­

Theo Levison

Suffolk One 19216

Interactive sandbox

Contents

[Section 1 – Analysis of the Problem 4](#_Toc529878897)

[Problem Identification 4](#_Toc529878898)

[Introduction to Problem 4](#_Toc529878899)

[Amenability to Computational Approach 4](#_Toc529878900)

[Computational Methods 5](#_Toc529878901)

[Stakeholders 5](#_Toc529878902)

[Identification of Stakeholders 5](#_Toc529878903)

[Appropriateness of Solution 6](#_Toc529878904)

[Research 6](#_Toc529878905)

[Survey 6](#_Toc529878906)

[Results and Analysis 7](#_Toc529878907)

[Existing Solutions 9](#_Toc529878908)

[Essential Features of the Proposed System 11](#_Toc529878909)

[Limitations of the Proposed System 11](#_Toc529878910)

[Specification of Proposed Solution 13](#_Toc529878911)

[Hardware Requirements 13](#_Toc529878912)

[Software Requirements 13](#_Toc529878913)

[Specification of Proposed Solution (With Justifications) 14](#_Toc529878914)

[Success Criteria 15](#_Toc529878915)

[Section 2 – Design of the Solution 16](#_Toc529878916)

[Top Down Design / Stepwise Refinement 16](#_Toc529878917)

[Diagram 16](#_Toc529878918)

[Justification of Approach and Core Module Functionality 16](#_Toc529878919)

[Interface Designs 18](#_Toc529878920)

[1st Set of Designs 18](#_Toc529878921)

[2nd Set of Designs 19](#_Toc529878922)

[Usability Features 20](#_Toc529878923)

[Data Structures 21](#_Toc529878924)

[Key Variables and Data Structures 21](#_Toc529878925)

[Validation Rules 22](#_Toc529878926)

[UML Diagram 23](#_Toc529878927)

[Algorithms 24](#_Toc529878928)

[Test Strategy 25](#_Toc529878929)

[Overall Approach to Testing 25](#_Toc529878930)

[Iterative Development Milestone Plan 26](#_Toc529878931)

[Data Sets for Testing 27](#_Toc529878932)

[Milestone Test Plans 28](#_Toc529878933)

[Stakeholder Testing 33](#_Toc529878934)

[Section 3 – Developing the Solution 35](#_Toc529878935)

[Milestone 1 - Extract height data from Kinect 35](#_Toc529878936)

[Development Log 35](#_Toc529878937)

[Test Plan 35](#_Toc529878938)

[Review of Milestone 35](#_Toc529878939)

[Milestone 2 - Get useable output from Unity after inputting height data 36](#_Toc529878940)

[Development Log 36](#_Toc529878941)

[Test Plan 36](#_Toc529878942)

[Review of Milestone 36](#_Toc529878943)

[Milestone 3 - On screen coloured height map, ready for projection 37](#_Toc529878944)

[Development Log 37](#_Toc529878945)

[Test Plan 37](#_Toc529878946)

[Review of Milestone 37](#_Toc529878947)

[Milestone 4 - Successful projection onto sand 38](#_Toc529878948)

[Development Log 38](#_Toc529878949)

[Test Plan 38](#_Toc529878950)

[Review of Milestone 38](#_Toc529878951)

[Milestone 5 - Projection updates after sand is moved 39](#_Toc529878952)

[Development Log 39](#_Toc529878953)

[Test Plan 39](#_Toc529878954)

[Review of Milestone 39](#_Toc529878955)

[Milestone 6 - GUI is built 40](#_Toc529878956)

[Development Log 40](#_Toc529878957)

[Test Plan 40](#_Toc529878958)

[Review of Milestone 40](#_Toc529878959)

[Milestone 7 - GUI allows altering of settings 41](#_Toc529878960)

[Development Log 41](#_Toc529878961)

[Test Plan 41](#_Toc529878962)

[Review of Milestone 41](#_Toc529878963)

[Milestone 8 - Hydrology features can be added manually 42](#_Toc529878964)

[Development Log 42](#_Toc529878965)

[Test Plan 42](#_Toc529878966)

[Review of Milestone 42](#_Toc529878967)

[Milestone 9 - Hydrology features can be added automatically 43](#_Toc529878968)

[Development Log 43](#_Toc529878969)

[Test Plan 43](#_Toc529878970)

[Review of Milestone 43](#_Toc529878971)

[Milestone 10 - Gestures can be detected 44](#_Toc529878972)

[Development Log 44](#_Toc529878973)

[Test Plan 44](#_Toc529878974)

[Review of Milestone 44](#_Toc529878975)

[Milestone 11 - Gestures can control feature generation 45](#_Toc529878976)

[Development Log 45](#_Toc529878977)

[Test Plan 45](#_Toc529878978)

[Review of Milestone 45](#_Toc529878979)

[Section 4 – Evaluation 46](#_Toc529878980)

[Testing to inform Evaluation 46](#_Toc529878981)

[Evaluation against Success Criteria 48](#_Toc529878982)

[Evaluation of Usability Features 49](#_Toc529878983)

[Maintenance 50](#_Toc529878984)

[Future Developments 51](#_Toc529878985)

[Conclusion 52](#_Toc529878986)

[Appendix 53](#_Toc529878987)

[Final Source Code 53](#_Toc529878988)

# Section 1 – Analysis of the Problem

## Problem Identification

### Introduction to Problem

Traditional maps are inherently 2-dimensional, making visualising landscapes difficult without specialist software, even more difficult is understanding how changes in terrain can lead to unintended effects on the environment. Without training and time invested in learning to use professional tools, often coming with a large price tag, many people cannot learn easily and quickly what effects terrain changes have.

Simulating and displaying landscapes can be used as an effective educational tool for many people, ranging from geographers to planning developers. My project will allow anyone to intuitively manipulate landscapes and see the effects on the ecosystem in real time.

The hardware of my system will be a sandbox with an Xbox Kinect camera and projector suspended above it, the camera will measure the height of the sand and feedback data to an attached computer.

As the user moves the sand into different shapes, the data will be processed by my software into a topographic map with different coloured elevations and generated physical features like water and rivers. The final map and animated water effect will be projected onto the sand where the user can see what affects their actions have had on the environment and how the water cycle effects the environment.

### Amenability to Computational Approach

Using a computational approach allows me to accurately measure the users input and quickly generate results and easily display them back to the user. Using augmented reality gives infinite reusability, enabling people to see physical systems like rivers without having to replace wet sand and other inconveniences.

Simulating the effects on the terrain will most likely involve complex mathematics that most people would not be able to do in a reasonable amount of time, nor would they be able to visualise their results in an easily understandable way. This approach can delegate all the time consuming calculations to the software and present the finished data quickly and simply.

The user will be able to simulate time passing to see the growth of rivers and how they evolve, that would be impossible without using a computer.

The interaction with hardware will be handled by the software so that it all works seamlessly together, without using a computational approach this would not work.

?????IMPROVE

### Computational Methods

Problem is quickly scanning heights and rendering map with modelled rivers etc

Find an efficient process, find some mathematical formulae for speed…..

Graphical Interface to control rainfall, time.

Divide and Conquer

The use of an object orientated language such as java allows me to easily have multiple physical processes that can all have complex behaviour, stated by a combination of simple classes that will be easy to maintain and implement. I will be able to instantiate nodes for rivers etc easily and manage their hierarchy in an effective manner.

By using thread handling and concurrency I will be able to generate topographical map and rivers simultaneously, so that they are projected at the same time and there is no lag between one appearing and the other. If there were delays between each scene the user could lose their sense of immersion and control, plus an added benefit of concurrency is the possibility of pre-rendering the next scenes as the current scenes are projected, which could increase the refresh rate of the projection.

All hardware-software interfaces will be abstracted away behind the GUI, so that the user cannot alter the settings of the hardware or otherwise break the system.

A user using the GUI will not have to see any calculations because they will be abstracted away…….

?????IMPROVE

## Stakeholders

### Identification of Stakeholders

The users of this system would primarily be students and educators, they would benefit the most from my project so it makes sense to primarily survey them. The sample will be representative of students from sixth form, where I expect the system can be best utilised. Many teachers use practical activities in their lessons to engage students and improve their knowledge; my system is the next step for bringing the world to life and encouraging an interest in geography, by taking real world concepts and bringing them into the classroom.

This means teachers would be the main users of the system which is why I chose to survey three of them, and students will be interacting with the system to learn geography concepts which is why I’m surveying ...

My stakeholders are sixth form geography teachers Abigail Lord and Rachael Hurst, and students …

I will be testing the final system in some of the teacher’s lessons so I will be able to collect feedback from more students than listed here for a general review of how practical the system is, this will inform any adjustments and the final review of the system.

### Appropriateness of Solution

Practical activities reinforce concepts and engage students more than learning from textbooks, but there is often limited opportunity for hands on learning. This problem is especially prevalent in Geography where, barring a field trip, most activities revolve around filling in sheets, a problem which my system aims to solve.

The system is perfect for the stakeholders because it’s portable enough to be used in classrooms, whilst being an engaging and effective tool for teaching. Aimed primarily as a teaching aid when learning about topics like the water cycle and river formation, the system is designed for ease of use and maximum educational potential with minimal effort.

Teachers will be able to craft a scenario for their students to analyse, discuss and manipulate themselves, the level of detail and complexity of the system will be controllable in the software in an intuitive and easy to use interface. Important features can be highlighted and altered so that teachers can isolate the critical information and ensure the students understanding.

## Research

### Survey

These are the questions that I will ask my stakeholders, however I may follow up the survey respondents to ask for elaboration on answers or follow up questions.

#### For Teachers

1. When teaching about the water cycle and rivers, did you use any practical activities?

2. If yes, what were they and do you think they improved the lesson?

3. Do you think being able to create a river system would be beneficial to your student’s understanding?

4. What features would you like to see?

5. Is there anything else that you would like to add?

I want to quantify how often practical activities are used in the classroom so that I can see how useful my system will be, and use any existing activities or requested features to inform my development of the system.

Questions one and two help me optimise the system for educational use by investigating what activities exist in the lesson and how they are used to aid students learning.

Question three checks if my system will be useful.

To inform the usability of the system, question four enquires what features the teacher thinks would be the most useful.

The final question gives the interviewee the opportunity to address any issues or say anything that the questions have not covered.

#### For students

1. When learning about the water cycle and rivers, did you have any practical activities?

2. If yes, what were they and do you think they improved the lesson?

3. Have you ever experimented with creating rivers or landscapes, on a computer or in real life?

4. If yes, what did you do?

5. What do you think about using augmented reality to learn?

6. What features would you like to see?

7. Do you have anything else to add?

I want to see how the student’s needs differ from the teacher so that the final product will strike the best compromise between each side’s requirements.

Questions one and two will tell me what activities the students remember and therefore what was most effective at helping them remember the lesson, I will be able to use that to refine the system to get the best results.

Questions three and four will give me a baseline of what the students will expect my system to provide and I can compare what features I think should be included to what other systems have.

Question five tells me if the student thinks my system would be effective.

The final question gives the interviewee the opportunity to address any issues or say anything that the questions have not covered.

### Results and Analysis

#### Teachers

Abigail Lord

1. When learning about the water cycle and rivers, did you have any practical activities?

“Yes”

2. If yes, what were they and do you think they improved the lesson?

“We use a tarsia to link concepts”

3. Do you think being able to create a river system would be beneficial to your student’s understanding?

“Yes; I used to have a paper mache model, I lost it when I moved house years ago. You could add water to it through different methods: it was good to show the cycle. I now use a concept model off you tube.”

4. What features would you like to see?

“It would be good to see snow capped mountains, forests and farmland, and then a delta. Within this needs to be tributaries feeding into the major river.it would be good to show the catchment of the drainage basin. Watershed. It would be good to have an idea around clouds and the atmosphere. It would be good to also show the water table level. If at all possible that is.”

5. Is there anything else that you would like to add?

“It would be good to add the % of water held in stores across the world in ice sheets, rivers, atmosphere, forest stores, surface water etc”

Rachael Hurst

1. When learning about the water cycle and rivers, did you have any practical activities?

Yes

2. If yes, what were they and do you think they improved the lesson?

I don't currently teach this element of the course; in the past I have got the students to re-enact the water cycle through role play. I have also used a boiling kettle holding Perspex above it.

3. Do you think being able to create a river system would be beneficial to your student’s understanding?

Yes, I think it would. It's such a large concept that it is often hard to grasp. We will often use youtube videos to show an animation to aid learning.

4. What features would you like to see?

Inputs and outputs. water storages. water transfers and percentages.

5. Is there anything else that you would like to add?

It would be good to be able to differentiate - so have different levels for different age groups

It is clear that practical activities are very useful in teaching the hydrological cycle and this system would be useful in a classroom setting. Many of the requested features such as inputs, rivers and drainage basins I had anticipated and planned for them to be included in the final product. A common request that I had not thought about was for displaying levels/percentages of water storage in different mediums such as water held in the atmosphere. Including a water storage level in the ground and rivers is something that could easily be accomplished with small infographics, however showing water in clouds and other more diverse places such as forests and ice sheets would clog up the amount of space, and it would all depend on how the sand was shaped or the topology would not make sense. It is possible that the teacher could create their ideal landscape to showcase everything but then the students would have less input, reducing the interactivity of the system and therefore it’s educational effect.

I will try to incorporate other types of water storage because I can see it is an important part of the curriculum.

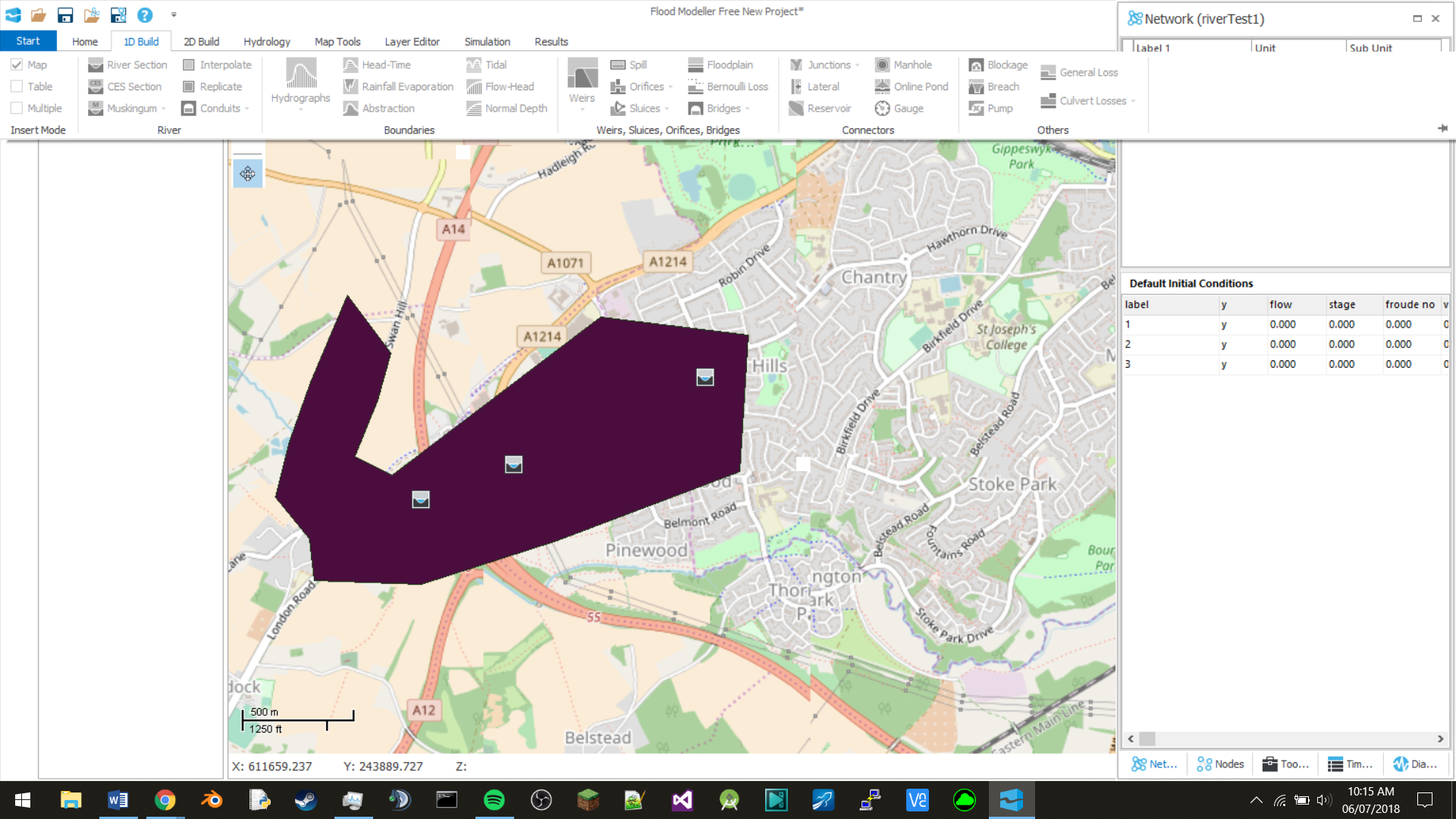
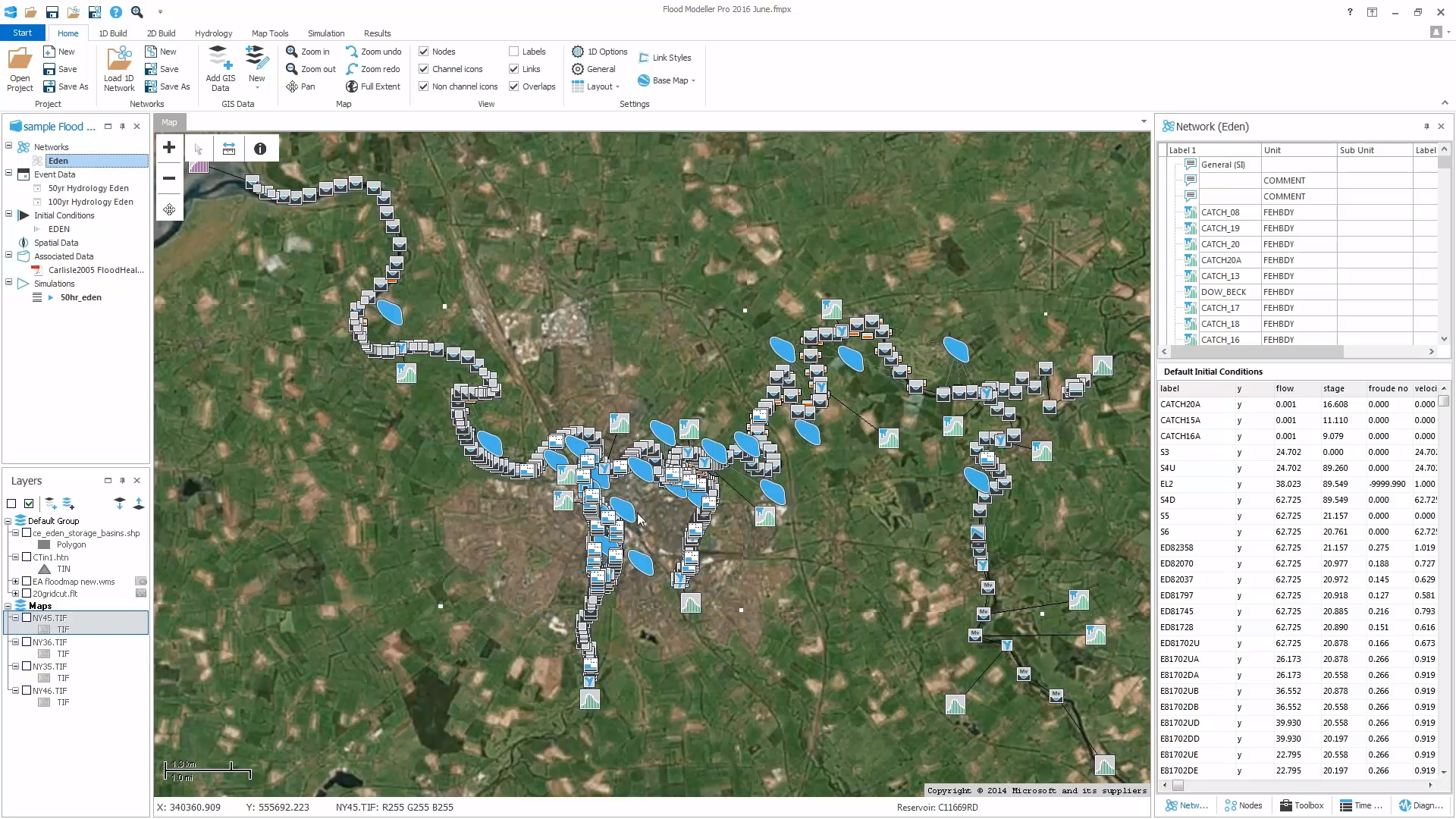
Overall analysis???????????????

### Existing Solutions

**Flood Modeller Pro**

***www.floodmodeller.com***

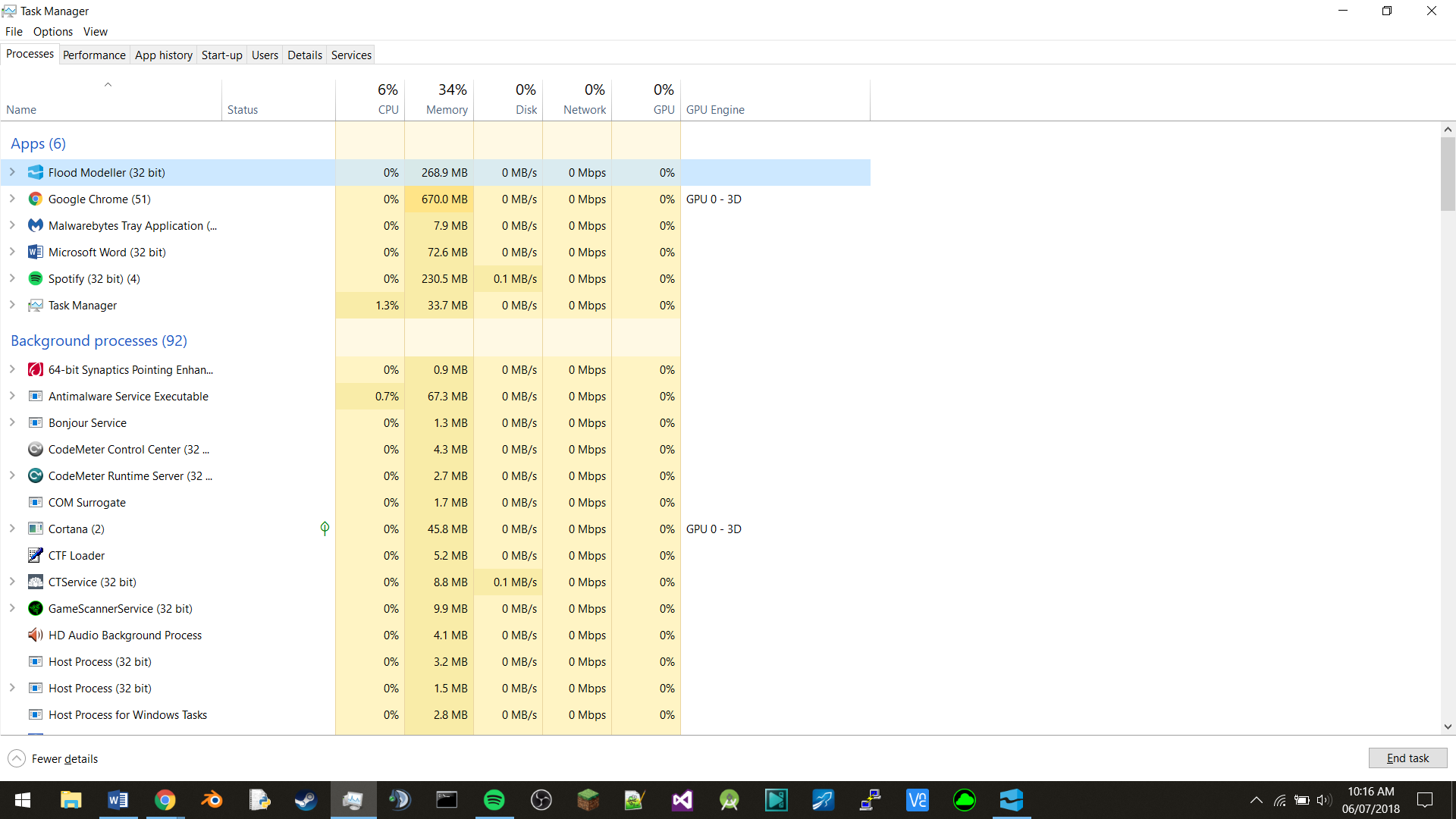
An existing solution is Flood Modeller, this is a professional program for modelling floods using real life map data and 1 2 and 3D tools. I will be testing the free version which is limited to a smaller map area and less dimensional nodes; however, this will still allow me to compare the limits and benefits of this approach to the problem.



#### Overview

There are a lot of advanced features on offer, but the main ones are integration with google maps that allow you to find data on the area you want quickly and easily, and the switching between different dimensions to improve simulation speed.

When detailed simulation is not required you can use 1D nodes and model a river as a line, this means you can have the effects of the river without increasing processing resource consumption by a large amount. Dimensions can be added when needed, this means complexity of the simulation is kept to a minimum but the capacity for detail is conserved.



The system requirements are not extreme; the cpu and gpu are only utilised intensively when the simulation is rendered and the ram is easily within modern computer specs.

The design of the GUI is not very intuitive, there are many complex options such that without reading all of the documentation, it is difficult to work out where to start. It is easy to become confused when dealing with the list of different dimension nodes, 8

#### Application to my project

I will utilise the different dimensions to improve processing speed and I will learn from their confusing menus and ensure that my GUI is easy to use. The minimum specs for my system will probably be higher because of the graphical projection aspect but it should not be extreme. The interface will be more abstracted than this program because the primary control is moving the sand around, any options I have on the computer screen will be obvious and simple to use.

**Interactive sandbox**

<https://www.youtube.com/watch?v=E9aL3HjZbcw>

My project was inspired by the potential of this system, but I wanted to improve upon and create a legitimate purpose for their idea. Sadly, I don’t have access to a physical version, however I can investigate the problems and advantages through youtube and photographs. This can help me to build upon their existing system and bring an effective educational tool to the classroom.

#### Overview

Similar to my project, this utilises hardware and software to create augmented reality, however it lacks the educational focus and is primarily aimed at entertaining people rather than teaching. The visual effects are very striking when combined with the three dimensions of the sand, so when added to the novelty of playing with such a responsive medium, the system is attractive and engaging to it’s users.

#### Application to my project

The sandbox is an easy medium to interact with, a property I have incorporated into my design; utilised in conjunction with more comprehensive software this will allow me to capture student’s attention more easily than traditional methods would allow.

I will use the idea of projected topographical data and build upon it, including a higher level of terrain simulation and extra features such as rivers and lakes, hopefully at only a slight performance cost.

This system is so successful because of how intuitive it is, I will aim to keep that property whilst adding increased functionality, whilst some will be operated through a GUI, it is important that I maintain the sand as the primary method of interaction.

### Essential Features of the Proposed System

After consulting my stakeholders, I have revised the features that the system will offer.

* **GUI that allows altering graphics settings, toggling of features and simulation parameters**

This will allow settings such as rainfall amount, river size etc to be selected, altered and inserted easily into the simulation, this gives the teacher more power to refine the system to suit their needs and ultimately increase effectiveness of the lesson.

If features are not needed for the lesson they can be disabled so that the system runs faster and the visual display is not cluttered unnecessarily.

Less demanding graphics will enable more hardware to run the system if settings are turned down, this would in turn reduce the price of the system and increase accessibility.

* **The option to speed up time**

This would enable the user to see how the water cycle affects the landscape over different time periods, this feature increases usability and educational potential because the students can see long term changes in small time periods, which would not be possible in real life.

* **Highlighting of important features**

Isolating and highlighting the critical information allows the teacher to ensure the students have a comprehensive understanding of the topic they are covering. Including this feature will improve information retention by the learner and give the teacher more power to optimise their lessons to teach the primary concepts of their topic.

* **Etc**

### Limitations of the Proposed System

None?? Probably is

The time taken to generate each frame? Every new feature increases delay between each refresh.

## Specification of Proposed Solution

### Hardware Requirements

Kinect Camera, to measure the height of the sand, there is no other easy way to do this without an infrared camera and this is an easily obtainable, reasonably priced commercial camera.

Projector, required to display the image on a surface, this is the only option for what is needed and is designed to interface with software so is suitable for the task.

I will be using the GPU to render most of the graphics so depending on settings selected and the specs of the GPU, this component will be the limiting factor in performance. The powerful GPU is required because of the complex maths equations that will need to be computed and for fast responsiveness

There will be a small amount of ram and CPU resources used but I do not expect it to be very much, the system should be able to run on any modern CPU and ram.

The software will not take up much space as there is not much to store, physical space is more of a concern because of the sand, box, projector and camera that all need to be stored and transported.

### Software Requirements

There must be fast processing of data so that the system is responsive and there is not a long wait time for your actions to take effect, without this a user could get bored and the immersion of creating the landscape will be broken.

There must be a GUI so that the user can easily interact with the system, this would allow them to toggle features that are needed for each particular lesson and alter virtual simulation settings to do things such as increase time dilation or rainfall.

### Specification of Proposed Solution (With Justifications)

1. My solution will have a physical sandbox that the user can play with as practical, hands on learning, is more effective than other methods.
2. I will use a Kinect camera to detect the shape of the sand, because it is a well-supported commercial product and is accurate enough for my needs.
3. The computer will generate graphics and project them onto the sandbox, so that the user can see info graphics that relate to how they have shaped the sand.

#### Stakeholder Feedback

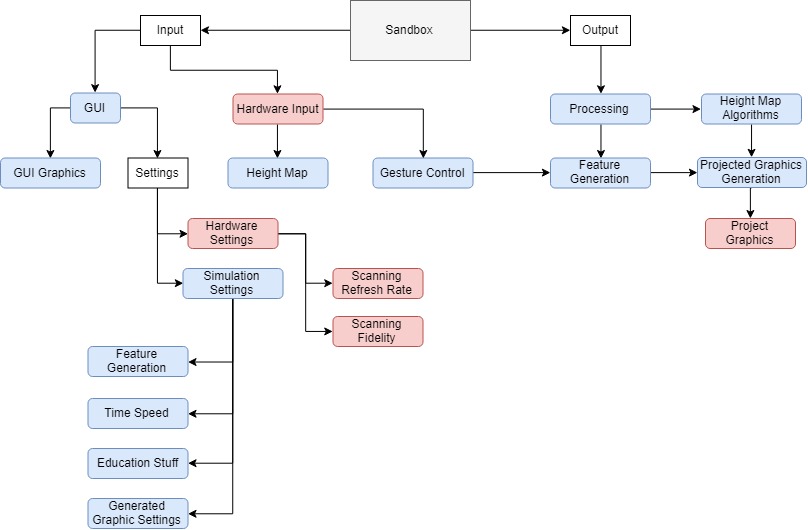
### Success Criteria

|  |  |  |
| --- | --- | --- |
| **Critera**  **No** | **Criteria** | **Evidence of meeting criteria** |
| 1 | Box that holds enough sand for the Kinect to detect, frame to suspend Kinect and projector from | Photographs of hardware installed in the system, box at least 1\*1 meters |
| 2 | Simple setup and calibration, that non-technical people can complete easily | Screenshot of on-screen instructions for an easy setup and calibration, with streamlined controls to control the Kinect’s scanning parameters, stakeholder feedback on ease of use. Rating out of 5 for ease of use |
| 3 | Kinect detects new arrangement of sand after interaction | Screenshots and console output, showing data received from Kinect and transmitted to software |
| 4 | Software processes Kinect data and extracts height data | Screenshots and console output, showing successful isolation of important data |
| 5 | Software can use external process (Unity 3D) to generate colour grading and topographical map ready for projection | Screenshots showing map generated from Unity, colour grading at least 1 shade per 10 cm of height |
| 6 | Addition of automatic hydrological features to the map | Screenshots showing features added to the map in Unity |
| 7 | Projection of map onto sand | Photos of map projected onto sand, showing colour grading and water features |
| 8 | Map is refreshed whenever new data is in the queue, waiting to be projected | Console output showing data waiting to be projected, film of map refreshing |
| 9 | Map refresh rate is a suitable speed and not very noticeable | Films of map refreshing and stakeholder feedback on whether the refresh rate is suitable and/or breaks the immersion, refresh speed at least 3 Hz |
| 10 | Kinect can detect gestures/hands above sand and program can detect when this happens | Screenshots and console output of Kinect data and program detecting when people deliberately use gestures, detection of hand at least half a meter above sandbox |
| 11 | Software can, when activated by gestures, add water to the map, for projection in the next map refresh | Screenshots and photos of gestures controlling water levels/rainfall |
| 12 | Option to set gestures and what they trigger | Screenshots and console output of users changing their gesture controls and their effects |
| 13 | GUI to control settings and start software | Screenshots of GUI showing options to alter settings such as refresh rate and graphics fidelity, rating out of 5 (by stakeholders) for ease of use |
| 14 | Instructions for users | Screenshots of instructions/tutorials on how to use the system |

# Section 2 – Design of the Solution

## Top Down Design / Stepwise Refinement

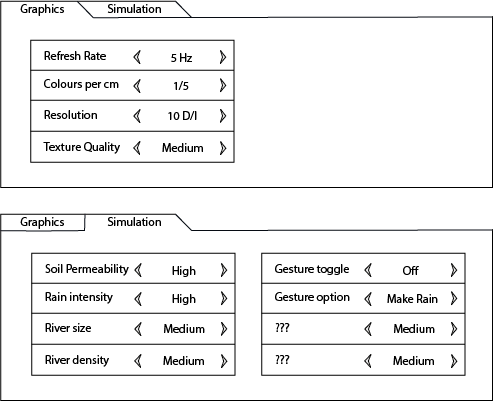
### Diagram

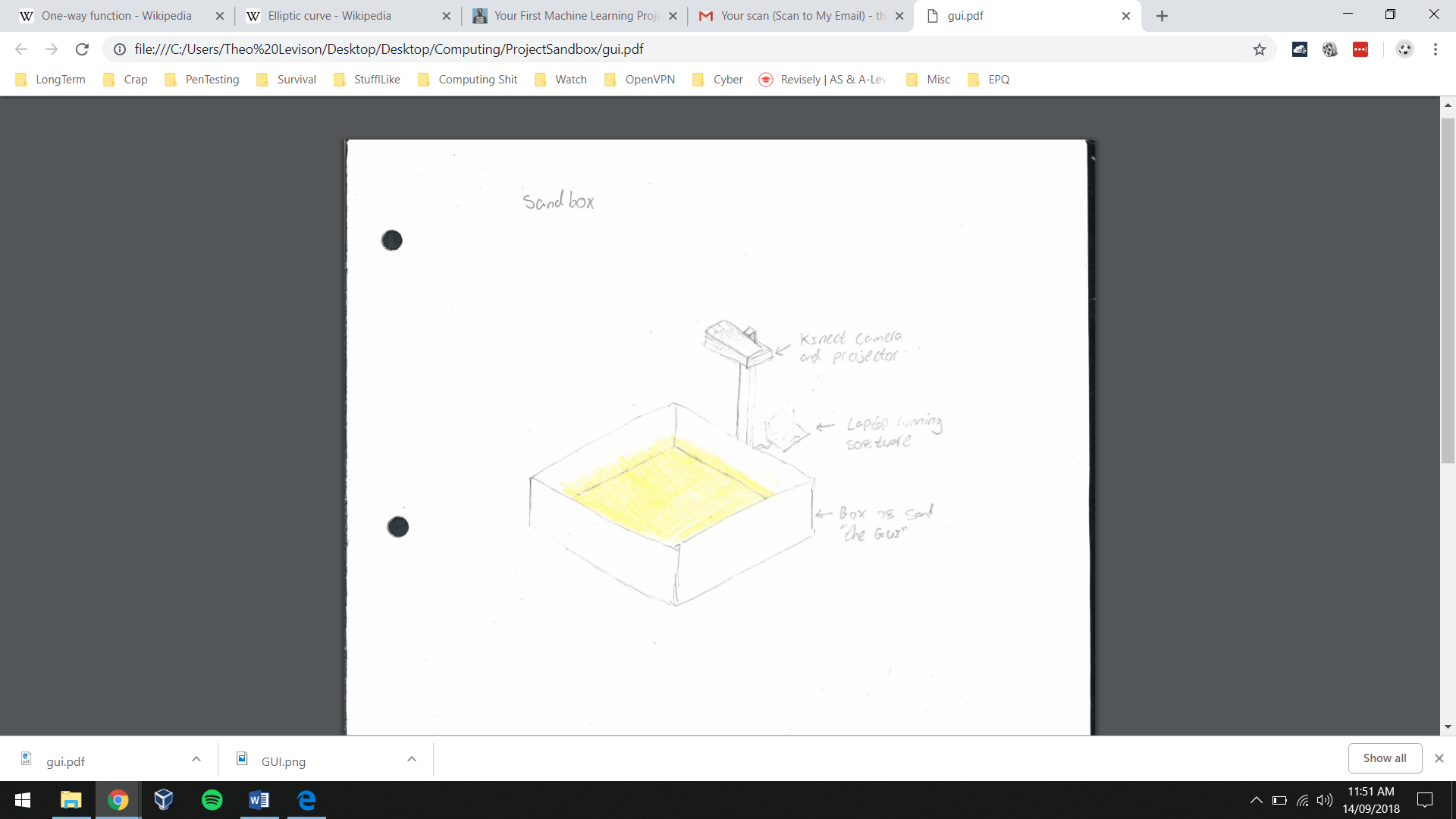


### Justification of Approach and Core Module Functionality

## Interface Designs

### 1st Set of Designs

  
Calibration Screeen !!!!!!!!!!!!!!!!!!!!!!!



#### Stakeholder Feedback

### 2nd Set of Designs

#### Stakeholder Feedback

### Usability Features

Moving sand around is likely to be something everyone has done at some point, this makes the primary interface immediately accessible and intuitive for the vast majority of people (word for relating to real life?). Using augmented reality removes any physical limit to the size of information being displayed, the projector can make words as big as are needed, meaning users that have difficulty reading small font can be easily accommodated.

The only part of the system that might require more technical knowledge is setting up and calibrating the camera-projector tower, I will aim to make this as easy as possible through the inclusion of documentation and a guide to follow to calibrate successfully.

## Data Structures

### Key Variables and Data Structures

Most of the data that I will be handling is coming from the Kinect, I will know the size of the sandbox and therefore how much data I will be receiving, this enables me to use a pre-defined 2-dimensional array, I chose this method of storing data because it is easy to traverse and will be easy to interpret as a heightmap. Other data will be coming from my graphics generation, however I don’t know how much I will have to store, so I will use an array list, the length which can change, so that I can store as much data as I need to.

### Validation Rules

I have used pre-set options in my GUI so that the users cannot give erroneous commands to the software, this means the only organic input that the user can give is through interaction with the sandbox. As all interaction with the sand is valid, unless the user is in the process of moving sand, I do not need to validate the Kinect data besides discarding all but the infrared camera frames.

If the user is still interacting with the sandbox I can save resources by not trying to generate graphics immediately that could be wiped away in the next few seconds, I can detect when a person is using the system because the height of their arms will be greater than the possible height of the sand, then delay the generation of graphics until they have finished.

### UML Diagram

#### Diagram

#### Justification of UML Diagram

## Algorithms

## Test Strategy

### Overall Approach to Testing

### Iterative Development Milestone Plan

#### Milestone 1 - Extract height data from Kinect

This is the first step in my project, this will include some validation of the data and the storage of relevant data in a 2D array

#### Milestone 2 - Get useable output from Unity after inputting height data

I will use Unity to handle colour grading and the 3D modelling of the topographical map, this enables me to easily create the map without worrying about generating my own graphics. The process should not take a long time to complete so that the refresh rate can be as high as possible

#### Milestone 3 - On screen coloured height map, ready for projection

Make sure Unity is outputting the correct colours for the height and inserting height lines, be able to view the map in the Unity editor to ensure it is correct

#### Milestone 4 - Successful projection onto sand

Finish developing the hardware interfaces, ensure Kinect to Projector data flow works so that there will be a useable product. Check if calibration of projector is correct and build calibration interface so that users can set it up themselves

#### Milestone 5 - Projection updates after sand is moved

Refresh rate at least 3Hz, increase refresh rate as much as possible without sacrificing too much graphics quality, check with stakeholders whether graphics is more desirable than refresh rate,

#### Milestone 6 - GUI is built

Users can interact with GUI in an intuitive fashion, ensure stakeholders are happy with the look, feel and usability of the interface

#### Milestone 7 - GUI allows altering of settings

Allow control of the software through the GUI, test that all options can be used together and that there are no errors due to combinations of settings. Limit settings to a useable level, high graphics are pointless if refresh rate is very slow

#### Milestone 8 - Hydrology features can be added manually

Insert objects such as rivers from the console, check that water is projected correctly and is put in the correct places in the terrain

#### Milestone 9 - Hydrology features can be added automatically

Insert objects after analysing height and existing water level, combine the water with the graphical map

#### Milestone 10 - Gestures can be detected

Software can differentiate between a hand above the sand and the surface, has a low margin of error of false positives

#### Milestone 11 - Gestures can control feature generation

Rain, rivers etc can be controlled by moving hand above the surface at a certain height

### Data Sets for Testing

Recorded Kinect data, for reliable repeatable height data?

### Milestone Test Plans

#### Milestone 1 - Extract height data from Kinect

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No | Test Explanation | Success Criteria Point/s | Input Dataset/s | Expected Outcome |
| 1 | Initialise Kinect |  |  | Red lights are illuminated on the Kinect |
| 2 | Frame handler for depth frames receives a frame that is not null |  |  | Prints the frame object received |
| 3 | Depth data array is extracted from depth frame | 4 | Data stream from Kinect | Prints depth values array |

#### Milestone 2 - Get useable output from Unity after inputting height data

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No | Test Explanation | Success Criteria Point/s | Input Dataset/s | Expected Outcome |
| 1 | Script compiles and is embedded in game object successfully |  |  | No errors are returned in the console |
| 2 | Depth data from infrared camera in converted to the correct data type |  | Depth frame from Kinect | Prints an array of float values |
| 3 | Unity terrain object is altered by the input data |  | Array of float values | Terrain in unity changes shape |
| 4 | Unity terrain object is smoothed and looks like the shape the camera is scanning |  | Depth frame from Kinect | Terrain in unity is not jagged, and is recognisably the shape the Kinect is pointed at |

#### Milestone 3 - On screen coloured height map, ready for projection

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No | Test Explanation | Success Criteria Point/s | Input Dataset/s | Expected Outcome |
| 1 | Unity creates splat map to texture and colour generated terrain | 5 | Depth frame from Kinect | Terrain has some kind of colour to it (probably grey) |
| 2 | Splat map changes colour as gradient increases | 5 | Depth frame from Kinect | Terrain colour changes as gradient does |
| 3 | Peak over a certain height has snow caps on | 6 | Depth frame from Kinect | Peaks of terrain over a certain height have white tops |
| 4 | Camera is positioned in Unity correctly |  | Depth frame from Kinect | Game view in Unity shows all of the terrain with correct colours on |

#### Milestone 4 - Successful projection onto sand

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No | Test Explanation | Success Criteria Point/s | Input Dataset/s | Expected Outcome |
| 1 | Projector projects game view | 7 |  | Picture from game view is projected |
| 2 | Projected image is correctly lined up and in proportion with sand |  |  | Projected image projects the right things onto the right places |

#### Milestone 5 - Projection updates after sand is moved

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No | Test Explanation | Success Criteria Point/s | Input Dataset/s | Expected Outcome |
| 1 | When someone pushes the sand around, the projection updates to fit the new sand configuration | 3, 8 | Data stream from Kinect | The projection will update and redraw the coloured topographical map |
| 2 | Updates only after the user has stopped moving the sound | 8 | Data stream from Kinect | The projection will not update when any part of the user is still in the camera space |
| 3 | The update speed is at least 3Hz | 9 | Data stream from Kinect | The projection updates at a suitable speed |

#### Milestone 6 - GUI is built

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No | Test Explanation | Success Criteria Point/s | Input Dataset/s | Expected Outcome |
| 1 | Menu in Unity is shown, before the projector is enabled |  |  | Box appears on the screen, projector does not project the game screen onto the sand |
| 2 | GUI has the layout shown in the design stages |  |  | Menu is on screen with graphics, buttons and labels |
| 3 | GUI has multiple pages with buttons and settings suitable for the sub-menu | 13 |  | Menu is on screen, with multiple selectable menus and suitable options on each page |

#### Milestone 7 - GUI allows altering of settings

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No | Test Explanation | Success Criteria Point/s | Input Dataset/s | Expected Outcome |
| 1 | Buttons are linked to variables in the terrain script and can alter the values | 13 |  | Buttons are clickable have an effect on the terrain script |
| 2 | Labels update to show the current values of the terrain script variables |  |  | Labels show the values in terrain script and update when buttons are pressed |

#### Milestone 8 - Hydrology features can be added manually

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No | Test Explanation | Success Criteria Point/s | Input Dataset/s | Expected Outcome |
| 1 | Console commands generate a river |  |  | Command line input can create a river on the terrain somewhere, coloured blue and with some curves |
| 2 | Multiple available features that can be selected and generated |  |  | More than one feature is placed on the map, you can select which ones you want from the command line |
| 3 | Features are generated in suitable places and look good |  |  | e.g. rivers are generated in valleys and lakes are generated in land locked areas |

#### Milestone 9 - Hydrology features can be added automatically

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No | Test Explanation | Success Criteria Point/s | Input Dataset/s | Expected Outcome |
| 1 | At least one features is generated automatically and projected onto the sand, based only off analysis of the height data | 6 | Data stream from Kinect | River is placed in the correct place with no input besides moving the sand |
| 2 | Multiple features are generated automatically | 6 | Data stream from Kinect | Multiple features are placed in suitable places with no input from the user |

#### Milestone 10 - Gestures can be detected

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No | Test Explanation | Success Criteria Point/s | Input Dataset/s | Expected Outcome |
| 1 | Hands are detected above the sand | 10 | Data stream from Kinect | When the user places their hands above the sand, the program outputs a confirmation onto the console |
| 2 | Specific shapes made by the users’ hands are detected and identified |  | Data stream from Kinect | After the user makes a correct shape a confirmation is printed to the console |
| 3 | The program detects the correct gestures at least 60% of the time |  | Data stream from Kinect | A greater than 60% success rate |

#### Milestone 11 - Gestures can control feature generation

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No | Test Explanation | Success Criteria Point/s | Input Dataset/s | Expected Outcome |
| 1 | If the user makes a gesture that corresponds to a feature, generate the feature on the projection | 11 | Data stream from Kinect | After a user makes a gesture above the sand, the correct feature is generated on the projection |
| 2 | User can create gestures and link them with something they wish to happen | 12 | Data stream from Kinect, User input in the GUI | Through the GUI, the user can make a gesture and link it to an effect |
| 3 | User created gestures work correctly with at least a 40% accuracy |  | Data stream from Kinect | User makes a gesture that they programmed, and it does what they want successfully |

### Stakeholder Testing

#### Test Plan (Set Tasks)

#### Questionnaire

# Section 3 – Developing the Solution

## Milestone 1 - Extract height data from Kinect

### Development Log

Initially I did not start programming in Unity, to understand and utilise the Kinect SDK I began prototyping in visual studio.

I had some issues with finding a working SDK because Microsoft has discontinued Kinect cameras, so many links had suffered from link rot, or were out of date. After finding a working version, I installed the SDK, which came with drivers and pre-created applications in c#, then tested that the Kinect was correctly connected and generating a picture.

INSERT KINECT THING HERE

Working from examples included in the SDK, I wrote code to initialise the Kinect and capture frames from the video stream.

I decided to create a multiple source handler because I wasn’t sure if I would need to handle any other types of frames. It made sense to make my code easily expandable if that situation arose.

#### Validation Routines

### Test Plan

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test No | Test Explanation | Success Criteria Point/s | Input Dataset/s | Expected Outcome | Pass/Fail |
| 1 | Initialise Kinect |  |  | Red lights are illuminated on the Kinect |  |
| 2 | Frame handler for depth frames receives a frame that is not null |  |  | Prints the frame object received |  |
| 3 | Depth data array is extracted from depth frame | 4 | Data stream from Kinect | Prints depth values array |  |

#### Testing Evidence

### Review of Milestone

## Milestone 2 - Get useable output from Unity after inputting height data

### Development Log

The next major challenge was converting the height data into a data type that could be used in Unity’s height map functions.

#### Validation Routines

### Test Plan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No | Test Explanation | Success Criteria Point/s | Input Dataset/s | Expected Outcome |
| 1 | Script compiles and is embedded in game object successfully |  |  | No errors are returned in the console |
| 2 | Depth data from infrared camera in converted to the correct data type |  | Depth frame from Kinect | Prints an array of float values |
| 3 | Unity terrain object is altered by the input data |  | Array of float values | Terrain in unity changes shape |
| 4 | Unity terrain object is smoothed and looks like the shape the camera is scanning |  | Depth frame from Kinect | Terrain in unity is not jagged, and is recognisably the shape the Kinect is pointed at |

#### Testing Evidence

### Review of Milestone

## Milestone 3 - On screen coloured height map, ready for projection

### Development Log

#### Validation Routines

### Test Plan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No | Test Explanation | Success Criteria Point/s | Input Dataset/s | Expected Outcome |
| 1 | Unity creates splat map to texture and colour generated terrain | 5 | Depth frame from Kinect | Terrain has some kind of colour to it (probably grey) |
| 2 | Splat map changes colour as gradient increases | 5 | Depth frame from Kinect | Terrain colour changes as gradient does |
| 3 | Peak over a certain height has snow caps on | 6 | Depth frame from Kinect | Peaks of terrain over a certain height have white tops |
| 4 | Camera is positioned in Unity correctly |  | Depth frame from Kinect | Game view in Unity shows all of the terrain with correct colours on |

#### Testing Evidence

### Review of Milestone

## Milestone 4 - Successful projection onto sand

### Development Log

#### Validation Routines

### Test Plan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No | Test Explanation | Success Criteria Point/s | Input Dataset/s | Expected Outcome |
| 1 | Projector projects game view | 7 |  | Picture from game view is projected |
| 2 | Projected image is correctly lined up and in proportion with sand |  |  | Projected image projects the right things onto the right places |

#### Testing Evidence

### Review of Milestone

## Milestone 5 - Projection updates after sand is moved

### Development Log

#### Validation Routines

### Test Plan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No | Test Explanation | Success Criteria Point/s | Input Dataset/s | Expected Outcome |
| 1 | When someone pushes the sand around, the projection updates to fit the new sand configuration | 3, 8 | Data stream from Kinect | The projection will update and redraw the coloured topographical map |
| 2 | Updates only after the user has stopped moving the sound | 8 | Data stream from Kinect | The projection will not update when any part of the user is still in the camera space |
| 3 | The update speed is at least 3Hz | 9 | Data stream from Kinect | The projection updates at a suitable speed |

#### Testing Evidence

### Review of Milestone

## Milestone 6 - GUI is built

### Development Log

#### Validation Routines

### Test Plan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No | Test Explanation | Success Criteria Point/s | Input Dataset/s | Expected Outcome |
| 1 | Menu in Unity is shown, before the projector is enabled |  |  | Box appears on the screen, projector does not project the game screen onto the sand |
| 2 | GUI has the layout shown in the design stages |  |  | Menu is on screen with graphics, buttons and labels |
| 3 | GUI has multiple pages with buttons and settings suitable for the sub-menu | 13 |  | Menu is on screen, with multiple selectable menus and suitable options on each page |

#### Testing Evidence

### Review of Milestone

## Milestone 7 - GUI allows altering of settings

### Development Log

#### Validation Routines

### Test Plan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No | Test Explanation | Success Criteria Point/s | Input Dataset/s | Expected Outcome |
| 1 | Buttons are linked to variables in the terrain script and can alter the values | 13 |  | Buttons are clickable have an effect on the terrain script |
| 2 | Labels update to show the current values of the terrain script variables |  |  | Labels show the values in terrain script and update when buttons are pressed |

#### Testing Evidence

### Review of Milestone

## Milestone 8 - Hydrology features can be added manually

### Development Log

#### Validation Routines

### Test Plan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No | Test Explanation | Success Criteria Point/s | Input Dataset/s | Expected Outcome |
| 1 | Console commands generate a river |  |  | Command line input can create a river on the terrain somewhere, coloured blue and with some curves |
| 2 | Multiple available features that can be selected and generated |  |  | More than one feature is placed on the map, you can select which ones you want from the command line |
| 3 | Features are generated in suitable places and look good |  |  | e.g. rivers are generated in valleys and lakes are generated in land locked areas |

#### Testing Evidence

### Review of Milestone

## Milestone 9 - Hydrology features can be added automatically

### Development Log

#### Validation Routines

### Test Plan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No | Test Explanation | Success Criteria Point/s | Input Dataset/s | Expected Outcome |
| 1 | At least one features is generated automatically and projected onto the sand, based only off analysis of the height data | 6 | Data stream from Kinect | River is placed in the correct place with no input besides moving the sand |
| 2 | Multiple features are generated automatically | 6 | Data stream from Kinect | Multiple features are placed in suitable places with no input from the user |

#### Testing Evidence

### Review of Milestone

## Milestone 10 - Gestures can be detected

### Development Log

#### Validation Routines

### Test Plan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No | Test Explanation | Success Criteria Point/s | Input Dataset/s | Expected Outcome |
| 1 | Hands are detected above the sand | 10 | Data stream from Kinect | When the user places their hands above the sand, the program outputs a confirmation onto the console |
| 2 | Specific shapes made by the users’ hands are detected and identified |  | Data stream from Kinect | After the user makes a correct shape a confirmation is printed to the console |
| 3 | The program detects the correct gestures at least 60% of the time |  | Data stream from Kinect | A greater than 60% success rate |

#### Testing Evidence

### Review of Milestone

## Milestone 11 - Gestures can control feature generation

### Development Log

#### Validation Routines

### Test Plan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No | Test Explanation | Success Criteria Point/s | Input Dataset/s | Expected Outcome |
| 1 | If the user makes a gesture that corresponds to a feature, generate the feature on the projection | 11 | Data stream from Kinect | After a user makes a gesture above the sand, the correct feature is generated on the projection |
| 2 | User can create gestures and link them with something they wish to happen | 12 | Data stream from Kinect, User input in the GUI | Through the GUI, the user can make a gesture and link it to an effect |
| 3 | User created gestures work correctly with at least a 40% accuracy |  | Data stream from Kinect | User makes a gesture that they programmed, and it does what they want successfully |

#### Testing Evidence

### Review of Milestone

# Section 4 – Evaluation

### Testing to inform Evaluation

#### Stakeholder Test Plan

|  |  |  |
| --- | --- | --- |
| Task | Success/Failure | Comments-Any problems encountered? |
| Setup the sandbox, Kinect and projector |  |  |
| Turn on the system |  |  |
| Calibrate camera |  |  |
| Change settings as required |  |  |
| Start the main program |  |  |
| Interact with sand and get a suitable topographical projection |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

##### Analysis

#### Questionnaire

##### Analysis

## Evaluation against Success Criteria

## Evaluation of Usability Features

## Maintenance

## Future Developments

## Conclusion

# Appendix

## Final Source Code