UNIVERSITY OF CAPE TOWN

MASTERS THESIS

A user interface for terrain modelling in virtual reality using a head mounted display

Author:
Timothy GWYNN

Supervisor:
Dr. James GAIN

A thesis submitted in fulfillment of the requirements for the degree of Master of Computer Science

in the

Department of Computer Science

September 25, 2020

UNIVERSITY OF CAPE TOWN

Abstract

Science

Department of Computer Science

Master of Computer Science

A user interface for terrain modelling in virtual reality using a head mounted display

by Timothy GWYNN

The increased commercial availability of virtual reality (VR) devices has resulted in more content being created for virtual environments (VEs). This content creation has mainly taken place using traditional desktop systems but certain applications are now integrating VR into the creation pipeline. Therefore we look at the effectiveness of creating content, specifically designing terrains, for use in immersive environments using VR technology.

To do this, we develop a VR interface for terrain creation based on an existing desktop application. The interface incorporates a head-mounted display and 6 degree of freedom controllers. This allows the mapping of user controls to more natural movements compared to the abstract controls in mouse and keyboard based systems. It also means that users can view the terrain in full 3D due to the inherent stereoscopy of the VR display. The interface goes through three iterations of user centred design and testing. This results in paper and low fidelity prototypes being created before the final interface is developed.

The performance of this final VR interface is then compared to the desktop interface on which it was based. We carry out user tests to assess the performance of each interface in terms of speed, accuracy and usability. From our results we find that there is no significant difference between the interfaces when it comes to accuracy but that the desktop interface is superior in terms of speed while the VR interface was rated as having higher usability.

Some of the possible reasons for these results, such as users preferring the natural interactions offered by the VR interface but not having sufficient training to fully take advantage of it, are discussed. Finally, we conclude that while it was not shown that either interface is clearly superior, there is certainly room for further exploration of this research area. Recommendations for how to incorporate lessons learned during the creation of this dissertation into any further research are also made.

Acknowledgements

Many people helped me both directly and indirectly during the course of this project. Of course nothing could have happened without my supervisor Prof J Gain who helped define the vision for this project and then guided it along the way. His feedback on my writing has been invaluable and I am especially thankful for him putting up with me pushing every deadline to its limit.

Thanks also to my family for being patient with me as this project has seemed at times to extend indefinitely and for always being ready to listen and provide advice on random issues that I probably provided far too little context for at the time.

To all of my friends that helped me with advice along the way particularly to Riesna has also always been supportive and put up with me cancelling plans at the last minute as well as listening to all my complaints when things were not quite going as smoothly as I thought they should. Thank you must also go to Jacques who I often went to with some of my more technical complaints, he has been a great sounding board for some of my ideas. Also to everyone in the lab and especially the VEX research group who have offered feedback on the fly as I wander around making a nuisance of myself. Finally, thank you to everyone who helped me review my writing in the final stages of this project.

Contents

List of Figures

List of Tables

List of Abbreviations

AAAD	Action at a Distance
CAD	Computer Assisted Design
CAVE	Cave Automatic Virtual Environment
DH	Dominant Hand
DOF	Degrees of Freedom
FOV	Field of Vision
FPV	First Person View
GLMM	Generalised Linear Mixed Model
GV	God View
HMD	Head Mounted Display
LMM	Linear Mixed Model
NDH	Non-Dominant Hand
POV	Point of View
PSSUQ	Post-Study System Usability Questionnaire
SHI	Single-handed Interaction
THI	Two-handed Interaction
UCD	User Centred Design
VE	Virtual Environment
VR	Virtual Reality
WIMP	Windows, Icons, Menus, Pointer

Chapter 1

Introduction

1.1 VR as an Emerging Medium

Virtual Reality (VR) has now existed as both a concept and a technology for a number of decades. In fact the term 'Virtual Reality' was coined in 1989 by Jaron Lanier after the basic concept of VR, and even working examples, had been in existence for some time [7] . The first working VR head mounted display (see figure 1.1), known as the Sword of Damocles, was created in 1968 [44]. Early VR equipment was large and heavy: the Sword of Damocles was so named because it had to be hung from the ceiling to support the weight of the headset. Because of this, as well as the expense, VR has mainly been used within the domain of academia and military simulations until recently [62].

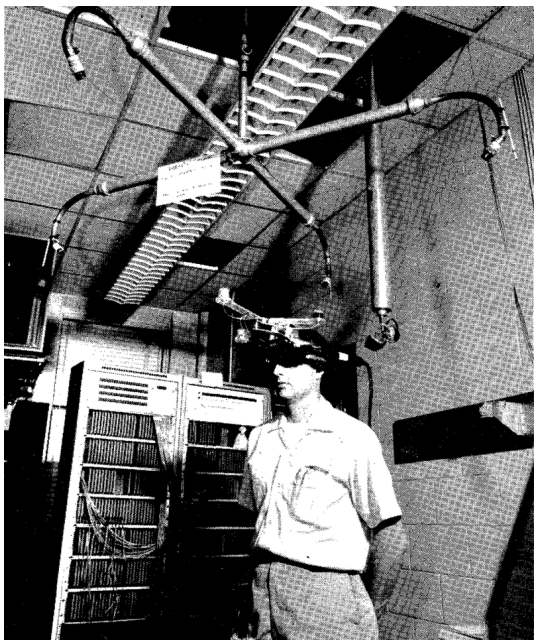


FIGURE 1.1: The Sword of Damocles - the first head mounted display [62]

However, with the release of products such as the Oculus Rift, HTC Vive and Playstation VR (see figure 1.2), often referred to as high end headsets, the technology reached the public consumer market. These devices are far more affordable than earlier technologies and are light and comfortable enough to allow for long periods of use without any weight supporting equipment. They also provide significantly greater definition and field of view (FOV) than earlier devices[49, 31] . These modern headsets or head mounted displays(HMDs) are also paired with hand held input devices which are tracked in 3D space,

allowing users to interact with virtual environments(VEs). These devices are often called 6-degree of freedom (DOF) controllers as they have 3 degrees of movement and 3 degrees of rotation. The input devices also include a number of control surfaces such as buttons, track pads and joysticks for more complex user input.

FIGURE 1.2: Modern VR stereoscopic headsets

In addition to these high-end headsets there are a number of low-end VR options generally termed *Mobile VR*. These often use a headset that users can slot a smartphone into as both a computing and display device, allowing users a more affordable experience of VR. However, they provide a less polished experience compared to the high end headsets.

The current wave of VR devices was initially marketed as an entertainment revolution. So far this has failed to materialise in terms of widespread adoption. There may be various reasons for this, such as the cost of the headsets themselves as well as the necessity of an expensive PC to run the devices. Additionally, prospective users have complained about the lack of a 'killer app' for the medium. Despite this the companies behind the initial headsets are continuing production and the variety of headsets is rising as the prices drop. Statistics from *Steam* (shown in figure 1.3), a popular gaming platform that supports VR, show that, although VR users constitute less than 1% of active user accounts, the number has doubled during 2018 from 0.4% to 0.8% [29]. As of 11 June 2019 this trend continues with the latest steam hardware survey (May 2019) showing .99% of users with VR headsets. [67]

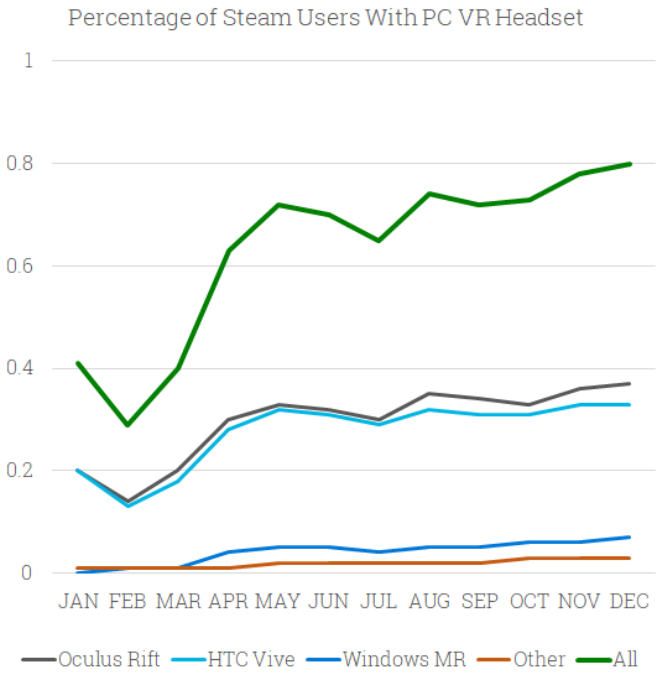


FIGURE 1.3: Change in number of Steam users with VR headsets during 2018 [29]

Furthermore, VR is not solely an entertainment medium. It is also a social experience with apps such as VRChat being popular with thousands of users. Additionally VR can be used within industry for a number of 3D digital tasks such as creating 3D models, presenting prototypes in 3D and much more.

1.2 Modelling in VR

VR has already found applications as a technology for modelling 3D objects. Applications such as Medium, Google Blocks and Tilt Brush allow users to create and colour objects within the 3D space of VR. Although these applications include many of the tools found in traditional desktop 3D modelling programs they were built specifically for use in VR. However, not all VR applications were originally intended for VR but rather had a VR mode added. For example, Unity3D, a popular game creation platform, now includes EditorVR which allows users to switch between a desktop interface and a VR interface.

FIGURE 1.4: Screenshots of the interface of Medium(top) and EditorVR(bottom)

These different approaches to interface design have their own strengths and weaknesses. Dual-interface applications, such as Unity3D/EditorVR, allow for fast switching between the interfaces and users who are familiar with the desktop version will likely be quick to pick up the controls for the VR interface. However, the downside is that trying to reproduce an existing desktop interface in VR can lead to clunky interaction in certain circumstances [28]. For example, Unity3D has many text-based interface elements that require the user to interact with a virtual keyboard in VR as shown in figure 1.4. Typing on this keyboard is slow as the user can only press one key at a time [28]. In contrast VR specific applications are designed to minimise the need for text input, relying mostly on slider widgets which can be grabbed and dragged to change numerical values. However, when using such a VR-only application the user cannot easily switch to a desktop interface should the need arise. It would therefore be necessary to save the 3D artefact and then open it in a separate program. This can be slow (loading complex 3D models from disk can be a lengthy process) and may lead to compatibility issues. This reduces the efficiency of collaborative work as only one user can effectively see the model and interface at a time.

Despite some of the weaknesses detailed above, there are some innate advantages to modelling in VR. By tracking hand movements and some gestures these systems allow for more natural spatial interaction with 3D objects. Additionally, due to the stereoscopic nature of VR headsets the user can see the virtual objects they are creating in full 3D rather than as a projection on a 2D screen. Finally, by tracking head position the applications can move the camera to match head movements or rotations which removes or reduces the need for the user to manage the camera movement manually. It is because of these advantages, among other potential benefits, that there has been significant interest in modelling in VR. The variety of existing applications is strong evidence of this.

Unfortunately, there is little hard evidence that undertaking modelling in VR is actually better than using traditional systems with regards to either work efficiency or user satisfaction. We therefore intend to determine whether this is the case or if, aside from the novelty of VR, users are better off sticking with desktop modelling applications. Since 3D modelling is a broad field with many sub-areas, including character modelling, building simulations etc., we chose to limit our research to a specific type of 3D modelling, namely virtual terrain creation.

1.3 Creating Realistic Terrains

Virtual terrains are a specific type of 3D model, characterised by their large size and need for realistic detail. This dissertation will specifically focus on terrains considered are bare earth terrains as shown in figure 1.5. Only the height of the terrain is represented by the model and surface details such as plant life, loose rocks or bodies of water are not included. Realistic terrains are useful in a number of areas, including special effects for movies, 3D games for PCs and consoles, and terrain-based simulations such as water runoff prediction.

Creating virtual terrains requires a specific tool-set that allows users to easily scale, move and rotate the terrain so that they can select features to modify. Users also need to be able to make fine adjustments to the terrain so that desired landscape elements can be accurately recreated. Because of the contrast between the scale of a terrain and the need for fine detail, creating virtual terrains can be a very time consuming process. Any terrain creation tool therefore needs to maximise the speed at which a user can work, while not sacrificing the achievable accuracy.

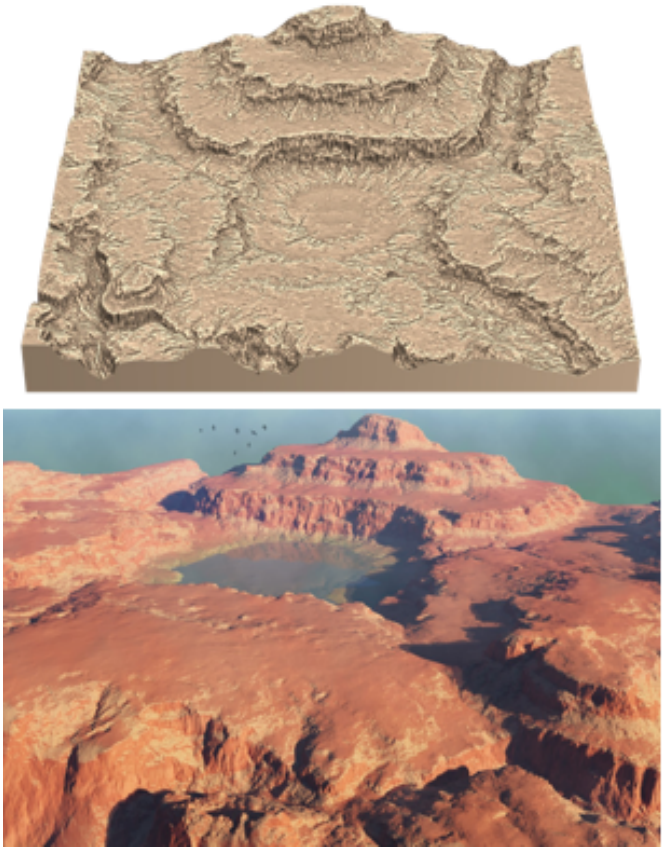


FIGURE 1.5: An example bare earth terrain model with textures added below [24]

1.4 Research Questions

Focusing only on modelling 3D terrains allows us to specify a well defined goal for this research. Specifically, we want to establish whether using a VR interface has advantages over using a "windows, icons, menus, pointer" (WIMP) desktop interface for interactively modelling terrain. The primary research question is therefore: *Is it advantageous to use a VR interface with a HMD and 6-DOF controllers as compared to a WIMP desktop interface for terrain modelling?* As discussed in the section above speed and accuracy are key to an effective terrain creation tool and usability is an important aspect of any human-computer interface. Therefore, in the context of this thesis, we define advantageous as faster, more accurate or more usable. These aspects can be measured independently and compared to one another and therefore we supply three sub-questions.

1.4.1 Modelling Time

This refers to the time it takes for a designer to create a terrain using a given system. Although simple to measure, it is somewhat difficult to establish when a particular design is finished. Our technique for determining this factor will be elaborated on in the methods section. Our research question is therefore: **Is it faster to create a terrain model with a predetermined set of features in VR or using a desktop?**

1.4.2 Modelling Accuracy

We define accuracy as how closely a model matches a concept image. Because we are concerned with faithfulness to concept rather than creating precise copies we will be measuring conceptual similarity by using a panel of human participants. This is opposed to using physical similarity, which would be measured by methods such as computing volume differences. The reason for this is that physical similarity could easily produce misleading results. If for example a user perfectly recreated a terrain but shifted all features uniformly a small distance left then a volume difference check would show a large discrepancy despite the terrain appearing very accurate visually. Our research question is therefore: **Are terrains models created in VR visually closer to a target terrain image than those created with a desktop system?**

1.4.3 Usability

Usability refers to how easy a system is to use for its intended purpose. This can be broken down further into the quality of the interface, the quality of the information provided to the user, the usefulness of the tools available, and overall user satisfaction with the system. With regards to usability our research question is: **Do users rate the usability of the VR system more highly than that of the desktop system?**

Although our research will focus on the specific questions above, it may also suggest further areas of research or investigation. For example, it may allow us to recommend effective VR interface design techniques for 3D design. Additionally, it may indicate that HMD technology is useful for content creation in general in the area of 3D design.

1.5 Outline

In the following chapters of this dissertation we will attempt to answer the research questions outlined above and shed light on the potential usefulness of VR as a medium for content creation. In our background chapter some of the research in areas related to creating virtual terrains is presented as well as research on creating effective interfaces for VR. The existing research which is most similar to this dissertation is then discussed in the Related Work chapter. Next we detail how the initial VR interface was designed based on previous examples and then improved through a user centred design process in the chapter on Interface Design. This is followed in the Experiment Design chapter by an explanation of our experimental procedure. The results of our experiment and a discussion thereof can be found in the Results chapter. Finally we summarise the value of this research in the Conclusions chapter as well as suggesting improvements and alternative paths for potential further research.

Chapter 2

Background

2.1 Introduction

As suggested by the title this research incorporates both terrain synthesis and VR interface design. Therefore, a solid understanding of these topics is required before we progress with our own research in this area.

In terrain synthesis the focus is often on various algorithms for effective terrain generation. However, there are a number of papers that also consider suitable interfaces for user control of the terrain [25]. This aspect is particularly relevant to us as it helps establish the criteria for an effective interface for terrain modelling. Similarly, there is a large body of research into the design of large scale VEs that, while not always directly relevant, nevertheless contains useful information. Even in cases where research has been carried out with now outdated technology or for different use cases, certain universal concepts still apply. For example, even though the risk of simulator sickness has been reduced by hardware improvements that have allowed for higher frame rates it is still an issue that must be considered during the design of VR applications. This section focuses on the areas of terrain synthesis and VR interface design research that are relevant to us.

2.2 Terrain Synthesis

Virtual landscapes are an important component in representing natural environments for applications such as computer games, film, and simulation [24]. While it may be sufficient to use scans of real world data for certain applications, such as simulation, this is not the case for games and film. For computer games in particular, game creators need to be able to author their own unique terrains with specific constraints, such as key visual features and navigable routes for players to traverse. Even in film, particularly in the fantasy and science-fiction genres, it may be necessary to create terrains that do not resemble anything found naturally.

In these cases it is necessary for a virtual terrain to be created from scratch. However, modelling the required micro-features manually can take a huge amount of time [12]. In addition, even unique terrains in games or fantasy films require a certain level of realism. The end user must believe that the terrain is a natural part of the virtual world. Any glaring inconsistencies, such as completely flat ground, are likely to reduce user immersion in the virtual experience. Therefore, there has been considerable research into both realistic terrain synthesis and methods for allowing users to efficiently control the output.

2.2.1 Synthesis Methods

A number of procedural terrain generation techniques have been developed to address the problem of adding realistic micro-features to virtual terrains. These can be broadly categorised into three major areas: procedural generation, simulation, and example-based methods [25].

Procedural generation techniques are based on algorithms not related to physical phenomena, such as erosion [25]. Noise-based synthesis [36, 21]is an example of procedural generation which uses a random noise function such as Perlin noise [51] to modify a height map.

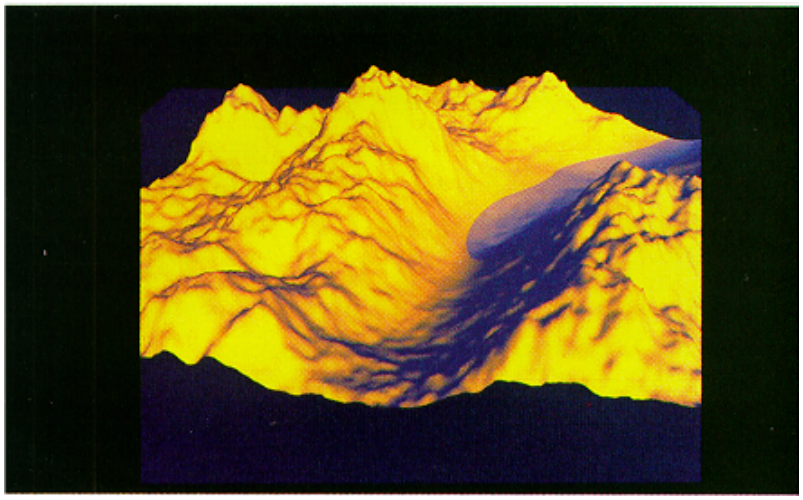


FIGURE 2.1: Procedural generation: An example of noise-based terrain synthesis [36].

An example of a simulation technique is erosion based synthesis [3]. This, and other simulation techniques, aim to achieve greater realism than noise-based synthesis by deforming terrain based on geomorphological effects. Erosion simulation, in particular, achieves this by simulating the hydraulic erosion of top-soil as shown in figure 2.2. One challenge with simulation methods is user control. Even though the user sets the initial conditions of the simulation and terrain shape it can be hard to predict how the evolving simulation will affect the final outcome [25]. As a consequence, the user may need to perform many trial and error runs of the simulation process before achieving a satisfactory result.

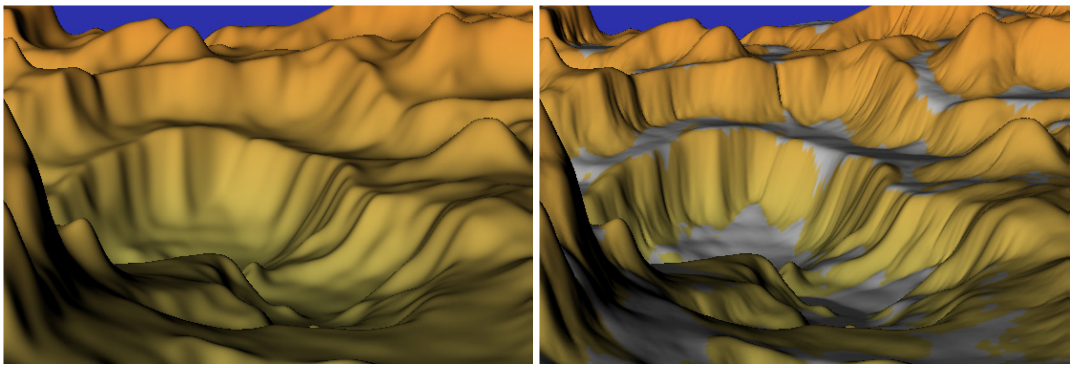


FIGURE 2.2: Simulation: A terrain before (left) and after(right) erosion based synthesis [3].

Finally, example-based generation techniques, such as patch or texture-based synthesis [16, 63], create unique terrains by merging patches of terrain drawn from a database of real world exemplar terrains an

example of which is shown in figure 2.3. This approach is able to simulate terrain that is indistinguishable from real terrain examples[24]. Another advantage of these techniques is the level of user control. Since the terrain patches can be drawn such that they match user-defined constraints they are unlikely to result in unpredictable artefacts. However, this does depend on the variety and suitability of the terrain examples in the database [25]. Specifically, it is impossible to reliably generate terrain features that do not exist in the database examples.

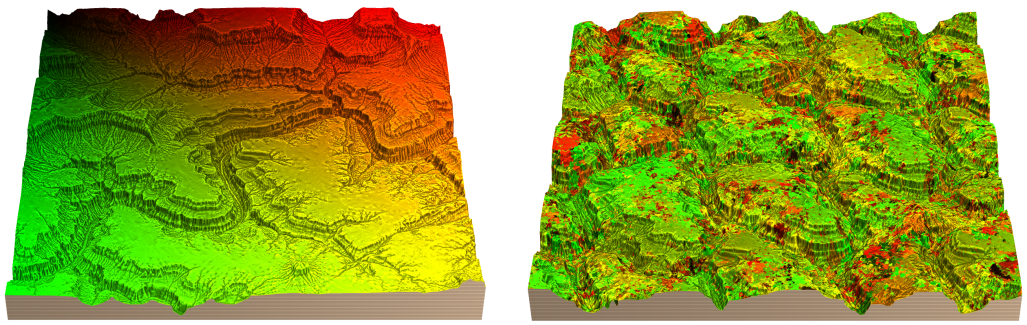


FIGURE 2.3: Example-based generation: An exemplar (left) is mapped to a modelled terrain (right) using texture-based synthesis. The colour marking shows how patches of the exemplar are split up and and used where necessary on the modelled terrain [24].

It is important to note that the more realistic methods often require more computational power. This makes them harder to optimise sufficiently to execute in real time, which is necessary for interactive terrain modelling.

2.2.2 Modelling Methods

Although synthesis allows the automatic generation of finely-detailed virtual terrains the user still needs to be able to specify the features they desire. Such features include mountains, ridges, valleys, or even a specific type of terrain. While interfaces generally allow the user to specify most of these features, while the synthesis system fills in the details, the controls often vary. We will discuss some of the common terrain modelling tools below.

Painting, is a popular method for allowing users to interactively model terrain. It can be used to set terrain types in models [24], edit the height of terrain elements [66] or even add layers of material onto a terrain [68] as shown in figure 2.4. This is due to the simplicity of the interface and the ability to quickly change between making large or small changes by adjusting the size of the paintbrush area. One downside, however, is that merging two different painting types (be this material, height, or some other attribute) can be challenging. For example, if a user wanted a certain area to be a mix of swamp and forest types they would need to manually specify a large number of small points very close to each other unless the interface specifically allowed the user to combine types together.

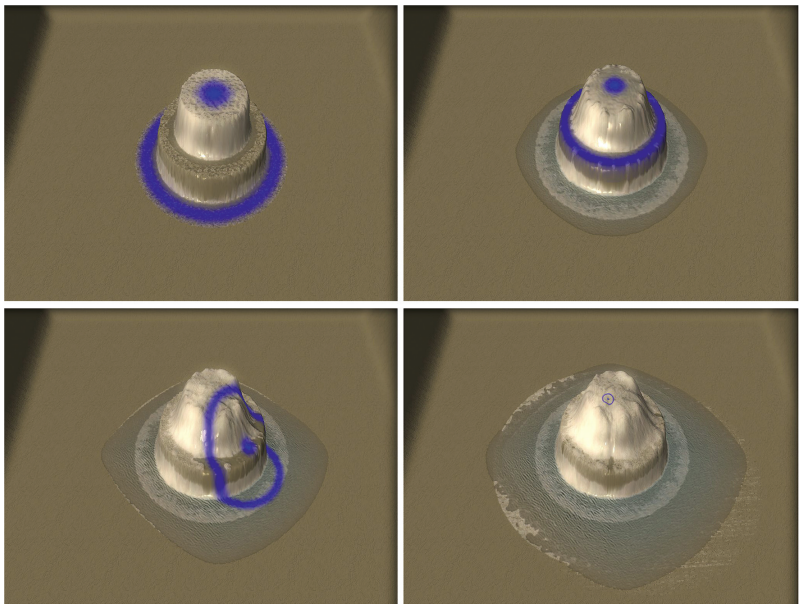


FIGURE 2.4: An example of a painting interface being used to adjust an area of terrain by simulating rainfall [68].

Other systems, such as that shown in figure 2.5, use vector fields to specify terrain features [13]. In these systems the user specifies a vector field that defines a certain portion of the terrain. This vector field can then be given particular properties such as height or even a type such as a 'road' vector field. This enables a wide variety of features to be added to the terrain through a single consistent process. It also allows for interaction between vector fields. For example if two roads cross an intersection can be created automatically. However, since each point of the vector fields must be specified manually it requires more user input than some other modelling tools.

FIGURE 2.5: An example of Vector fields outlining bodies of water in a terrain. The position of water bodies can be adjusted by moving the points defining the vector field. [13]

Both painting and vector field specification methods are effective for making changes to terrain, but require a top-down view of the terrain. To allow users to edit terrain from a first person point of view (POV) a different tool is required. Silhouette sketching is a technique that allows users to sketch a horizon curve, which the system then uses to create a number of constraints to be obeyed by the synthesis process [23]. This technique can either use depth cues to place the sketched terrain features at the correct depth from the user [64] or allow the user to select from a number of possible interpretations of the sketched elements [50]. While this method is suited for a first person POV it can also be modified for use in a top-down view, which gives the user additional control over the terrain features being sketched as shown in figure 2.6 [23].

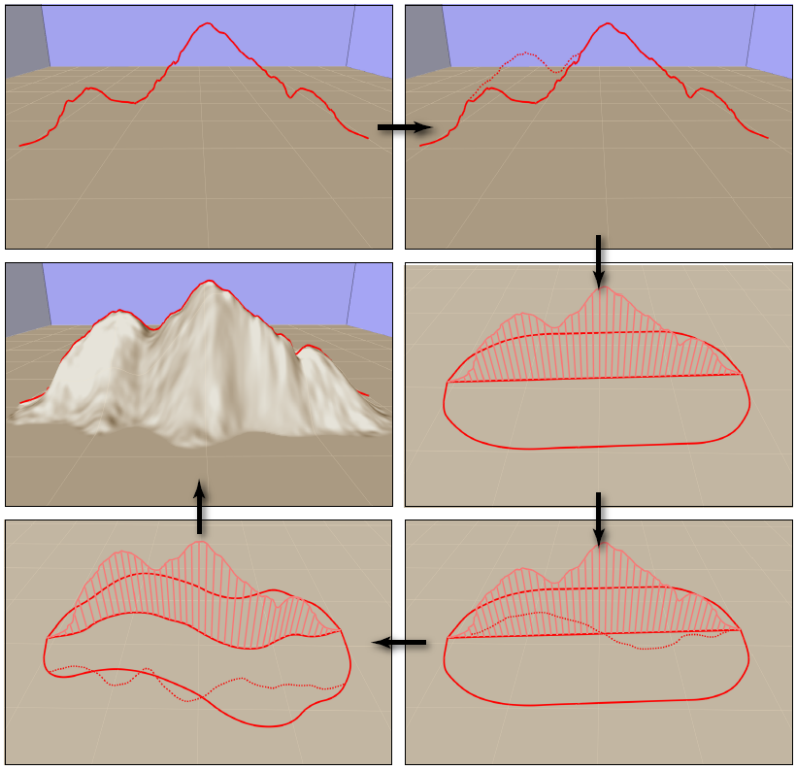


FIGURE 2.6: An example of a terrain sketching system interface [23].

Gain et al.'s terrain synthesis system [24] allows for the addition of constraints into the terrain synthesis process using a number of tools. Users can add and modify constraint points and curves on a landscape, shown in figure 2.7, thereby creating landscape features such as hills, ridges and valleys. This facilitate the specification of the macro features of the terrain, while an automated texture synthesis process adds realistic detail. While the controls are simple, consisting of points and curves which can be modified in terms of height, width and angle, a user can nevertheless create complex geographical features such as calderas and plateau-topped mountains.

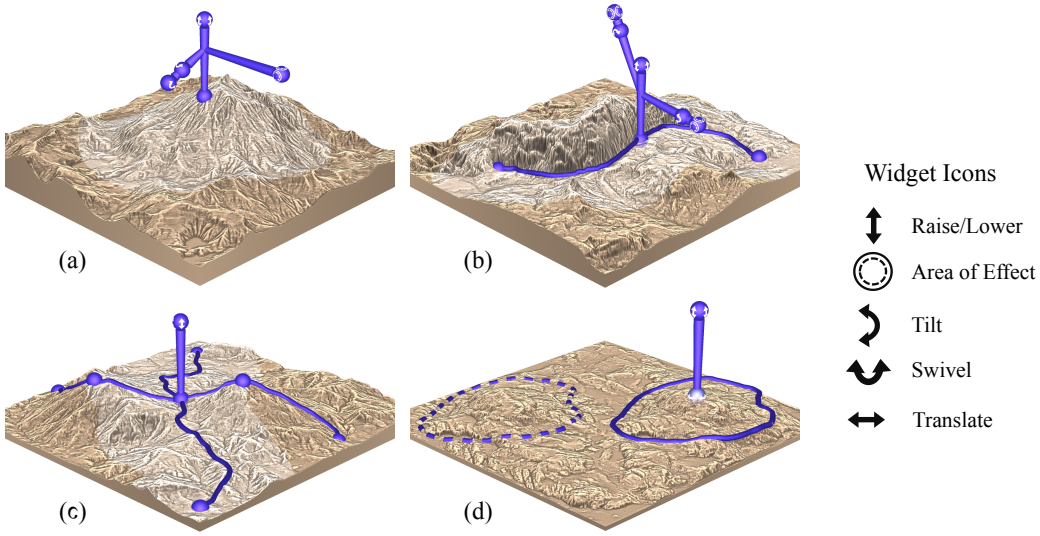


FIGURE 2.7: Some examples of the constraint widgets used in Gain et al.'s system [24].

The system also uses type constraints, which can be painted onto the landscape to define areas of a certain type, such as 'canyons', 'rolling hills', etc. This allows for the modification of how the texture synthesis process is applied. For example an area classified as 'canyon' will draw from exemplar terrains also classified as 'canyon' type. To apply these constraints users have access to a combination of sketching, painting and 3D widget interface tools [23].

2.3 Large Scale VE Interface Design

The scale on which terrain synthesis occurs is unavoidably large, a typical virtual terrain may represent a 100km^2 area. It is therefore important to know how to design an interface which is suitable for such a scale. This is opposed to a small scale interface which might be used to create much smaller objects such as props for a game or animated film. Specifically, an interface that allows interaction with a virtual space much larger than the user, is needed. A review of the relevant literature suggests there are three important aspects to consider when designing such a large scale VE interface: User navigation and movement, Interface interaction and Environment interaction [10]. Respectively, these allow the user to move within the VE, select their mode of interaction and then interact with elements of the VE. Below we look at each of these aspects in detail and consider some of the successful, and less successful, implementations in each case.

2.3.1 Navigation

Within a large scale VE, *navigation* refers to the user's ability to determine their location in the VE and to change this location in a controlled manner [10]. During content creation, in particular, it is essential that the user is always aware of their position and can move around within the VE rapidly. This allows the user to ensure that any elements they create are placed correctly and that they can move from one area of work to another efficiently. It also allows them to accurately simulate the planned end-user experience and view-point.

Unfortunately, it is common for users to become disorientated and lost in a VE [18]. There may be a number of reasons for this, such as the user being unfamiliar with the environment or insufficient location and orientation information being available.

Using landmarks is a common and natural method for everyday physical navigation. Consequently, there has been considerable research into effectively using them as navigational aids within VEs. One study investigated whether users orientate themselves via local or global landmarks [60]. Global landmarks, such as distant mountains, are visible from a distance and can be used in much the same way as a compass to give directional information. Local landmarks, are only useful from small distances and are often used for relative positioning. An example of using both might be: "Head north towards the tall mountain(global) until you get to the red building(local) and then your destination is two doors to the left". It was found that while users do not consistently use a single type of landmark, they rarely use both local and global types simultaneously. This suggests that users tend to use the most visually distinct landmarks and prefer landmarks that are not occluded. Figure 2.8 shows user preference with regard to local or global landmarks when the landmarks in a known area were adjusted to conflict with one another. In location A the global landmarks were not occluded and the local landmarks were buildings with few identifying features, while in location B the global landmarks were partially occluded and the local landmarks were visually distinctive buildings. This implies that for navigation, interface design should provide a variety of landmarks, both global and local, that allow the user to always have a distinctive point of reference. It also suggests that global landmarks un-occluded by local geometry should be preferred.

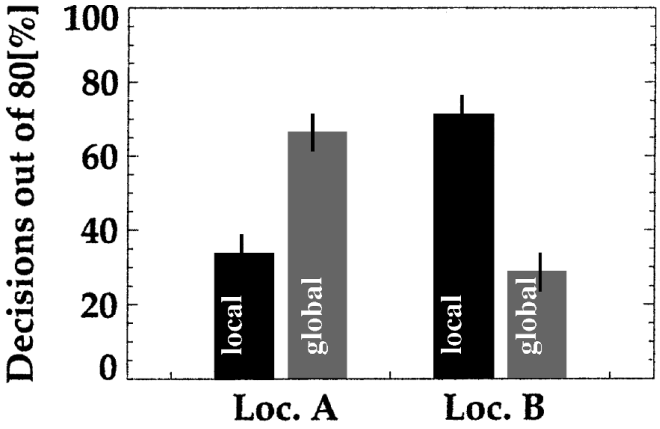


FIGURE 2.8: A graph from Steck et al.'s [60] research showing the type of land mark on which users based their decisions. Location A had clearly visible global landmarks while in Location B global landmarks where sometimes occluded and local landmarks where more distinct than in location A.

Users also tend to favour landmarks that resemble man-made artefacts [69] in natural environments. This supports the findings of the previous study that the distinctiveness of a landmark is an important property. Additionally, it was suggested that artificial landmarks in a VE should all be orientated the same way and marked according to this orientation. This allows the user to extract orientation information as well as location information from a single landmark.

Another study compared a variety of tools for navigation in VEs. These included landmarks, breadcrumb markers (which allow users to drop visual markers along their path) and maps [18]. They associated their techniques with natural human and avian navigational behaviours. They found that the map in particular allowed for straightforward navigation, although this may have been related to the navigation space being a 2D plane. Another tool that let users fly vertically, allowing them to observe

the navigational space from above, also facilitated a simple sequence of actions for accurate navigation. They also found that, while landmarks were effective for distinguishing certain areas, they provided little directional information. When they added a synthetic sun and shadows in the test case with landmarks user performance increased significantly. While the relative effectiveness of different techniques was not evaluated, they did detail the common behaviour of the test subjects for each technique. By observing which techniques lead to simple or complex behaviour some conclusions can be drawn about ease of use.

These results reinforce the idea that a combination of local and global landmarks is useful for navigation [18]. In addition, the ease of use of both map and flying techniques suggest that incorporating either will aid navigation considerably. We suggest that in a 3D environment where 3D movement is possible a 3D map is better suited to navigation. This will allow the user to read their height directly from the map and possibly aid the user in determining the scale of objects.

Although being able to determine your current position in large scale VEs is important, it is also necessary to be able to move around effectively. To aid in the design of user mechanics for movement Bowman et al. have created a list of aims for effective travel techniques [9]. These include: speed, accuracy, spatial awareness, ease of learning, ease of use, information gathering, and presence. We suggest that user comfort should also be taken into account, especially when designing a VR interface for 6-DOF controllers.

In the same paper travel techniques are categorised into two groups: Gaze-directed steering and gesture-directed steering [9]. Gaze-directed steering refers to a control scheme where the user's navigation is directed via a tracked HMD, while in gesture-directed steering the user controls their direction of movement through gestures (with or without props). Their findings suggest that gesture-directed steering is both more comfortable and allows the user to continue observing the VE while moving through it allowing for superior spatial awareness and information gathering. This makes it easier to use, particularly in cases where the user needs to move relative to a landmark that is not directly ahead. Experiments specifically showed that tool-directed navigation was faster than gaze-directed navigation for navigating relative to objects and equal for navigating to an absolute point in space [9].

A modern example of tool-directed navigation is Google Earth VR, which allows users to interact with the popular desktop application via VR. Google engineers went through a number of iterations of navigational controls to optimise usability and user comfort. Instead of using teleportation users can drag a point on the world towards themselves, making the camera fly forward from the user's POV thus preventing the loss of context associated with teleportation [35]. However, this technique introduced an issue where users could accidentally drag themselves through objects especially when moving upwards. This was solved by dynamically scaling the user over time based on the height difference of their current location and their selected destination. The other issue associated with continuous movement in VR is simulator sickness [57]. Google Earth VR attempts to reduce this by introducing three artificial camera effects in their application, namely a high contrast horizon line, a subtle grid displayed to represent the floor of the physical space the user is in and a tunnel vision or vignette effect. These three effects do not move with the camera and provide visual cues to the user to reduce the feeling of moving visually despite being physically still [35].

2.3.2 Interface Interaction

Interface interaction refers to a user modifying the state of the system or mode of interaction [10]. Typical examples include menu interaction and tool selection. These tasks, which are often 2D or 1D in nature, are ill suited to 3D environments [10, 28]. In particular, interacting with 2D surfaces in a VE can lead to a significantly higher error rate and issues of occlusion that need to be considered [28].

It is therefore necessary to consider novel interface elements. For example, a ring menu allows for user navigation with only 2 degrees of freedom, vertical and horizontal [28]. Other alternatives to traditional menu interaction include gesture-based shortcuts [32] or speech input [17, 10, 28]. Furthermore, combining interaction methods can also be effective. For example, gesture-based shortcuts and radial menus can be combined to allow users to utilise muscle memory to select tools from the radial menu using gestures without requiring graphical display [37]. Finally, it is important to keep in mind the physical controls available to the user. For example, the Oculus touch controllers have a number of buttons, triggers, and even a joystick on each controller. The joysticks, in particular, can be used for menu navigation as they have only two degrees of freedom and so are a good match to a ring menu both in terms of degrees of motion and in motion trajectory as demonstarted in figure 2.9.

FIGURE 2.9: A visual example of how the joystick on an Oculus touch controller naturally maps to a ring menu (from the popular game CS:GO).

“Clutching" is another important consideration in interface design. Clutching, involves the user pausing their input to allow themselves to reset to a comfortable position. An example would be when a PC user picks up a mouse and places it in the middle of their mouse-pad. This is required when a user must pause movement tracking so as to reset to a comfortable position [28]. Crucially, this needs to be done without interfering with the current interaction. For example, a reset is required when a user intends to rotate an object further than is comfortable in a single movement. Typically, this can be achieved by providing the user with a button to toggle motion tracking on and off. However, this is annoying for the user and feels unnatural [28]. One solution is to utilise a direct mapping of controls onto virtual objects combined with two-handed-interaction(THI). This can reduce or remove the need for artificial clutching mechanisms [28]. Hinckley et al. argue that utilising THI reduces the need for clutching by tracking gestures of the dominant hand relative to those of the non-dominant hand [30]. They also suggest that an alternative is to use a physical 3D prop, such as those used by Ware et al. for their eye-in-hand metaphor[72].

The depth positioning of interface elements is another challenge unique to VR that simply does not arise in desktop systems [2]. This is because HMD devices use stereoscopy to create depth, while traditional desktop systems simply place interface elements on the 2D screen. Therefore, it is necessary to avoid having interface elements that are too close to the user’s visual field, as these are hard to focus on. Likewise, elements should not be placed beyond comfortable reach. These constraints combine to leave a somewhat restricted volume in which to place interface elements as displayed in figure 2.10. In addition to interface elements, all text that the user needs to read must be placed in this restricted volume. Text must also be rendered larger than it would be in a desktop application as the smaller effective resolution

of current VR devices means small text can be hard to read and trying to focus on this can cause user discomfort [2].

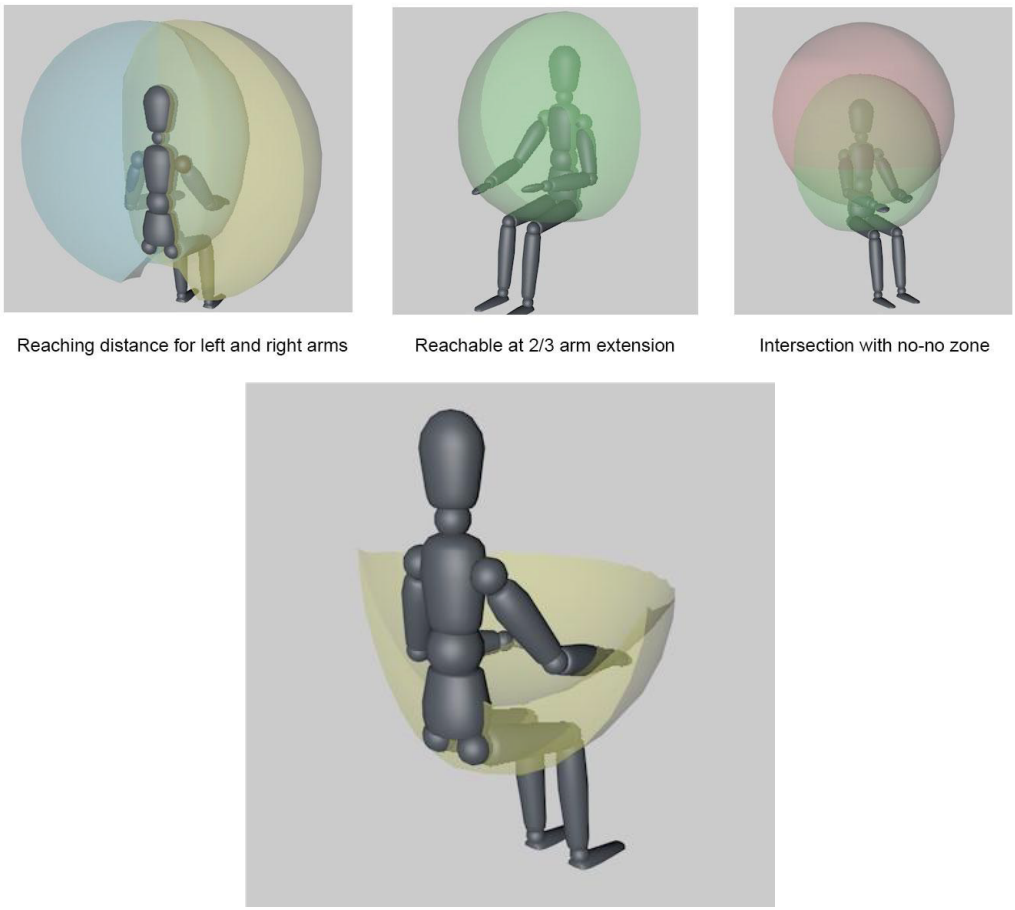


FIGURE 2.10: A 3D diagram showing the various restrictions on where interface elements can be placed and how these combine to create a comfortable working zone [2].

As well as considering where interface elements are placed in the virtual space users should also be provided with an indication of the location of objects in their physical environment. This helps reconcile the differences between the virtual reality a user is immersed in and their surroundings [20]. Reasons to show some of the user's physical environment include: allowing them to more easily interact with physical devices such as keyboards and other objects, preventing inadvertent physical contact that may cause injury or distraction, and being able to interact with other people without exiting VR completely.

2.3.3 Environment Interaction

Interaction with the environment involves the user making changes to the virtual world through selection and manipulation of objects. To do this users should, at a minimum, be able to select, position and rotate objects [10]. Research in this area addresses the wide range of hardware devices created for interacting with VEs, as well as the related software solutions. Common tools include a motion-tracked glove [73], spaceballs [28], and pen or wand controllers [56]. As opposed to WIMP interfaces interaction in VR is often based on direct manipulation, where the user 'grabs' an object and then manipulates it as one would manipulate a malleable physical body.

There have also been investigations of the advantages of THI over other interaction methods. People are better able to judge the relative, rather than absolute, position of their hands [11, 14]. Moreover, users have a preference for THI [14] and perform certain basic manipulation tasks, such as rotation and translation, more efficiently when using two hands [56, 5]. Therefore, interfaces should be designed for bi-manual interaction [30] where possible.

Another avenue of investigation in environment interaction is how to interact with objects far away from the user, commonly referred to as action at a distance (AAAD). This is necessary in our case to allow users to manipulate points on the terrain that are far away from their viewpoint. Techniques to address this include: the go-go hand technique[53], where the hand is tracked and then non-linearly mapped into the VE, and a number of ray-casting techniques [10]. These methods roughly correspond to the function of a cursor in a 2D environment. Hand et al.[28] suggest that although cursors are a necessary metaphor in a 2D environment they do not correspond well with natural gestures for use in VR.

Although there are a number of ways to perform AAAD, users also need depth cues to manipulate or select objects with precision [56]. Some suggested methods for this include representing the user's hand as a transparent outline in the VE [30], drawing a line connecting the user's hands in the VE [56] or creating a plane of action [45] on which the user acts.

To further simplify environment interaction, particularly when working with stereoscopic depth, studies suggest that the degrees of interaction freedom should be restricted as much as possible [10]. For example, when moving an object in 3D space a widget can be provided such that the user grabs one of 3 orthogonal axes. The user can then only move the object along the axis they have selected. This allows for more precise placement as the user does not need to be concerned with unintentionally moving the object along the other axes, i.e. when trying to move the object vertically they will not accidentally move it towards themselves due to the restricted degrees of freedom.

2.4 Summary

From the existing research discussed above we can draw a few key lessons. Terrain synthesis is a wide field of study with many different methods for terrain generation and user interaction. Different generation techniques trade computational resources for realism where the most advanced, realistic techniques are often difficult to implement in real time. Methods for user interaction with the terrains need to consider the level of control needed by users as well as other contextual information such as how the user camera is positioned.

The three important areas to consider when carrying out large scale VE interface design are navigation, interface interaction and environment interaction [10]. Implementing an effective navigational system means that the user is never confused about their position and can easily move about as needed. Interface interaction can be tricky within VR as traditional interface components such as 2D menus are not well suited to a 3D environment. It is important to bear this in mind and design the user interface to work within the constraints of the VR technology a user has access to. Environment interaction needs to allow the user to modify their environment in the most intuitive manner possible. To do this we need to consider possible use cases for the particular system as well as the problems the user might need to overcome such as not being able to reach a certain component of the environment.

Chapter 3

Related Work

3.1 Introduction

In the previous chapter we discussed terrain synthesis and general VR interface design, both of which pertain to the creation of our VR interface. However, it is also important to explore research that is more directly relevant, specifically: creating 3D models in VR and comparing user performance in VR against desktop interfaces. This chapter explores existing work in these particular areas. By analysing previous research in creating virtual objects in VR we can avoid creating ineffective authoring systems. This chapter also covers research that has directly compared user performance across desktop and VR interfaces, which indicates where one system is likely to have an advantage over the other. This helps identify areas where VR systems tend to be weak and, consequently, which interface elements require extra care to implement effectively. Finally, exploring such research allows us to develop an understanding of the area. This will help us proceed in an efficient and sensible manner when developing our own initial system and establishing our experimental methods.

3.2 Designing for VEs Within VR

It is only in recent years that the technology to effectively create content in VR has reached maturity. However, there are already a variety of systems for modelling 3D objects and environments in VR. Early research into the design of VR systems has also revealed some basic principles that can be incorporated into modern systems. By examining solutions in existing systems and incorporating general VR design principles, we can address some of the common problems inherent to working in VR.

3.2.1 Hand Controls

When considering how to build an effective VR content creation system, user interaction is an obvious priority. One important aspect of this, in the case where interaction occurs using tracked hand controllers, is how a user employs their hands to control the system. In particular, natural interaction where users are only minimally aware of the interface should be encouraged [28], but user errors should be prevented. To do this we need to consider a variety of factors, such as how closely interactions mirror everyday tasks, restrictions on the user's interactions, and the cognitive load on the user.

An example of how these factors may affect each other is the choice of two-handed interaction (THI) over single-handed interaction (SHI). It might seem that restricting the user to one hand reduces the cognitive load of any particular task. However, as stated in the previous chapter users have a preference for THI over SHI [14]. This is likely because we naturally use both hands to perform everyday manipulation tasks such as moving objects. As most users have already developed this type of 'muscle memory' for

common interactions [14], an effective control system should not unnecessarily limit them to unnatural interactive techniques.

However, it is important to remember that findings in specific studies may not necessarily generalise to universal rules or guidelines. For example, more recent research by Mine et al. [45] has shown that in certain scenarios users performed better with a specific hand dedicated to a specific task, rather than using both hands for a variety of tasks. This may be because as the number and complexity of tasks increases, reducing the cognitive load required by the user to memorise controls becomes more important than enforcing a natural interaction metaphor. We must therefore carefully consider the context in which user interactions take place when implementing interaction metaphors.

Though using THI may not always be the optimal solution, it is important to not discount its potential benefits. For example, early research in THI found that people have an innate awareness of the relative position of their hands [11, 14]. Some modern systems take advantage of this to solve the issue of 2D menu depth in a scene. Specifically, menus are often attached to the user's Non-Dominant Hand (NDH) either as a physical touch screen [71, 45] or as a virtual representation [34]. Because the menu is always at a constant position relative to the user's NDH, they are able to reliably interact with it using their Dominant Hand (DH). This reduces the need for the user to visually track the depth of 2D surfaces within a 3D virtual environment thus reducing possible errors and frustration [28]

An alternative technique for designing interactive menus is to have a ring menu (as discussed in the previous chapter). This allows the user to either select menu elements by moving in the correct direction [19], which is a 2-DOF action, or by rotating the elements for selection, which is a 1-DOF rotation action [28]. Although using the relative position of hands is an effective measure to reduce the difficulty associated with depth perception, using a ring menu system removes the depth aspect of menu interaction entirely. It also allows for menu interaction using only one hand so that the user can continue interacting with their environment while the menu is open.

3.2.2 Action at a Distance

While it is possible to remove or reduce the impact of stereoscopic depth on interface interaction, it cannot be ignored when considering environmental interaction. While interface elements should always be within comfortable reach of the user [45] this may not be possible in the case of environmental elements. To address this, systems designed for large-scale VEs must allow for some form of Action At A Distance (AAAD).

Systems that involve the creation and editing of smaller objects can rely solely on direct interaction [19, 34]. This allows the user to work directly on an object with high precision provided it is within reach. However, this is not practical when working with objects or environments that are much larger than the user. To resolve this, systems often allow objects to be selected via raycasting [71, 52]. Although this is suitable for object selection and placement, it is less precise than direct manipulation, thus making it unsuitable for object manipulations such as rotation and scaling.. A useful analogy would be trying to perform a precise task using a very long stick. Some systems make the distinction between raycasting activities and direct interaction explicit by having two modes where users can create objects at a direct interaction scale and then switch to raycasting to place and select objects in a larger environment [71, 52, 34].

Although raycasting is the most common approach to AAAD, other techniques, such as the go-go hand exist [53]. This technique is used to give the user the illusion of direct interaction with objects beyond arms length. This works by mapping the virtual hand representation to the user's hand position in a non-linear manner, causing hand movements far away from the body or tracking origin to result in larger changes in the position of the virtual hand as shown in figure 3.1. This gives the user a much

larger area that they can reach into. However, this technique causes a disconnect between the position of the user's virtual hand representation and the position of their actual hand. As stated in the Oculus best practices guidelines [48], any disconnect between the virtual avatar and the user's real body may lead to discomfort.

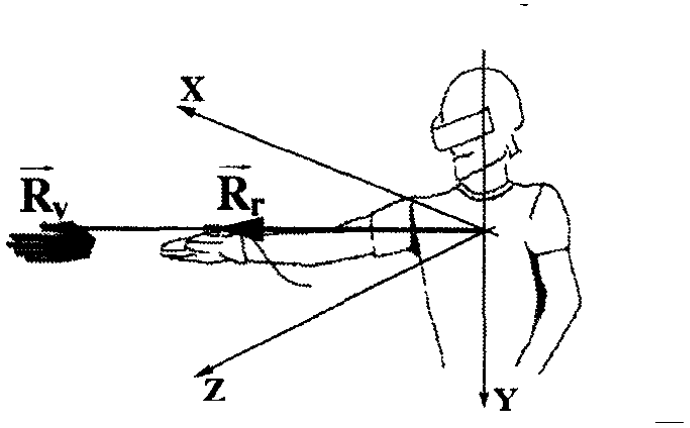


FIGURE 3.1: The go-go hand technique: The position of the user's hand, $\vec{R}_r$ is mapped to $\vec{R}_v$, the position of the virtual hand [53]

An interesting variant of the go-go hand technique, shown in figure 3.2 allows the user to select an action plane within a scene, essentially setting a specific depth for interaction, and then manipulate objects around a point on this plane [45]. This combines the advantage of the go-go hand, where users can interact with objects beyond their natural reach, with a reduction in the DOF of action by locking into a single plane. It also avoids the issue of the users virtual hand not being directly mapped to their real hand. However, the user still needs to manually position the action plane every time they want to change the depth of interaction. This extra action slows down user interaction.

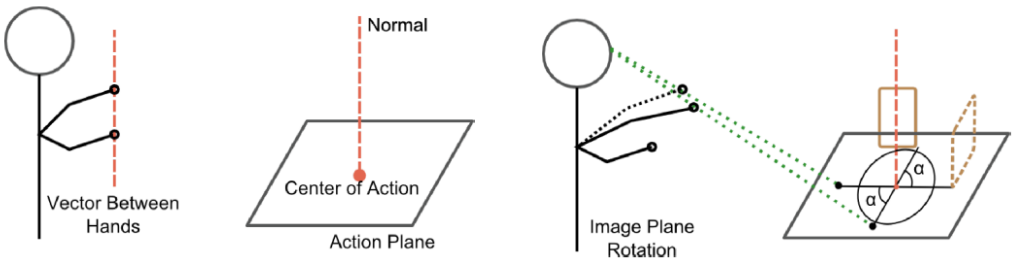


FIGURE 3.2: Positioning an action plane and then using it to effect a rotation [45].

3.2.3 Mapping Virtual Functions onto Physical Controllers

Once a set of controls has been chosen, the challenge of mapping them onto physical devices remains. A variety of such mappings have been explored in previous research with a multitude of physical devices. In some cases the user's hands are tracked directly [19], while in others hand-held devices are used. In the case of direct hand tracking the user is free to make gestures as necessary and these can be mapped to specific actions. However, this requires accurate hand tracking in order to support a reliable user experience. The positional tracking of devices enables high accuracy while presenting a less complex

technical challenge than hand tracking. Using physical devices also means that logical controls can be mapped onto the physical controls of the device, such as buttons, track pads or touch screens.

The input of textual and numerical data is another challenge to consider when mapping physical inputs to virtual functions. To address this, a number of systems use voice input to compensate for the absence of a keyboard [52, 65]. An implementation of this in the *Placeholder* system [38] allows users to create audio notes at certain locations for play-back at any time. An alternative to this, where speech input is not practical, is to reduce or remove any reliance on manually entering such data. For example, sliders can be used to select numerical values without inputting a specific number.

Modern VR systems that incorporate THI for 3D modelling suggest keeping the functions of each controller independent unless otherwise necessary [45]. For example, using the NDH to control movement while the DH is used for interacting with the environment. This means that users do not have to use both controllers for every action. This seems to contradict previous studies that suggest users favour using two hands whenever possible [14, 30]. However, by separating tasks, users are able to simultaneously perform two independent actions. Mine et al. [45] also found that having a small number of buttons dedicated to common tasks is effective. Physical buttons allow the user to find them by sense of touch, which reduces the focus required to memorise controls that would be otherwise be placed on a flat track pad or touch screen [45].

3.2.4 User Comfort

User comfort is essential to the usability of any interface. Because the physical devices and actions available to a VR user are so different from a traditional desktop, particular attention must be paid to the issue of user comfort. Below we discuss techniques designed to reduce user fatigue and discomfort in the arms, neck and body.

Fatigue can be a serious problem when users are required or even encouraged to hold out their arms in an unsupported position for long periods of time. Techniques where the user can bring geometry to a comfortable position instead of reaching out, tend to minimise arm fatigue issues [34]. Alger et al. [2] suggest that users should not regularly have to extend their arms to more than 2/3 of their maximum length. In an experiment using a handlebar metaphor to control the camera, users consistently reported feeling fatigued as early as 20 minutes from start of use [58]. Song et al. [58] also note that when extended-arm interactions require a high level of precision they can be even more fatiguing. This is likely because users naturally tense up while trying to ensure precision [58].

While arm position is probably the most important or at least most easily overlooked aspect of user comfort in VR, both head and body position should be considered. Ergonomics research into the optimal viewing angle suggests that the entire viewing area should be placed between 15° and 50° below eye level [4] as shown in figure 3.3. However, this research focuses on desktop systems and does not take into account the weight of an HMD, which places additional strain on the neck. There seems to be a lack of similar relevant research specifically for HMD devices. This may be because the weight of such devices varies as hardware improvements allow for progressively lighter headsets. In the absence of an alternative, we should endeavour to place the majority of important visual content within the area described above as optimal by Ankrum et al.'s [4] research.

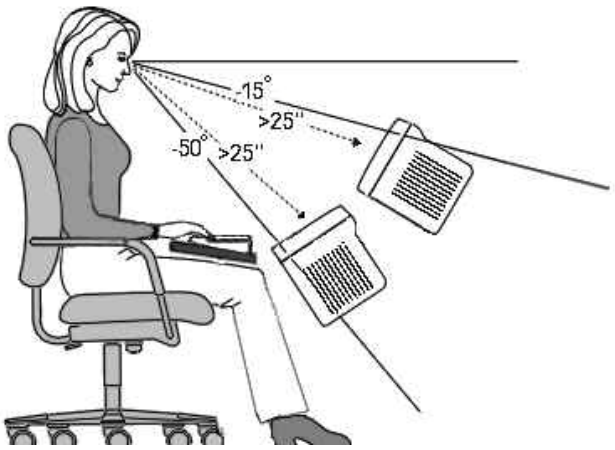


FIGURE 3.3: The optimal viewing volume for desktop systems [53].

With regards to body position, the most obvious consideration is whether the user is sitting or standing. Research by MacEwen et al. shows that standing desks in the workplace can offer physiological and mental benefits, although it is noted that standing only desks can lead to fatigue and discomfort [42]. One benefit of using VR is that the user is no longer linked to a static screen and may therefore sit or stand depending on their preference. However, certain applications and interfaces may specifically require the user to stand or sit.

The risk of simulator sickness, similar in effect to motion sickness, is an important consideration when discussing user comfort in VR. A primary cause of simulator sickness is vection [57]. This occurs when visual movement does not match that detected by the balance organ. About 30% of the population is susceptible to this issue and it should therefore be avoided as much as possible [57]. The easiest way to do this is to avoid the need for showing movement visually. This can be achieved by ensuring the user never needs to move or alternatively by using a teleportation metaphor when movement is necessary.

3.3 Comparison of Desktop and VR Applications

The primary motivation for designers to adopt a VR-based workflow is to achieve increased productivity. It is therefore important to consider areas in which VR systems perform significantly better than corresponding traditional desktop systems. Unfortunately, there have been few studies that undertake such a direct comparison.

Comparisons between Desktop and VR tend to be mainly in the area of entertainment or gaming, where user immersion or flow is the primary measure. Another area where a number of comparative studies have been conducted is in education and training. Here the effectiveness of VR as a training tool is the focus. For the most part, VR tools for content creation or editing have been evaluated in isolation or against other VR/AR variants. This may because it is only recently that suitable tools for practical content creation have become available or because it is difficult to create a system that works well enough on both VR and desktop platforms for meaningful comparison.

3.3.1 Object Manipulation

Object manipulation is a key aspect of content creation. Without this the user has no way to interact with objects in VR. Studies in this area have mostly focused on very clearly-defined manipulation tasks [56, 55].

A study conducted by Scali et al. [55] investigated the effectiveness of using a 6-DOF controller and stereoscopic vision for the rotation and translation of 3D objects in VR. They compared a number of conditions, including a traditional WIMP interface, a stereoscopic setup with a 6-DOF controller shown in figure 3.4, and a monoscopic condition with a 6-DOF controller. They also included conditions that varied the use of haptics and a virtual snapping tool for object placement. A later study by Schultheis et al. [56] performed similar tests, varying the user controls between mouse and keyboard, a single 6-DOF wand controller, and Dual 6-DOF hand-held controllers displayed in figure 3.4. They also tested both stereoscopic and monoscopic conditions for each control device.

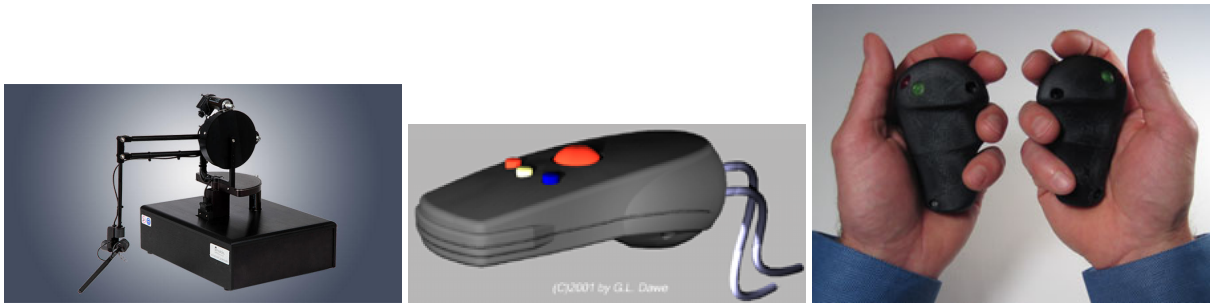


FIGURE 3.4: Some of the input devices in object manipulation experiments. From Left: The 6-DOF controller used by Scali et al. [55] capable of force feedback, the wand device and 6-DOF hand-held controllers used by Schultheis et al. [56]

Scali et al. had users perform a series of tasks that involved translating, rotating and scaling objects to fit onto pre-defined surfaces [55]. They found that the 6-DOF systems outperformed the WIMP interfaces with regards to time for task completion, perceived workload and usability. In addition, they found that, although haptics and stereo vision did not significantly affect task completion times, haptics had a significant effect on perceived workload and usability.

In the later study by Schultheis et al., users had to perform more complex tasks, demonstrated in figure 3.5, such as docking during which users placed shaped objects into matching openings in a cube [56]. Users could manipulate both the shapes and the cube simultaneously when using dual controllers. In a separate task users also had to construct a target pattern using a set of shapes. Again, it was shown that users were much more efficient using the 6-DOF controllers and that dual controllers led to a further gain in user performance [56].

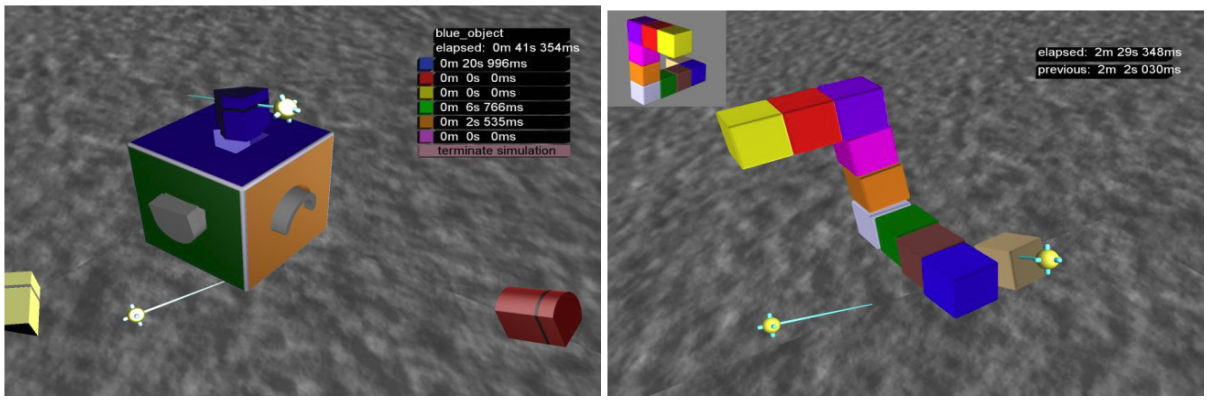


FIGURE 3.5: The tasks assigned to users in Schulteis et al.'s study. On the left users had to slot the coloured shapes into matching holes in the cube. On the right users had to build the image in the top left of the screen using the blocks provided [56].

Both studies noted that it is essential to incorporate user training prior to evaluation, with Schulteis et al. demonstrating that expert users could achieve task completion up to 8 times faster when using dual 6-DOF controllers. However, neither study showed that the stereoscopic condition significantly improved user performance. Schulteis et al. do suggest that having more participants would have allowed them to show significant improvement [56]. In addition, participants in both studies self reported a preference for the stereoscopic condition [56, 55].

Although the above results are promising, they only apply to specific tasks. The performance gains in these areas may be insufficient to offset the potential negatives of using VR for complex 3D modelling applications. For example, neither study addresses the well known difficulties of menu interaction within VR [56, 55, 10, 28]. Furthermore, both studies were performed with hardware that is now outdated. In fact, both experiments used projection-based systems for the stereoscopic condition rather than HMDs. By using modern hardware it is possible to improve upon the results of the above studies.

A more complex study, performed by Toma et al. [65], required users to complete modelling and assembly tasks using Computer Assisted Design (CAD) software. Unfortunately, this study consisted of only 8 participants. Their results, shown in figure 3.6, suggest that while users could perform assembly much faster in their VR condition, an immersive Cave Automatic Virtual Environment (CAVE) with user tracking and data gloves with gesture recognition, than in the traditional desktop condition, they performed similarly for the modelling task. They also noted that the VR system required much greater movement from users, which could lead to fatigue during extended use. Despite this, users reported that the VR interfaces were more intuitive and easier to use than the desktop systems, despite there being no significant difference in overall task performance.

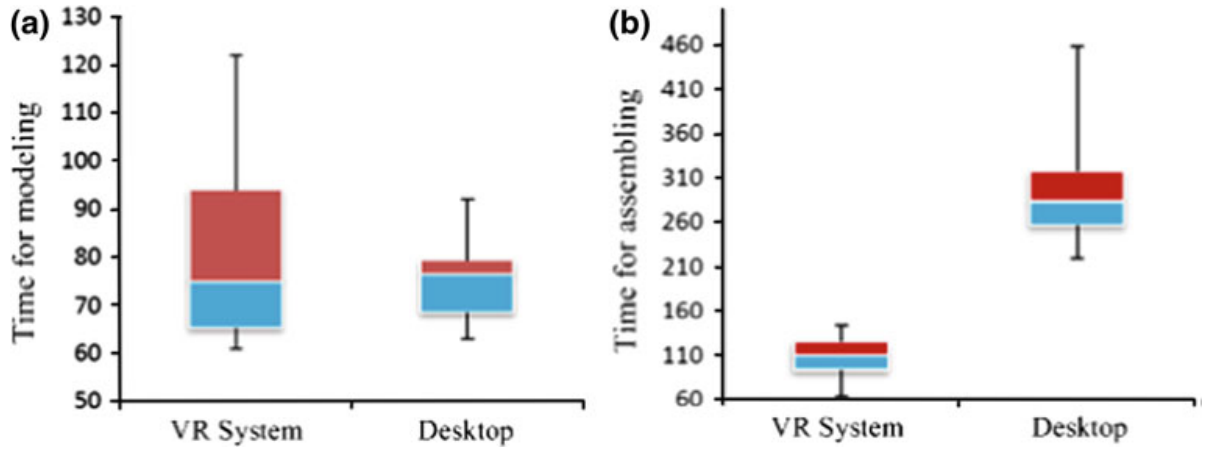


FIGURE 3.6: Toma et al.'s results with regards to task completion time [65]. Users completed two different tasks: In the modelling task they created a simple 3D model from scratch, during the assembly task they moved existing models to specific positions and orientations relative to each other

A larger study, also based on a CAVE VR condition, integrated both navigation and object manipulation [27]. Users were required to map a safe path for oil-well drilling through a mature oilfield, with existing oil wells to be avoided. This represented a complex 3D environment shown in figure 3.7. Because of the size of the oilfield, users needed to navigate around the terrain before creating a new oil well path by positioning and rotating a virtual object such that it reached a target location and did not intersect any existing oil wells. The study showed that participants were both faster and more accurate in the VR condition than in the desktop condition. However, it should be noted that both tasks were relatively simple, with navigation being unconstrained and the task itself constituting the placement of a single object. Another study showed that in complex environments users tend to navigate better in desktop conditions than in VR [59]. Specifically, it was shown that when navigating through an indoor maze environment in order to pick up as many objects as possible in a limited time users performed better in a desktop environment by maintaining a higher average speed and thus moving further.

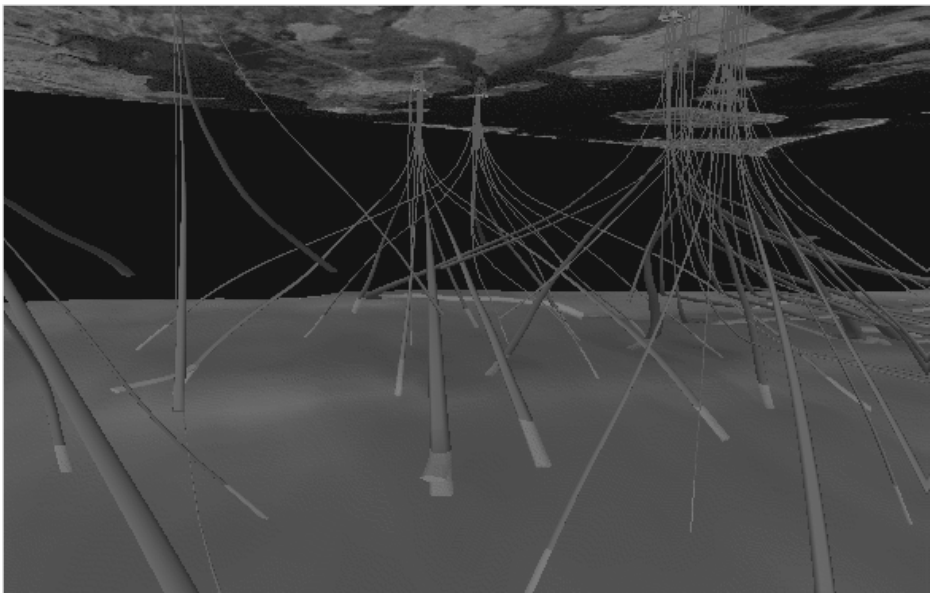


FIGURE 3.7: A 3D representation of a mature oilfield [27].

3.4 Summary

In this chapter we reviewed existing work in the areas of creating 3D models in VR and comparing user performance when using desktop interfaces interfaces. The importance of choosing effective hand control systems was explored. This, together with choosing the correct manner to map virtual functions, such as AAAD, onto the physical controllers, constitutes the core element of creating a useful VR interface. We also looked at user comfort as this is an essential part of any human-computer interface.

Research that directly compared VR and desktop interfaces was also discussed. It was noted that much of the existing research focuses on highly specific user interactions which may not necessarily apply to everyday use cases. Overall the research did show that VR interfaces were more effective for these actions although, as the interactions became more complex, these advantages became smaller.

Through review of this related work we are able to determine an initial plan for building our initial VR interface in an effective manner. We can also make some educated predictions on how the performance of our VR interface will compare to the desktop one. Finally, we can use the lessons learned in previous experiments to create a methodology for our own tests.

Chapter 4

Interface Design

The design of the interface was perhaps the most challenging part of this research. It is essential that the VR application be of a similar quality to the existing desktop one. Specifically, no component of one interface should be superior to its equivalent in the other in terms of design. For example, widget manipulation must be present in both interfaces and should be implemented such that it maximises user efficiency in each implementation. Additionally, it is important to balance using best design principles for the VR interface against maintaining the core components of the original system. All these considerations are important in ensuring that the comparison between VR and Desktop implementations is fair and does not favour a particular interface simply because the basic design is better.

This chapter explains how we started from the existing desktop interface and then used user centred design (UCD) together with existing VR design principles to create and improve on an initial design concept. The chapter concludes with a detailed presentation of the resulting final interface.

4.1 Desktop Interface

With the above in mind, we first consider the existing desktop interface created in native C++. This interface was designed during previous research into controllable terrain synthesis by Gain et al. [24]. The interface includes painting, sketching and 3D widgets, which allow users to shape and interact with the terrain (see section 2.2.2 for a more detailed explanation). While all three of these basic interaction types are included in our final interfaces some of the specific functionality (eg., copy-paste) was removed due to scope concerns. This interface was initially designed for Gain et al.'s research but was further developed for a pre-commercialisation project. Therefore, it was felt that the quality of this interface was sufficient for our desktop implementation and, other than some feature removals as mentioned above, no changes were made.

However, as existing research has made clear, simply adding VR interactions to a desktop interface is far from optimal [28]. This meant it was necessary to create a high quality VR specific interface for our implementation according to existing best practices, while supporting the functionality of the original application. Moreover, since the interface should be designed around the users and the tasks they need to perform a consultative approach is advisable. Thus, we also chose to follow a UCD process.

4.2 Interface Design Process

The technical details of this UCD process are summarised below and visually represented in figure 4.1. In brief overview, three design iterations took place before the final interface was created. The initial

design was based on the existing desktop interface and design principles drawn from similar VR applications. This design was implemented as a paper prototype and the subsequent design iteration was based on resulting user feedback. After the second design iteration was complete it was implemented as a high fidelity prototype using the Unity game engine. Another round of user feedback was gathered which allowed a final iteration of design leading to our final interface implementation.

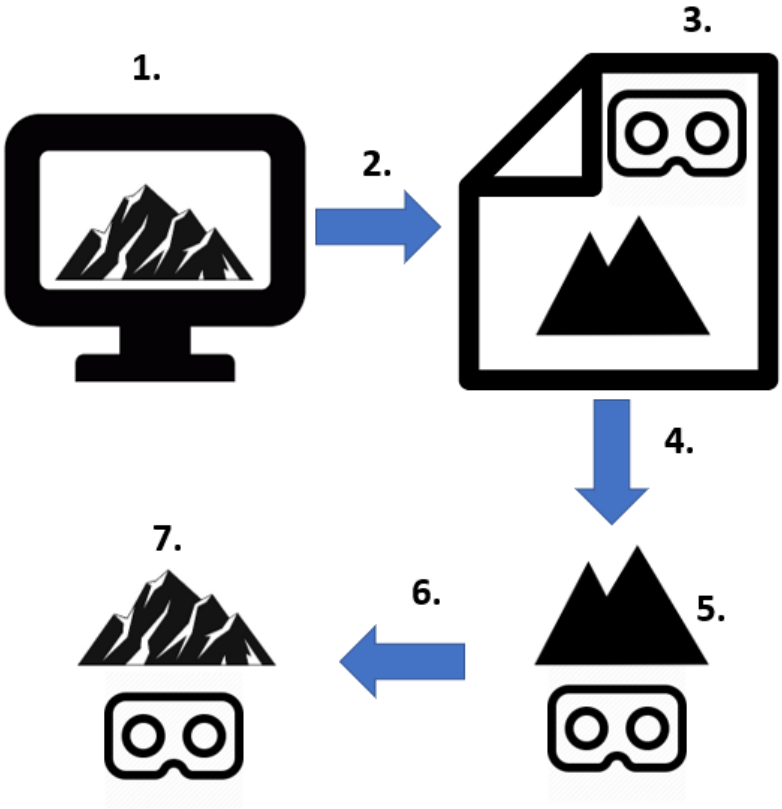


FIGURE 4.1: 1. The initial desktop program, 2. Design based on existing research, 3. Paper prototype of initial design concept, 4. User testing of paper prototype and development of high fidelity prototype, 5. High fidelity prototype created in Unity, 6. User testing of high fidelity prototype and development of final application, 7. Final VR terrain editing application.

In this section we cover each iteration of our design in detail as well as discussing user feedback and how this influenced changes to our original design.

4.2.1 Overview of UCD

User centred design (UCD) is a way of involving users in the design of artefacts [1]. In our case, the design of a user interface for the creation of terrains in VR.

UCD includes a number of techniques that can be implemented at various stages of the design [1] as shown in figure 4.2. In the early stages these include: Interviews and questionnaires to determine the exact purpose of artefact, focus groups to establish requirements for the artefact amongst all stakeholders, and on-site observation to establish the usage environment. Including users from as early on as

possible reduces the need for additional work in the later stages. Incorporating all necessary features from the beginning is more time- and cost-effective than trying to work them into an existing design later.

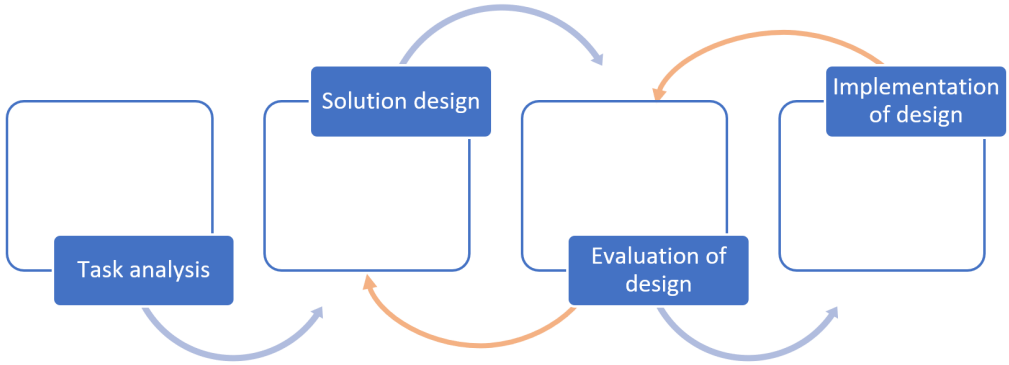


FIGURE 4.2: A diagram showing the basic UCD design cycle

Later stages form a cycle of design and feedback where concepts are introduced, tested and then modified based on feedback. The techniques used in these stages include: role playing, walkthroughs and simulations to explore alternative designs in a prototype format [1]. Generally the quality of the prototypes being tested increases over a number of cycles. This means that cheap, flexible prototypes can be used when many parts of the design are uncertain and many changes are still likely to be made. Higher quality prototypes that are more time consuming to develop are used once the design is more mature to isolate the more subtle usability issues that would not be apparent in very low fidelity prototypes [43]. Lastly, techniques such as usability testing and interviews can be used to collect quantitative and qualitative data regarding the final artefact [1].

Since we are working from an existing application and the basic functionality of the VR interface is therefore already established, we did not employ the UCD techniques typically used in the earlier stages [1]. However, we did apply simulation and usability testing techniques in the prototype design stages. We also used a usability questionnaire for our final evaluation of the interface design.

By testing in the prototyping design stage it is possible to address most of the usability issues before the final interface evaluation [47]. This ensures that the VR interface being compared to the existing desktop interface is robust and usable. This is necessary to ensure that a fair comparison is made between the VR and desktop applications and prevents a flaw in interface design affecting our quantitative experiments.

4.2.2 Initial Design Concepts

Since our interface is based on an existing desktop application there is already a basis for the initial design concept. However, the interface medium is different: a HMD and two 6-DOF controllers, as opposed to a desktop with mouse and keyboard. This requires that certain interface elements be modified and new interaction metaphors adopted.

Many of our initial changes to design are based on findings in the literature regarding the effective design of VR interfaces. Examples of this include menu design and performing actions at a distance (see sections 2.3 and 3.2 for detailed discussion of VR interface designs). Using these guidelines we are able to formulate a complete interface concept, which can be broken down into the categories of interface interaction, navigation interaction and environment interaction [10].

Interface Interaction

One of the main interface interaction tasks a user needs to perform is tool selection, which implicitly specifies the mode of environment interaction. We chose to enable users to perform this using a single wedge menu similar to that used by Mine et al. [45], which could be toggled on and off with a single dedicated button. Users are then able to select an item from the menu by moving the joystick on one of the Oculus controllers in the corresponding direction. This provides a 2-DOF solution for tool selection as opposed to the 3-DOF of a floating virtual menu with buttons [28]. In addition, since all menu elements are consistently positioned users are able to learn the positions of specific tools over time. This allows experienced users to summon the menu and select a new tool without looking away from their current task.

We also considered having the menu rotate when the user moved the joystick left or right, which would be a 1-DOF task as only movements in the horizontal plane are required as opposed to selecting a direction from the horizontal and vertical planes combined. In this scenario the tool at the top of the menu is the one selected. However, we chose to implement the prior design as we were concerned that requiring multiple rotations per tool selection might lead to user fatigue.

We chose to implement type selection, which lets the user select a terrain type such as wetland or canyon to paint onto the terrain, in the same manner. Users are able to toggle between the main tool menu and the type selection menu with a dedicated button on the Oculus controllers. We felt that having the same menu design throughout the interface would help maintain consistency and reduce cognitive load on users, in accordance with basic usability principles [46].

Navigation

The main difficulty with navigation lies in allowing the user to move around in a first person viewing (FPV) mode. In particular, this requires switching between the standard God View (GV), which allows the user to look down onto the terrain from a large distance above, and FPV. See figure 4.4 (bottom) for an example of God View. This is complicated by the fact that it is not a symmetrical action. When shifting from FPV to GV the user's view point simply needs to be moved backwards sufficiently far. However, when shifting into FPV the user must first choose the location to which their viewpoint is moved. Fortunately, selecting points on the terrain is part of the core functionality of environment interaction. Therefore, we can use a selection tool in GV to specify the position to which the user will teleport on switching to FPV. After a position had been selected a user can then switch between view types using a single button. Th switch between view types was implemented as a teleport, the view would fade to black and then come back in the new position, rather than a zoom because of concerns for simulator sickness. A zoom mechanic, would create the illusion of rapid movement which, without the associated physical indicators of movement, would lead to cognitive dissonance and potentially simulator sickness and disorientation [6]. Teleportation avoids this as no visible movement occurs.

Within FPV we chose a walking metaphor for user travel. This choice allows designers to explore the terrain in the same way that an end-user will experience it in a virtual game. Additionally, since it has less freedom of rotation than a flying metaphor (rotation is limited to the horizontal plane while walking) it is less likely to cause significant vection which would lead to simulator sickness [57]. It also only requires the use of a single joystick for movement, as opposed to a more complex targeting and action selection teleportation travel metaphor.

In GV we implemented a scene-in-hand metaphor for viewpoint manipulation [72]. This allows the user to position the terrain freely by grabbing it and moving it relative to the their viewpoint. They can then place the terrain in a comfortable position for interaction, which reduces potential fatigue [58]. In

addition, the viewpoint also moves to match the user's head movements as tracked by the Oculus Rift headset.

Lastly, it was also necessary to ensure that users would not become disorientated or lost while in FPV. To prevent this, we incorporated a tool for use in GV allowing users to place landmarks at chosen locations [18]. These landmarks were designed to stand out from the terrain itself [69] and provide both positional and directional information [18] while also being visible from any point on the terrain while in FPV due to their height.

Environment Interaction

In terms of environment interaction, a specific set of user actions needs to be supported. These actions include: **selection**, either of a point on the terrain or an existing node; **drawing**, either of a single node or a continuous curve; **painting** and **placement of landmarks** [24].

For selection, two of the approaches mentioned in sections 2.3.3 and 3.2.2 for action at a distance are suitable. Either the user could select points using a virtual laser pointer through ray-casting [10] or reach out to distant points using the go-go hand technique [53]. We choose to implement the ray casting technique as it is less likely than the go-go hand technique to cause user fatigue, since it does not require the user to extend their arms. Specifically, the user can point to a node with a controller and press the index trigger to make a selection. In addition, prior experience with VR applications indicates that it should be sufficiently accurate for our intended use.

However, once a node is selected via ray-casting it may be necessary for the user to make fine adjustments to the variables associated with the node. In such cases, ray-casting at a distance is not suitable, as small hand movements can lead to large adjustments, depending on the distance and angle at which the user is working. We therefore decided to incorporate direct manipulation [15]. A selected node is mapped directly to the position and orientation of the user's hand. The user can then move their hand in order to adjust the node height, slope angle, rotation angle, and area of influence. This is possible given that the controller is tracked with 6-DOF.

For drawing we also use the ray-casting technique. While in drawing mode the user can either point to a location and tap the index trigger to create a node or hold the trigger while moving their hand to sketch a curve.

Painting is implemented in a similar manner. While in paint mode the user can paint an area around where they are pointing by pressing and holding the index trigger. The size of the area to be painted is shown as a preview at all times while in paint mode and can be adjusted by moving the controller joystick up or down.

Finally, the placement of landmarks is also carried out through ray-casting. The user can simply select a point on the terrain and press the index trigger to place a landmark while in landmark mode.

All actions are mapped to the same trigger event, specifically, the index trigger on the user's dominant hand (DH) controller. The effect of the trigger press is determined by the mode the user has selected via the menu as discussed in section 4.2.2. This is motivated by having a consistent control scheme across all interactions to reduce the cognitive load on users [46].

4.2.3 Paper Prototype

Paper prototyping involves creating a prototype using cheap tools such as paper and other common objects [70]. This allows the prototype to be cheaply and quickly created. Since a human controls most interactive elements it also allows greater flexibility in adjusting certain elements of the prototype. This

means that the prototype iteration is cheap and fast [70]. However, since the interaction is faked by human intervention, paper prototypes may miss some usability issues [70]. Additionally, it is important that users are able to imagine the prototype being mapped into the intended medium for them to provide useful feedback. For example, we had to have users imagine that a 2D piece of paper represented their view in a VR headset.

Our paper prototype was the first iteration of the interface design implementation. The aim was to create an interactive interface that corresponded to the initial design concept. It was constructed using a number of parts to represent various interface elements. The displayed terrain itself was represented by a flat sheet of paper on a table surface. This allowed UI elements to be placed onto the terrain surface, thereby approximating the functionality of the VR interface. Users were asked to keep in mind that the terrain image is simply representative of the VR display. A number of UI elements are represented by small cardboard tokens, which can be placed on the terrain or “view area". Two wedge menus, displayed in figure 4.3, are included: the main menu, with which users can select tools, and a terrain type menu, which users can use to select which terrain type is active in painting mode.

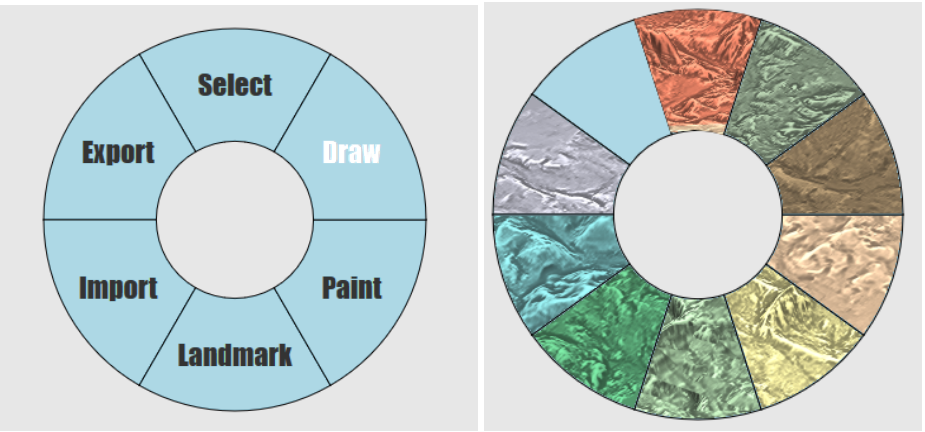


FIGURE 4.3: The wedge menus used in the paper prototype: the main menu on the left and the terrain menu on the right

Users do not wear an HMD while interacting with the prototype but are given the 6-DOF hand-held devices intended for the final interface. This allows the prototype to represent the final interactive experience with greater fidelity than normally present in a paper prototype. The hand-held devices are not tracked using software, so users need to state out loud what actions they are performing with the devices. An experimenter is then able to modify the interface representation according to the intended action. For example: The user points to a certain position on the terrain and reports pressing the button assigned to creating a constraint. The experimenter then places the representation of a constraint at that point on the terrain.

Although a paper prototype is quick and cheap to create, the fidelity of the prototype is limited. Specifically, even though the interface elements reflect user input it is not feasible to actually update the terrain model. Thus, users are required to imagine how the terrain changes based on verbal feedback.

Evaluation

The paper prototype was evaluated using a cognitive walk-through approach [8]. This involves the user stepping through common tasks and then evaluating how well the application supports each step. This is particularly suited to evaluating the usability of a system for first time users [8].

This approach is intended especially to help understand the usability of a system for first-time or infrequent users, that is, for users in an exploratory learning mode.

Users were asked to perform a number of basic actions with the help of textual instructions available in appendix A. This represents the typical first-time user experience with the interface. Users verbally reported on problems and insights while performing these actions. Each experiment was video recorded for later review.

We used this method of evaluation as it does not require our participants to be representative of the ultimate end users [8]. This made it easier to recruit participants. It also requires less time and expertise from participants than a heuristic evaluation. It was decided that evaluating the paper prototype quickly and minimising turnaround time was most appropriate given its low fidelity. This allowed us to efficiently ascertain the most significant usability issues.

In total 5 participants evaluated the paper prototype. While they gave a range of feedback, there were a number of common issues. These included: users wanting to be able to select menu items with the DH trigger rather than needing to toggle the menu off, the instruction set was insufficiently rail-roaded (users often fell out of the tutorial by failing to take a specific action) and not having a preview of the controller layout visible to verify the location of buttons/triggers referred to in the instructions. Additionally, all users felt that having selected nodes directly mapped to user hand movement was confusing and did not find the controls for adjusting node variables intuitive. Another common issue was that users would need to switch modes more often than expected, resulting in more interface interaction than anticipated. Users found this slow and annoying. They also wanted to have a persistent visual indicator of the current mode. A full record of user feedback can be found in appendix B

Changes to the Design

Based on the feedback from the paper prototype evaluation, a number of design changes were incorporated. These were primarily related to the user instructions, widget interaction and menu design. Other changes were also made but these were less dramatic.

The user instructions were streamlined to prevent users falling out of the tutorial halfway through. In addition, the order in which instructions are presented was modified to ensure that users would never be unable to follow a new instruction due to a lack of prior information. The clarity of some instructions was also improved to prevent possible user confusion.

The way in which users interacted with the widget controlling the point constraints was clearly counter-intuitive and many users had difficulty with this. We therefore completely redesigned the widget implementation. The original concept preserved the design of three independent axes that could be manipulated. However, this does not translate well into a 3D environment where there are six degrees of freedom. We therefore changed the widget to a single control for height and a second control that simultaneously defines the area of the constraint and the slope angle as labelled in figure 4.4. The height control is restricted to movement along the y-axis but the area/angle control can freely be manipulated in 3D space. This simplified the control scheme and took advantage of the 6-DOF interaction devices.

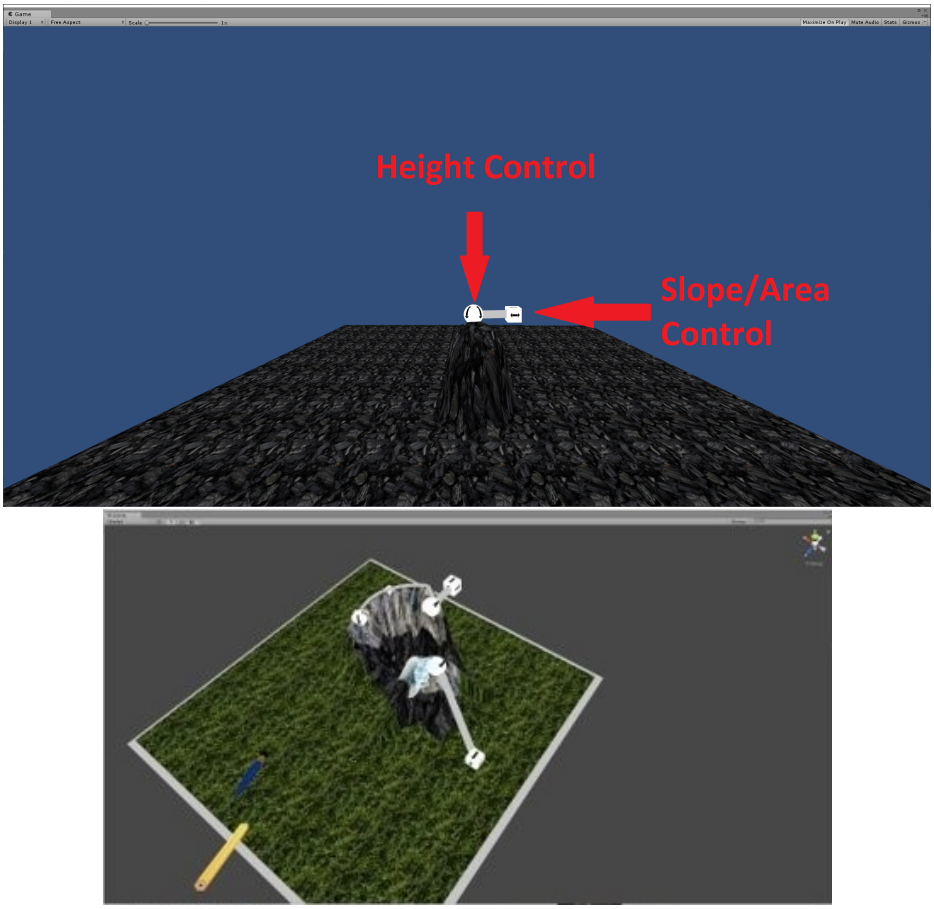


FIGURE 4.4: Some examples of the widgets used in the high fidelity prototype

Another usability issue arose with regard to the menu design. Specifically, users needed to use menus too often with most steps of each task requiring menu interaction. To solve this problem we replaced a large part of the main menu functionality with 3D props. This meant that users could simply pick up the desired tool from the environment rather than being forced to enter and exit a menu to select an action mode. This also removed the issue of users becoming confused as to what mode they were in as the 3D prop in hand indicated this. The menu selection was also changed such that the user could rotate it and the top item would be selected. This change was included because in pilot testing for the high-fidelity prototype it was found that moving the joystick in the direction of the object to be selected was not sufficiently precise and resulted in too many user errors. Finally, as requested in the paper prototype evaluation, users were able to confirm a menu selection via trigger press, which would also dismiss the menu.

4.2.4 High Fidelity Prototype

This prototype was created using Unity and is based on the paper prototype, together with modifications resulting from the paper prototype evaluation. It allows real-time terrain interaction and supports the use of an Oculus Rift HMD together with two 6-DOF Oculus touch controllers. It closely mimics the expected functionality of the final interface. However, the visual quality and realism of the terrain is reduced.

The prototype was created using Unity as it provided better support for the Oculus Rift than the native C++ the desktop application was created with. This allowed the prototype to be developed over a shorter period. Integrating the HMD interface into the existing desktop application within the desired time period was deemed unlikely due to a lack of support and the researcher's lack of experience. As delays were encountered in other parts of the project we decided that the faster and more reliable route should be taken, even if this meant a loss of visual fidelity. Also, when working in Unity it is much easier to rapidly explore different options through pilot testing. This was the case with both the wedge menu design change (mentioned in the previous section) as well as the control scheme of the 3-D widget for node editing, which went through several iterations before being fully realised. The flexibility we had in the design of the high fidelity prototype allowed us to quickly develop the most promising version of our interface for evaluation while avoiding the complications that may have arisen when making numerous changes to the existing C++ application.

Apart from the design changes the largest difference between this prototype and the paper prototype is the real-time terrain interaction. This means the user receives instant feedback from their actions without going through an experimenter to explain what is happening. This was expected to greatly increase the speed at which a user is able to understand effective methods of interaction and reduce user frustration.

Evaluation

The High fidelity prototype was evaluated through a heuristic evaluation approach [8]. We chose the expert evaluator method with a set of heuristics designed for VR [61]. This is a cost effective method of usability evaluation [47]. Although any number of evaluators can be involved, three to five is the recommended range [47].

The method of evaluation differs slightly from the process for expert evaluation recommended by Nielsen et al. [47, 61]. Specifically, an initial technology audit step is introduced, which allows the evaluator to establish what the VR technology is capable of as a baseline. This allows the identification of usability issues that are an inherent to the VR equipment. For example: there is no haptic feedback so users will not be able to feel objects in the VE. The technology audit aims to identify issues in four principle areas: user presence, haptic feedback, interactive techniques and realistic graphics. User presence, as defined by Sutcliffe et al., refers to how the user is represented within the VE [61]. This will be our working definition from now on although others do exist. Haptic feedback refers to the touch feedback provided by the VR technology, interactive techniques cover how the user interacts with objects in VR, and realistic graphics addresses the realism of the graphics in the VE.

Prior to evaluation, the procedure is explained to participants. This includes describing the technology audit, going over the heuristics and the way in which they are applied and how to rank the severity of issues. Users also complete the standard Oculus introduction to the touch controllers. This is the introductory experience any first time user goes through when setting up the Oculus Touch controllers. It is assumed as the bare baseline of proficiency when using them. Table 4.1 explains the heuristics included and was used to familiarise users with the heuristics prior to the application evaluation [61].

This method and set of heuristics was designed by Sutcliffe et al. specifically for the evaluation of VR applications [61]. They are based on the Nielsen's usability evaluation heuristics [47] but have been modified to better suit VR applications. They were evaluated through a number of case studies which showed them to produce meaningful results in an efficient manner [61].

Guideline	Explanation
Natural engagement	Interaction should approach the user's expectation of interaction in the real world as far as possible. Ideally, the user should be unaware that the reality is virtual. Interpreting this heuristic will depend on the naturalness requirement and the user's sense of presence and engagement.
Compatibility with the user's task and domain	The VE and behaviour of objects should correspond as closely as possible to the user's expectation of real world objects; their behaviour, and affordances for task action.
Natural expression of action	The representation of the self/presence in the VE should allow the user to act and explore in a natural manner and not restrict normal physical actions. This design quality may be limited by the available devices. If haptic feedback is absent, natural expression inevitably suffers.
Close coordination of action and representation	The representation of the self/ presence and behaviour manifest in the VE should be faithful to the user's actions. Response time between user movement and update of the VE display should be less than 200 ms to avoid motion sickness problems.
Realistic feedback	The effect of the user's actions on virtual world objects should be immediately visible and conform to the laws of physics and the user's perceptual expectations.
Faithful viewpoints	The visual representation of the virtual world should map to the user's normal perception, and the viewpoint change by head movement should be rendered without delay.
Navigation and orientation support	The users should always be able to find where they are in the VE and return to known, preset positions. Unnatural actions such as fly-through surfaces may help but these have to be judged in a trade-off with naturalness (see heuristics 1 and 2).
Clear entry and exit points	The means of entering and exiting from a virtual world should be clearly communicated.
Consistent departures	When design compromises are used they should be consistent and clearly marked, e.g. cross-modal substitution and power actions for navigation.
Support for learning	Active objects should be cued and if necessary explain themselves to promote learning of VEs.
Clear turn-taking	Where system initiative is used it should be clearly signalled and conventions established for turn-taking.
Sense of presence	The user's perception of engagement and being in a 'real' world should be as natural as possible.

TABLE 4.1: Table listing heuristics for user evaluation of the high fidelity prototype [61]

After the technology audit is performed the user then familiarises themselves with the application, before carrying out a number of typical user tasks. During these tasks the user notes any issues encountered. Since users are wearing an HMD during this part of the experiment they are asked to verbally describe the issues. These descriptions are recorded by an experimenter.

After the tasks are completed the user leaves VR. They then associate each issue with a heuristic. Where an issue is related to more than one heuristic the most applicable heuristic is chosen with a note made relating to the other applicable heuristics [61]. Finally, the severity of the issues discovered is ranked by heuristic. This ranking reflects the number and seriousness of issues assigned to a particular heuristic. The severity scale, as described by Sutcliffe et al. [61]. and as presented to participants, is shown in table 4.2 .

Severity	**Description**
Severe.	The problem encountered would make it impossible to complete the task successfully.
Annoying.	The problem would disrupt the user's task but most users would learn how to overcome the error given an explanation, and some might find a work-around with time.
Distracting.	The problem would disrupt the user's tasks but most users would discover the fix relatively quickly given a hint.
Inconvenient.	The problem could disrupt the user's task but most users would discover the fix unaided.

TABLE 4.2: Table explaining the severity scale used by participants [61]

The same 5 users that evaluated the paper prototype also evaluated the high fidelity prototype. They provided a variety of feedback on usability issues with this version of the interface. Many of these issues were directly related to the visual fidelity of the prototype. Also raised were various aspects of how the terrain representation behaved in unexpected ways during interaction. Many of these issues will be solved in the final interface by virtue of integrating the VR interface with the existing terrain synthesis application. In the Changes to Design section below we will present a complete list of usability problems captured during the evaluation as well as the steps taken to solve them in implementing the final interface. The individual evaluation details can be found in appendix D

Changes to Design

The table below lists all usability problems captured during the evaluation of the high fidelity prototype. The problems are grouped into classes of design features [61], allowing an identification of those areas requiring the most work.

Category	**Problem**	**Solution**
Graphics	Instruction text is fuzzy	Use instruction images with text included, to avoid relying on text rendering
	Textures (Terrain type) do not scale correctly	Use the terrain textures as implemented in the existing desktop application

Category	Problem	Solution
	Texture rendering and terrain update delays	Use the terrain textures as implemented in the existing desktop application
Presence	Zooming causes some unrealistic effects	Instead of zooming the camera forward simply adjust the offset of the view cameras to create the same effect
	Instructions in the sky in FPV	FPV not implemented in final interface
	Simulator sickness from rotating in FPV	FPV not implemented in final interface
Presence	Could see user avatar hands in the sky while in FPV	FPV not implemented in final interface
Interaction	Could not grab terrain at all points	Adjust the grabbable area of terrain based on the height map
	No Undo	Use the undo functionality from the desktop application
	Point constraints not centred on hill-tops	This is also the case in the desktop application, however, higher graphical fidelity and cleaner widget representations may reduce this issue
	No way to measure scale	Use contour functionality from desktop application
	No way to move constraints once created	Desktop application provides this functionality and integrating with it will solve this issue
	No haptic feedback when interacting with objects	Did not solve
	Ridge lines stuck to non-terrain objects	Artefact of naïve Unity implementation, integrating with desktop application will fix this.
	Accidentally grabbing the wrong objects	Separate the grab controls for the terrain/tools from the grab tools for control nodes. The most common issue is grabbing a node instead of the terrain or vice a versa. The change solves this
	Cannot rotate the terrain using both hands	This would require being able to grab the terrain with both hands, which can have other negative effects.
	Dropping the tools accidentally when finished actively using them (let go of both triggers instead of just one)	Did not solve (too rare)

Category	Problem	Solution
Interaction	Accidental creation of line constraint instead of point constraint	Spend time balancing the minimum distance for a line constraint (based on terrain scale)
	Cannot change paintbrush shape	Not possible in desktop application either. Not solved
	No minimum scale for terrain	Set minimum and maximum scales
	Instructions skip too easily	Had user manually choose to go to next instruction instead of skipping automatically
Environmental features	Instructions clip on terrain	Attach instructions to a static plane.
	Terrain does not scale in a 1-to-1 manner	Adjust scaling to achieve this
	Textures painted on terrain do not match the paintbrush tool preview	Unexpected issue, solved when integrating with desktop application
	No indication of what terrain is traversable in FPV	FPV not implemented in final interface
	Ridges do not scale correctly	Clumsy implementation issue solved by integrating with desktop application
Controls	Menu controls not consistent (selection)	Make all menus consistent
	Cannot toggle constraint visibility	Integrate transparency features from desktop application
	Hard to remember all controls	Instructions must be accessible at all times and include controller diagrams
	Need to scale /move terrain before relevant instruction appear	Change instruction order
	Text instructions too close to user	Solve by attaching instructions to a static plane. (allows user to move closer/further)
	Circle menu sometimes does not rotate and was not animated	Unexpected bug, will avoid in final interface implementation
	Colour Menu on opposite hand to related button	Because of limited buttons this will be retained

Category	Problem	Solution
	Accidental painting of terrain when trying to select menu option	Disable terrain interaction while any menu is open
	Blue menu persists after option selected	Dismiss menu when option selected
	Non-intuitive to be selecting a grey coloured menu option	Use different colour scheme in selection menu
Controls	Using the pencil to TP is unintuitive	FPV not implemented in final interface

TABLE 4.3: Table detailing usability issues and solutions from high fidelity prototype evaluation

4.3 Final Interface Design

4.3.1 Interface Interaction

The menu design from the paper prototype, specifically a wedge menu, was retained. However, based on feedback, a rotational menu was chosen over a directional menu. Thus, instead of moving the joystick towards an item to select it, the user rotates the menu as a whole by moving the joystick left or right. The item at the top of the menu becomes the selected item. Additionally, the final interface only includes one wedge menu, shown in figure 4.5, without a main menu with a vertical layout and buttons selected by ray-casting, which was used in the high-fidelity prototype.

The mode selection menu was made obsolete by the addition of 3D virtual objects that the user could grab and the removal of the FPV mode. The 3D objects consist of a pen, which users grasp to activate the creation of point or line constraints, and a paintbrush to add type constraints.

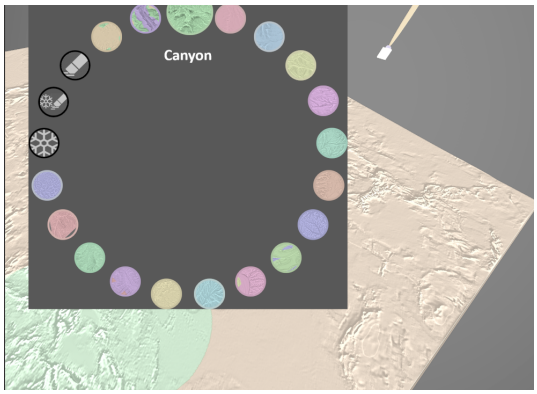


FIGURE 4.5: The terrain type selection menu used in the VR interface

Finally, the instruction set available to users went through a number of iterations before being finalised as a static plane with a number of displayable pages that a user can scroll through with dedicated buttons. This system is also used to display further instructions or views of an example terrain for the speed and accuracy tasks (explained in the next chapter). In addition, controller diagrams have been added to the instructions as appropriate to increase clarity. An example of these instructions is displayed in figure 4.6.

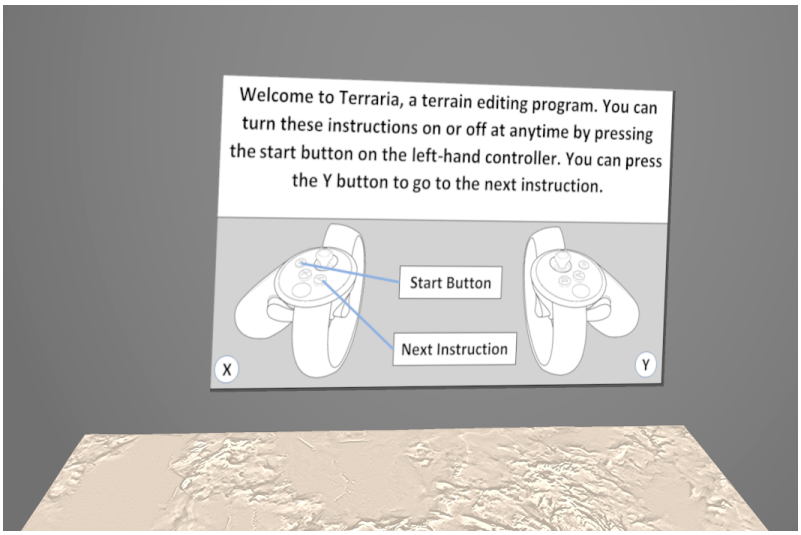


FIGURE 4.6: An example of the instructions available to user

4.3.2 Navigation

For the final interface all aspects of the FPV functionality that existed in previous prototypes were removed. There were a number of reasons for this. Primarily, the desktop application had no equivalent functionality so including it in the VR application would only increase the discrepancy between the two. It was also noted in the prototype testing that users did not try to use the FPV functionality without being directed to do so and generally spent a minimal amount of time using it. Finally, due to delays in other areas of development it became necessary to reduce the scope of the project.

Therefore, previous navigation techniques for use in FPV do not apply to the final interface. However, the original world-in-hand metaphor for viewpoint manipulation while in god view is retained from the initial design. Additionally, users have the ability to scale the environment by grabbing two points in the virtual space with the touch controllers and either bringing them together to reduce the scale or moving them apart to increase it. This mirrors the pinch/spread action to zoom used by most touch screen devices, which means most users should be familiar with the metaphor.

4.3.3 Environment Interaction

Environment interaction takes three forms depending on which tool the user is holding.

When holding the pen a user can add point or line constraints by using the tool to ray-cast onto the terrain, and a ray is visualised from the pen tip (shown in figure 4.7) to aid with this. Pressing the index trigger will then create a constraint. A single tap creates a point constraint while a hold and drag is used to draw a line constraint.

Similarly, while holding the paintbrush the user can select a point on the terrain by ray-casting with the tool and then paint the selected area by pressing the trigger. Since an area around the point is affected, a preview of the selected zone is highlighted while the paintbrush is in use and pointed towards the terrain. The user can also adjust the size of the area by pushing the joystick on the hand holding the paintbrush forwards (to enlarge the area) or backwards (to diminish the area). This change is reflected in the area preview.

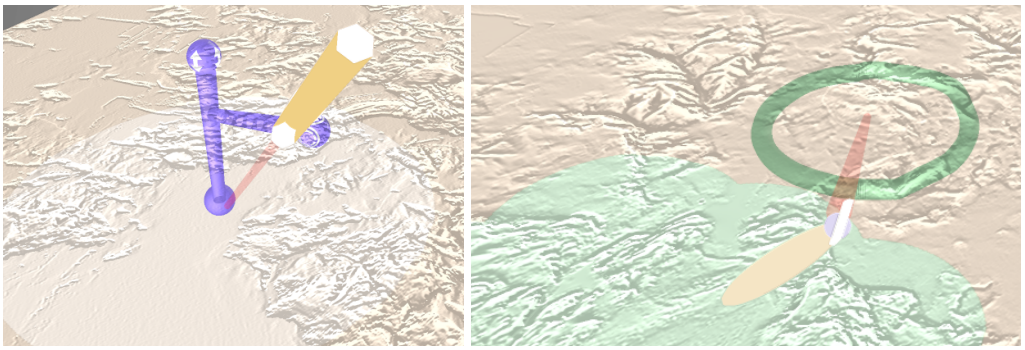


FIGURE 4.7: Left: An example of a point constraint being created; Right: Screenshot showing the painting preview while holding the paintbrush tool

When not holding a tool the user can directly interact with the constraint controls. For point constraints these comprise a vertical manipulator that can be dragged up or down and a free-moving horizontal manipulator, which controls the area of affect, slope angle and slope direction of the constraint (See figure 4.8). Specifically, a user can move the second component further from or closer to the constraint centre to adjust the area. They can move it either higher or lower relative to its default position to increase the angle of the slope in a specific direction. Lastly, the user can rotate it around the constraint centre to specify the direction of the angle.

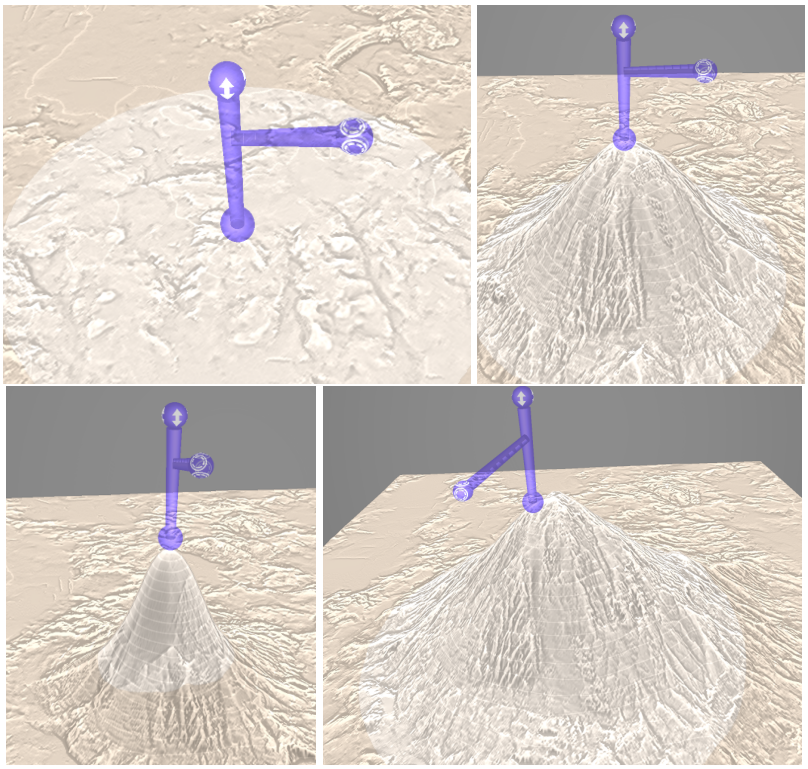


FIGURE 4.8: **Top Left:** An example of a new point constraint
Top Right: The same constraint with the height adjusted up
Bottom left: The constraint with a reduced area of effect
Bottom right: The constraint with the slope angle adjusted to be steeper on the left

4.4 Summary

Based on the existing desktop interface we created an initial design concept using findings from previous research to avoid many of the common issues related to designing VR interfaces. This initial design was realised as a paper prototype and underwent user testing by cognitive walkthrough [8]. Based on the feedback from these tests the design of the 3D widgets for terrain control was overhauled and the user instructions were improved as well as many other changes. A high fidelity prototype was then created using Unity 3D. This allowed for real-time terrain interaction in VR. This prototype was heuristically evaluated by expert evaluators. Although there were fewer major changes to be made after this iteration of testing, the data we collected was still useful in identifying smaller usability issues. The final design was then created based on the high fidelity prototype together with improvements to address the issues that arose during testing. This design was implemented using the same terrain rendering software as the original desktop application with the modified interface for VR.

Chapter 5

Experiment Design

In this section we review our research questions and explain how we plan to answer them.

5.1 Measures

Recall that the primary research question of this research is: *Is it advantageous to use a VR interface with a HMD and 6-DOF controllers over a WIMP desktop interface for terrain modelling?* We defined advantageous as faster, more accurate or more usable in section 1.4. Each of these components can be independently measured and compared allowing for the formation of three sub-questions.

5.1.1 Modelling Time

Is it faster to create a terrain model with a predetermined set of features in VR or using a desktop? To answer this question we need a consistent method to measure the speed at which users can perform tasks with each interface. Specifically, we want to compare the time that would be required for a typical use case in each interface.

Asking users to simply recreate a particular terrain and then self report on the time required would be insufficiently consistent since there is no precise definition of completion. In this scenario the time of completion would heavily depend on the standards the user felt they should reach in terms of precision. To avoid this issue we gave users a list of clearly defined items and asked them to complete the list. This meant that task completion occured when all items were completed. By clearly defining the tasks it was then possible for the evaluator to visually follow the participants progress and manually stop the timer after the final item was completed. It also enabled greater specification of a user's actions, allowing coverage of all typical use cases for the interface.

5.1.2 Modelling Accuracy

Because we are concerned with creating terrains based on the user's conceptualisation rather than creating precise copies we measure similarity by using a panel of human participants. This is as opposed to using precise physical similarity which might be measured by computing volume differences.This would not be reflective of conceptual accuracy as a terrain that is very similar to the copy target but has all features consistently shifted in a particular direction should score highly according to conceptual accuracy but would have a very large volume difference. Our research question is therefore: **Are terrains models created in VR visually closer to a target terrain image than those created with a desktop system?**

5.1.3 Usability

Usability refers to how well suited a system is to use for its intended purpose. Usability is a function of the quality of the interface, the quality of the information provided to the user and the usefulness of the tools available which contribute to the overall user satisfaction with the system [40]. With regards to usability our research question is: **Do users rate the usability of the VR system more highly than that of the desktop system?**

5.2 Participants

5.2.1 Recruitment

Participants were recruited from graduates of the UCT Computer Science honours 4-year undergraduate program. Specifically, participants were required to have completed the Computer Games Development major during their undergraduate studies. This ensured that they had some familiarity with 3D content creation, having built a 3D game as part of their studies. This was important for two reasons. Firstly, it meant that our participants were more representative of the target population. While not experts they did have a similar basic skill set to professional 3D content creators. Secondly, since there would be limited time avaialable for training, having prior knowledge of the basic components of 3D interfaces meant that they would only need to learn the particularly unique aspects of the terrain applications.

Recruitment took place via a group email sent to the honours class and an incentive of R30 (ZAR) per participant was provided. To increase the pool of participants, students from previous years were personally contacted. The same incentive was offered to these individuals.

5.2.2 Physical Apparatus

To carry out the experiment we used a desktop PC with a GTX 1080 graphics card, an Intel i7 processor and 16 GB of RAM. For the desktop interface treatment participants used a single computer screen with a standard qwerty keyboard and mouse. For the HMD treatment participants use the Oculus Rift CV1 headset along with the Oculus Touch hand controllers. The user was tracked with two Oculus trackers. They were able to move in a square area of: 2.5 X 3.5 metres. All training and experiments took place in the same room where only the participant and evaluator were present. There was some minor noise from foot traffic outside but this was judged to not be distractingly loud.

5.3 Experimental Procedure

5.3.1 Training

Prior to beginning the experiment, participants were required to take part in a 30 minute training session. This helped reduce the influence of disparities in previous experience with VR equipment. Furthermore, research has shown that basic training with VR can greatly improve user performance in VR applications [56]. During training users participated in the Oculus introductory experience, which Oculus Rift users complete when setting up the hardware. Also, users were given 10 minutes to familiarise themselves with the desktop and HMD terrain creation interfaces, respectively.

5.3.2 User Tasks

Accuracy Task

For each treatment, participants were given a terrain to replicate as faithfully as possible. They were provided with 2D images of the target terrain from a top down, front, back and 2 side views. In the VR interface users had these images available in a heads up display within the headset, while for the desktop interface printed versions of the images were made available. They were then required to replicate the terrain as closely as possible within 15 minutes. The images provided to participants can be found in appendices E,F.

Speed Task

Participants were given a list of feature descriptions and accompanying visuals of what to include in a terrain. Again, these instructions were available via heads up display or printed pages for VR and desktop treatments respectively. The features were chosen to make full use of all the interactive tools available to participants. The feature lists can be found in appendices G,H. Participants were then verbally instructed to create a terrain, that included all the features listed, as quickly as possible. The instructions were:

1. You have 30 seconds to read the feature list.
2. Once 30 seconds are up you must create all the features as quickly as possible.
3. All features must be completed.
4. There is no accuracy requirement for this task: so long as the features are identifiable.
5. Alert the evaluator once all features have been completed

Usability Task

After using a given interface for the accuracy and speed tasks as well as the training session, participants completed a usability questionnaire, specifically the Post-Study System Usability Questionnaire (PSSUQ) [39]. The PSSUQ has been shown to be both reliable and adaptable [22]. Which makes it suitable to measure both desktop and VR systems. Sections of the questionnaire have also been used to evaluate augmented reality interfaces [33]. Additionally, the three sub-measures provided by the questionnaire allow us to further analyse experimental data. We did not use a specialised VR usability questionnaire for the VR treatment as we wanted to be able to compare results across treatments.

5.3.3 Protocol

The following procedures were adopted for training and the experiment itself.

Training

1. Experimenter informed the participant that some VR participants do experience discomfort and nausea and that they would be allowed to stop at any time.
2. Participant signed informed consent for the experiment.

3. Experimenter instructed the participant on how to put on the VR equipment, as well as how to adjust it for comfort.
4. Participant undertook the Oculus introductory experience.
5. Participant removed VR equipment and opened the desktop terrain editing interface[1].
6. The participant worked through a short set of tasks from a written set of instructions using the desktop interface. The task list can be found in appendix I
7. Once the Participant completed the tasks using the desktop interface they put the VR equipment back on.
8. The participant completed a similar set of tasks using the VR terrain editing interface. Task instructions were displayed in VR rather than being written down so that the participant was not required to remove the headset.
9. Once the VR tasks were complete the training session was over.

Experiment

1. The experimenter briefly explained the tasks and the nature of the experiment.
2. Desktop Treatment[2]:
   (a) The participant was given reference materials for the accuracy task and the goal of task was explained.
   (b) The system was initialised with the default terrain (flat with no features) for the desktop interface.
   (c) The participant manipulated the terrain as desired until they were satisfied or 15 minutes had passed.
   (d) The completed terrain was saved for later evaluation by the evaluator.
   (e) The participant was given reference materials for the speed task and the required minimum level of accuracy was explained (i.e., that it must be visually apparent that all instructions had been followed)
   (f) Again, the participant started with the initial default terrain (flat with no features) on the desktop interface.
   (g) Timing started when the user first clicked anywhere on the interface.
   (h) The participant manipulated the terrain until they had completed all required features.
   (i) Timing stopped.
   (j) The completed terrain was saved for confirmation of validity and the time taken to completion was recorded.
   (k) Participant filled out the PSSUQ.
3. HMD Treatment:
   (a) The participant was given a short time to adjust the HMD equipment to a comfortable fit.

[1]Note that the ordering of tasks 3-4 and tasks 5-6 was swapped for 50% of participants. This was to ensure that no advantage was given to either interface based on the order in which training was conducted.
[2]Note that the ordering of tasks 2 and 3 swapped for 50% of participants.

(b) The participant performed tasks a-j from the desktop treatment but used the VR interface with the HMD equipment in place of the Desktop interface. Reference materials were available as a heads up display in the virtual environment.[3]

(c) The participant filled out the PSSUQ.

For the speed and accuracy tasks, in addition to the treatment factor of VR or Desktop interface, there is a secondary factor, hereafter referred to as *task variation*. This factor takes on a values of 1 or 2 depending on the instruction set participants were provided with or the terrain they were required to reproduce for the speed and accuracy tests, respectively. Task variation was counterbalanced with respect to treatment to reduce the possibility of a bias towards a specific treatment. This resulted in 4 possible orderings for participants, as described in table 5.1. Both the task variation and the varied ordering of treatments are used to prevent potential learning effects. By varying the treatment order we reduce the chance of participants learning how to use the interface better across treatments. By varying the tasks slightly we prevent the participants from learning the task itself and therefore gaining a benefit when completing it a second time.

Permutation	Treatment Order	VR variation	Desktop variation
1	VR then Desktop	2	1
2	Desktop then VR	1	2
3	Desktop then VR	2	1
4	VR then Desktop	1	2

TABLE 5.1: The possible task permutations for participants in our study.

For the usability task there was no task variation as the usability questionnaire was the same for both conditions.

5.3.4 Data Gathering

Accuracy Task

During the experiment the terrain created by the participant was saved but not immediately evaluated. After all experiments were complete a panel of two judges was asked to rate the accuracy of the output terrains. As mentioned in section 5.1.2 we felt that using human evaluators would give us more meaningful evaluations. A panel was used to reduce the chance of personal biases in the evaluator skewing the results. The judges discussed each terrain before deciding on a single score. This avoided a scenario where each judge scored differently based on their personal standard of quality. The judges were unaware of which participant's terrain they were marking and which treatment was used to create the terrain. All the terrains of task variation 1 were judged first in randomised order followed by all the terrains of task variation 2. Both judges were professional 3D artists, which meant that they had experience with evaluating and designing 3D models. Each terrain was compared to the the reference images provided to the participants. The judges were asked to score accuracy on a scale of 1-5 according to:

- Presence of major features;
- Height or depth of features;

[3]The tasks were varied slightly between the treatments. See table 5.1 and accompanying explanation for further details

- Volume of features;
- Slope angles of features
- Positioning of features
- Level of overall detail

The rubric for assessing terrains can be found in the Appendix J

Speed Task

The time taken to complete the speed task was automatically recorded by the terrain synthesis software and saved with the completed terrain. After the participant had completed the experiment the experimenter labelled and stored the files for later reference. The terrain was also checked by the experimenter to ensure that all necessary features were present. This was done post experiment and the results for this section were discarded for those participants who did not have all features present.

Usability Assessment

We used the Post-Study System Usability Questionnaire (PSSUQ), a 19-item, 7 point scale questionnaire [39], to measure the usability of our system. The PSSUQ allowed us to collect results on overall satisfaction, system usefulness, information quality and interface quality.

Chapter 6

Results

6.1 Introduction

In this chapter we present the results of our user study based on 21 participants. All participants, except one, were male and were aged between 19 and 30. They all had recent (within 2 years) experience with 3D content creation due to our selection criteria in section 5.2.1. The participants undertook the training and experimental processes discussed in the previous chapter. None of them chose to stop the study due to ill effects, such as simulator sickness, and thus 21 full sets of data were available for analysis.

Since we have a within subjects design, and our speed data has two factors, a linear mixed model(LMM) tool is used for the analysis of the data which is normally distributed. The accuracy data also includes 2 factors but is not normally distributed, therefore a generalised linear mixed model (GLMM) is used. For our usability data *Paired-sample t-tests* and *Wilcoxon signed-rank tests* are suitable since there is only the single factor of treatment present. Using a within-subjects design means fewer participants are needed overall since we do not need to consider the variability between participant populations. However, there is the risk that a within-subjects design may introduce learning effects as a confounding variable [26]. The treatment order and task variation factors were introduced to minimise this effect. Additionally, all data is tested for significant differences across treatment order and in all cases our counterbalancing was measured as sufficient.

6.2 Accuracy

In the accuracy task participants were required to reproduce a terrain with as much accuracy as possible, based on a set of 2D images. Their attempts were then graded by a panel of professional 3D artists according to several criteria shown in appendix J which combined gave the participant a total score out of 30 for the terrain. The lowest overall score among our participants was 6, which constituted the minimum possible grade. The maximum score achieved was 21 in variation 2 of the VR treatment. The median score was 11 for all conditions except variation 2 of the desktop treatment which had a median of 10.5. A summary of the data can be found in table 6.1. The full data can be found in Appendix K. The distribution of the accuracy data, separated by treatment, is shown in figure 6.1.

Variation	Treatment	Min	Max	Mean	Lower	Upper	SD
1	D	8	17	11.54	10	13	2.91
1	VR	8	15	11.2	10	12	2.15
2	D	6	17	11.4	6.5	13	3.92
2	VR	6	21	12.64	9	18	5.18

TABLE 6.1: Participant accuracy for different task variations and treatments.

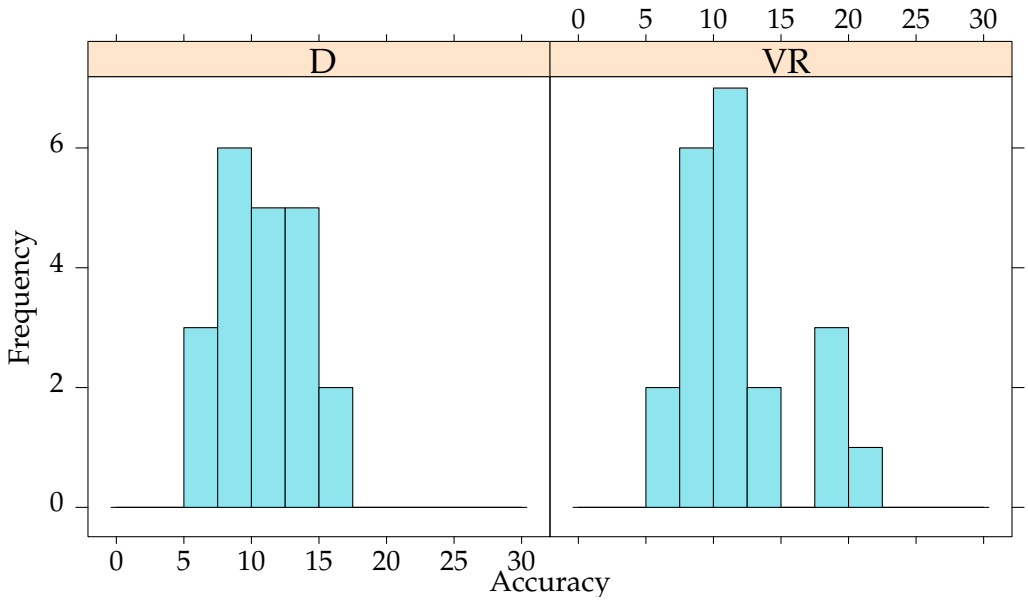


FIGURE 6.1: The distribution of accuracy scores among participants in all conditions.

We performed a Chi-squared goodness of fit test to establish whether our data was Poisson distributed. The data was separated by both treatment and variation. The results of these tests are summarised in table 6.2

Factor	Df	N	χ^2	Sig(p)
Treatment = Desktop	2	21	1.1014	0.577
Treatment = VR	3	21	5.119924	0.163
Variation = 1	3	21	7.923097	0.048
Variation = 2	3	21	7.663724	0.053

TABLE 6.2: Poisson goodness of fit for Accuracy data.

Despite our data from variation 1 not strictly conforming to the Poisson distribution with $p = 0.0476$ it was still the best fit compared to other distributions. The GLMM was therefore carried out with a Poisson link function. First, potential order effects were tested for across our factors with the following results: $\chi^2(1) = 0.970$, $p = 0.325$ for variation order and $\chi^2(1) = 0.933$, $p = 0.334$ for treatment order. This means that our counter-balancing is sufficient to remove any potential ordering effects between conditions.

We then proceed to test for our main effects again using a GLMM test. The results of this test are summarised in table 6.3 below.

Factor	Df	N	χ^2	Sig(p)
Variation	1	21	0.0078	0.93
Treatment:Variation	1	21	0.8462	0.358
Treatment	1	21	0.8041	0.37

TABLE 6.3: Main effects of Accuracy comparisons.

The first row of table 6.3 shows that task variation has no significant effect on participant accuracy. This is expected as the tasks were purposefully kept similar to avoid any unwanted effect. This result is visually confirmed in figure 6.2 (left), which shows the accuracy data across variations. The second row in table 6.3 indicates that there is no significant cross-over effect between our two factors. Since neither factor had a significant effect in itself, this is not surprising.

Finally, we find that the treatment itself has no significant effect, i.e. that there was no significant difference in participant accuracy between the VR and desktop interfaces. Figure 6.2 (right) visually confirms this. Given how similar the data is across treatments it seems unlikely that increasing the participant pool would result in a significant result.

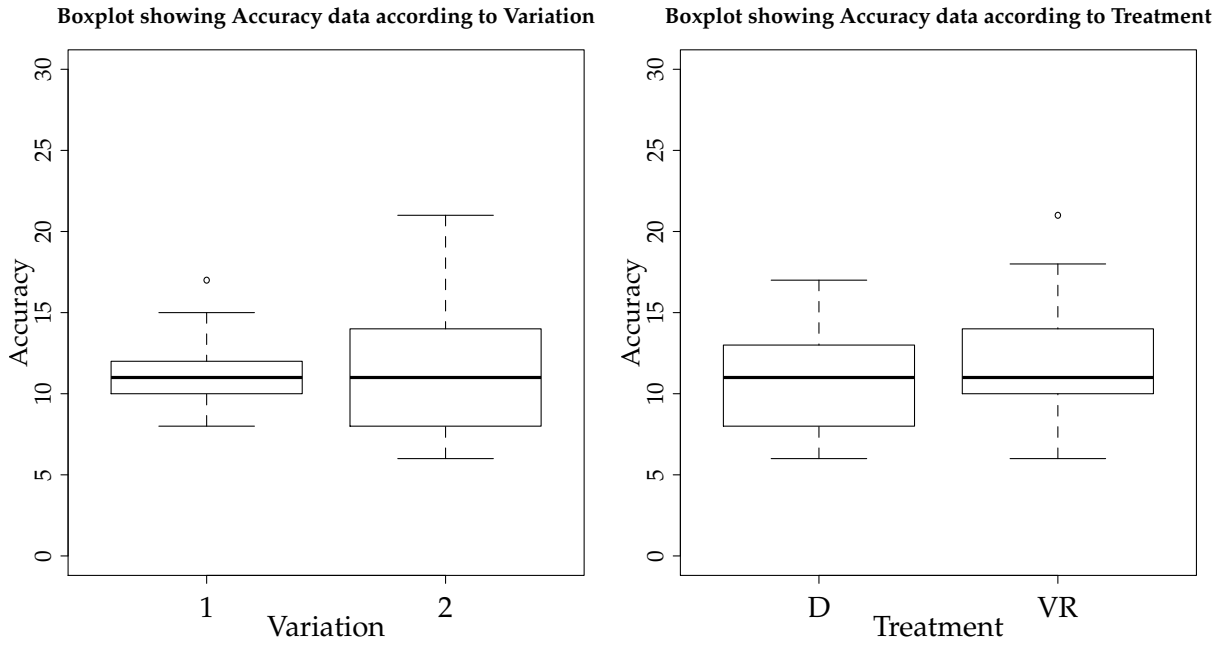


FIGURE 6.2: Accuracy scores among participants by Task Variation (left) and treatment (right)

Although we required familiarity with 3D modelling, we did not recruit 3D artists specifically. This meant that participants may have not had sufficient experience with reproducing 3D models from 2D images. This is supported by the generally low accuracy scores, with an average score of just 11.48 out of 30. The limited time participants were given (15 minutes) may also have contributed to this outcome as many participants spent the majority of their time on a single feature and did not complete the rest of the terrain. Additionally, even though participants had some practice using both systems,

many still seemed unfamiliar with using the applications, regularly referring back to the instructions from the training session which were made available to them. This may have been due in part to the delay between the training and experiment sessions, but was probably mainly a function of the limited time spent training with each interface. Unfortunately, we were not able to allow longer training due to a number of factors, including: deadlines for completion of data collection and difficulty in recruiting participants.

Another factor that may have contributed to the uniformity of the data as a whole is the use of a panel to judge the accuracy of the terrains. Our evaluators, while professional 3D artists, were not used to having to act as judges or markers. This may have led them to be more lenient towards obviously bad submissions and more critical when evaluating some of the better ones.

6.3 Speed

For this task participants had to complete a set of items as quickly as possible. For each participant the time from their first interaction with the terrain to self reported completion of all items was recorded. It was planned that if participants did not complete all items their data would be discarded. However, this did not occur and there was no need to remove any speed data. The fastest time achieved was 256.46 seconds while the longest time taken by any participant was 865.25 seconds (All times in this section are reported in seconds). The median time across all conditions was 531.4s. A summary of the data can be found in table 6.4.

Variation	Treatment	Min	Max	Mean	Lower Quartile	Upper Quartile	SD
1	D	299.58	771.66	456.03	321.38	537.21	171.41
1	VR	458.72	778.88	583.12	514.51	625.71	101.83
2	D	256.46	865.25	506.72	397.46	608.28	177.88
2	VR	383.97	783.25	565.6	515.78	596.07	119.69

TABLE 6.4: Participant speed for different Task Variation and Treatments.

When analysing the data we first tested for normality and homoscedasticity of variance using the Wilks-Shapiro and Brown-Forsythe tests, respectively. This confirmed that our data, pictured in figure 6.3 below, adhered to these assumptions. The graph uses 100 second buckets to group results. Larger time windows would have resulted in the majority of participants being grouped in a single bucket while smaller time windows would result in only 1-3 participants per bucket. This decision was made purely to create a useful visualisation and had no effect on the statistical tests.

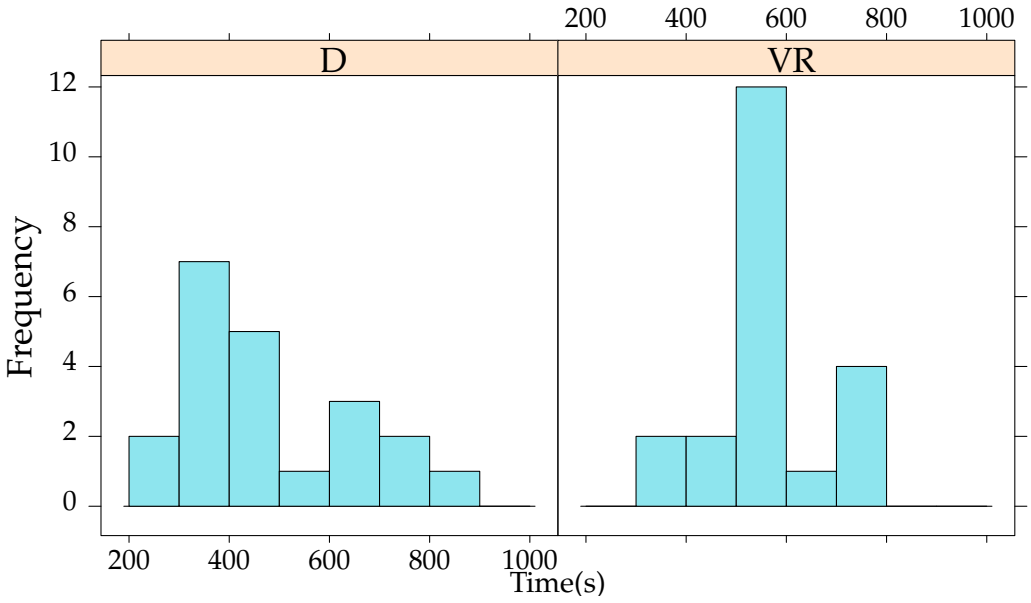


FIGURE 6.3: The distribution of speed times among participants in all conditions.

Again, we first tested for ordering effects against both variation and treatment, but used a linear mixed model (LMM) test since our speed data is parametric. The results of this are: $F(1, 19) = 0.111, p = 0.742$ for variation order and $F(1, 19) = 0.084, p = 0.776$ for treatment order. This means that our counterbalancing is sufficient to remove any potential ordering effects between conditions.

We then proceeded to test our main effects, again using an LMM test. The results of this test are summarised in table 6.5 below.

Factor	Df	Df res.	F	Sig(p)
Variation	1	19	0.223	0.642
Treatment:Variation	1	19	0.406	0.532
Treatment	1	19	7.023	0.016

TABLE 6.5: The main effects of Speed comparisons

The first row of table 6.5 shows us that variation has no significant effect on the time participants took to complete this task. Again, the variations were kept similar to avoid any unwanted effects, so this was expected. This result is visually confirmed in figure 6.4 (left), which shows the speed data across task variations. The second row indicates that there are no significant cross-over effects from our two factors. Since variation had no significant effect this is not surprising.

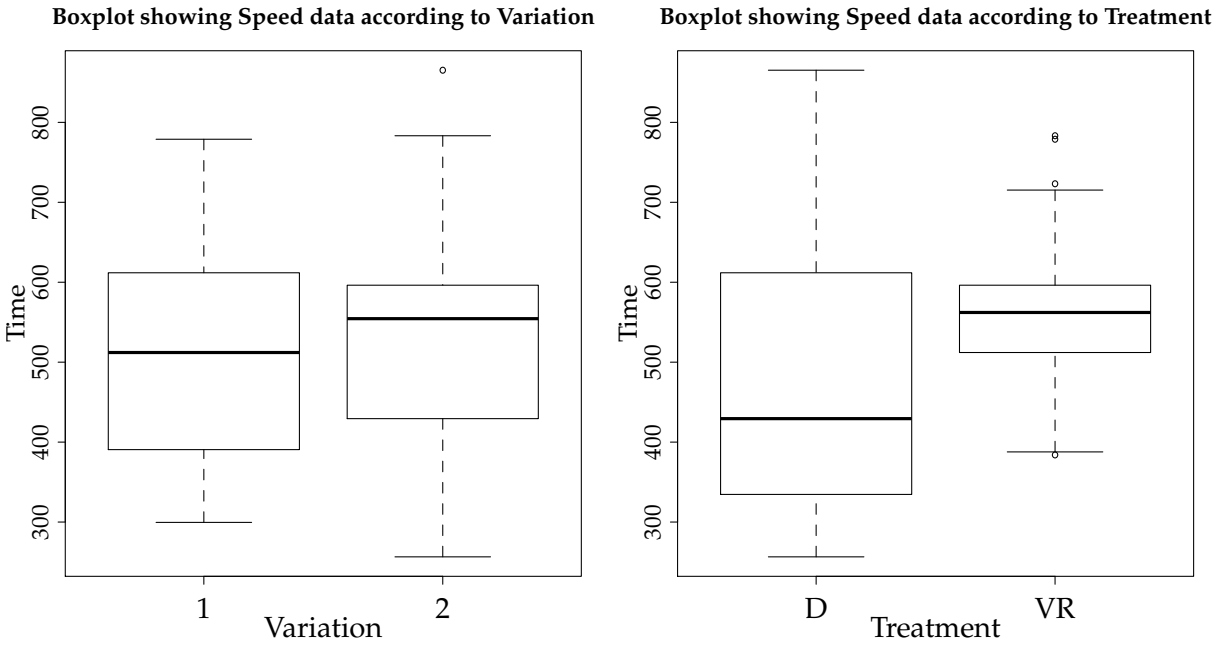


FIGURE 6.4: Speed test results among participants by Task Variation (left) and treatment (right)

In the final row of table 6.5 there is a significant value across treatment conditions. Figure 6.4 (right) shows that this significance occurs because the VR condition takes longer than the desktop condition. Although participants are generally more consistent in their times when using the VR interface there are also a number of outliers for the VR condition, mainly from those who took much longer than the average.

That VR terrain modelling takes longer is not the expected result, given that previous research suggests that participants are faster when performing basic 3D manipulation in VR [56, 55]. There are a number of possible reasons for this.

For instance, the existing research that shows VR to be faster only applies to specific tasks and does not take into account other actions. The speed gains in these areas may be insufficient to make up for actions where VR tends to be slower, such as menu interaction [10, 28]. However, it seems unlikely that this entirely accounts for the difference in task completion times between the treatments. Another possible contributing factor is the slower frame rates during terrain editing. Because the terrain synthesis is a costly operation the program lags slightly whenever a participant makes changes to the terrain. Although this lag is uniform across treatments, since it is due to the synthesis operation rather than any difficulty rendering the scene, it may have had a larger impact in the VR treatment. Specifically, in tasks that require direct manipulation this delay leads to a momentary disparity between physical and virtual (avatar) hand positions. This is potentially more disruptive to the task than having a mouse cursor lag slightly. In particular, most PC users have experienced some sort of lag on a desktop but are unlikely to have experienced a disconnect between their actual and visual depiction of its position.

Another way in which direct manipulation in the VR condition may have caused further delays for participants was viewpoint manipulation. For the desktop condition participants typically placed the terrain in a position from which they could carry out all tasks without needing to re-adjust it. Since there is no accuracy component to this task participants did not need to move the viewpoint to check the appearance of features from multiple angles. This meant that for the desktop condition participants spent very little time on viewpoint manipulation, a task which can be quite cumbersome on a desktop as opposed to in VR, where a user merely needs to move their head. However, in the VR condition

participants could not reach all points on the terrain without regularly adjusting its position relative to themselves. Although participants could decrease the scale of the terrain, most found that they were then unable to perform many of the direct manipulation tasks at small scales, such as grabbing specific widgets. This meant that participants spent more time and effort moving the terrain around in the VR condition than in the desktop condition. It is worth noting that this may not reflect a more general use case.

There may also be other aspects, separate from the interfaces, that caused the desktop treatment to be faster. As mentioned previously, despite the initial training session, many participants still seemed unsure of the controls, particularly when using the unfamiliar Touch controllers, which they could not see. Although participants had spent an additional 15 minutes completing the accuracy task prior to the speed task, some may have still not felt comfortable with the controls. The VR interface would have been more affected by this due to its novelty given that very few participants had previously done 3D modelling in VR and the fact that the controllers were not visible during the task. The evaluator observed that this led to more errors while participants performed the speed task for the VR treatment. The most common example of this was participants pushing the wrong button and accidentally deleting a previously completed constraint. Another potential issue with the VR interface is that some participants struggled to identify the depth of objects despite the inherent stereoscopy [54]. This meant that they struggled to be consistent during actions that required direct interaction, often needing multiple attempts to grab objects. This slowed down some participants significantly and is likely the reason for the outliers in the VR condition (see figure 6.4 (right)).

Finally, it may have been that, despite our efforts to ensure the quality of the VR interface, it was simply inferior to the desktop version. However, results in the following section suggest that this was not the case.

6.4 Usability

As stated previously, we use the PSSUQ questionnaire to evaluate the usability of our system [40]. Participants filled in the questionnaire for each treatment immediately after completing the relevant speed and accuracy tasks. The PSSUQ gives 4 different measures based on subsets of the 18 questions [40]. These measures are Overall score (items 1-18), System Usefulness (items 3-9), Information Quality(items 13-18) and Interface Quality(items 1, 10 and 11) [40]. System Usefulness reflects how easy the system was to use to complete the tasks it was designed for. Information quality encompasses the help provided by the system to guide the user when errors occur or when information is needed to complete a task. Interface quality is based on how much users enjoy interacting with the system. Each item in the questionnaire uses a 7 point Likert scale. All items are consistently aligned, so we calculate the score of a particular measure by taking the average score across all items pertaining to the measure [40]. This approach is sufficiently robust that we can include subject data where a single item is unanswered [41]. It is fortunate that this is the case as a number of participants did not answer item 13 of the questionnaire: *The system gave error messages that clearly told me how to fix problems*. Some participants commented that they did not encounter any errors so could not answer the question. We did, however, have two test subjects who had multiple unanswered items and therefore had to exclude them from our usability evaluation.

In this section we report the results of each measure separately and give possible reasons for the outcomes. As an overview of the data analysis we have provided table 6.6, which details the different tests for each measure and whether there was a significant difference between the treatments. A box plot showing data for each measure by treatment is also present below, in figure 6.5. Note that lower scores indicate a more favourable rating in the PSSUQ, i.e. the best possible score is 1 while the worst is 7.

Factor	Parametric	Significance	Test used
Overall	No	Yes	Wilcoxsign test
System Usefulness	lognormal	No	t test on log of scores
Information Quality	Yes	No	t test
Interface Quality	lognormal	Yes	t test on log of scores

TABLE 6.6: Summary of usability measures.

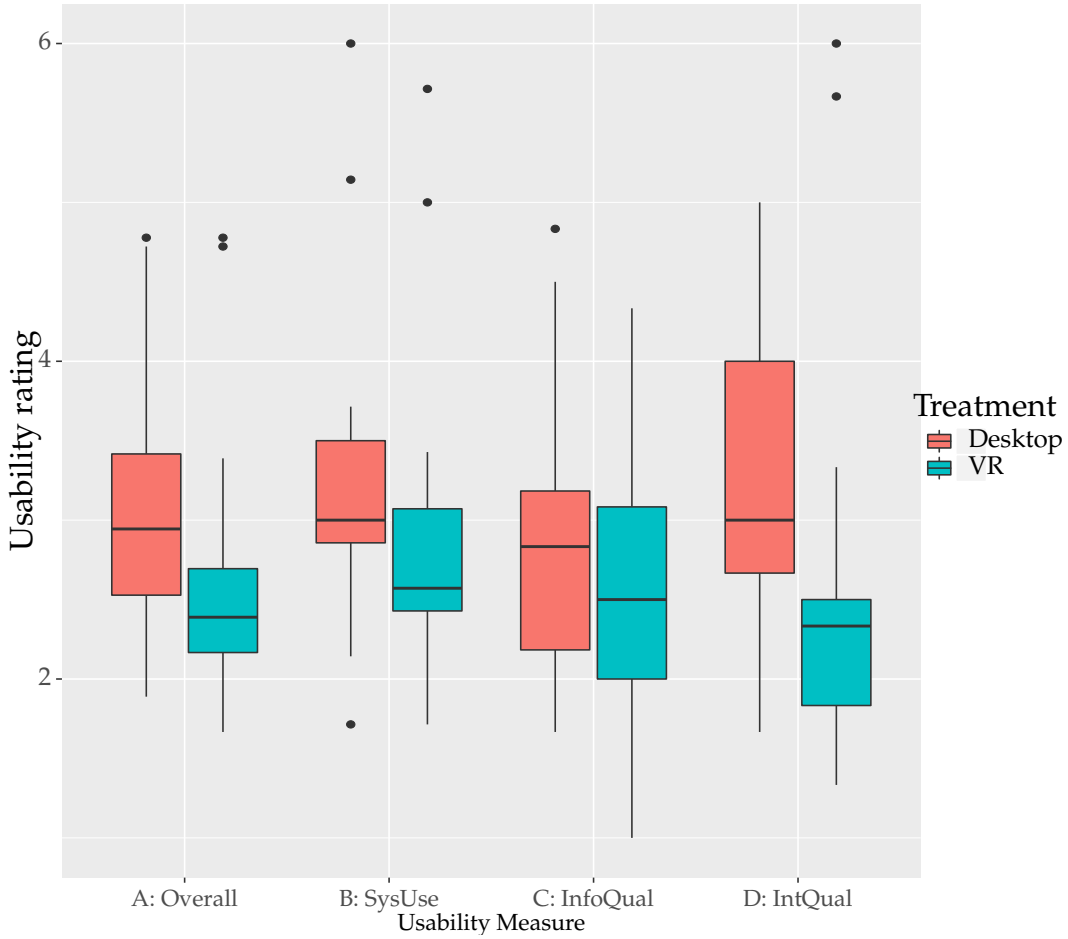


FIGURE 6.5: All usability results by measure : A - Overall score, B - System usefulness score, C - Information quality score, D - Interface Quality

6.4.1 Measure: Overall

Table 6.7 summarises the data pertaining to the overall usability of the interfaces in our experiment. This data was arrived at by averaging the scores across all items of the PSSUQ. As stated above, lower scores indicate greater usability. We can see that the mean VR score is 0.38 lower than the mean desktop score. Additionally, the scores for the interfaces fall on either side of the mean score of 21 other studies using the PSSUQ which is 2.82 [41]. This suggests that the average usability of the interface was of a similar standard as other studies using the PSSUQ.

Treatment	Min	Max	Mean	Lower	Upper	SD
D	1.89	4.78	3.03	2.53	3.42	0.8
VR	1.67	4.78	2.65	2.17	2.69	0.86

TABLE 6.7: An overall summary of usability data

In figure 6.6 we can see that while the scores for the desktop treatment seem normally distributed this is not the case for the VR treatment. Specifically, there is an unusually high number of participants who gave the VR treatment a score between 2 and 2.5 for overall usefulness.

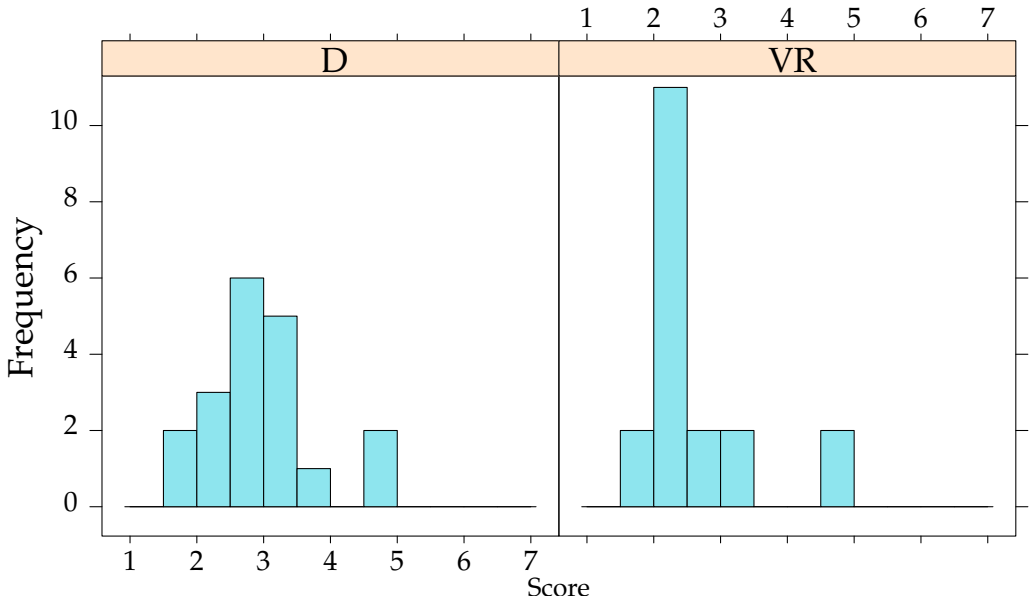


FIGURE 6.6: Distribution of Overall Scores by Treatment

The overall measure data was found to not be parametric for the VR treatment, although the desktop treatment was. We therefore used the non-parametric Wilcoxon signed-rank test to analyse our data. We first tested for any order effect and found no significance, as in previous sections: $Z = 1.028$, $p - value = 0.317$. We then tested for and found a significant difference by treatment: $Z = 2.378$, $p - value = 0.016$. From figure 6.5 we can see that this means the VR treatment had a significantly lower overall score and is therefore better than the desktop treatment with regards to overall usability.

This preference by participants corresponds to their verbal comments both during and after the experiments. Despite the fact that no favour was shown to either system during the recruitment or experiment, the novelty of the VR system unavoidably points to it being the newer system on which research was being conducted and participants may have felt pressure to favour it. While these concerns must be considered, there is some evidence in the measure for information quality that in cases where the systems did not differ there was correspondingly no significant difference in usability scores.

If the preference for the VR system is based on genuine advantages, one key component is likely to have been the camera controls. This was as much an effect of participants finding the VR viewpoint manipulation fast and intuitive as them finding the desktop interface frustrating when it came to camera manipulation. In a similar way participants may have felt less frustrated overall with the VR system. Particularly when it came to adjusting the widgets, since in cases where multiple adjustments needed

to be made in the desktop system a single action was normally sufficient in the VR system. This is a potential benefit of the more natural and direct interactions with the virtual objects offered by VR.

6.4.2 Measure: System Usefulness

Th measure for system usefulness is based on items 3-9 of the PSSUQ. Table 6.8 provides a summary of the data. The mean VR score is 0.43 lower than the mean desktop score, a smaller difference than exhibited by the overall score. We can also see that the mean scores for both treatments are slightly higher, and therefore worse, than the mean overall scores.

Treatment	Min	Max	Mean	Lower	Upper	SD
D	1.71	6	3.22	2.86	3.5	1
VR	1.71	5.71	2.86	2.43	3.07	1

TABLE 6.8: System Usefulness data summary

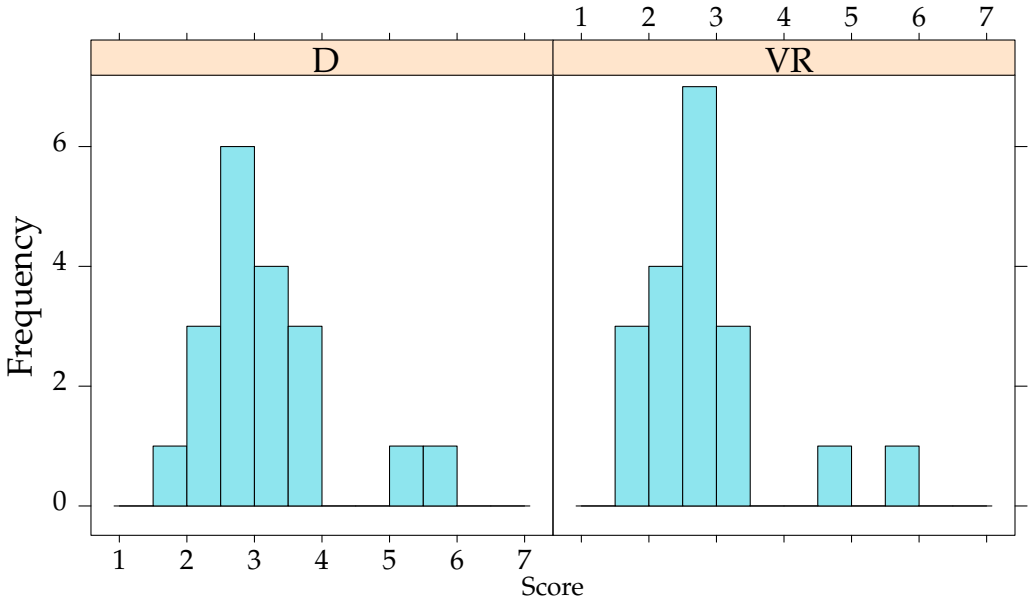


FIGURE 6.7: Distribution of System Usefulness Scores by Treatment

Here we found that our data for both treatments was log-normal. We also checked for that our data was homoscedastic using the Brown-Forsythe test: $F(1, 36) = 0.002, p = 0.963$. This allowed us to perform a paired samples t-test once we converted our data by taking the log of participant scores for this measure. As usual we first tested for any order effect but found none: $t(18) = 1.585, p = 0.13$. We followed this with our comparison by treatment, finding: $t(18) = 1.657, p = 0.115$

Although no significant difference between treatments was found for this measure, figure 6.5 shows that the VR treatment is visibly different from the desktop treatment. It may be that with more participants we could have achieved significance in this measure. A possible reason for the lack of significant difference is that both systems have the same tools even though they are implemented differently. This means that the actual tasks which each treatment supports are the same. Although this does not correspond entirely to usefulness, it may explain why there was no significant difference for this measure despite there being one for the overall measure.

6.4.3 Measure: Information Quality

Table 6.9 below provides a summary of the data for the information quality measure, based on items 13-18 of the PSSUQ. We can see the difference between the mean scores of the treatments is only 0.27, less than half that of the overall scores' difference of means and lower than any other measure.

Treatment	Min	Max	Mean	Lower	Upper	SD
D	1.67	4.83	2.84	2.18	3.18	0.87
VR	1	4.33	2.57	2	3.08	0.78

TABLE 6.9: Information Quality data summary

Figure 6.8 shows few high scores for this measure, unlike the other measures. If we refer to figure 6.4 we can see that the Information quality had the fewest outliers of any measure.

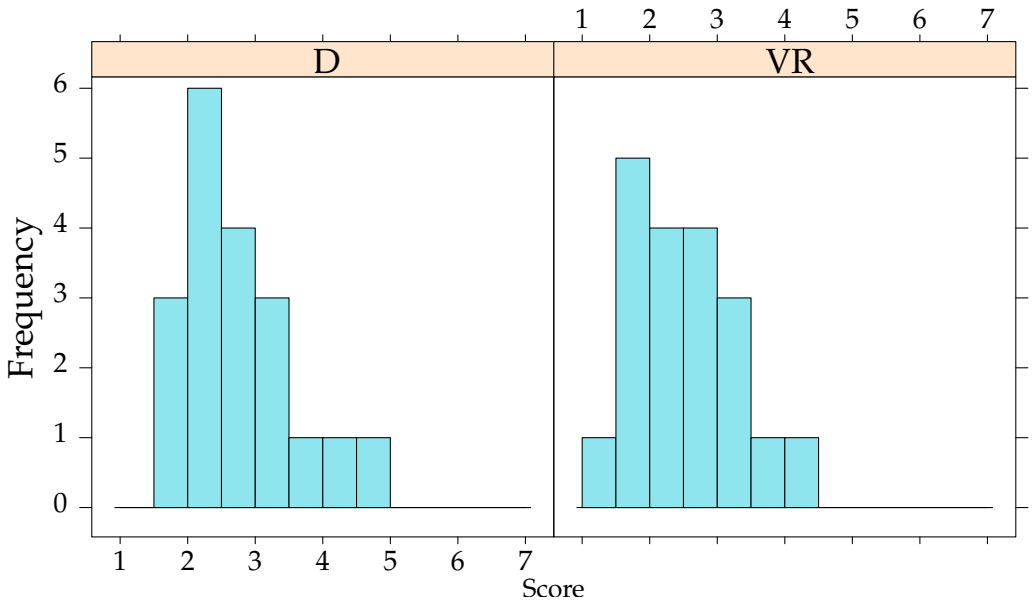


FIGURE 6.8: Distribution of Information Quality Scores by Treatment

For this measure our data was found to be normal for both treatments as well as homoscedastic across them. A Paired samples t-test was therefore suitable for analysis. We found no significant difference across the order of treatments $t(18) = 1.694, p = 0.108$ and also no significant difference across treatments: $t(18) = 1.531, p = 0.143$

This is not an entirely unexpected result. The information provided to participants in both treatments was the same even though the format was, of necessity, different. In figure 6.5 it can be seen that the scores for this measure are the most similarly distributed across treatments. This similarity also suggests that the case for participant bias as an explanation of the overall measure is not particularly strong. A strong participant bias for VR would have significantly favoured that treatment in all measures, including those where the systems were very similar.

6.4.4 Measure: Interface Quality

Table 6.10 below provides a summary of the data for the interface quality measure based on items 1,10 and 11 of the PSSUQ. The difference between the mean scores of the treatments is 0.77, the largest difference in means of any of the measures. It is also worth noting that the means for both treatments are higher than those for the overall score, with the mean of the desktop treatment being larger by a greater margin.

Treatment	Min	Max	Mean	Lower	Upper	SD
D	1.67	5	3.3	2.67	4	0.94
VR	1.33	6	2.53	1.83	2.5	1.27

TABLE 6.10: Interface Quality data summary

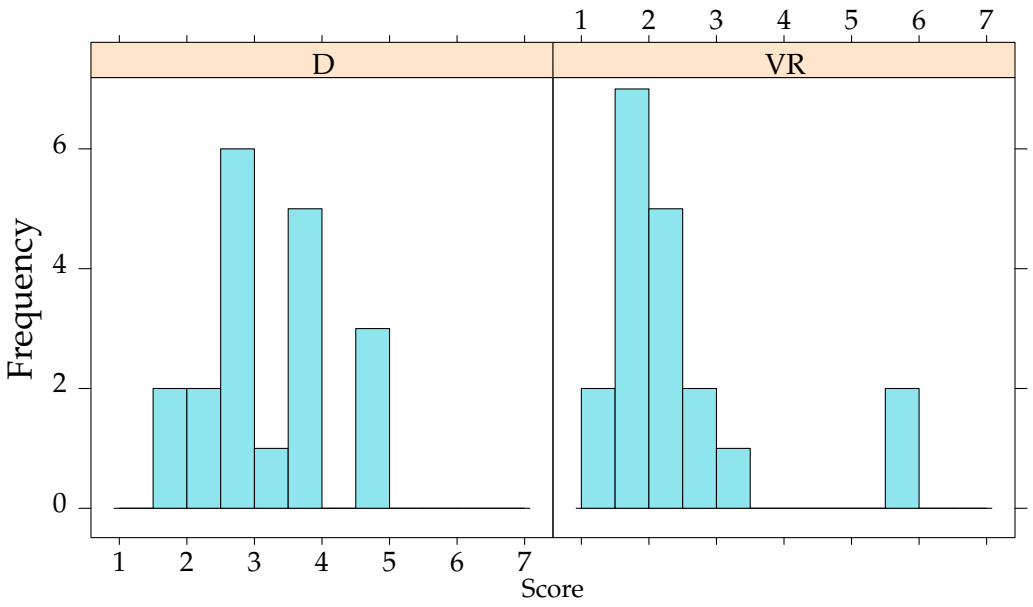


FIGURE 6.9: Distribution of Interface Quality Scores by Treatment.

Although the VR data was significantly different from log-normal: $W = 0.88481, p < 0.05$, both the desktop data and the residuals by subject and treatment were log-normal. The log of the interface quality data was also homoscedastic across treatments. On this basis, performing a paired samples t-test for our data analysis is justified. As in previous cases there was no significant difference across the order of treatments: $t(18) = 0.13, p = 0.899$. When testing across treatments however, we did find a significant difference: $t(18) = 3.284$, $p = 0.004$. Figure 6.5 shows that this was the measure that most heavily favoured the VR treatment over the desktop treatment.

This measure perhaps most directly reflects the participant's general attitude towards the system. Where items in other measures are often quite specific, the interface quality items include questions such as: *The interface of this system was pleasant* and *I liked using the interface of this system.* These questions seem more likely to reflect an emotional response not based on any particular aspect of the system, but rather on overall impression. There are two likely reasons that this may have so heavily favoured the VR treatment. Firstly, because of the vagueness of the questions participant bias in favour of VR may have been more strongly present. The other explanation is that participants simply felt less frustrated overall

with the VR system and because of the frustrations with camera control and the need for more actions to adjust widgets in the desktop system, participants scored the desktop treatment more poorly.

6.5 Discussion

For our results as a whole there is evidence that the VR interface did not perform as well as might have been expected based on the existing literature [56, 55, 65]. There are a number of possible explanations, which we explore in a more holistic manner in this section.

Training plays a large part in user performance in VR. It also affects desktop applications, although to a lesser extent. There are two major factors that cause this disparity in the need for training in VR and desktop treatments. The first is the relative novelty of VR. Because our participants were all trained programmers and had experience with 3D modelling, they were already familiar with basic concepts, such as camera control and widget manipulation, in the desktop treatment. However, this was not the case in VR. Users struggled to memorise the controls, primarily because they were seeing them for the first time. Another reason is that in VR the user is unable to see the external world. This makes it necessary to memorise controls based on their position. Because there is no visual reinforcement, participants struggle to remember where specific controls are, even when they can find the buttons easily. Including controller diagrams in the instructions helped with this but did not entirely alleviate it, especially when the user was not looking at the relevant instructions.

As mentioned previously, the complexity of the tasks participants performed in our experiments was greater than generally found in previous research. Although this is most relevant to user speed, it also affects accuracy and usability measures. In the case of our accuracy task, the complexity arose from the number of aspects a participant had to consider when transforming 2D images into a single 3D terrain with many features. This complexity may explain why our participants generally performed poorly on this task and is a reason why professional 3D artists should be used in any future testing. With regards to usability, the higher complexity of tasks performed by users introduced them to more of the features and potential flaws present in the application. This means the usability scores are more applicable to industry use cases than they would be if only very focused testing in specific conditions had been carried out, since they are based on a more holistic evaluation.

Although these reasons and others, such as the lag discussed in section 6.3 , may have caused the unexpected results, there are some aspects we know did not have any significant effect on participant performance. Specifically, we showed in our statistical analysis of results that neither the order of treatments or the task sets resulted in a significant difference between participants. This suggests that we successfully avoided any learning effect between treatments by the introduction of different task sets but also kept them similar enough that no other unwanted effects were introduced.

The discrepancy between subjective and objective data was interesting. Participants had a clear preference for using the VR application, with some needing to be told multiple times to exit the training session for that treatment. In fact, based on self reports during experiments and during pilot tests, it was not obvious that the desktop system was better than VR. It was only once objective results were gathered that it became evident that the VR system had significant drawbacks, namely speed. This suggests that participants were more concerned with the ease of functionality and the intuitive controls of the VR interface as opposed to the increased efficiency of the desktop interface. With extensive use, however, this attitude might change. It is hard to predict whether regular users might improve with the VR interface to the point where they can work as efficiently with it as with the desktop interface or if they will end up preferring the efficiency of the desktop interface.

Chapter 7

Conclusions

In this dissertation we have presented our research into the effectiveness of a VR interface for terrain modelling when compared to a desktop interface. We began by considering the recent adoption of VR devices in the consumer market and some of the potential reasons users might want to use them for 3D modelling. Methods for terrain synthesis and the requirements for modelling realistic terrain were also investigated. This allowed us to further define what was meant by *effectiveness* in the context of this research. Specifically, there were three core questions that need to be investigated when comparing VR and Desktop applications for terrain modelling: *Which is faster?*, *Which allows greater accuracy?* and *Which is more usable*?

This initial outline of the dissertation topics was followed by analysis of relevant existing research. Terrain synthesis methods were investigated in section 2.2 to confirm that our chosen desktop application was acceptable and we also looked at the creation of VR interfaces for large environments in section 2.3. In addition, in our related work chapter 3, VR interfaces for 3D modelling were investigated, along with previous experiments that compared Desktop and VR systems .

Once all relevant existing research had been considered we explained the interface design for use in our experiments (Chapter 4) and how those experiments were undertaken (Chapter 5).

7.1 Summary of Outcomes

This research focused on three research questions (as mentioned above and outlined in section 1.4):

1. *Is it faster to create a terrain model with a predetermined set of features in a VR or desktop application?*
2. *Are terrains models created in VR visually closer to a target terrain image than those created with a desktop application?*
3. *Do users rate the usability of the VR application more highly than that of the desktop one?*

In order to answer these questions experimental participants perform two sets of three tasks; one set using the VR system and the other using the desktop system. These tasks were:

1. Completing a representative set of terrain modelling tasks as rapidly as possible;
2. Recreating a terrain based on 2D images;
3. Completing a Post study system usability questionnaire (PSSUQ)

While participants performed the same general tasks in each treatment, the details of each task were changed to reduce any possible learning effect in our within subjects design. To avoid bias towards a specific treatment, the two sets of tasks were alternated between treatments. This meant that half the participants completed task set 1 in the VR treatment and task set 2 in the desktop treatment and the

other half did task set 1 on the desktop and task set 2 in VR. During the analysis of results (Chapter 6) we examined whether the task sets themselves had any significant effect on participant performance, but found that there was none. A summary of our results comparing the separate treatments can be found in the table 7.1 below.

Factor	Significant result	Treatment favoured
Speed	Yes	Desktop Interface
Accuracy	No	N//A
Usability	Yes	VR Interface
-System Usefulness	No	N//A
-Information Quality	No	N//A
-Interface Quality	Yes	VR Interface

TABLE 7.1: Table showing summary of results. All significant values were at a p<0.05 level of confidence.

These results were based on 21 participants who completed our experiment. Each participant was required to have previous experience with 3D modelling at a tertiary education level. Unfortunately, this requirement prevented the recruitment of more participants within the available period. Our participants were given 20 minutes training split between the two interface types. However, they were prevented from starting the experiment immediately after training to avoid introducing an unwanted fatigue effect. Although allowed to do so, no participants cancelled participation during the experiment. However, two participants did not fully complete the usability tasks and it was therefore necessary to remove their data for that task during data analysis.

7.2 Contributions

Over the course of our research an interface for terrain modelling suitable for use with VR equipment was developed. Additionally, during the process of designing this interface we demonstrated the effectiveness of user-centred design techniques for developing VR applications. The lessons learned may inform design decisions for future work in the area of 3D modelling in VR. In particular, the effectiveness of replacing traditional menus, which are not suitable for VR, with 3D virtual props was shown. This greatly reduced the user's reliance on menu interaction while also providing a visual indicator as to the selected mode of interaction. Finally, users mentioned that using virtual tools felt comfortable and natural and required less cognitive effort than menu navigation.

7.2.1 Data Gathering

The research also contributes towards the growing, but still small, collection of quantitative data comparing desktop and VR systems for use in creative design. Although other comparative studies exist, they are mainly within the areas of immersion or presence in games and the effectiveness of virtual training tools. In 3D modelling and content creation little research is publicly available. What does exist often focuses on very specific tasks, such as object manipulation. This is also often carried out using hardware that is not representative of the current industry standards, either using custom or outdated devices. To the best of our knowledge no other research exists comparing desktop and VR systems for

full industry use cases, such as terrain modelling, that also utilises mainstream commercial hardware such as the Oculus rift or HTC Vive.

7.2.2 Findings

Design Process

As stated above we found that user centred design techniques were effective when applied to design problems in VR. In particular, despite the complex technology in VR systems, it was possible to design a paper prototype that successfully captured a large number of the usability problems. The benefit of this was that a prototype could quickly be created and adapted without relying on expensive hardware components. Also, working with a paper prototype allows more people to simultaneously be involved in the design process as opposed to creating a full VR prototype which only allows a single user at a time. This potentially facilitates group brainstorming sessions during the design of VR systems.

However, we would not recommend that system designers proceed directly from a paper prototype to final product development. As the feedback on our high fidelity prototype showed there are some issues that will not be picked up by users unless they experience VR fully. For example, the perceived depth of components such as user instructions is hard to accurately recreate in a paper prototype. Avatar interactions are also difficult to simulate and may need to be altered due to technology constraints in the final implementation. Because of these issues creating and testing the high fidelity prototype was essential to our process of interface design.

Training

Apart from the lessons learned during the design process our final experimental process also revealed a number of interesting considerations. As mentioned in previous research, we found that training was key to user performance, particularly in VR [56, 55]. Unfortunately, due to time and participant recruitment constraints it was only possible to have a 30 minute training session with each participant. While some were able to learn the basic functionality of the applications in this period, the majority still seemed uncertain about some aspects when starting the main experiments. In an ideal situation participants should be able to train with each interface as long as is required to feel comfortable. Furthermore, unless given a specific training structure and motivation users may not use training time effectively. When our participants were given a short time of 'free exploration' they tended to explore the limits of the system, which did not necessarily translate to practical skills for normal use cases. A common scenario was a participant trying to see how large a mountain they could make.

Application Complexity

Possibly due to the lack of training our participants took longer to complete the speed test in VR than in the desktop treatment. However, there are other possible explanations for this as discussed in section 6.3. It is interesting that this conflicts with some earlier research [55, 56]. That research, however, was based on very specific actions that users were required to perform. Our results seem more in line with later research performed using a CAVE system that showed participants having similar times in both VR and desktop conditions when modelling using CAD software [65]. However, the latter experiment only had 8 participants so the data is not statistically reliable. It seems most likely that the complexity and variety of tasks in our experiment is the cause of weaker speed performance in the VR condition. Additionally, we feel our experiments are more representative of a practical use case than previous research which focused on a very narrow sets of actions.

Practical Use

Although participants did prefer the VR system in terms of usability, we would not recommend using VR over the desktop system in practical workplace situations based on our results. This is mainly because of the disadvantage of the VR system in terms of speed. Neither system performed better when it came to accuracy, which in a workplace scenario favours the desktop system as it requires no special hardware or reserved VR areas to use effectively. However, with some improvements to the VR application, particularly in terms of performance, and the benefit of longer periods of training it is possible that VR could significantly outperform a traditional desktop system in the areas we have looked at.

7.3 Shortcomings and Further Research

Due to a variety of challenges faced there are some aspects of this study that take away from how meaningful our results might otherwise be. These challenges occur mainly in the development of our applications and the recruitment and testing of participants.

7.3.1 Application Functionality

Unfortunately, during the process of building the final VR application a number of issues were encountered that required us to reduce the overall scope of the application. Most notably the removal of any first person point of view mechanic in the final interface. In other cases we had to remove functionality from the existing desktop application to achieve feature parity between the systems. For example the desktop application contained a tool to copy areas of terrain and paste them elsewhere. Because of time constraints it was not possible to implement this in the VR condition and it was therefore also necessary to remove it from the desktop application.

Another consequence of needing to bound the scope was that the VR interface was not as polished as it could have been. One aspect of this that may have had a particular impact on our results was the lack of sufficient depth cues for users who could not rely on stereoscopy to perceive depth. Research has shown that 4% of people are completely unable to use stereoscopy and an additional 10% struggle to reliably use stereoscopic depth cues [54]. This issue may have been picked up earlier had we tested with more users during the user centred design phase. As it was, none of our initial testers reported this issue and we were unaware of it until some participants in the final experiment encountered it.

Finally, one of the biggest issues with both desktop and VR applications was the lag generated by the terrain synthesis step. Because the desktop application itself was created a number of years ago it is not well optimised for current generation GPUs despite the synthesis algorithm itself being state of the art in the area of terrain synthesis [24]. Had there been sufficient time it may have been possible to optimise the back-end of the application to significantly reduce the delay effect. This would have brought the application closer to commercial VR modelling applications such as *Google Blocks* or *Medium*.

7.3.2 Recruitment and testing

Because of time constraints and availability of participants we faced a number of challenges during our testing of the final applications. Although more participants would have been preferable (as is usually the case), the larger issue was with the skill requirements for participants. As our accuracy results revealed, having experience with 3D development using engines such as Unity or Unreal Engine does not necessarily translate to the ability to create good 3D models. It would have been far better,

in retrospect, to have recruited participants from a design college that specialised in 3D design and modelling.

A further issue was that participants were insufficiently trained for the interface. Although we were aware of the importance of training for VR interfaces we had to avoid too large a time commitment from users. Additionally, the time required to achieve an adequate familiarity with the interface was underestimated. Unfortunately, our training was not adequate and this may have unfairly biased the results towards the desktop application, particularly in terms of speed.

7.3.3 Future Research

Despite these shortcomings we believe that this research shows that complex VR applications do not behave as might be expected based on evidence from specific VR tasks. We therefore suggest that there is room for more testing in related areas that may produce novel and useful results. In particular, testing that provides quantitative data from many participants can provide evidence for the strengths and weaknesses of VR as a medium for content creation in industry.

This research should preferably be done using professionals in the applicable area as experimental participants. Without suitable participants, test results are ambiguous and may reflect the weaknesses of novice users rather than of the systems being tested. It is rare to be in the position where one has the freedom to test on a large number of suitable participants but we believe there would be significant benefits to carrying out such an experiment.

Finally, it is still essential that the systems being tested are of as high a quality as possible. In our tests participants were negatively affected by the lag generated by the terrain synthesis. We recommend that any further research makes it a priority that the application to be tested runs smoothly at all times. This is especially true in VR treatments and even more so when any sort of user avatar or visible tracked object is used. Having an avatar that is out of sync with the user even slightly can cause significant difficulty and discomfort. Finally, we learnt that relying purely on stereoscopic depth cues in VR is insufficient as a small number of users are unable to process these. Any future research should therefore integrate other cues, such as shadows and occlusion, to broaden the potential user base.

Appendix A

Paper Prototype Instructions

Paper Prototype Instructions

1. Welcome! These instructions will guide you through the program. Nod or perform the action described to dismiss the instruction
2. Press the Menu Button on the left-hand controller to bring up the menu.
3. To select an option move your hand in the corresponding direction. Once an option has been selected press the menu button again to exit the menu.
4. Select the back option to return to select mode.
5. To create a point, select a location using the laser and tap the right index trigger. To draw a line, keep the trigger held down.
6. To select an object/point, point to it using the laser and tap the right index trigger.
7. Widget instructions:
    a. Move your right hand up and down to adjust widget height
    b. Left and right movements will increase/decrease the area of effect of the widget
    c. Tilt your hand forward to increase the slope steepness and backwards to decrease steepness
    d. Point up and rotate around the y-axis to change the slope direction
8. To de-select an object/point double tap the right index trigger.
9. To delete a selected object/point tap the right hand trigger.
10. To move the entire terrain with your hand hold down the left index trigger.
11. To teleport to a selected object/point tap the left hand trigger.
12. To move around use the right-hand joystick.
13. To switch back to god view tap the left hand trigger again.
14. To create a landmark, select a location using the laser and tap the right index trigger.
15. Landmarks will be numbered in order of creation
16. To Undo an action tap the B button on the right controller.
17. Press the Menu Button on the left-hand controller to bring up the terrain-selection menu.
18. To paint an area, select a location using the laser and hold the right index trigger.
19. To change the paint brush width use the left hand joystick

Appendix B

Paper Prototype Feedback

Paper Prototype Feedback:

Participant 1:

- Wants ambidextrous controls
- Concerned about nod sensitivity for dismissing instructions
- Needs a controller layout tutorial.
- It needs to be specified which hand is used for menu control
- Wanted to use trigger to select menu options
- Instructions need to be more rail-roaded
- Inform user that you select a widget to manipulate it.
- There needs to be feedback as to what interaction mode the user is in
- How widgets are manipulated is confusing, particularly rotation
- While interacting with widgets the user often resets hand positions without 'letting go' of the widget
- Lack of on-screen hand was disconcerting
- There was a lack of two-handed interaction
- Explanation for how to change the size of the paintbrush in instructions was not clear.
- Was concerned about the menu button being overloaded
- Would prefer to switch back to menu rather then select mode when exiting a certain mode
- Switching to select mode is too common and annoying to need to do constantly
- It was not made clear which trigger was being referred to in some cases (index or hand)

Participant 2:

- Menu interaction needs to be clearly explained
- Need to make it clearer how to change the paintbrush size
- Wanted to use trigger to select menu options.
- Instructions need to be more rail-roaded
- It is counterintuitive that widget points cannot be moved with 3 -DOF
- The teleport point is not intuitively selected
- Need a controller layout tutorial.
- Instructions should be togglable
- Felt that flying would be a good way to move around in FPV
- Point manipulation needs to be togglable like point

Participant 3:

- Needed to peep to visually confirm location of Menu button
- Adjusting the steepness using the point control widget was confusing.
- Felt that the trigger should be held down while performing manipulations
- Tried to select the lines between points for direct manipulation
- Found that deleting a point was too complex an action
- Didn't find it intuitive to switch to FPV

Participant 4:

- Wanted sound effects to provide audio feedback for actions
- Wanted to select menu options using trigger button

- Wanted the mode the user was in to be clearly displayed
- Triggers were not well differentiated between
- Felt that it was conventional to move with the left-hand joystick
- Pointed at objects to select them
- Need a controller layout tutorial
- Was confused about difference between deselection and deletion
- Menu selection movements were often broken down into vertical and then horizontal movements
- Widget elements look to much like a traditional 3D movement axis
- Didn't realise deleting constraints would remove the terrain influences they were having
- Tried moving the paintbrush with the trigger held down
- The order in which paint brush instructions were given was bad
- Confused by overloaded menu buttons
- Tended to not change more than one variable at once
- Found the constraint widget confusing in general

Participant 5:

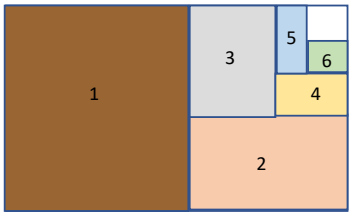
- Confused as to the location of triggers
- Wanted a controller diagram
- Got lost when she fell off the instruction set
- Wanted to use triggers to select menu options.

Appendix C

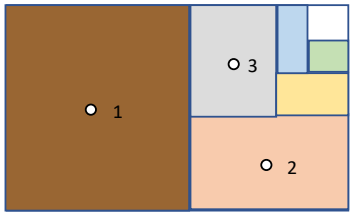
High Fidelity Instructions

Task 1: Scale and position the map such that it is the maximum size you can touch any point on the terrain comfortably.

Task 2: Paint the terrain using six different colours of choice. The left half should be one single colour. The right bottom quarter should be another colour. Of the remaining uncoloured portion, the left half should be one single colour and the right bottom quarter should be another colour. Of the remaining uncoloured portion, the left half should be one single colour and the right bottom quarter should be another colour.



Task 3: In three largest sections of the terrain create a single point constraint used to create hills.



Two of the hills in sections 2 and 3 should be the same height, the other in section 1 should be double this height.

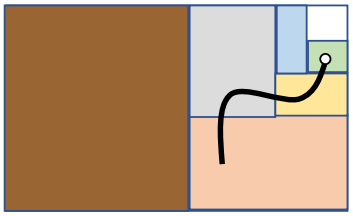
The tall hill in section 1 should be double the width of the low hill in section 2 and the low hill in section 3 should be half the width of the low hill in section 2.

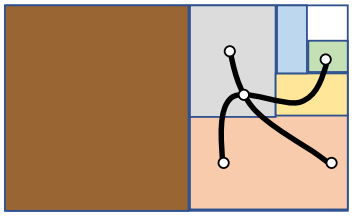
The tall hill in section 1 should have a steeply sloped North face (Top of the image above)

The low hill in section 2 should have a gently sloped West face (Left of the image above)

The low hill in section 3 should be approximately symmetrical in all directions.

Task 4: Create a ridge that passes through at least 4 coloured sections of the terrain. Adjust the endpoints such that they have uniform width but the start point is double the height of the end point





Task 5: Create a second ridge line such that it crosses the original ridge at any point. Raise the point of intersection such that it is the highest point on both ridges.

Task 6: Teleport onto the map anywhere in section 1. Navigate to the uncoloured section of the terrain using any tools available to you.

Appendix D

High Fidelity Feedback

High fidelity prototype Evaluation 1

1. Technology audit

Feature	Presence	Associated Problems
Operation of the users presence	Present	Awkward to see hand avatars in the sky while in 1st person POV
Lack of Haptic Feedback	Present	Would have like haptic feedback when successfully grabbing/ dropping objects
Interactive Techniques	Present	Ray casting worked well. Would have like a snap to effect for grabbing objects
Realistic Graphics	Mixed	Instructions were too close to user and too low resolution making them hard to read.

2. Heuristics rating and interpretation of problems encountered

	Heuristic	**Rating**	**Problems Encountered**
1	Natural engagement	S	No undo functionality, Could not grab all points of the terrain, point constraints were not exactly centred on top of hills, did not finding using the pencil to TP intuitive, Zoom affected FPV scale as well (also 12), Zooming seemed to affect ridges incorrectly (also 12)
2	Compatibility with the user's task and domain	-	No Problems
3	Natural expression of action	S	No way to measure scale, no way to move point constraints, No haptic feedback
4	Close coordination of action and representation	D	Walking in FPV was too slow (also 12)
5	Realistic feedback	S	Textures to not scale correctly(also 1,12), Could teleport onto constraints in FPV
6	Faithful viewpoints	-	No Problems
7	Navigation and orientation support	A	Reset tools orientation and distance was confusing
8	Clear entry and exit points	N/A	
9	Consistent departures	-	No Problems
10	Support for learning	S	Skipped instructions accidently
11	Clear turn-taking	N/A	
12	Sense of presence	D	Problems from 1,4,5

3. Classification of problems encountered with severity ratings and suggested design improvements

Feature	Problem Description	Problem rating	Design Change
Graphics	Textures did not scale correctly	Inconvenient	Use specialized terrain simulation software
Presence	Zooming caused some unrealistic effects	Annoying	Use specialized terrain simulation software
Interaction	Could not grab terrain at all points, No undo, Point constraints not centred, No way to measure scale, No way to move constraints, No haptic feedback when interacting	Severe	Make all points on the terrain grabbable. Add undo and delete features, add user measuring tool, vibrate controller on successful grab
Environmental features	Textures and ridges scaled incorrectly	Annoying	Use specialized terrain simulation software
Controls	Using the pencil to TP is unintuitive	Distracting	Specialized TP tool / Change tool visually depending on action selected
Hardware	N/A		

High fidelity prototype Evaluation 2

1. Technology audit

Feature	Presence	Associated Problems
Operation of the users presence	Present	No problems
Lack of Haptic Feedback	Present	No problems
Interactive Techniques	Present	No problems
Realistic Graphics	Present	Textures behave strangely but do not interfere with using the software

2. Heuristics rating and interpretation of problems encountered

	Heuristic	**Rating**	**Problems Encountered**
1	Natural engagement	S	Can't change paintbrush Shape (also 1), no delete/undo
2	Compatibility with the user's task and domain	A	See 1
3	Natural expression of action	-	No problems
4	Close coordination of action and representation	-	No problems
5	Realistic feedback	A	Tools positioning obscured ray-casting of other tools
6	Faithful viewpoints	-	No Problems
7	Navigation and orientation support	0	No problems
8	Clear entry and exit points	N/A	
9	Consistent departures	-	No Problems
10	Support for learning	S	Could not grab constraints due to positioning but hadn't seen scaling/moving terrain instructions yet
11	Clear turn-taking	N/A	
12	Sense of presence	-	No problems

3. Classification of problems encountered with severity ratings and suggested design improvements

Feature	Problem Description	Problem rating	Design Change
Graphics	No Issues	-	-
Presence	No Issues	-	-
Interaction	Can't change paintbrush shape, Ray-casting issues, No delete/Undo	Severe	Add undo and delete features, only detect ray-casting hits on the terrain
Environmental features	No Issues	-	-
Controls	Needed to scale/move terrain before relevant instruction appeared	Severe	Standardize menu controls, allow constraint visibility to be toggled, have a controller guide diagram accessible from main menu
Hardware	N/A		

High fidelity prototype Evaluation 3

1. Technology audit

Feature	Presence	Associated Problems
Operation of the users presence	Present	No problems
Lack of Haptic Feedback	Present	No problems
Interactive Techniques	Present	No problems
Realistic Graphics	Present	Textures behave strangely but do not interfere with using the software

2. Heuristics rating and interpretation of problems encountered

	Heuristic	**Rating**	**Problems Encountered**
1	Natural engagement	S	Instruction text is fuzzy, clips unexpectedly and moves badly, Instructions are too close to the user, making reading the edges hard, menu on right hand uses left hand button, grabbing paintbrush and adjusting area of effect on the same hand was too many actions together, No minimum scale of terrain
2	Compatibility with the user's task and domain	S	Not able to delete constraints, Not able to toggle visibility of constraints
3	Natural expression of action	-	No problems
4	Close coordination of action and representation	D	Trigger for menu selection not consistent
5	Realistic feedback	A	Rotating circular menus left is sometimes unresponsive
6	Faithful viewpoints	-	No Problems
7	Navigation and orientation support	S	First person rotation with thumb stick can cause nausea (see also 1)
8	Clear entry and exit points	N/A	
9	Consistent departures	-	No Problems
10	Support for learning	S	Instructions skip too soon, Hard to remember all the controls
11	Clear turn-taking	N/A	
12	Sense of presence	D	Could see user avatar in the sky in FPV

3. Classification of problems encountered with severity ratings and suggested design improvements

Feature	Problem Description	Problem rating	Design Change
Graphics	Textures did not scale correctly, Instruction text is fuzzy	Annoying	Use specialized terrain simulation software, use higher resolution text
Presence	Could see god hands in FPV	Irritation	Disable user avatar in FPV
Interaction	No minimum scale for terrain, can't delete constraints, Instructions skip too easily	Severe	Set minimum terrain size/scale, add undo and delete features, use a manual next instruction tool
Environmental features	Textures scaled incorrectly, instructions clip on terrain	Annoying	Use specialized terrain simulation software
Controls	Menus not consistent, can't toggle constraint visibility, hard to remember all the controls	Annoying	Standardize menu controls, allow constraint visibility to be toggled, have a controller guide diagram accessible from main menu
Hardware	N/A		

High fidelity prototype Evaluation 4

1. Technology audit

Feature	Presence	Associated Problems
Operation of the users presence	Present	Hand avatars were present, these were sufficient and necessary
Lack of Haptic Feedback	Present	Passive haptics present from controllers and button presses. No further haptic feedback was needed
Interactive Techniques	Present	Mix of grab and ray cast selection. Appropriate to type of actions. Some lag occurred during interaction which will cause simulator sickness
Realistic Graphics	Mixed	Depth perception could be improved with better depth cues. Eg. Shadows, fog to give better shape perspective

2. Heuristics rating and interpretation of problems encountered

	Heuristic	**Rating**	**Problems Encountered**
1	Natural engagement	S	User instructions were too close to user making them hard to read. No animation when menu wheel spins making it hard to tell which direction it moves in. Scaling of map is not 1-to-1 with hand movements (see also 4). Initial terrain size is too large. Walking is slow and jerky which may lead to sim sickness (see also 7, 12). No sense of scale of the terrain or depth cues (see also 12)
2	Compatibility with the user's task and domain	-	No Problems
3	Natural expression of action	S	Can accidently draw ridge instead of creating point constraint. Rotating through the circle menu sometimes not reactive. Changing the size of the terrain type constraint painter too slow. No undo functionality. Texture writes are too slow. Tended to drop pencil when creating a constraint. Constraint sometimes lagged and continued to move after released. Using the pencil to select teleport location seemed unintuitive.
4	Close coordination of action and representation	A	Texture painting does not match circle indicator. Not all points on the Terrain are grabbable.
5	Realistic feedback	D	Display of area of effect indicator on the paintbrush tool is incorrect when painting small areas.
6	Faithful viewpoints	A	Constraint widgets are too densely packed when the terrain is scaled down
7	Navigation and orientation support	A	Location markers do not scale with the terrain
8	Clear entry and exit points	N/A	
9	Consistent departures	-	No Problems
10	Support for learning	I	Not clearly indicated that the circle menus rotate continuously
11	Clear turn-taking	N/A	
12	Sense of presence	S	Rotating while in the first person POV is too fast

3. Classification of problems encountered with severity ratings and suggested design improvements

Feature	Problem Description	Problem rating	Design Change
Graphics	Texture rendering delays and terrain update delays	Distracting	Use better optimized terrain simulation software
Presence	Simulator sickness from lag and rotating in FPV	Annoying	Use better optimized terrain simulation software, Do not let user rotate smoothly in FPV
Interaction	Could not grab terrain at all points, Dropped the pencil accidently. Accidently created wrong constraint type. No undo.	Severe	Make all points on the terrain grabbable. Have a setting to make the tool grab a toggle rather than a persistent grip. Add undo and delete features
Environmental features	Terrain did not scale in a 1-to-1 manner. No sense of scale or depth cues in terrain. Textures painted on the terrain did not match the paintbrush tool preview.	Annoying	Adjust scaling to be 1-to-1. Paint textures correctly within preview.
Controls	Text instructions too close to user. Circle menu rotation sometimes unreactive and not animated.	Annoying	Place instructions on a static surface further away from the user. Animate circle menu rotation
Hardware	N/A		

High fidelity prototype Evaluation 5

1. Technology audit

Feature	Presence	Associated Problems
Operation of the users presence	Present	No Problems
Lack of Haptic Feedback	Present	Hard to evaluate whether hand was in the correct area to grab an object
Interactive Techniques	Present	No Problems
Realistic Graphics	No	Hard to evaluate distances and scales

2. Heuristics rating and interpretation of problems encountered

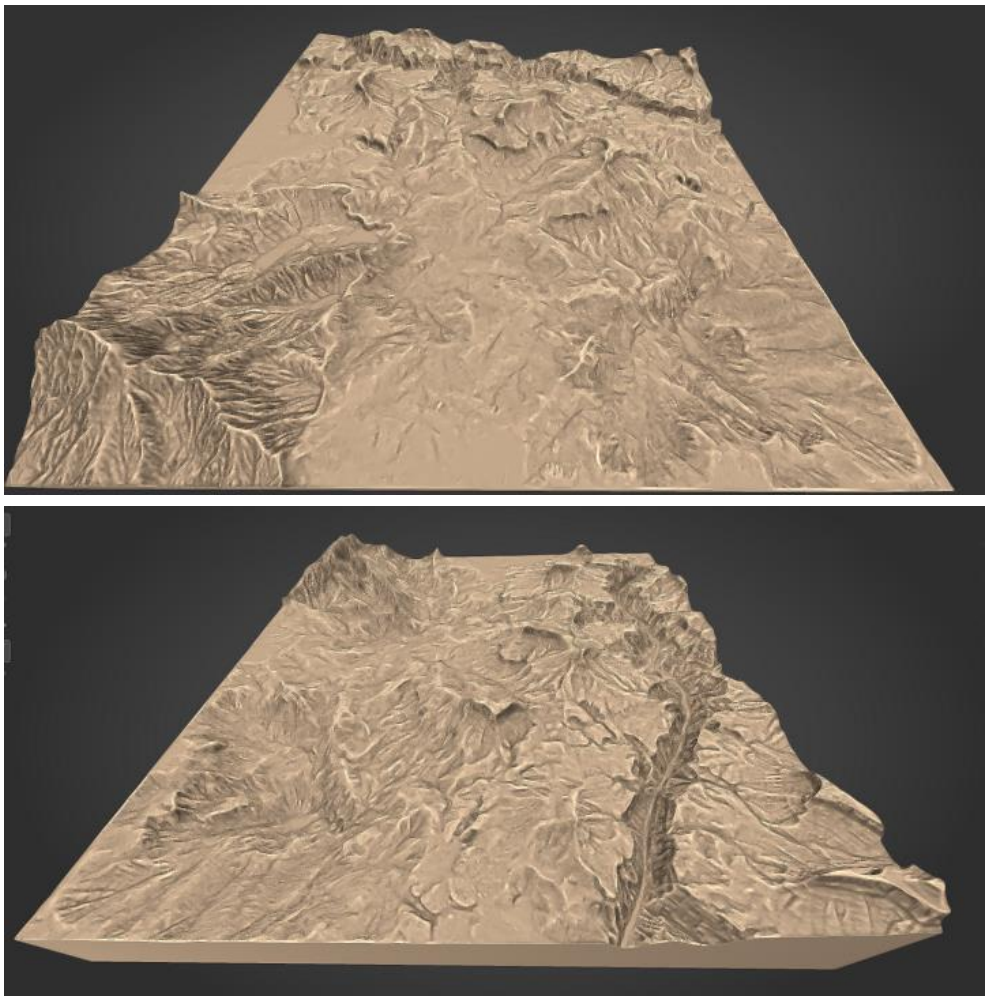
	Heuristic	**Rating**	**Problems Encountered**
1	Natural engagement	A	Instruction text was too close to the user and attached to camera (see also 5,10), Instructions did not appear properly in FPV (see also 12)
2	Compatibility with the user's task and domain	A	No Undo (see also 10)
3	Natural expression of action	A	Colour menu on the opposite hand to the button that opens it is unintuitive, painted the terrain accidently when trying to select a colour (see also 2), felt unnatural to select grey option in tool menu, ridge line sticks to other widgets, cannot grab constraint and terrain at the same time (see also 7).
4	Close coordination of action and representation	D	Created ridge instead of point constraint (see also 5), lag when holding constraints (see also 6,7), Blue menu does not disappear when an option is selected (see also 12).
5	Realistic feedback	A	Texture does not scale with terrain (see also 1,2), texture filling area does not match paintbrush preview (see also 3), can accidently incorrectly grab ground/constraint when the two are in close proximity (see also 2,4), cannot identify which areas of the terrain are traversable in FPV (see also 10)
6	Faithful viewpoints	D	Constraints do not disappear in FPV
7	Navigation and orientation support	D	Cannot rotate the terrain using both hands, cannot grab all points on the terrain.
8	Clear entry and exit points	N/A	
9	Consistent departures	D	Was unintuitive to select options from blue menu using A button
10	Support for learning	A	See problems from 1, 2, 5
11	Clear turn-taking	N/A	
12	Sense of presence	D	Problems from 1,4

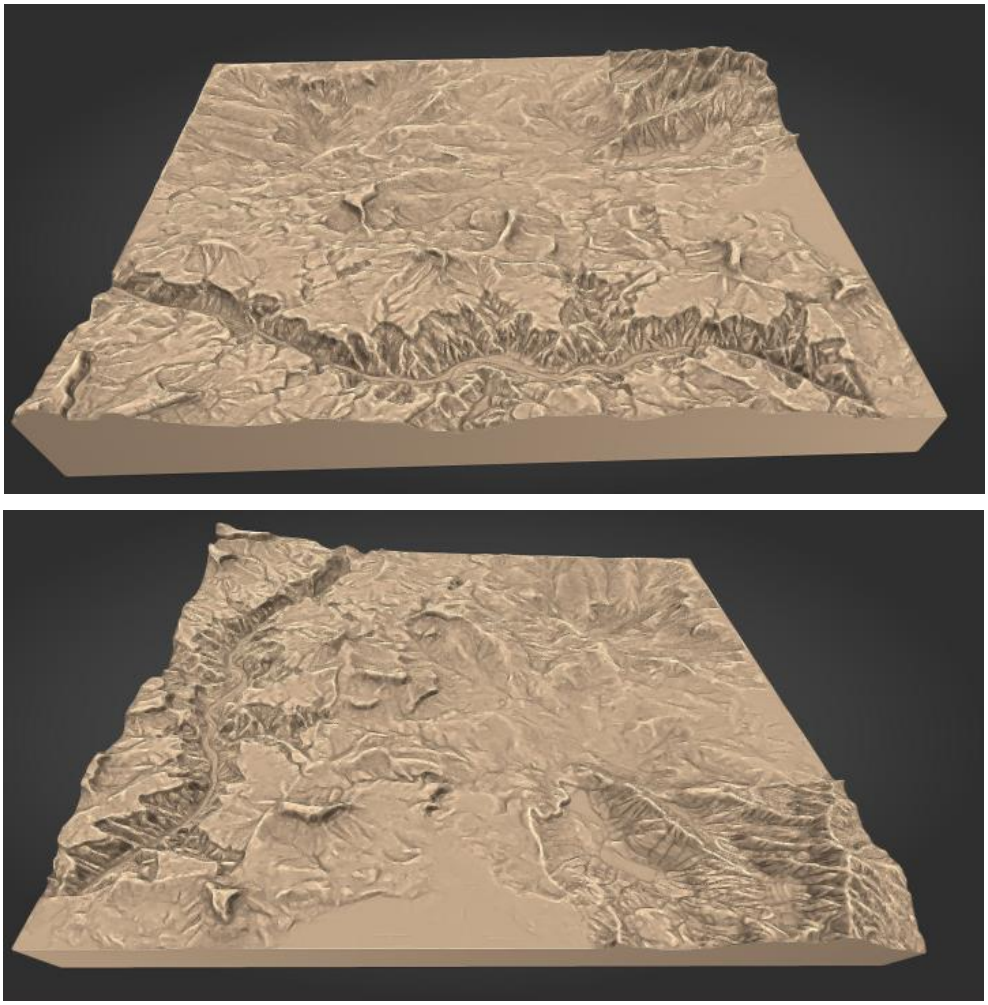
3. Classification of problems encountered with severity ratings and suggested design improvements

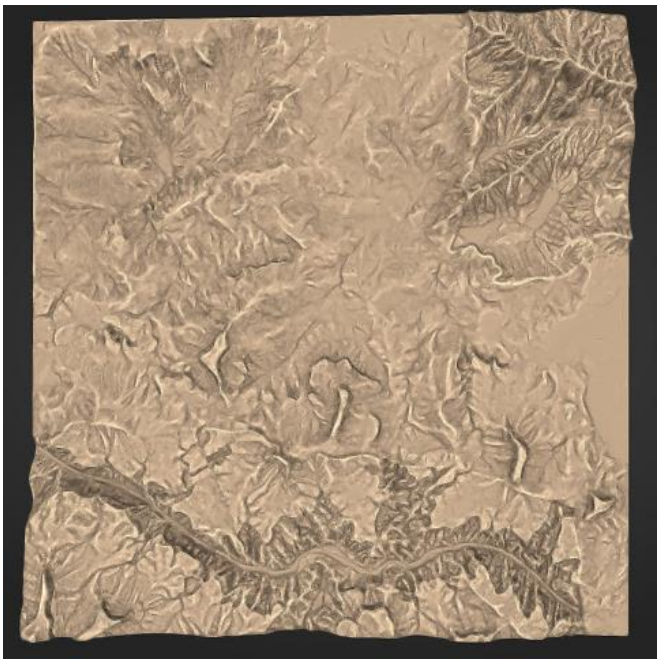
Feature	Problem Description	Problem rating	Design Change
Graphics	Textures did not scale correctly	Inconvenient	Use specialized terrain simulation software
Presence	Some interface elements did not behave in an expected manner	Distracting	Improve how instructions are shown and make menus consistent
Interaction	Could not grab terrain at all points, No undo, Ridge lines stuck to non-terrain objects, created wrong constraint types, accidently grabbed the wrong objects, cannot rotate the terrain with both hands	Severe	Make all points on the terrain grabbable. Add undo and delete features, offset widgets further from terrain, allow scaling feature to be turned off
Environmental features	Textures scaled incorrectly, No indication of what terrain is traversable in FPV	Annoying	Use specialized terrain simulation software, Use a teleport mechanic to move in FPV
Controls	Colour Menu on opposite hand to related button, Painted terrain when trying to select menu option, Blue menu persists after option selected, Blue menu was inconsistent with other menus, unintuitive to be selecting a grey coloured menu option	Annoying	Match menu buttons to hand they appear on, standardize menu controls, disable interaction with terrain while menus are open
Hardware	N/A		

Appendix E

Accuracy Task Images (Variation 1)

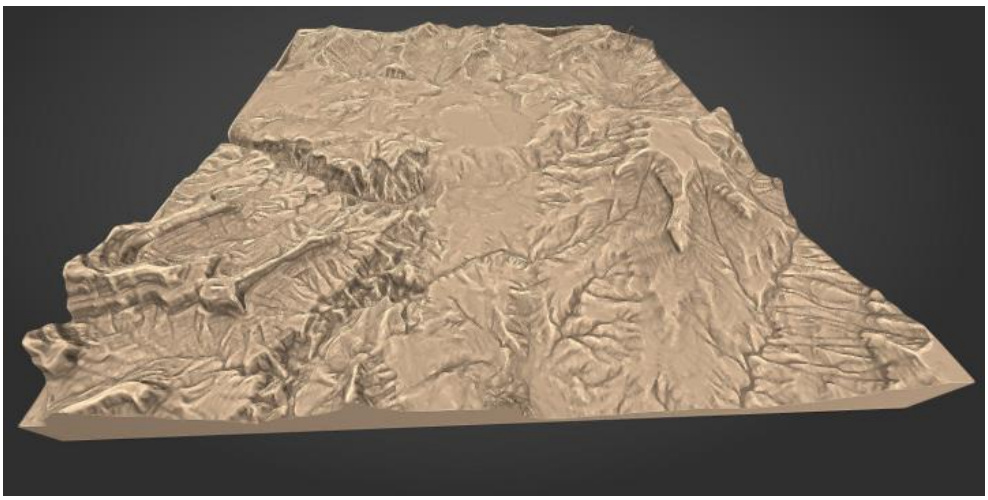


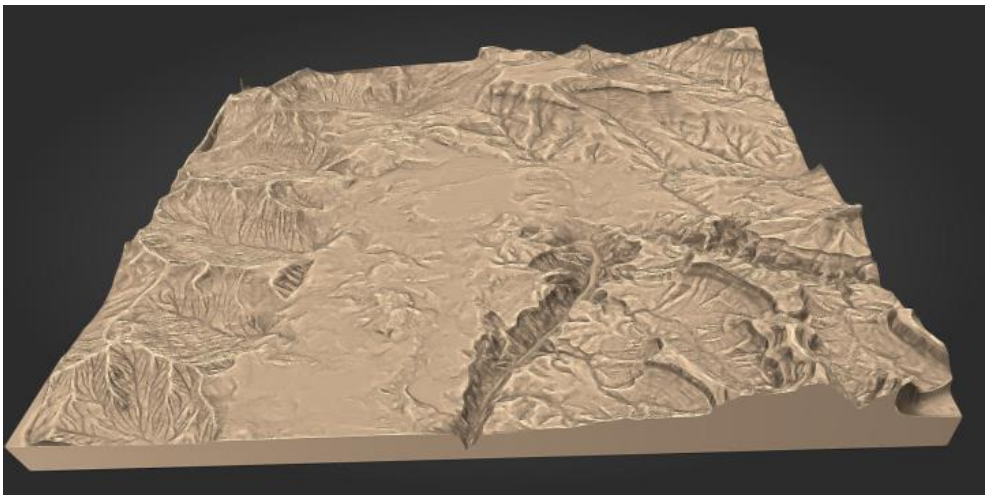


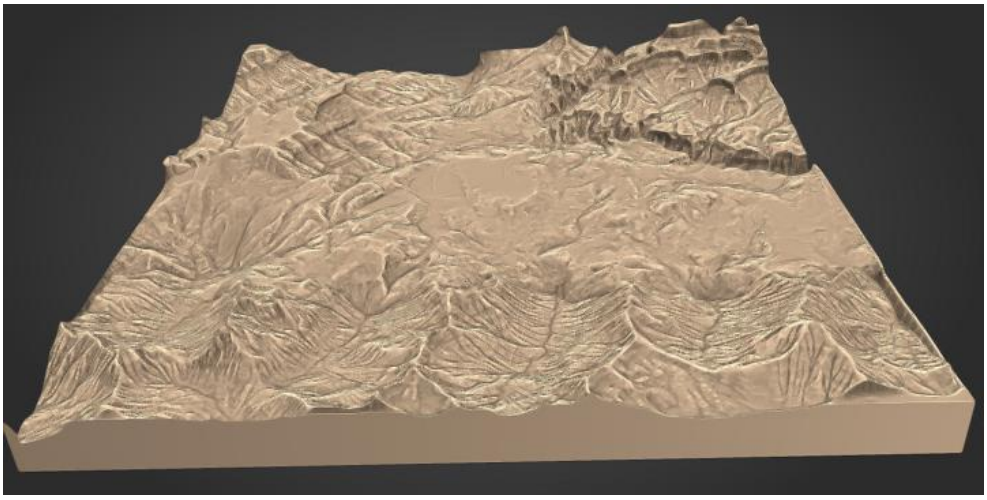


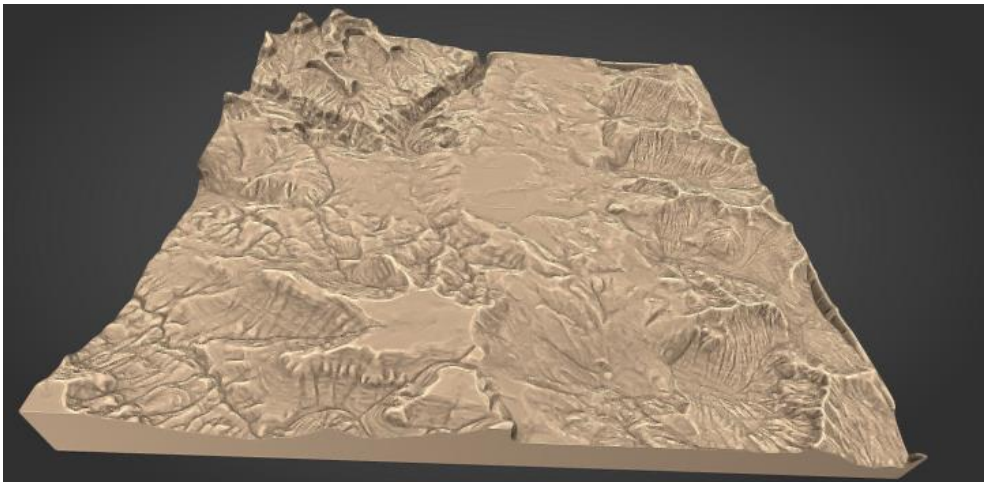
Appendix F

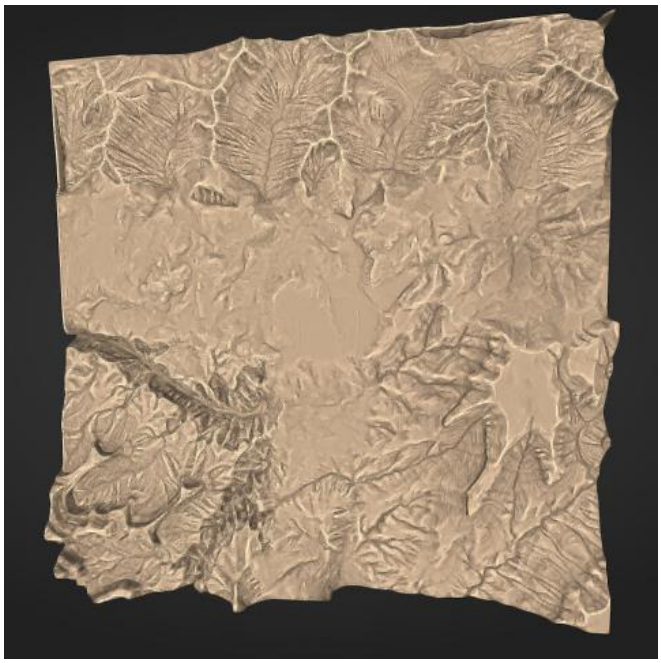
Accuracy Task Images (Variation 2)







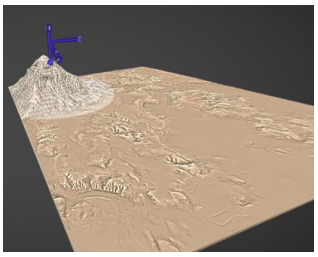




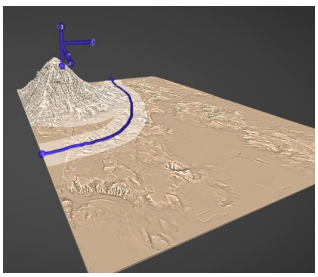
Appendix G

Speed Task Instructions (Variation 1)

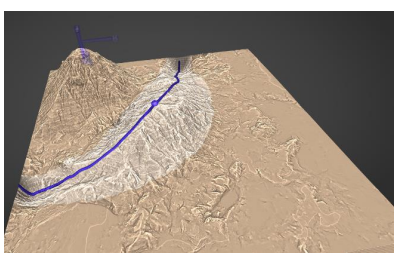
1. **Create a point constraint in the far-left corner of the terrain. Use this constraint to create a hill. Adjust the slope and tilt angle so that the hill slope is steeper towards the centre of the terrain.**



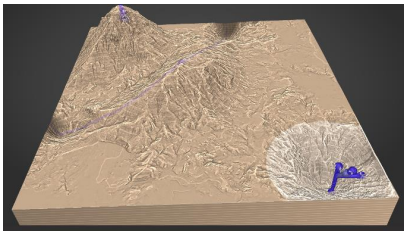
2. **Create a ridge constraint that encircles the hill towards the centre of the terrain but ends at the edges of the terrain.**



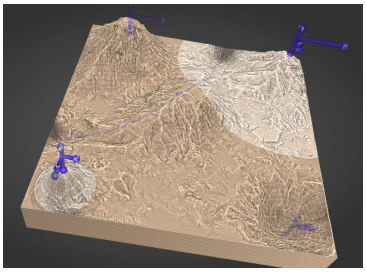
3. **Create an extra point along the middle of the ridge constraint.**
4. **Use the ridge constraints to make it dip at either end but rise in the middle, the resulting valleys should also get narrower near the terrain edges.**



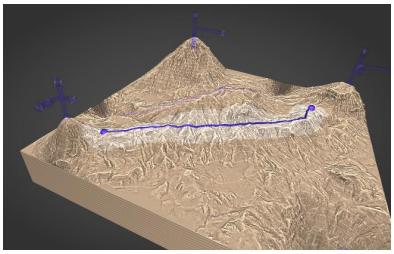
5. **Create a second point constraint in the opposite corner of the terrain to the hill. Use this constraint to make a hollow in the landscape. Adjust the constraint so that the slope is steeper towards the corner of the terrain.**



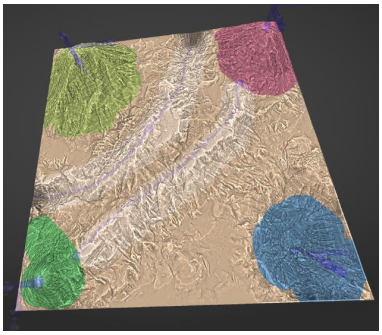
6. **Create two more point constraints in the remaining corners of the terrain.**
7. **These constraints should be used to create hills of the same height. However, one should be much wider than the hill you created in the beginning and one should be much narrower the hill you created in the beginning.**



8. **Create a ridge constraint from between the bases of these two hills. Adjust it so the ridge is approximately half the height of the hills for its entire length.**



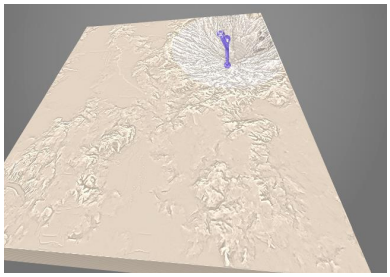
9. **Adjust the ridge so that the radius on the side of the hollow is much larger than the opposite side.**
10. **Set the terrain type of each hill to be unique. Set the terrain type of the hollow to be any blue terrain type. (Note: Ensure all slopes of a hill are set to the same terrain type, do not only paint one face of the hill)**
11. **Use the 'freeze' terrain type to freeze both ridges in place**



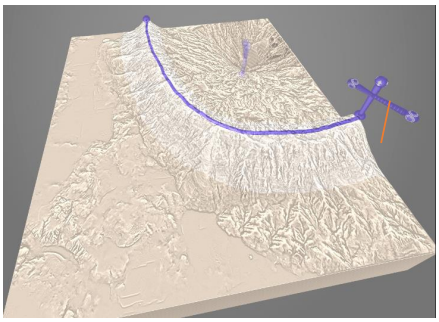
Appendix H

Speed Task Instructions (Variation 2)

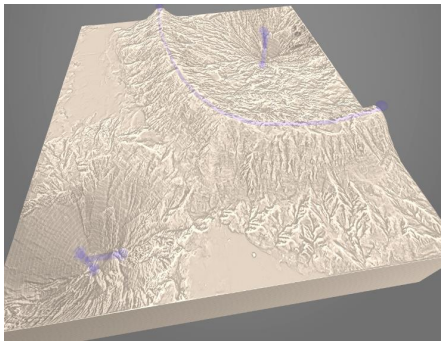
1. Create a point constraint in the far-right corner of the terrain. Use this constraint to create a hollow in the landscape. Adjust the slope and tilt angle so that the hill slope is steeper towards the centre of the terrain.



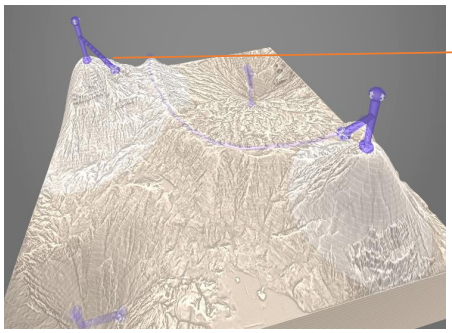
2. Create a ridge constraint that encircles the hollow towards the centre of the terrain but ends at the edges of the terrain. Raise the ridge so that it is a constant height.
3. Adjust the ridge radius so it is wider towards the centre of the terrain.



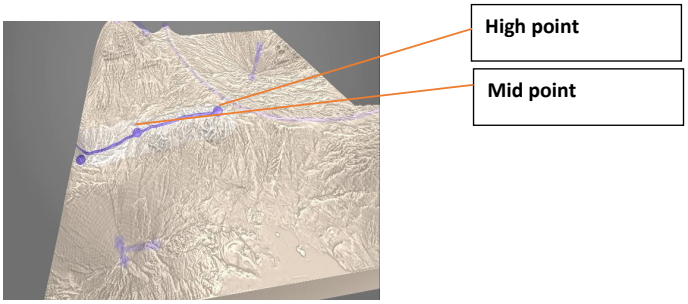
4. Create a second point constraint in the opposite corner of the terrain to the hollow. Use this constraint to make a hollow in the landscape. Adjust this constraint to be much deeper than the first hollow.



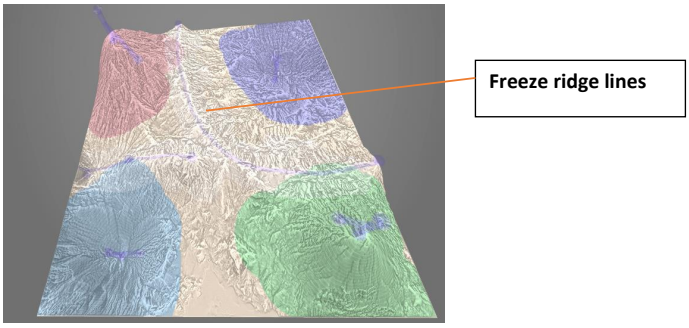
5. Create two more point constraints in the remaining corners of the terrain.
6. These constraints should be used to create hills of the same height. However, one should be much wider than the other.
7. Adjust the slope and tilt angle of both hills so that their slopes are steeper towards the centre of the terrain.



8. Create a ridge constraint from the centre of the terrain to any edge of the terrain
9. Create an extra point along the middle of the ridge constraint you have just created.
10. Adjust the ridge so that it is above the base terrain at the centre and below it at the terrain edge. The point you added should be at the base terrain height.



11. Set the terrain type of each hollow or hill to be unique. The terrain type of the hollows must be any blue terrain type. (Note: Ensure all slopes of a hill or hollow are set to the same terrain type, do not only paint one face of the slope)
12. Use the 'freeze' terrain type to freeze both ridges in place

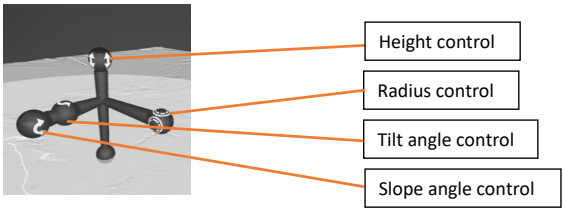


Appendix I

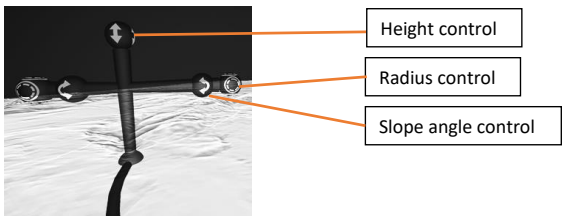
User instructions for final interface

1. Welcome to Terraria, a terrain editing program.
2. To create a point constraint, select the point icon  on the left menu and click anywhere on the terrain.
3. To create a ridge constraint, select the ridge icon  on the left menu and draw a line anywhere on the terrain.
4. To add a new constraint point along a ridge, click anywhere along an existing ridge while the ridge icon is selected.
5. To select a constraint, click on the constraint or its listing on the right side of the screen

6. Change the height of a point constraint by dragging the component with straight vertical arrows up or down using your mouse.
7. Change the radius of a point constraint by dragging the component with a dotted circle closer or further from the constraint centre.
8. Change the slope angle of a point constraint by dragging the component with curved vertical arrows up or down.
9. Change the tilt angle of a point constraint by dragging the component with curved horizontal arrows up or down.



10. Change the height of a ridge constraint by dragging the component with straight vertical arrows up or down using your mouse.
11. Change the radius of a ridge constraint by dragging the component with a dotted circle closer or further from the constraint centre.
12. Change the slope angle of a ridge constraint by dragging the component with curved vertical arrows up or down.



13. To delete a constraint right click it on the list on the right of the screen and select delete
14. To deselect all constraints, click empty space on the constraint list on the right of the screen
15. To undo or redo actions use the undo and redo buttons on the left of the screen
16. To create a terrain type constraint, select the paintbrush icon  on the left menu then select a terrain type from the scrolling list on the bottom right of the screen. Then use the mouse to drag the paintbrush tool over the terrain.
17. To adjust the paintbrush, use the slider in the bottom right of the screen
18. To move the terrain right-click on it with mouse and drag

Appendix J

Rubric for Accuracy assessment

Subject #	Presence of major features (5)	Height/Depth of features (5)	Volume of features (5)	Slope angles of features with height/depth (5)	Positioning of features (5)	Level of overall detail (5)	Total (30)
1							
2							
3							
4							
5							
6							
7							
8							
9							
10							
11							

Subject #	Presence of major features (5)	Height/Depth of features (5)	Volume of features (5)	Slope angles of features with height/depth (5)	Positioning of features (5)	Level of overall detail (5)	Total (30)
12							
13							
14							
15							
16							
17							
18							
19							
20							
21							

Appendix K

Full results of Accuracy tests

Variation 1

Subject	Presence of major features	Height/Depth of features	Volume of features	Slope angles of features with height/depth	Positioning of features	Level of overall detail	Total
1	3	3	3	3	3	2	17
2	3	3	2	2	2	2	14
3	2	2	2	2	2	2	11
4	1	1	1	2	2	1	8
5	2	2	1	1	1	1	8
6	3	2	2	2	3	2	15
7	3	2	1	2	2	1	11
8	3	2	1	2	2	1	11
9	2	2	1	1	1	1	8
10	3	1	1	2	2	1	10
11	2	2	1	2	2	2	11
12	3	1	1	1	2	1	9
13	3	2	2	2	3	3	15
14	3	1	1	1	3	2	11
15	3	2	1	1	2	2	11
16	3	1	1	1	3	1	10
17	2	2	2	2	1	1	10
18	3	2	1	1	3	2	12
19	2	1	2	1	2	2	10
20	3	1	2	1	3	2	12
21	3	2	2	3	3	2	15

Variation 2

Subject	Presence of major features	Height/Depth of features	Volume of features	Slope angles of features with height/depth	Positioning of features	Level of overall detail	Total
1	5	4	3	2	2	3	21
2	5	3	2	2	2	3	17
3	4	1	2	1	1	2	11
4	4	2	2	2	2	2	13
5	3	1	1	1	1	3	10
6	3	3	1	2	2	2	14
7	4	2	3	2	2	4	18
8	2	1	1	2	1	1	8
9	4	2	2	1	1	2	11
10	3	2	2	2	2	2	12
11	1	1	1	1	1	1	6
12	1	1	1	1	1	1	6
13	5	3	3	2	2	3	18
14	3	3	2	2	2	2	13
15	2	3	2	2	2	1	11
16	1	1	1	1	1	1	6
17	2	1	1	2	2	1	8
18	1	1	1	1	1	1	6
19	2	1	1	1	1	1	7
20	3	1	1	1	2	2	9
21	5	3	2	2	3	2	18

Appendix L

Full results of Speed tests

Variation 1

Subject	Pencil	Paintbrush	Widgets	Point	Ridge	Painting	Total
1				149,837	133,724	124,454	420,321
2	30,43	89,55	132,453				638,004
3				334,163	185,259	202,19	730,419
4	52,007	53,02	55,23				562,181
5				210,593	264,13	128,776	611,829
6	46,284	38,874	82,899				512,084
7				187,934	112,83	89,42	390,617
8	48,969	65,939	149,547				723,174
9				0	124,156	67,485	320,316
10	49,827	111,132	121,499				506,658
11				0	359,548	154,821	771,66
12	44,345	64,922	81,378				458,724
13				0	34,839	75,244	322,434
14	49,591	41,27	94,824				521,779
15				110,14	105,259	83,219	299,582
16	38,828	57,315	89,724				540,933
17				122,482	158,463	100,164	381,158
18	61,354	87,77	151,473				778,875
19				128,737	91,648	84,971	305,435
20	55,158	105,631	74,327				588,828
21				165,235	186,829	67,7	462,583

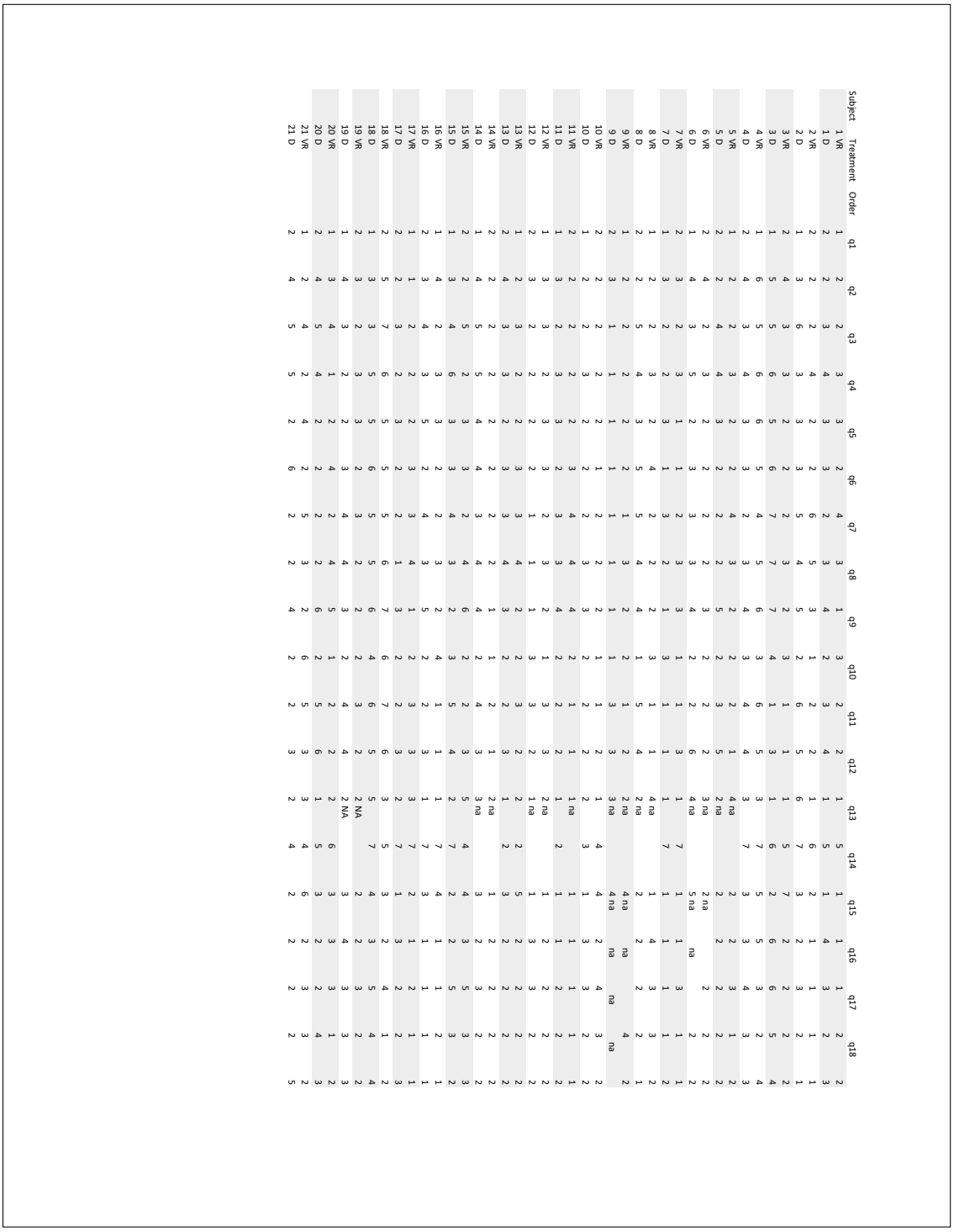
Variation 2 516,5521

Subject	Pencil	Paintbrush	Widgets	Point	Ridge	Painting	Total
1	56,203	85,978	149,974				576,327
2				141,11	196,193	144,673	495,522
3	39,727	81,96	159,751				783,25
4				127,243	160,416	77,669	386,82
5	46,187	49,63	165,096				595,844
6				0	316,23	109,559	554,438
7	24,66	31,884	91,238				383,972
8				0	47,32	124,004	662,692
9	41,996	49,385	129,448				541,699
10				0	179,957	116,126	455,86
11	47,088	66,87	94,455				596,291
12				85,35	97,02	74,039	256,464
13	64,427	47,413	127,063				589,206
14				114,429	111,655	94,181	334,574
15	36,54	44,668	85,396				387,747
16				329,566	284,794	224,42	865,249
17	39,11	85,253	95,342				562,155
18				153,267	132,244	139,089	429,361
19	45,535	94,114	85,372				489,855
20				201,967	194,955	215,912	626,227
21	43,196	82,63	121,847				715,288

537,5639

Appendix M

Full results of Usability tests



Subject	Treatment	Order	q1	q2	q3	q4	q5	q6	q7	q8	q9	q10	q11	q12	q13	q14	q15	q16	q17	q18
1	VR	1	2	2	3	3	3	2	3	1	3	3	2	1	5	1	1	3	2	2
1	D	2	2	3	4	3	3	2	3	4	2	2	4	1	5	1	4	3	2	3
2	VR	2	2	2	4	2	2	2	5	2	1	2	2	1	6	2	1	1	1	1
2	D	1	3	2	6	3	3	2	6	2	2	2	2	1	7	3	2	1	2	1
3	VR	2	4	3	3	2	2	2	7	2	2	1	2	1	5	2	2	2	2	2
3	D	1	5	5	6	5	6	2	7	2	4	1	2	1	6	2	2	2	2	4
4	VR	1	6	5	6	6	5	4	5	6	3	6	5	3	7	5	5	3	2	4
4	D	2	4	3	4	3	3	2	3	4	3	4	4	3	na	3	3	4	3	3
5	VR	1	2	2	3	2	2	4	3	2	2	3	5	4 na	na	2	2	3	1	2
5	D	2	2	4	3	3	2	4	3	5	2	3	5	2 na	na	2	2	2	2	2
6	VR	2	2	4	2	3	2	2	2	3	2	2	2	3 na	na	2 na	2	2	2	2
6	D	1	4	3	5	2	2	3	3	4	2	2	6	4 na	na	5 na	na	2	2	2
7	VR	2	3	2	3	1	1	2	2	3	1	1	3	1	7	1	1	1	1	1
7	D	1	3	2	5	3	3	3	2	1	3	1	1	1 na	7	1	1	1	1	2
8	VR	1	2	2	3	2	2	2	2	2	2	3	1	4 na	na	2	2	2	2	2
8	D	2	2	5	4	3	3	2	4	4	1	3	4	2 na	na	2	2	2	2	1
9	VR	1	2	2	2	2	2	1	3	2	2	1	2	2 na	4 na	4 na	na	4	4	2
9	D	2	3	1	2	2	2	1	3	2	2	1	2	3 na	na	4 na	na	na	na	
10	VR	2	2	2	2	2	2	2	2	2	1	2	2	1	4	4	2	4	3	2
10	D	1	2	2	3	2	2	2	3	3	2	2	3	2	3	1	3	3	2	2
11	VR	2	2	2	2	2	3	2	4	4	2	1	2	1 na	4	1	1	1	1	1
11	D	1	3	3	2	3	2	3	3	4	2	2	2	1	2	1	1	2	1	2
12	VR	1	3	3	3	3	3	2	3	2	1	3	3	2 na	na	2	2	2	2	2
12	D	2	3	2	2	2	2	2	3	1	3	3	2	1 na	2	1	1	3	2	2
13	VR	1	2	3	2	2	3	2	4	2	2	2	2	2	2	3	2	2	2	2
13	D	2	4	3	3	2	3	3	4	3	2	2	3	1	2	3	2	2	2	2
14	VR	2	2	2	2	2	2	2	2	1	2	2	1	2 na		3 na	2	3	2	2
14	D	1	4	5	5	4	4	3	4	4	2	4	3	3 na			2	2	2	2
15	VR	2	2	5	2	3	3	2	4	6	2	2	3	5	4	4	3	3	3	3
15	D	1	3	4	6	3	3	4	3	2	3	5	4	2	7	4	2	5	3	2
16	VR	1	4	4	3	2	2	2	3	2	4	2	1	1	7	4	1	1	1	1
16	D	2	3	4	3	3	2	4	3	5	2	2	3	1	7	3	1	2	1	1
17	VR	1	1	2	3	2	3	3	4	1	2	3	3	3	7	2	2	2	1	1
17	D	2	2	3	2	2	2	2	4	3	2	2	3	2	7	1	1	2	1	1
18	VR	2	5	7	6	5	5	5	6	7	4	7	6	3	5	3	2	4	1	2
18	D	1	3	3	5	5	6	5	5	6	4	6	5	2 NA	7	4	2	5	3	2
19	VR	2	3	2	3	2	2	4	2	2	2	3	2	2 NA		2	2	3	2	2
19	D	1	4	3	2	3	2	4	4	4	2	4	4	2 NA		3	2	3	2	3
20	VR	1	3	4	1	2	2	2	4	5	1	2	2	2	6	3	3	3	1	2
20	D	2	4	5	4	2	2	2	2	6	2	2	6	1	5	4	2	2	3	3
21	VR	1	2	4	2	4	2	5	3	2	2	5	3	3	4	6	2	2	2	4
21	D	2	4	5	5	2	6	2	2	4	2	2	3	2	4	2	2	2	2	5

Bibliography

[1] Chadia Abras, Diane Maloney-Krichmar, and Jenny Preece. "User-centered design". In: *Bainbridge, W. Encyclopedia of Human-Computer Interaction. Thousand Oaks: Sage Publications* 37.4 (2004), pp. 445–456.

[2] Mike Alger. *Visual Design Methods for Virtual Reality*. http://aperturesciencellc.com/vr/VisualDesignMethodsforVR_MikeAlger.pdf. [Online; accessed 22-April-2019]. 2015.

[3] Nguyen Hoang Anh, Alexei Sourin, and Parimal Aswani. "Physically based hydraulic erosion simulation on graphics processing unit". In: *Proceedings of the 5th international conference on Computer graphics and interactive techniques in Australia and Southeast Asia*. ACM, 2007, pp. 257–264. DOI: 10.1145/1321261.1321308.

[4] Dennis R Ankrum. In: *Occupational Health and Safety* 68.7 (1999), pp. 64–70.

[5] Ravin Balakrishnan and Gordon Kurtenbach. "Exploring bimanual camera control and object manipulation in 3D graphics interfaces". In: *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 1999, pp. 56–62. DOI: 10.1145/302979.302991.

[6] Frank Biocca. "Will Simulation Sickness Slow Down the Diffusion of Virtual Environment Technology?" In: *Presence: Teleoperators and Virtual Environments* 1.3 (1992), pp. 334–343. DOI: 10.1162/pres.1992.1.3.334.

[7] Chuck Blanchard, Scott Burgess, Young Harvill, Jaron Lanier, Ann Lasko, Mark Oberman, and Mike Teitel. "Reality built for two: a virtual reality tool". In: *ACM SIGGRAPH Computer Graphics*. Vol. 24. 2. ACM, 1990, pp. 35–36. DOI: 10.1145/91394.91409.

[8] Doug A. Bowman, Joseph L. Gabbard, and Deborah Hix. "A Survey of Usability Evaluation in Virtual Environments: Classification and Comparison of Methods". In: *Presence: Teleoperators & Virtual Environments* 11.4 (Aug. 2002), pp. 404–424. DOI: 10.1162/105474602760204309.

[9] Doug A. Bowman, David Koller, and Larry F. Hodges. "Travel in Immersive Virtual Environments : An Evaluation of Viewpoint Motion Control Techniques Georgia Institute of Technology". In: *Proceedings of the 1997 Virtual Reality Annual International Symposium* (1997), pp. 45–52. DOI: 10.1109/VRAIS.1997.583043.

[10] Doug A Bowman, Ernst Kruijff, Joseph J LaViola Jr, and Ivan Poupyrev. "An Introduction to 3-D User Interface Design". In: *Presence: Teleoperators and Virtual Environments* 10.1 (2001), pp. 96–108. DOI: 10.1162/105474601750182342.

[11] Doug A. Bowman, Jean Wineman, Larry F. Hodges, and Don Allison. "Designing animal habitats within an immersive VE". In: *IEEE Computer Graphics and Applications* 18.5 (1998), pp. 9–13. DOI: 10.1109/38.708555.

[12] John Brosz, Faramarz F Samavati, and Mario Costa Sousa. "Terrain synthesis by-example". In: *Advances in Computer Graphics and Computer Vision*. Vol. 11. 1. Springer, Berlin, Heidelberg, 2007, pp. 58–77. DOI: 10.1007/978-3-540-75274-5_4.

[13] Eric Bruneton and Fabrice Neyret. "Real-time rendering and editing of vector-based terrains". In: *Computer Graphics Forum*. Vol. 27. 2. Blackwell Publishing Ltd, Apr. 2008, pp. 311–320. DOI: 10.1111/j.1467-8659.2008.01128.x.

[14] William Buxton and Brad Myers. "A study in two-handed input". In: *Proceedings of the SIGCHI conference on Human factors in computing systems*. Vol. 17. 4. ACM Press, 1986, pp. 321–326. DOI: 10.1145/22627.22390.

[15] Brookshire D. Conner, Scott S. Snibbe, Kenneth P. Herndon, Daniel C. Robbins, Robert C. Zeleznik, and Andries van Dam. "Three-dimensional widgets". In: *Proceedings of the 1992 symposium on Interactive 3D graphics*. ACM, 1992, pp. 183–188. DOI: 10.1145/147156.147199.

[16] Leandro Cruz, Luiz Velho, Eric Galin, Adrien Peytavie, and Eric Guérin. "Patch-based Terrain Synthesis". In: *International Conference on Computer Graphics Theory and Applications*. Proceedings of the 10th International Conference on Computer Graphics Theory and Applications, Mar. 2015, 6 pages. DOI: 10.5220/0005360201890194.

[17] Andries van Dam. "Post-WIMP user interfaces". In: *Communications of the ACM* 40.2 (1997), pp. 63–67. DOI: 10.1145/253671.253708.

[18] Rudy P. Darken and John L. Sibert. "A toolset for navigation in virtual environments". In: *ACM symposium on User interface software and technology* (1993), pp. 157–165. DOI: 10.1145/168642.168658.

[19] Bruno R. De Araùjo, Géry Casiez, and Joaquim A. Jorge. "Mockup Builder: Direct 3D Modeling on and Above the Surface in a Continuous Interaction Space". In: *Proceedings of Graphics Interface 2012*. GI '12. Canadian Information Processing Society, 2012, pp. 173–180.

[20] Thierry Duval, Thi Thuong Huyen Nguyen, Cédric Fleury, Alain Chauffaut, Georges Dumont, and Valérie Gouranton. "Improving awareness for 3D virtual collaboration by embedding the features of users' physical environments and by augmenting interaction tools with cognitive feedback cues". In: *Journal on Multimodal User Interfaces* 8.2 (2014), pp. 187–197. DOI: 10.1007/s12193-013-0134-z.

[21] David S Ebert, F Kenton Musgrave, Darwyn Peachy, Ken Perlin, and Steven Worley. *Texturing and Modeling: A Procedural Approach*. Morgan Kaufmann, 2003.

[22] Ann Fruhling and Sang Lee. "Assessing the reliability, validity and adaptability of PSSUQ". In: *AMCIS 2005 Proceedings* (2005), p. 378.

[23] James Gain, Patrick Marais, and Wolfgang Straßer. "Terrain sketching". In: *Proceedings of the 2009 symposium on Interactive 3D graphics and games* (2009), pp. 31–38. DOI: 10.1145/1507149.1507155.

[24] James Gain, Bruce Merry, and Patrick Marais. "Parallel, Realistic and Controllable Terrain Synthesis". In: *Computer Graphics Forum*. Vol. 34. 2. 2015, pp. 105–116. DOI: 10.1111/cgf.12545.

[25] Eric Galin, Eric Guérin, Adrien Peytavie, Guillaume Cordonnier, Marie-Paule Cani, Bedrich Benes, and James Gain. "A Review of Digital Terrain Modeling". In: *Computer Graphics Forum* (2019). DOI: 10.1111/cgf.13657.

[26] Anthony G Greenwald. "Within-subjects designs: To use or not to use?" In: *Psychological Bulletin* 83.2 (1976), pp. 314–320.

[27] Kenny Gruchalla. "Immersive well-path editing: investigating the added value of immersion". In: *IEEE Virtual Reality 2004*. IEEE, pp. 157–164. DOI: 10.1109/VR.2004.1310069.

[28] Chris Hand. "A Survey of 3D Interaction Techniques". In: *Computer Graphics Forum* 16.5 (1997), pp. 269–281. DOI: 10.1111/1467-8659.00194.

[29] David Heaney. *Share Of VR Headsets On Steam Doubled In 2018*. https://uploadvr.com/vr-steam-grew-2018/. [Online; accessed 11-June-2019]. 2019.

[30] Ken Hinckley, Randy Pausch, John C. Goble, and Neal F. Kassell. "A Survey of Design Issues in Spatial Input". In: *Proceedings of the 7th Annual ACM Symposium on User Interface Software and Technology*. New York, NY, USA: ACM, 1994, pp. 213–222. DOI: 10.1145/192426.192501.

[31] *HTC Vive*. https://www.vive.com/us/. [Online; accessed 22-January-2019].

[32] Takeo Igarashi, Takeo Igarashi, Satoshi Matsuoka, and Hidehiko Tanaka. "Teddy: A Sketching Interface for 3D Freeform Design". In: *ACM SIGGRAPH 2007 Courses*. SIGGRAPH '07. New York, NY, USA: ACM, 2007. DOI: 10.1145/1281500.1281532.

[33] Javier Irizarry, Masoud Gheisari, Graceline Williams, and Bruce N. Walker. "InfoSPOT: A mobile Augmented Reality method for accessing building information through a situation awaren". In: 33 (2013), pp. 11 –23. DOI: https://doi.org/10.1016/j.autcon.2012.09.002.

[34] Jason Jerald, Paul Mlyniec, Arun Yoganandan, Amir Rubin, Dan Paullus, and Simon Solotko. "MakeVR: A 3D world-building interface". In: *IEEE Symposium on 3D User Interface*. IEEE, 2013, pp. 197–198. DOI: 10.1109/3DUI.2013.6550246.

[35] Dominik P Käser, Evan Parker, Adam Glazier, Mike Podwal, Matt Seegmiller, Chun-Po Wang, Per Karlsson, Nadav Ashkenazi, Joanna Kim, Andre Le, et al. "The making of Google earth VR". In: *ACM SIGGRAPH 2017 Talks*. 2017, pp. 1–2.

[36] F. Kenton Musgrave, Craig E. Kolb, and Robert S. Mace. "The Synthesis and Rendering of Eroded Fractal Terrains". In: *SIGGRAPH Comput. Graph.* 23.3 (July 1989), pp. 41–50. DOI: 10.1145/74334.74337.

[37] Gordon Kurtenbach and William Buxton. "The Limits of Expert Performance Using Hierarchic Marking Menus". In: *Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems*. New York, NY, USA: ACM, 1993, pp. 482–487. ISBN: 0-89791-575-5. DOI: 10.1145/169059.169426.

[38] Brenda Laurel, Rachel Strickland, and Rob Tow. "Placeholder: Landscape and Narrative in Virtual Environments". In: *SIGGRAPH Comput. Graph.* 28.2 (1994), pp. 118–126. DOI: 10.1145/178951.178967.

[39] James R Lewis. "IBM Computer Usability Satisfaction Questionnaires : Psychometric Evaluation and Instructions for Use". In: *International Journal of Human-Computer Interaction* 7.1 (1995), pp. 57–78. DOI: 10.1080/10447319509526110.

[40] James R Lewis. "Psychometric evaluation of the post-study system usability questionnaire: The PSSUQ". In: *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*. Vol. 36. 16. SAGE Publications Sage CA: Los Angeles, CA. 1992, pp. 1259–1260.

[41] James R Lewis. "Psychometric evaluation of the PSSUQ using data from five years of usability studies". In: *International Journal of Human-Computer Interaction* 14.3-4 (2002), pp. 463–488.

[42] Brittany T. MacEwen, Dany J. MacDonald, and Jamie F. Burr. "A systematic review of standing and treadmill desks in the workplace". In: *Preventive Medicine* 70 (Jan. 2015), pp. 50–58. DOI: 10.1016/j.ypmed.2014.11.011.

[43] Deborah J Mayhew. *The usability engineering lifecycle: a practitioner's handbook for user interface design*. Morgan Kaufmann, 1999.

[44] Hilary McLellan. "Virtual realities". In: *Handbook of research for educational communications and technology* (1996), pp. 457–487.

[45] Mark Mine, Arun Yoganandan, and Dane Coffey. "Making VR Work: Building a Real-world Immersive Modeling Application in the Virtual World". In: (2014), pp. 80–89. DOI: 10.1145/2659766.2659780.

[46] Rolf Molich and Jakob Nielsen. "Improving a human-computer dialogue". In: *Communications of the ACM* 33.3 (1990), pp. 338–349.

[47] Jakob Nielsen and Rolf Molich. "Heuristic Evaluation of User Interfaces". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. New York, NY, USA: ACM, 1990, pp. 249–256. DOI: 10.1145/97243.97281.

[48] *Oculus Guidelines*. https://developer.oculus.com/design/latest/concepts/book-bp/. [Online; accessed 18-February-2019].

[49] *Oculus Rift*. https://www.oculus.com/rift/. [Online; accessed 30-March-2017].

[50] Vladimir Alves dos Passos and Takeo Igarashi. "LandSketch: A First Person Point-of-view Example-based Terrain Modeling Approach". In: (2013), pp. 61–68. DOI: 10.1145/2487381.2487382.

[51] Ken Perlin. "An image synthesizer". In: *ACM Siggraph Computer Graphics* 19.3 (1985), pp. 287–296.

[52] Kevin Ponto, Ross Tredinnick, Aaron Bartholomew, Carrie Roy, Dan Szafir, Daniel Greenheck, and Joe Kohlmann. "SculptUp: A rapid, immersive 3D modeling environment". In: *IEEE Symposium on 3D User Interface*. IEEE, 2013, pp. 199–200. DOI: 10.1109/3DUI.2013.6550247.

[53] Ivan Poupyrev, Mark Billinghurst, Suzanne Weghorst, and Tadao Ichikawa. "The go-go interaction technique". In: *Proceedings of the 9th annual ACM symposium on User interface software and technology*. ACM Press, 1996, pp. 79–80. DOI: 10.1145/237091.237102.

[54] Whitman Richards. "Stereopsis and stereoblindness". In: *Experimental Brain Research* 10.4 (1970), pp. 380–388.

[55] Silvia Scali, Mark Wright, and Ann Marie Shillito. "3D Modelling Is Not for WIMPs ". In: *Proceedings of HCI International*. 2003, pp. 2–6.

[56] Udo Schultheis, Jason Jerald, Fernando Toledo, Arun Yoganandan, and Paul Mlyniec. "Comparison of a two-handed interface to a wand interface and a mouse interface for fundamental 3D tasks". In: *IEEE Symposium on 3D User Interfaces 2012*. IEEE, 2012, pp. 117–124. DOI: 10.1109/3DUI.2012.6184195.

[57] Richard H Y So and W T Lo. "Cybersickness: an experimental study to isolate the effects of rotational scene oscillations". In: *Proceedings IEEE Virtual Reality*. IEEE. 1999, pp. 237–241. DOI: 10.1109/VR.1999.756957.

[58] Peng Song, Wooi Boon Goh, William Hutama, Chi-Wing Fu, and Xiaopei Liu. "A Handle Bar Metaphor for Virtual Object Manipulation with Mid-air Interaction". In: (2012), pp. 1297–1306. DOI: 10.1145/2207676.2208585.

[59] Beatriz Sousa Santos, Paulo Dias, Angela Pimentel, Jan Willem Baggerman, Carlos Ferreira, Samuel Silva, and Joaquim Madeira. "Head-mounted display versus desktop for 3D navigation in virtual reality: A user study". In: *Multimedia Tools and Applications* 41.1 (2009), pp. 161–181. DOI: 10.1007/s11042-008-0223-2.

[60] Sibylle D. Steck and Hanspeter A. Mallot. "The Role of Global and Local Landmarks in Virtual Environment Navigation". In: *Presence: Teleoperators and Virtual Environments* 9.1 (2000), pp. 69–83. DOI: 10.1162/105474600566628.

[61] Alistair Sutcliffe and Brian Gault. "Heuristic evaluation of virtual reality applications". In: *Interacting with Computers* 16.4 (Aug. 2004), pp. 831–849. ISSN: 09535438. DOI: 10.1016/j.intcom.2004.05.001.

[62] Ivan E. Sutherland. "A head-mounted three dimensional display". In: *Proceedings of the December 9-11, 1968, fall joint computer conference, part I on - AFIPS '68 (Fall, part I)*. New York, New York, USA: ACM, 1968, p. 757. DOI: 10.1145/1476589.1476686.

[63] Flora Ponjou Tasse, James Gain, and Patrick Marais. "Enhanced texture-based terrain synthesis on graphics hardware". In: *Computer Graphics Forum*. Vol. 31. 6. Wiley Online Library. 2012, pp. 1959–1972. DOI: 10.1111/j.1467-8659.2012.03076.x.

[64] Flora Ponjou Tasse, Arnaud Emilien, Marie-Paule Cani, Stefanie Hahmann, and Adrien Bernhardt. "First Person Sketch-based Terrain Editing". In: (2014), pp. 217–224.

[65] Mădălina Ioana Toma, Florin Gîrbacia, and Csaba Antonya. "A comparative evaluation of human interaction for design and assembly of 3D CAD models in desktop and immersive environments". In: *International Journal on Interactive Design and Manufacturing* 6.3 (2012), pp. 179–193. DOI: 10.1007/s12008-012-0144-1.

[66] Marc Treib, Florian Reichl, Stefan Auer, and Rüdiger Westermann. "Interactive editing of GigaSample terrain fields". In: *Computer Graphics Forum*. Vol. 31. 2. May 2012, pp. 383–392. DOI: 10.1111/j.1467-8659.2012.03017.x.

[67] Valve. *Steam Hardware & Software Survey: May 2019*. https://store.steampowered.com/hwsurvey. [Online; accessed 11-June-2019]. 2019.

[68] Juraj Vanek, Bedřich Beneš, Adam Herout, and Ondřej Šťava. "Large-scale physics-based terrain editing using adaptive tiles on the GPU". In: *IEEE Computer Graphics and Applications* 31.6 (Nov. 2011), pp. 35–44. DOI: 10.1109/MCG.2011.66.

[69] Norman G. Vinson. "Design Guidelines for Landmarks to Support Navigation in Virtual Environments". In: CHI '99 (1999), pp. 278–285. DOI: 10.1145/302979.303062.

[70] Miriam Walker, Leila Takayama, and James A Landay. "High-fidelity or low-fidelity, paper or computer? Choosing attributes when testing web prototypes". In: *Proceedings of the human factors and ergonomics society annual meeting*. Vol. 46. 5. SAGE Publications Sage CA: Los Angeles, CA. 2002, pp. 661–665.

[71] Jia Wang, Owen Leach, and Robert W. Lindeman. "DIY World Builder: An immersive level-editing system". In: *IEEE Symposium on 3D User Interface*. IEEE, 2013, pp. 195–196. DOI: 10.1109/3DUI.2013.6550245.

[72] Colin Ware and Steven Osborne. "Exploration and Virtual Camera Control in Virtual Three Dimensional Environments". In: *SIGGRAPH Comput. Graph.* 24.2 (Feb. 1990), pp. 175–183. DOI: 10.1145/91394.91442.

[73] Thomas G. Zimmerman, Jaron Lanier, Chuck Blanchard, Steve Bryson, and Young Harvill. "A Hand Gesture Interface Device". In: *Proceedings of the SIGCHI/GI Conference on Human Factors in Computing Systems and Graphics Interface*. New York, NY, USA: ACM, 1987, pp. 189–192. DOI: 10.1145/29933.275628.