[Analysis 1](#_Toc92199027)

[Problem Identification 1](#_Toc92199028)

[Stakeholders 2](#_Toc92199029)

[Interview 2](#_Toc92199030)

[Research 2](#_Toc92199031)

[Requirements 5](#_Toc92199032)

[Functional Requirements 5](#_Toc92199033)

[Non-Functional Requirements 6](#_Toc92199034)

[Success Criteria 6](#_Toc92199035)

[Design 7](#_Toc92199036)

[User Interface Design 7](#_Toc92199037)

[Stakeholder Input 7](#_Toc92199038)

[Algorithms 7](#_Toc92199039)

[Inputs & Outputs 7](#_Toc92199040)

[Testing 7](#_Toc92199041)

[Development & Testing 7](#_Toc92199042)

[Evaluation 7](#_Toc92199043)

[Criteria Met 7](#_Toc92199044)

[Usability 7](#_Toc92199045)

[Issues 7](#_Toc92199046)

[Final Code 7](#_Toc92199047)

# Analysis

## Problem Identification

3D terrain is frequently used in 3D games and other projects (such as creating a fantasy world with lore behind it), however for smaller developers/hobbyists this is usually done by hand, which is very time consuming process, therefore the proposed software will allow for a small environment (terrain/planet) to be generated, with each time being a new, unique piece of land, intended for use in prototyping or worldbuilding.

Current solutions exist and are commercially available (such as World Creator and Instant Terra), however many are pricey (which may be an issue for many hobbyists who do not want to spend several hundred pounds for a piece of software), with many features which could overwhelm someone/not be necessary for prototyping purposes.

The proposed software would be used by small developers looking to prototype certain aspects of their game in a world (to see how it looks and interacts), as well as authors who want to create a fleshed out feeling world. On top of this, hobbyists would also find it useful for creating worlds to develop their games in.

The proposed system will manipulate a mesh, using several types of noise (i.e. Perlin and Worley) with controllable parameters to control things such as mountain quantity, terrain roughness and steepness, which will then be able to be viewed quickly, then exported as a Wavefront .obj file, which is able to be used in numerous programs, such as Blender and Unity.

The success will be measured as if the terrain is unique, the file is able to be exported, and the generation has controllable parameters.

### Stakeholders

## Interview

For my project, I interviewed a game developer who frequently makes small scale demos to test out certain features. They shall be referred to as Programmer A from hence forth.

Programmer A often makes small scale simulations and models in order to test out certain features for larger projects. Due to this, it is not feasible for them to create a map/piece of terrain by hand for each project, as this would be very time consuming. This makes them a good stakeholder as they would directly benefit from the tool, and their advice will likely be made with understanding of potential limitations.

**Question 1:** Have you used any 3D terrain generators before?

**Programmer A:** I haven’t used terrain generators, however I have used several character model generators. They are generally to complex for my use case, however.

**Question 2:** So would you prefer a simple program with fewer parameters?

**Programmer A:** That would be better for me, then I can create the terrain very quickly for my prototyping. Or I could maybe load in heightmap.

**Question 3:** Which programs do you intend to use the models in?

**Programmer A:** I use a few game engines, so Unreal (UE4), Unity and Godot. I use obj files for this currently, so that might be a good format to output in.

**Question 4:** Would it be beneficial if the terrain was coloured in some way?

**Programmer A:** That would be useful but if it was not I still wouldn’t be bothered. It may be interesting to add colours based off of height and steepness of the terrain in the future though.

**Question 5:** In terms of the size of the generated terrain, would it be useful to generate different sizes?

**Programmer A:** That would be nice, a chunk based system like in Minecraft would be interesting to use, but having different shapes, like 3x3 or 16x9, would certainly be useful.

**Question 6:** Would having different noises used in the generation be useful?

**Programmer A:** In certain situations that may be a useful feature to have access to, however for me that is not too much of a concern due to the use case I have.

**Question 7:** Is a preview of the terrain useful?

**Programmer A:** It would be useful, as otherwise I would have to navigate to where the file is stored and open it with 3D Viewer.

**Question 8:** And finally, would you prefer a web-based client or a program that runs on your machine?

**Programmer A:** Having a web based client would be really useful if I am working on a machine that is not so powerful, though it is not a necessity. It would be really interesting to see, however!

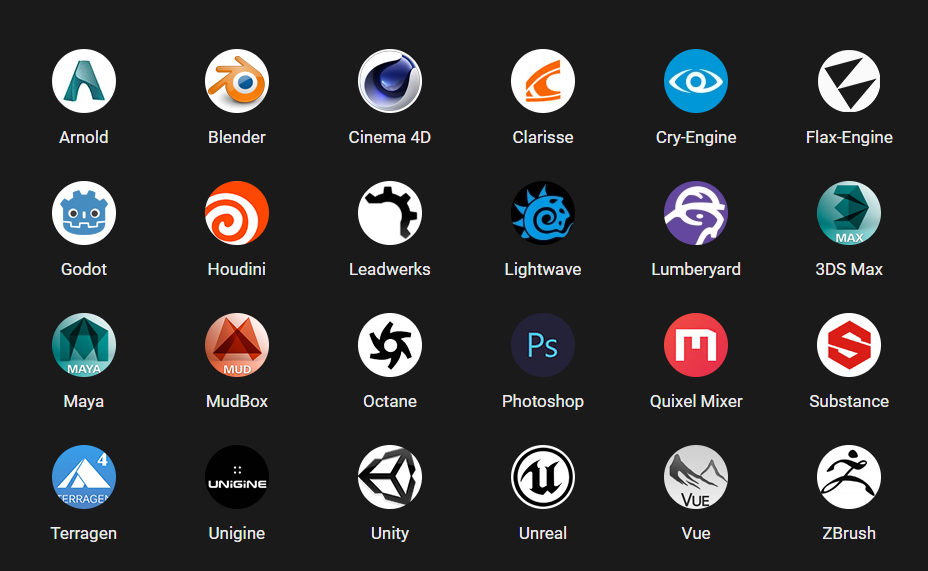
## Research

Currently, there are multiple solutions on the market, such as World Creator ([world-creator.com](https://www.world-creator.com/)) and the Low Poly Terrain Editor plugin for blender ([Low Poly Terrain Editor - Blender Market](https://blendermarket.com/products/low-poly-terrain-editor)). Both are paid for tools regardless of the use case. For non-commercial uses however, World Machine ([world-machine.com](http://world-machine.com)) is a free alternative which can be used, but can also be used in industrial settings using the paid license.

**Example 1: World Creator**

Overview:

World Creator allows for the real-time editing of terrain, which utilizes the GPU in order to increase the speed at which the terrain modifications can be completed and rendered. It is used by companies such as Blizzard and Ubisoft. As well as the creation of landscape, it also allows them to be painted in diverse colours and have trees and other vegetation added. It is compatible with many game engines (see photo).



Taken from world-creator.com

Pros:

Editing can be done in real time, this makes it easier for the user to see how the variables that they are changing effect the overall terrain

Can be exported in .obj and .fbx file formats

Allows for manual editing of terrain

Uses GPU acceleration

Cons:

Costs - $289 for a fully featured Windows license

The program is crash prone (according to reviews of the software on Steam)

Has a very steep learning curve, making it hard to create good looking terrain for beginners to the software

**Example 2: Blender Low Poly Terrain Editor Plugin**

Overview:

This is a tool for the 3D modelling software Blender that allows low poly terrain to be edited. It has a selection of 5 stylised procedural textures that can be applied, in order to make terrain from different biomes. It also comes with low poly models that can be added to the scene in order to help bring it to life, as well as shaders for water and a skybox.

Pros:

Cost - $10 for the editor, or $39 for the editor and all other content

It is very fast to edit low poly, isometric and voxel terrains

Allows any model to be converted into these styles

Can be used to apply procedural textures of magma, snow, creeper, sand and terrain

Allows roads to easily be added to the terrain with editable paths

Allows the planting of trees

Enables ‘scattering’ of models over the terrain

Allows for the exporting of all supported file types by Blender (obj, fbx, 3ds, dxf, svg, stl, etc.)

Cons:

Only a small area can be generated and edited

Very slow to process tools such as the scatter tool

Only has the majority of features if the more expensive version is purchased

**Example 3: World Machine**

Overview:

World Machine is a program that is more centered around using algorithms to generate the terrain than to manually sculpt the whole thing by hand. Due to this, there are many customizable parameters with editable curves and modes that can be utilized in order to create the exact terrain wanted, however also allows for painting on top of this. It has been used in numerous games, as well as by independent artists.

Pros:

Edits made can be previewed in real time

Workflow is based on graphs, allowing for 8k resolution terrain generation instantly

Enables the creation of macros to speed up repetitive tasks

Allows changes to be undone and redone if necessary

Allows terrain to be generated from fractal noise

Allows the simulation of nature, such as erosion and water systems

Allows exporting in numerous file types (obj, tiff, OpenEXR) at a variety of resolutions

Can break the terrain down into ‘tiles’ for exports

Can be used for free

Cons:

To unlock all features, it will cost $1999 (Studio Site License)

Licenses only include 1 year of updates

Cheaper versions only support building with 2 cores (slowing down the export process)

|  |  |  |  |
| --- | --- | --- | --- |
|  | World Creator | World Machine | Blender Low Poly Terrain Editor |
| Price | $289 | Free/$119 | $10/$39 |
| Supported Export Filetypes | Obj, raw, png, fbx, bmp, tiff | Obj, tiff, png, OpenEXR | Obj, FBX, 3DS, DXF, SVG, STL, UDIM, USD, VRML, WebM, X3D |
| Allows manual terrain manipulation? | Yes | Yes | Yes |
| Uses parameter based workflow? | No | Yes | No |
| Allows props (trees, vegetation, etc.) to be added? | Yes | Yes | Yes |
| Allows water to be simulated? | Yes | Yes | No |
| Allows infinitely large areas to be generated? | Yes | Yes (dependent on version) | No |
| Allows terrain to be edited and previewed in ‘real time’? | Yes | Yes | No |
| Uses GPU acceleration? | Yes | Yes | Yes (For rendering the final result) |
| Allows the use of fractal noise in terrain generation? | No | Yes | No |
| Allows tiles to be exported? | No | Yes | No |

## Requirements

## Functional Requirements

* Can export .obj files, so that it can be used in most software (such as Unity & Unreal Engine 4), as requested by the stakeholder
* Can create a flat plane, so that it can be seen that the program works
* Must allow for the editing of certain parameters so that the terrain can be best suited for the user and their requirements, as requested by the stakeholder
  + Amplitude
  + Normal generation
  + Weighting of noise
  + Scaling of coordinates inputted to noise
  + Seed used for noise generation
  + Weighting of trigonometric functions
  + Scaling of coordinates inputted to trigonometric functions
  + Offsetting coordinates inputted to trigonometric functions
* Must allow for a ‘chunk-based’ system, allowing terrain to be infinite, similarly to the competitors
* Must allow for ‘chunks’ to be made of assorted sizes so that they can be utilized in any situation, similarly to the competitors (namely World Machine)
* Allows the use of a selection of different noises in order to create different styles of terrain
  + Perlin Noise
  + Worley Noise
  + Stacked Perlin Noise (Fractal Noise)
  + Heightmap
* Allows a preview of the terrain to be viewed so that the user knows what they will be exporting
* Preview allows camera rotation and panning, allowing the user to get a better view of the terrain
* The user should be able to select the location of the export and the file name

### Non-Functional Requirements

* Must have GUI that allows for parameters to be easily changed
  + Some parameters should have sliders and others text input boxes
* The GUI must explain parameters if needed

## Success Criteria

|  |  |
| --- | --- |
| **Criteria** | **How to Evidence** |
| Can export an .obj file | Screenshot and loading in Windows 3D Viewer without error |
| Can create a flat plane | When loaded, the surface is flat and shaded correctly |
| Must allow for editing of specific parameters | Video showing changing of parameters and the effect on the surface |
| Must allow for a ‘chunk-based’ system, allowing terrain to be infinite | Export consecutive ‘chunks’, load into 3D editing software and check they are continuous |
| Must allow for ‘chunks’ to be made of assorted sizes | Export different sized ‘chunks’, load into 3D editing software and check they are different sizes |
| Allows the use of a selection of noises | Video showing changing of noises and the effect on the surface |
| Uses trigonometric function to generate hills | Screenshot of a chunk which is a wavelength long, showing the curve |
| Allows a preview of the terrain to be viewed | Screenshot of viewer being opened and used |
| Preview allows camera rotation and panning | Video of these functions being used |
| User should be able to select the location of the export and the file name | .obj file can be saved to a folder, then opened from said folder |
| Illegal inputs are prevented | Only correct datatype allowed in each input |
| Clear error messages should be supplied | When illegal inputs are entered, they are clearly highlighted |

# Design

## User Interface Design

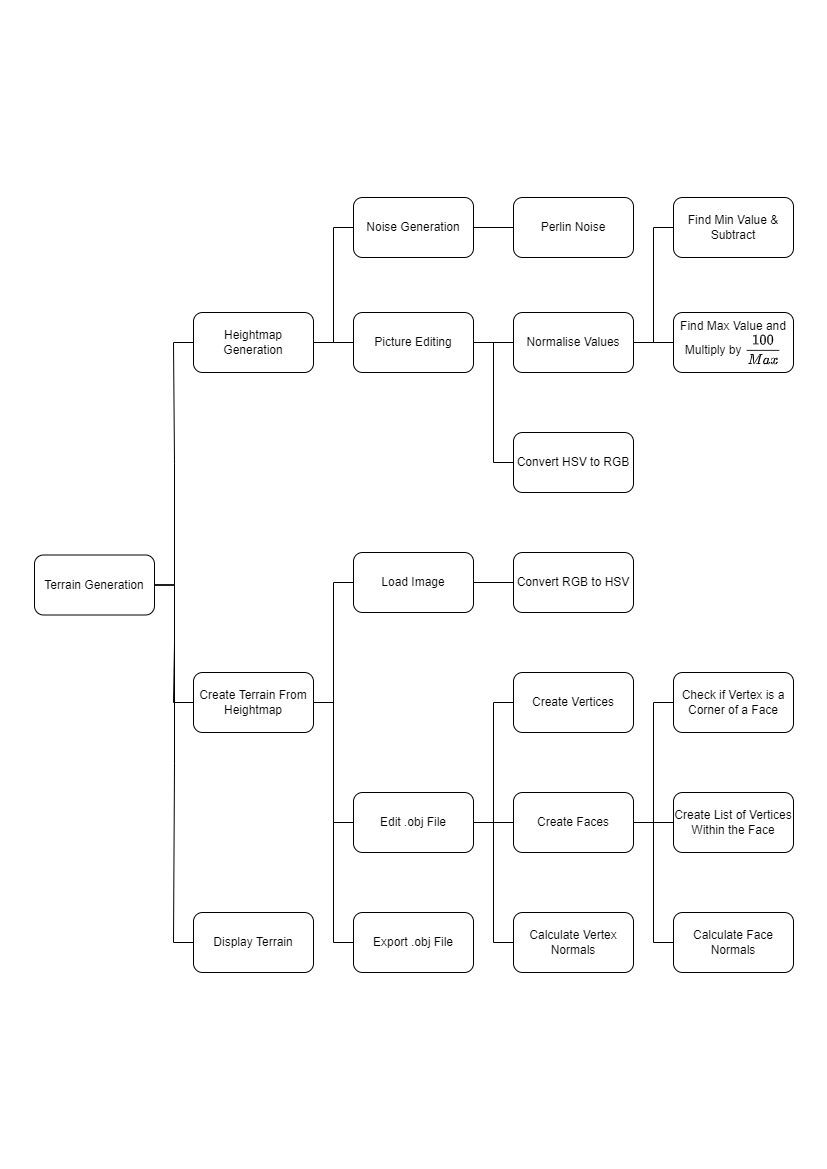
## Stakeholder Input

## Decomposition

This method of decomposing the problem allows me to easily work on the three main components one at a time.

This will be imperative when I am working on the terrain editing (see “Create Terrain From Heightmap” on the diagram), as having separate heightmap generation will prevent me from accidentally breaking the Perlin Noise, for example.

As the terrain will also be generated from a heightmap, it will be easy to implement the functionality of using a predetermined heightmap, as this image will simply be a parameter that could be passed in. Whilst this is not on the success criteria table, Programmer A (the stakeholder) has requested this feature.



## Algorithms

## Inputs & Outputs

## Testing

# Development & Testing

# Evaluation

## Criteria Met

## Usability

## Issues

# Final Code