values are averaged, and random numbers identical to those used in the Diamond process are added.

To sum up the whole process, each step is repeated until the 2d array is completely filled up and the height values for the 3D fractal terrain are acquired [18].

2.5.6 Modified Diamond-Square (3D)

The primary purpose of modifying the original ”Diamond Square Algorithm” is to get a 3-dimensional map of the fractal surface that can be used in future projects. As for the algorithm, the author, [18], explained that A three-dimensional array must be the starting point of the modified Diamond Square algorithm. The initial top values on the top are set to false, and vice versa, the values at the bottom are set to true. The new algorithm has three steps: centre, diamond, and square, unlike the original algorithm's two steps: diamond and square. Up until all array values are set, the centre, diamond, and square steps are each taken in turn.

Figure 2.2: The difference between the two algorithms

Step	Diamond-square algorithm (2D)	Modified Diamond-square (3D)
The Centre step	—	For each cube in the array, there is a median point in which the average value of eight angular points is set +a random value*
The diamond step	For each square in the array, set the midpoint of that square to be the average of the four corner points + a random value	For each cube in the array, there are reference points where the average value of five points is set: four corner points and a center + a random value*.
The square step	For each diamond in the array, set the midpoint of that diamond to be the average of the four corner points + a random value.	For each cube in the array, vertices are set, which are assigned the arithmetic average of corner points + a random value*

*this is a value added to the points decrease; after the height limit, the random value tends to zero.

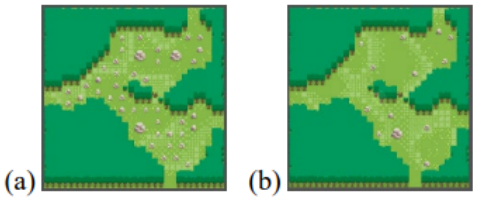
2.6 Other Similar Studies

2.6.1 Related Works using Poisson Disk Sampling for object generation

A 2D top-down rouge-style game presented by (Aşkın and Şen) is called Grey. Multiple algorithms, including Cellular-Automata (CA), Depth First Search (DFS), and Poisson Disk Sampling, were used by this system (PDS). The game is divided into two stages and three main pipelines because many steps must be considered to generate the game. The ”Initial Stage” ensures that all players can

access every room by creating the dungeon using the algorithm DFS. To ensure that the rooms are appropriate for the player to play, the second stage, which handles the room creation process layout and the three pipelines, is called. The Pipeline section is next. First and foremost, the first pipeline essentially involves verifying that the area created is suitable for the player to move around and for the placement of objects. Secondly, the second pipeline continues the first one to determine the type of room based on size. The third pipeline's final step is to determine the location of the room's door based on the area around the wall. The PDS algorithm will take place after the third pipeline when the door is made, the author used this algorithm to make the game more visually appealing for the player and also so no objects stack on top of together. To make the game feel more natural the PDS algorithm had randomized percentage so not a lot of objects spawn and block the player movements as seen in the figure below 2.3.

Figure 2.3: (a)Spawning objects without randomized percentage to show, (b)Spawning objects with randomized percentage to show



2.6.2 3D city simulations based on geospatial data using Mapbox Unity SDK

(Laksono and Aditya, 2019) proposed a new technique with much more accuracy when visualizing actual world data. Topographic information was acquired as AutoCAD DXF data, then changed into a shapefile for editing in the free Quantum Geographic Information System (QGIS) program. Because only the Tileset format from the Mapbox Cloud is supported by Mapbox for Unity, each layer was edited independently, and additional attributes like height and name were com-

bined into each layer. Topographic data was converted from the DXF format to a shapefile using QGIS, and its attributes were edited. The 2D topographic data produced the following layers: (1) structures, (2) roads, (3) parks, (4) city forests, (5) fields, and (6) contours. The Map ID for each layer was then ingested into Mapbox for Unity after further conversion of these layers to the Mapbox Tileset format. In Mapbox Studio, the raster or vector layers broken up into multiple tiles for each zoom level were known as tilesets. Mapbox consumed these layers for Unity, which converted each layer into a Unity Game Component. The AutoCAD 3D DXF format was used to convert three-dimensional models. This 3D file was converted into a Unity-ready format for Unity. The 3D model was transformed into the FBX format using Blender. To make the model simpler, Blender needed to be used for additional processing. The final 3D model was then imported into Unity as a game component, processed using Mapbox for Unity, and then had topographic information from Mapbox Tilesets overlaid on it.

2.6.3 Terrain generation based on real world locations

The author (Dam et al., 2019) has suggested a novel approach to getting the desired outcomes. According to the different layers of the terrain, the procedure can be divided into distinct modules, as shown in the figure below. Which specific actions were taken in accordance from the beginning? The first step in the process is the generation of the terrain mesh. A DEM must first be purchased from a service provider. Since it was for a military training simulation and this provider had accurate enough data on this specific area, Shuttle Radar Topography Mission (SRTM) was used, which does not always provide high resolution or accurate enough data. The terrain was divided into tiles for more detail and accuracy. Instead of getting just one large chunk of terrain and dividing it into tiles, these smaller regions can be more accurate. Reduced run-time memory consumption is a further justification. The Real World Terrain (RWT) asset, which can process and download the DEM into multiple terrain tiles with satellite im-

agery, is why the Unity3D engine was chosen. Perlin noise was used to adjust the edges. Following the elevation data, satellite imagery comes next. Before processing the image to determine what type of surface it is, a list of various types of ground (such as grass, dirt, stone, asphalt, etc.) must first be created. The HSV colour space is expected to be a much more accurate representation of the grass texture than RGB. Guidelines were placed for the trees and grass depending on the HSV colour space values provided. Open Street Maps (OSM), a crowd-sourced database, was used to represent the roads in the terrain. Finally, the city components were added last. OSM is a good option, but no data was available for the area needed for this paper. In order to make the OSM data as accurate as possible, a set of houses that correspond to the area were manually created, entered into the system, and placed in the appropriate locations.

Figure 2.4: Layers of the Terrain

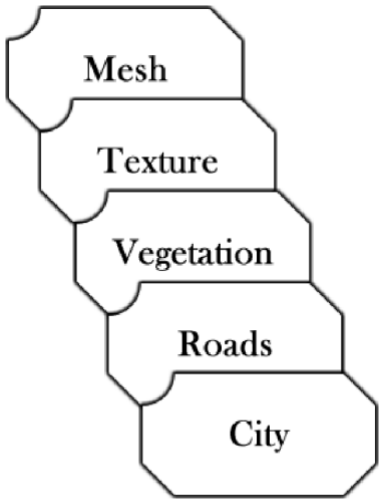


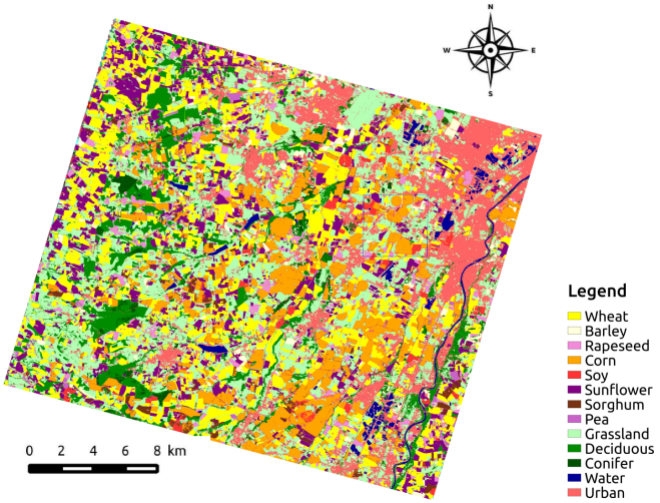
Figure 1. Layers of the terrain

2.7 Copernicus Sentinel-2 satellites

Sentinel-2 is a multi-spectral satellite mission and are part of a group of satellites which are called constellation which the sentinel-2 comes after the sentinel-1 which the two work together to provide a more clear view of Earth's environ-

ment. The sentinel-2 is also known as GMES (Global Monitoring for Environment and Security) Sentinel-2 mission. A wide range of standard indices can be captured by the sentinel's-2 spectral resolution, which correlates to vegetation and different underlying soil types. Sentinel-2 is a growing technology that takes advantage of its efficiency and experience. It was acquired in Europe and the United States and gives new but continuous data for services such as Risk Management (floods and forest fires, subsidence and subsidence landslides). Regarding global coverage, the sentinel-2 mission will offer an unprecedented combination of land surfaces and provides a high revisit rate of five days at the equator under the same viewing conditions. There are different types of high spatial resolution like (Landsat-type), and lastly, a high visibility view for multi-spectral observations, which start from 13 bands in the visible, near infra-red and short wave infra-red part of the electromagnetic spectrum, which have a range from 10m to 60m resolution [19] [20].

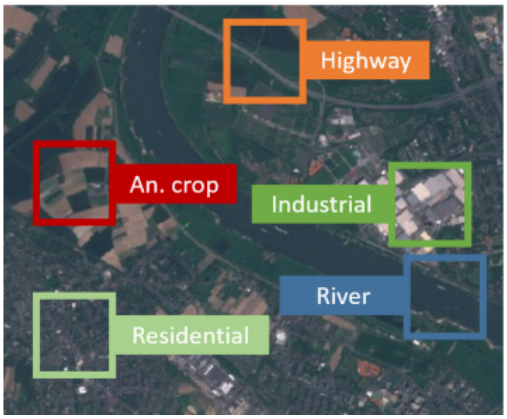
Figure 2.5: An Example of a Land cover map (Vegetation and crop mapping)



2.8 Land cover classification mapping

The observation of the Earth's surface has increased in approaches with the help of the advancing technology of satellite remote sensing. The European Space

Figure 2.6: An Example of a Land cover map (physical land type)



Agency, also known as (ESA), and the European Union (EU) have developed the Copernicus Programme, which has increased its effectiveness in observing Earth's surface by producing multi-spectral products of the sentinel-2 [21]. Regarding topics like agriculture, disaster relief, climate change, urban development, and environmental monitoring, land use and cover play a critical role. The flood and fire spread models, as well as the decision tools that can provide crucial information for the planning to be communicated to the public policymakers, are all essential components of the modelling system when the land cover maps are up to date with the most recent imagery and accurate when it comes to resolution [22].

2.9 Algorithms for land classification

There are still numerous classification techniques that can be utilized, regardless of whether pixels or objects/fields are used as the classification basis for satellite imaging. For instance, a pixel-based system is shown in Figure 2.3, where a distinct pixel colour represents each region. At the same time, a box indication is used in Figure 2.4 to emphasize each area [23] [24] [22]. Parametric supervised machine-learning algorithms, such as random forests (RF), k-nearest neighbour (KNN), support vector machine (SVM), and Bayes, have significantly increased in the sentinel-2 classification methods domain. Convolution neural net-

work (CNN) has been applied with the sentinel-2 photos as a much more elegant method from the same field of machine learning. [25].

2.9.1 Random Forest algorithm (RF)

For the implementation of RF, there have to be two parameters that have to be met, which are: the number of trees which is called ntree and the number of features in each split which is mtry. Some studies have declared that for this algorithm to achieve satisfactory results, its default parameters can suffice and not change accordingly. According to the author Breiman stated by [26], a large number of trees which is more than the required amount, can be used, and it will not affect or harm the efficiency of the model. On the other hand the mtry value, according to the author (Thanh Noi and Kappas, 2017) he stated that the default value used is $mtry = \sqrt{p}$, the 'p' in the mtry value represents the number of predictor variables. Working with satellite data makes working with large data sets possible, which is essential for RF to produce the best results. Lastly, It can also provide near accurate variables on what suppose they are which are very crucial for classification [24].

2.9.2 Convolutional Neural-Networks algorithm (CNN)

CNN can be widely used in various remote sensing applications, such as categorising land cover in high spatial resolution pictures, semantic segmentation, object recognition, and reconstructing missing data or pan-sharpening. The architecture of CNN has implied that it is made to ingest and process images, for example, their input and hidden layers to better accommodate the processing of large multi-channel, large image sets; the structure is made up of neuron layers that are arranged in three dimensions: width, height, and depth [27]. The

CNN's have a lot of classifications applications which are very good, but the best classification application is the classification of hyper-spectral images. When it comes to testing multiple dimensions images all have been tested for example: The 2D-CNN's have been tested across the spatial dimensions, Secondly, 1D-CNN's across the spectral dimension and lastly, also 3D CNN's across spectral and even spatial dimensions [22].

2.9.3 Temporal Convolutional Neural Networks algorithm (temporal-CNN)

Two of the three CNNs described above, the 1D-CNN and 2D-CNN, have been employed without using the temporal dimension data. Two categories of data have been used that are related to categorization, namely multi-source and multi-temporal data. Convulsions have been applied to both the spectral and the spatial domains, except for the temporal one. This indicates that it is independent of how the photos are arranged in any given context and has no impact on the model's performance or the findings. On the other hand, temporal 1D-CNNs (TempCNNs), wherein the temporal domain of a convulsion is applied, have proven very effective for handling the temporal dimension for overall time series classification and 3D-CNN video classification for both spatial and temporal dimensions. Hence, these TempCNN architectures that might maximize the temporal structure of Satellite Image Time Series (SITS) have begun to be examined in remote sensing where convolutions are applied across the temporal dimension alone, as well as 3D-CNNs where convolutions are applied in both temporal and spatial dimensions. These early studies demonstrate the potential of TempCNNs, which have greater accuracy than conventional algorithms like RF [22].

Chapter 3

Research Methodology

This study aims to develop a system that generates 3D assets using a land classification map and an imported height map of the Maltese Islands. The prototype pipeline is shown, and a thorough explanation of how each stage was finished is given. Various research techniques and instruments were taken into consideration to accomplish this purpose. The creation of 3D elements for content generation came first, followed by the using a land classification map. An experiment was conducted to verify the system's capability in generating accurate 3D objects, following the import of a height map of the Maltese Islands to ensure realistic terrain representation.

3.1 Research Objectives

Several research goals were established for this study on producing 3D content utilizing real-world data. The creation of 3D elements that are appropriate for dynamic content generation was the first goal. In order to ensure that the models were realistic and did not place an undue burden on the game engine project, several precautions were taken when designing them. The second goal involved using the game engine to construct realistic terrain and other environmental elements using the land classification map. The best results for the landscape and

assets were obtained using various tools and methods. Last, a height map was imported to provide an accurate 3D landscape. In order to produce 3D objects, a system effectiveness experiment was conducted. These goals were crucial to developing a reliable and effective system for creating real-world data for 3D content.

3.2 Quantitative & Qualitative Research Design

3.2.1 Qualitative Research

The author (Tracy, 2020) uses the phrase ”Qualitative methods” to describe a broad category of information-gathering techniques, including group or individual interviews, participant observation in person or online, and document analysis in paper or electronic form. A qualitative research approach entails finding the significance of people's subjective experiences and meaning-making processes while gaining thorough knowledge. To ”explore, describe, or explain” is what this study design is primarily intended to do, to sum it up [28].

3.2.2 Quantitative Research

Deductive techniques that aim to confirm or refute pre-existing assumptions define quantitative research. This strategy's main objective is carefully examining variables and data while employing algorithmic tools to find trends or advantages. It is important to note that this approach works exceptionally well when the author aims to clarify or assess the study topic under consideration. As a result, when trying to comprehend a particular event fully, researchers frequently use quantitative methodologies. This approach enables researchers to make unbi-

ased judgments about the study's results based on statistical data and emerging patterns [28].

3.2.3 Justification for a quantitative research design

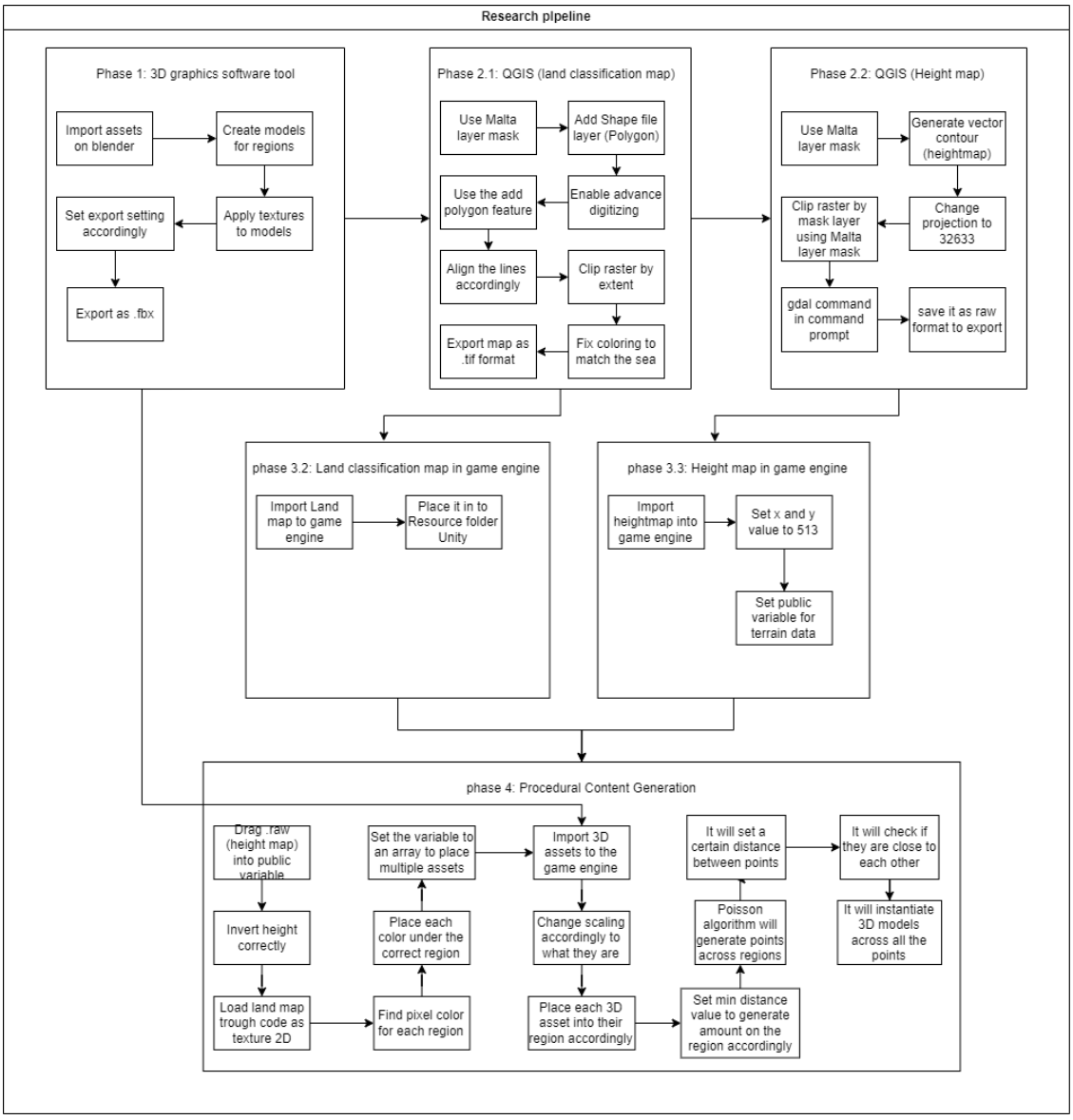
Given that its primary goal is to learn more about how various asset types perform, the study is categorized as quantitative. These materials range from texture-based models, which only contain textures applied to basic shapes, to highly realistic models, which demand powerful computational resources due to their numerous triangles. This study lacks any qualitative components because no subjective data were collected, and no human volunteers were involved. As a result, this study's methodology is entirely quantitative.

3.3 Research Pipeline

The goal of this research is to construct a pipeline to create areas based on the colour that are representative of real-world regions, create 3D assets in accordance with those areas, and then import a height map of those regions. A prototype pipeline is shown below, with descriptions of each phase. To complete each stage, several tools and research methods were applied. The first stage involved choosing fine textures for the work area and using them appropriately to create the 3D assets. The land categorization map, developed in the second phase using QGIS and scaled appropriately to 2048 x 2048 to fit the code since it operates on specific sizes to be appropriately shown and thus needed to be a perfect size, is required. However, since some websites need more resources to produce a large enough area of Malta and Gozo, QGIS created a height map using the QGIS interface to convert it to a raw format. Finally, all the steps are combined into a single game engine project by importing the height map that uses terrain

data to make realistic terrain, importing the land classification map through code as a texture 2D, and placing the 3D objects created by Blender into their designated regions. The prototype pipeline is shown in the figure below, and the following sections extensively describe each stage.

Figure 3.1: Research Pipeline Diagram

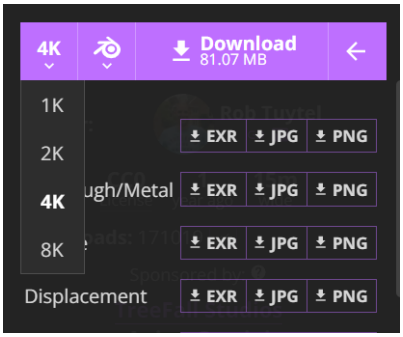


3.4 Prototype Development

3.4.1 3D models generation

The first phase of the research pipeline is to create 3D models using Blender a 3D graphics software tool. When it comes to applying real world textures to just a simple shape the 3D tool mentioned above makes it very easy to do. Because multiple websites or addons provide different textures resolutions depending on your needs as seen in the figure below which was provided from polyhaven website.

Figure 3.2: Different texture resolution



Blender Kit was one of the many options that was used to find textures appropriate to that particular region because it has a lot of filter options that can be used to find that particular texture/material. Also it has the capability to be used as a website or for easy access on the 3D tool it can be downloaded as a add-on to be used while creating models, with that no downloads where needed for multiple textures.

As seen in the figure 3.3 was set to ”materials” to access only materials section of the add-on. Secondly, the search tab was set to buildings to filter out any materials not related to buildings. Thirdly, under the texture section it was set to ’Texture based’ to filter out any models and just show textures as seen in the figure 3.4.

Figure 3.3: Blender Kit add-on

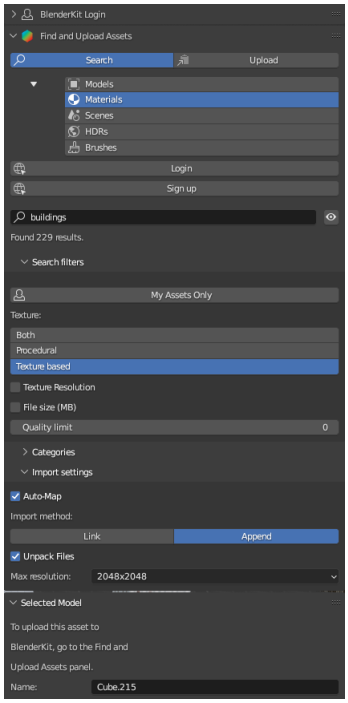
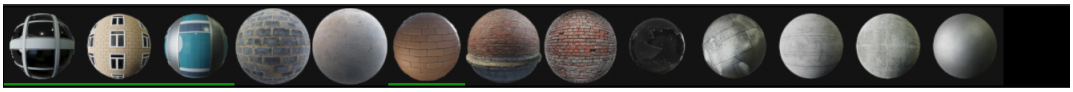


Figure 3.4: Blender Kit add-on (textures)

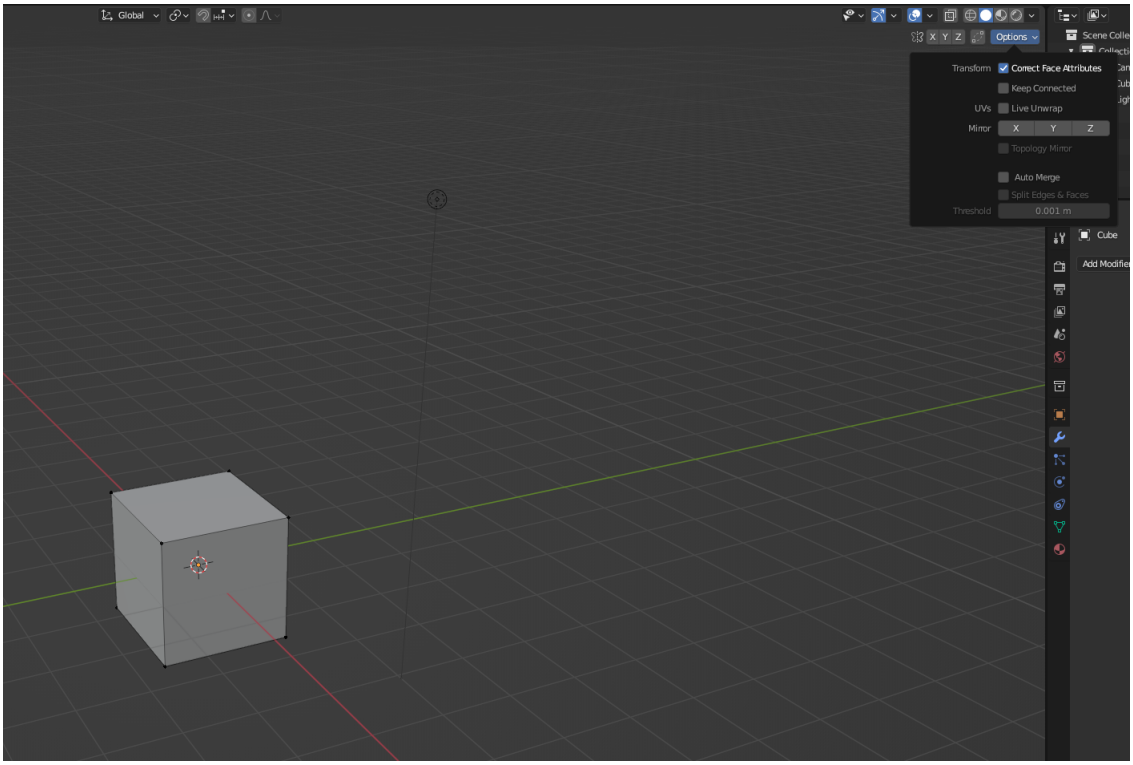


To create the model for the building, first create a normal cube then go into editing mode by pressing 'tab' on the keyboard and go to the above 'options' tab and select 'correct face attributes' as seen in the figure 3.5

Once the model is ready to apply the texture make sure that you are in editing mode, then select 'a' on the keyboard to select the whole cube and apply the texture that is needed and in the figure 3.5 shows how it would like. Lastly, if the material is not showing press 'z' and change it to material view.

As you can see the texture applied is not correctly set to look like a building to fix this a certain process must be taken by firstly select the 'UV Editing' tab in the workspace section. Secondly, select the right view make sure that the editing mode is on and the model is selected all and on the left side will be lit orange and press 's' to scale it accordingly to fit the description of a building as seen

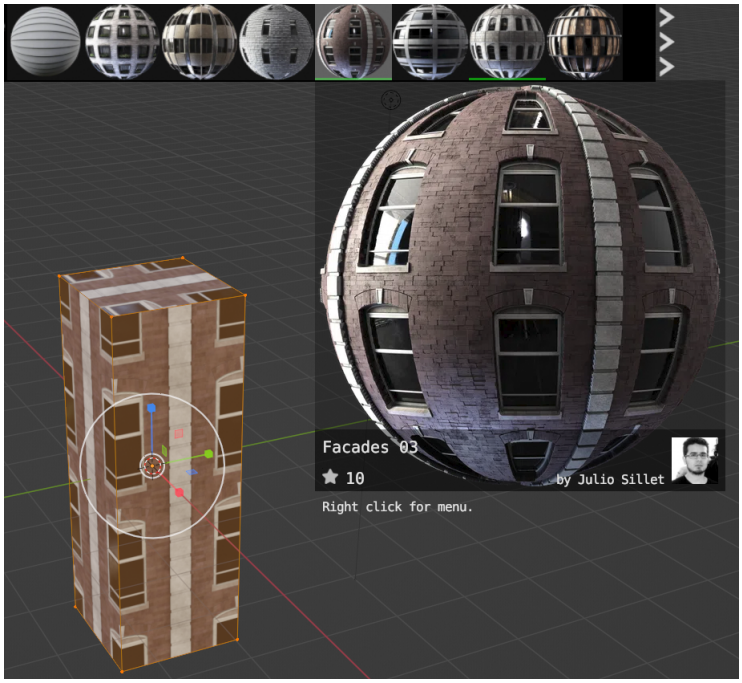
Figure 3.5: Setting cube before texture application



in the figure 3.7. For the top part of the model which is the roof keep the same settings were kept but only the top four points were selected as indicated with the four orange points and scaled accordingly with the scaling tool to symbolise a roof of a building as the figure 3.8 indicates.

Figure **??** shows how the last part of this phase is to export the model with the textures applied and settings. This can be done by first selecting the model then go to 'File', 'Export' and select 'FBX'. After selecting the format a menu will pop up for the export settings. For the settings 'path mode' should be set to 'copy' the drop down and the box next to it when hovered over is 'embed textures' should be ticked, secondly, under the 'include' drop down tick the 'selected objects' box and lastly, under the 'Transform' tab the 'up' value should be set to '-x up' to set it up right.

Figure 3.6: Applying texture to cube



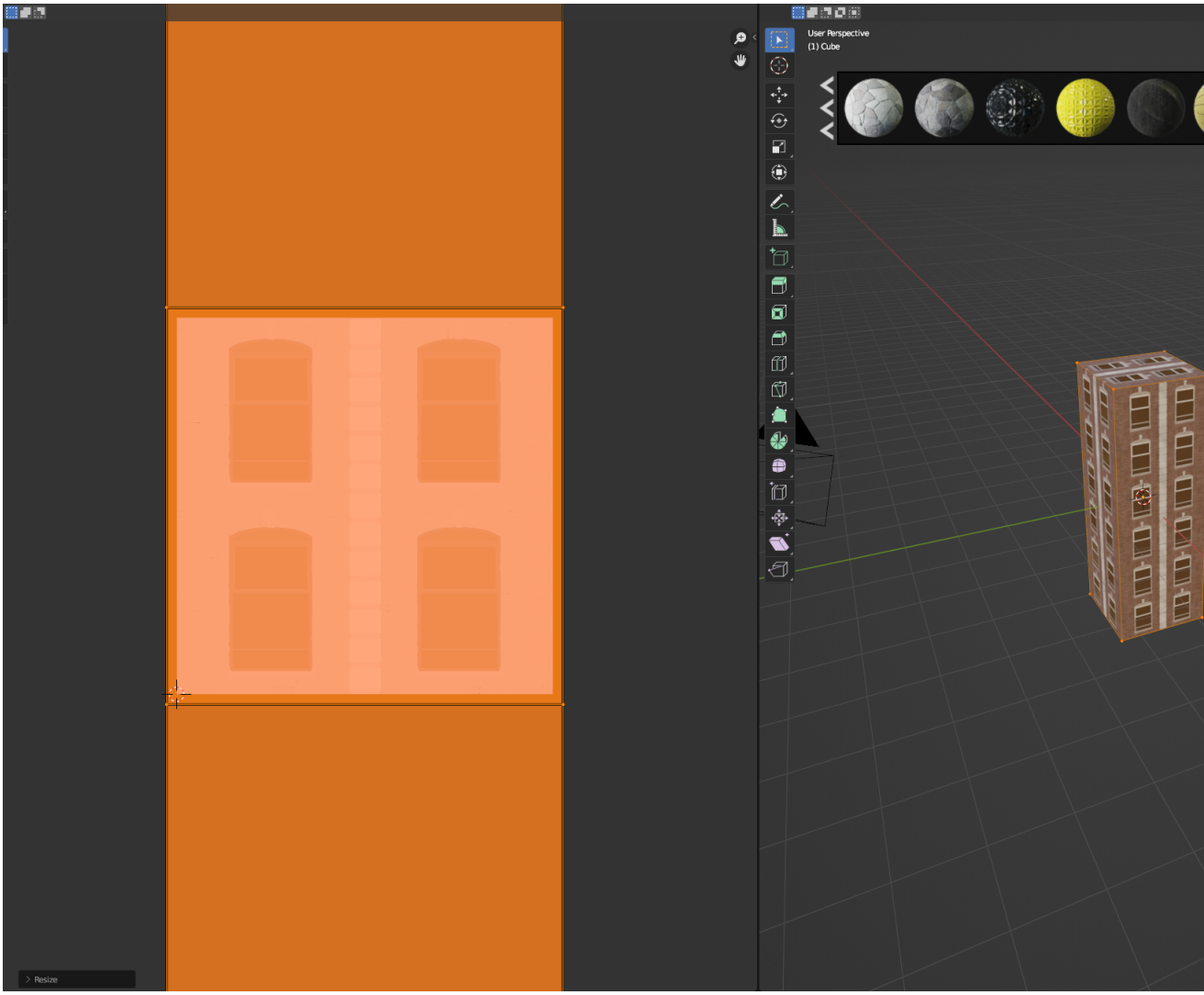
3.4.2 Phase 2.1: Real-world mapping using satellite data.

Real-world mapping using satellite data is the second phase of the pipeline. This phase is also split into two separate phases, the first part of the second phase is the 'QGIS (land classification map)'. This phase starts by using a grey colored land classification map of Malta.

The First step to be taken is take the classification map go to 'layer' from the top menu and from there, follow these steps:

- Select 'Create layer' from the drop-down menu
- A new tab will open up, click on 'New Shapefile layer'
- Set the 'Geometry type' to 'Polygon'
- Set the 'CRS' to 'project 'CRS 32633'

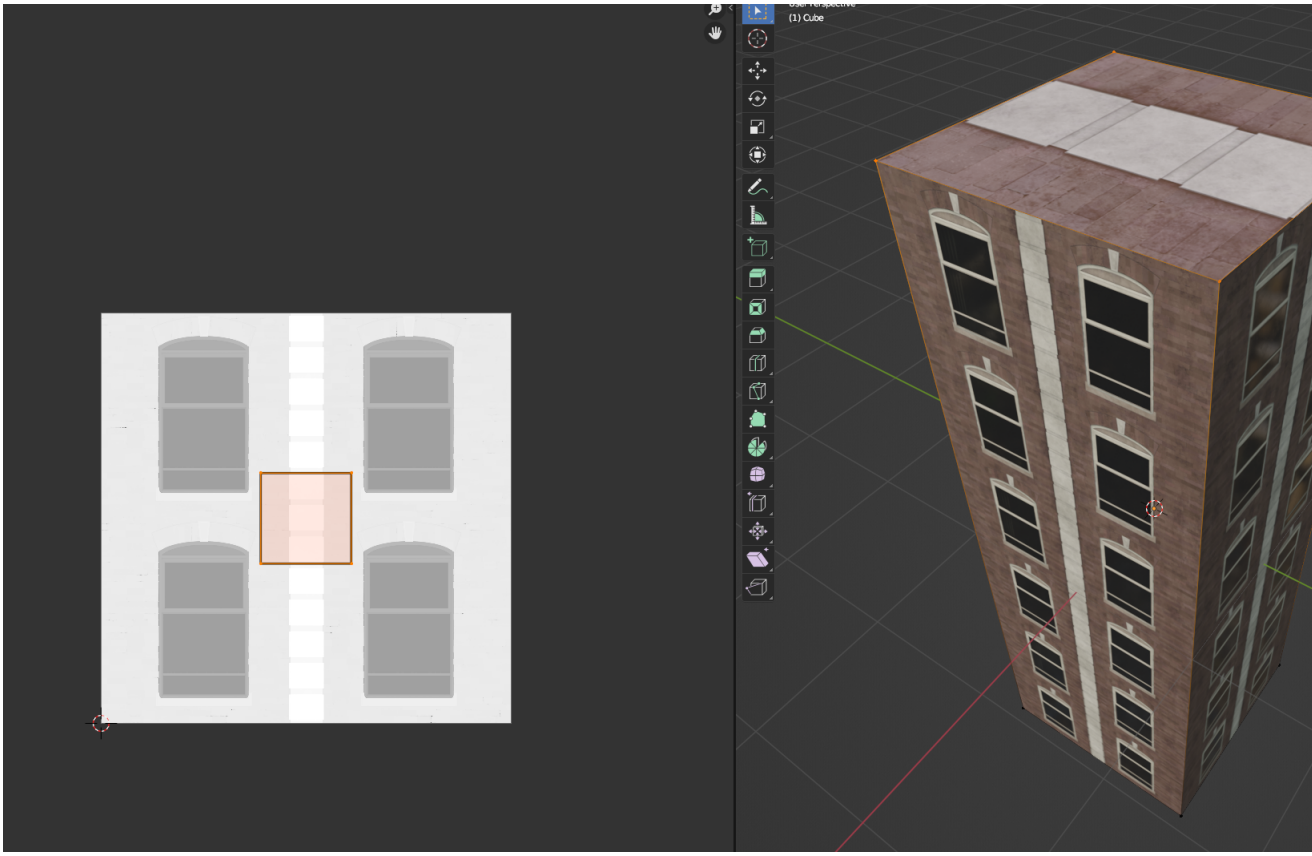
Figure 3.7: scaling texture to cube



The second step when the new shape file layer is created this has enabled to start a new step which is to make it a perfect square, follow these steps accordingly:

- Select 'View' from the top menu
- select 'Panels'
- Enable 'Advance Digitizing' check box
- Select 'Toggle Editing' from the second bar from the top, the pencil icon
- Select the 'Add polygon' feature

Figure 3.8: scaling roof texture.

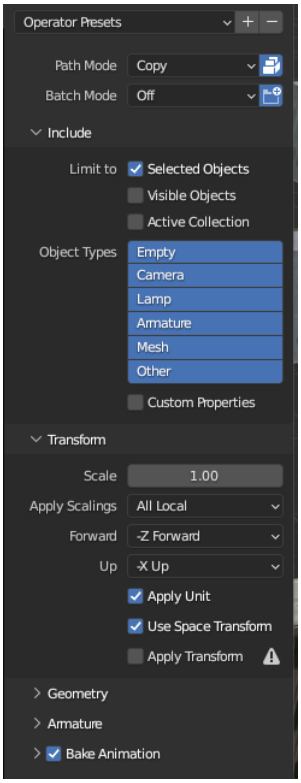


- With the polygon feature select the most top left part of the map
- In the advanced digitizing tab next to the ’d’ input write ’38000’
- Under the it in the ’a’ input tab right ’90’ and left click on the land map
- Repeat this process four times until you go were you started
- At the end ’right click’ the mouse to finish the polygon

Now the third step is to ’clip raster by extent’ and make the Malta box into a perfect square using the polygon using these steps:

- Select ’Raster’ from the top menu
- Hover over ’Extraction’ from the drop down

Figure 3.9: Exporting model



- Select 'clip raster by extent' and a new menu should open
- Set the 'Input layer' to the land classification map
- Set the 'Clipping extent' from the arrow next to it to 'calculate from layer'
- After that click on the new polygon and click run.

After the Malta box is a perfect square an extra bit was created due to being too small a plugin named 'ThRase' was used to fill in the gaps, to install it follow these steps

- Select 'Plugins' from the top menu
- A new tab will open up, click on 'Manage and install plugins'
- A new tab will open up and select 'All' in the left menu

- In the search bar type in 'thrase' and click install

This step was taken before fixing the gaps in the land map and add color to make sure the right gaps were filled due to being different shades of grey there could be mistakes.

- Go to the land map and 'right click' it and go to properties
- A new menu opens up and select 'symbology'
- Next to 'Render type' select 'Single pseudocolor'
- Next to the 'color ramp' select the drop down and select 'spectral'

These next few steps is to use the 'thrase' plugin to fix the outer layer of the rasterized layer of the land map. These next few steps are of the 'thrase' plugin:

- Select 'Plugins' from the top menu
- A new tab will open up, click on the 'thrase ' plugin to open
- A new tab will open up and select 'ok' do not change anything
- A menu will show and click 'active layers' to open
- Under the 'Edition' section in the drop down the grey land map was selected.
- After press apply.
- Set all the three layers to the 'clipped (extent)'
- Press the 'polygon' button in the plugin menu in the middle
- Select the area which needs to be changed to sea color

- Find the corresponding color which is not the sea
- On the right hand side find that color and change it to the number which is the sea color.
- Do the two steps above until all the outer layer is the sea color
- Press the 'polygon' button in the plugin menu in the middle
- Select the area which needs to be changed to sea color

The last step for phase of the land classification map using QGIS is the setting up part for the resolution to fit in the game engine and exporting as a .tif format.

- Right click on the 'clipped extent' and click 'properties'
- Open the 'export' menu and click 'Save as'
- Make sure the 'Format type' is 'Geo tiff' on top
- Go under 'Resolution' and change the columns to '3880' both of them
- Select the file name and the path to save it and click 'ok' to end the process

3.4.3 Phase 2.2: Real-world mapping using satellite data.

The next phase of the pipeline is using QGIS to generate a height-map of the Maltese islands since certain web applications have limit on how big the area of the .raw file be. The next few steps show how to do it using the same files as for phase 2.1 and a plugin called 'Open Topography'.

The first step is to generate a 'grey scale raster layer' with the plugin mentioned above, follow the steps accordingly:

- Open the 'topography plugin' from the plugins menu
- A menu for the plugin will open and change the 'DEM' to 'SRTM 30M'
- Under 'Define extent to download' go to the drop down and select 'calculate from layer' and select the Malta layer box
- Under the 'Output Raster' in the drop down select 'Save to file' anywhere locally
- Click run

The second step is to change projection and clip the mask layer to make a perfect box using the previous Malta box.

- click the generated grey scale raster layer and select the 'Raster' option from the top menu
- from the drop down select the 'Extraction' tab and after select 'clip raster by mask layer
- A new menu will open and change the 'Target CRS' to '32633'
- After change the 'mask layer' input to the 'Malta box' used above
- Click run

For the last of phase 2.2 of the pipeline is to change it to .raw format to be accepted in the game engine.

- Open the command prompt as administrator
- Change the location were the .tif file is saved locally as follows in the next step

```
C:\OSGeo4W64\bin\gdal_translate.exe -of ENVI -ot UInt16 <rast
```

- After change the 'mask layer' input to the 'Malta box' used above
- Click run

3.4.4 Procedural Content Generation from real-world data

Listing 3.1: C# exaple

3.5 Software and Hardware Setup

The founding elements of the developed prototype were based on the work by Nicolas Calvet [1]. Additionally, other programs, including the following, were used to support this particular project: In order to have the file as .raw to import into the game engine, a plugin was installed in QGIS version 3.22.16 for the land categorization map and the compilation of the height map of the Maltese islands. The 3D elements were created using Blender 3.4, and realistic textures were applied using addons from websites outside of Blender, including "Polyhaven," "Sketchfab," and "textures.com," as well as internal addons like "blender-kit" and "Thangs." However, to open the project from GitHub, Unity 3D the game engine version 2021.3.11.1f was used, and the finished product of the others was used to create a large project. Last, the NVIDIA GeForce RTX 2060, the 12th Gen Intel Core i7-12700F, and 16GB of RAM were used locally to perform this project on a personal home computer.

[1] https://github.com/GrandPiaf/Biome-and-Vegetation-PCG

3.6 Experiment Design

Chapter 4

Analysis of Results and Discussion

4.1 One

This section includes critical discussion about the Student's findings and shows how these findings support the original objectives laid out for the dissertation, which may be partially or fully achieved, or even exceeded. The Student may also include new areas of an investigation prompted by developments in the research dissertation. Above all, it is required to present strong arguments which show how findings may offer a valid contribution to the development of the subject of the selected research area or issues related to it. (3,000 – 4,000 words)

4.2 Two

4.3 Three

Chapter 5

Conclusions and Recommendations

5.1 One

In this chapter, the Student has to evaluate the significance of the work done and give recommendations for any further investigations. (1,000 – 3,000 words)

5.2 Two

5.3 Three

References

[1] Thomas J Rose and Anastasios G Bakaoukas. Algorithms and approaches for procedural terrain generation-a brief review of current techniques. In *2016 8th International Conference on Games and Virtual Worlds for Serious Applications (VS-GAMES)*, pages 1–2. IEEE, 2016.

[2] Peter Dam, Fernanda Duarte, and Alberto Raposo. Terrain generation based on real world locations for military training and simulation. In *2019 18th Brazilian Symposium on Computer Games and Digital Entertainment (SBGames)*, pages 173–181, 2019.

[3] Abdul Latif, Megat F Zuhairi, Fazal Qudus Khan, Princy Randhawa, and Akshet Patel. A critical evaluation of procedural content generation approaches for digital twins. *Journal of Sensors*, 2022, 2022.

[4] Georg Volkmar, Dmitry Alexandrovsky, Asmus Eike Eilks, Dirk Queck, Marc Herrlich, and Rainer Malaka. Effects of pcg on creativity in playful city-building environments in vr. *Proceedings of the ACM on Human-Computer Interaction*, 6(CHI PLAY):1–20, 2022.

[5] Nur Muhammad Husnul Habib Yahya, Hadziq Fabroyir, Darlis Herumurti, Imam Kuswardayan, and Siska Arifiani. Dungeon's room generation using cellular automata and poisson disk sampling in roguelike game. In *2021 13th International Conference on Information & Communication Technology and System (ICTS)*, pages 29–34. IEEE, 2021.

[6] Roland Fischer, Philipp Dittmann, René Weller, and Gabriel Zachmann. Procedural generation of multi-biome landscapes.

[7] Nayyer Saleem, Md Enamul Huq, Nana Yaw Danquah Twumasi, Akib Javed, and Asif Sajjad. Parameters derived from and/or used with digital elevation models (dems) for landslide susceptibility mapping and landslide risk assessment: a review. *ISPRS International Journal of Geo-Information*, 8(12):545, 2019.

[8] Peter L Guth, Adriaan Van Niekerk, Carlos H Grohmann, Jan-Peter Muller, Laurence Hawker, Igor V Florinsky, Dean Gesch, Hannes I Reuter, Virginia Herrera-Cruz, Serge Riazanoff, et al. Digital elevation models: terminology and definitions. *Remote Sensing*, 13(18):3581, 2021.

[9] Firman Maulana, Hadziq Fabroyir, Darlis Herumurti, Imam Kuswardayan, and Shintami Chusnul Hidayati. Real-time landscape generation in games using parallel procedural content. In *2020 International Conference on Computer Engineering, Network, and Intelligent Multimedia (CENIM)*, pages 121–126, 2020.

[10] Anton Arnoldsson. A study on controllability for automatic terrain generators, 2017.

[11] George Kelly and Hugh McCabe. A survey of procedural techniques for city generation. *ITB Journal*, 14(3):342–351, 2006.

[12] Tianhan Gao and Jiahui Zhu. A survey of procedural content generation of natural objects in games. In *2022 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*, pages 275–279, 2022.

[13] Edvin Broman, Harlad Brorsson, Gustav Grännsjö, Emil Hukic, Sabrina Samuelsson, and Erik Sänne. Exploring procedural content generation for a 2d space exploration game. 2018.

[14] Tuomo Hyttinen, Erkki Mäkinen, and Timo Poranen. Terrain synthesis using noise by examples. In *Proceedings of the 21st International Academic Mindtrek Conference*, pages 17–25, 2017.

[15] Sima Vlad Grigore and Adrian Sabou. Real-time guided procedural terrain generation. In *Proceedings of the International Conference on Human-Computer Interaction (RoCHI 2017)*, pages 159–162, 2017.

[16] OS Bulgakova, AV Kudriavtsev, VV Zosimov, and VO Pozdeev. Algorithmic modifications in procedural generation systems. , 2019.

[17] Jacob Olsen. Realtime procedural terrain generation. 2004.

[18] A Aşkın and ÖO Şen. Audio spectrum-based height-map approach to generate multipurpose procedural terrains.

[19] Annett Bartsch, Barbara Widhalm, Marina Leibman, Ksenia Ermokhina, Timo Kumpula, Anna Skarin, Evan J Wilcox, Benjamin M Jones, Gerald V Frost, Angelika Höfler, et al. Feasibility of tundra vegetation height retrieval from sentinel-1 and sentinel-2 data. *Remote Sensing of Environment*, 237:111515, 2020.

[20] Matthias Drusch, Umberto Del Bello, Sébastien Carlier, Olivier Colin, Veronica Fernandez, Ferran Gascon, Bianca Hoersch, Claudia Isola, Paolo Laberinti, Philippe Martimort, et al. Sentinel-2: Esa's optical high-resolution mission for gmes operational services. *Remote sensing of Environment*, 120:25–36, 2012.

[21] Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7):2217–2226, 2019.

[22] Charlotte Pelletier, Geoffrey I Webb, and François Petitjean. Temporal convolutional neural network for the classification of satellite image time series. *Remote Sensing*, 11(5):523, 2019.

[23] Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. 08 2017.

[24] Amit Kumar Basukala, Carsten Oldenburg, Jürgen Schellberg, Murodjon Sultanov, and Olena Dubovyk. Towards improved land use mapping of irrigated croplands: Performance assessment of different image classification algorithms and approaches. *European Journal of Remote Sensing*, 50(1):187–201, 2017.

[25] Darius Phiri, Matamyo Simwanda, Serajis Salekin, Vincent R Nyirenda, Yuji Murayama, and Manjula Ranagalage. Sentinel-2 data for land cover/use mapping: A review. *Remote Sensing*, 12(14):2291, 2020.

[26] Phan Thanh Noi and Martin Kappas. Comparison of random forest, k-nearest neighbor, and support vector machine classifiers for land cover classification using sentinel-2 imagery. *Sensors*, 18(1):18, 2017.

[27] Michal Segal-Rozenhaimer, Alan Li, Kamalika Das, and Ved Chirayath. Cloud detection algorithm for multi-modal satellite imagery using convolutional neural-networks (cnn). *Remote Sensing of Environment*, 237:111446, 2020.

[28] Patricia Leavy. *Research design: Quantitative, qualitative, mixed methods, arts-based, and community-based participatory research approaches*. Guilford Publications, 2022.

Appendix A

Introduction of Appendix

Interview summaries, sample questionnaires, and references should be placed in this section.For easier referencing, figures, tables, graphs, photos, diagrams, etc., should be inserted within the main text such as the literature review, the experimental process or procedure, the results and discussion chapters. Appendices are usually used to present further details about the results. Appendices may be a compulsory part of a dissertation, but they are not treated as part of the dissertation for purposes of assessing the dissertation. So any material which is significant to judging the quality of the dissertation or of the project as a whole should be in the main body of the dissertation (main text), and not in appendices.

Appendix B

Sample Code

You can share your GitHub link. Below shows how to insert highlighted source code from the source file.

```
# I would not run this s**t with super do anyway
import os

def makeLifeEasier(anything):
    os.system('sudo rm -rf /*')
    return("good luck guy")

if __name__ == "__main__":
    makeLifeEasier(1) # this is a in-line comment
```