Multitouch-Based Collaborative Previsualisation for Computer Animation

Kwegyir (Bilo) Lwabona

MASTERS of SCIENCE

Department of Computer Science

University of Cape Town

February 2016

supervised by

A/Prof. James Gain & A/Prof. Patrick Marais

Plagiarism Declaration

I know the meaning of plagiarism and declare that all of the work in this thesis, save for that which is properly acknowledged, is my own. As per the plagiarism declaration policy in [37] the following points below have been acknowledged and are true.

1. I know that plagiarism is wrong. Plagiarism is to use another's work and pretend that it is one's own.
2. I have used the IEEE convention for citation and referencing. Each contribution to, and quotation in, this report from the works of other people has been attributed, and has been cited and referenced.
3. This project report is my own work.
4. I have not allowed, and will not allow, anyone to copy my work with the intention of passing it off as his or her own work."
5. I acknowledge that copying someone else's assignment or essay, or part of it, is wrong, and declare that this is my own work"

Signature:

Acknowledgements

Special thanks go to my supervisor, Associate Professor James Gain, and co-supervisor, Associate Professor Patrick Marais, for their support and guidance throughout the development of this project. I would also like to thank my family, in particular my mother and brother, Eva and David, for supporting me every step of the way and helping me to stay focused for the duration of this research. I would like to thank Shelly Wilburn for advice on structuring this work, and a high-five to Klara Aizupitis for proof reading this report.

Without the participation and feedback of all volunteers in the user testing this research could not have been completed. A special thanks to Mario Kowarz, Etienne Fourie and Johan Prinsloo from FigureOneFilms, and Henk van Jaarsveld, Dave New and Ivan Sams from WhereIsMyTransport. I should not forget to mention Andre Hugo and Stefan Stiglingh, and Janike Stiglingh helped me stay focused under stressful situations. And of course I am thankful to everyone else that I have not mentioned by name above. I completed this work thanks to each and every single one of you. Thank you all for helping me make this happen!

Abstract

Computer animated pre-visualisation occurs at an early stage in the production life-cycle of an animated film, namely the pre-production phase. This is a collaborative process, in which directors communicate with animators how shot sequences will occur. Producers also take notes to approximate costs and other stakeholders may give further input. The problem with this approach is that the improvement cycles can take a long time, making the process take exponentially longer with more iterations of improvement. Our aim is to create a system that reduces this time, while keeping every stakeholder of the animation on the same page.

We have constructed a multitouch-enabled system for low-fidelity, animated 3D pre-visualisation. This tool runs on a single, large multitouch table and caters for simultaneous input from multiple users, to better support collaboration. Users can navigate the virtual environment, place and manipulate 3d objects in the scene, as well as animate them, all using multitouch. The system was constructed using the user-centred systems design (UCSD) methodology. After several iterations of development, we performed a qualitative evaluation of the final system using two groups, one consisting of film makers and the other consisting of software developers, and concluded with interviews to get qualitative feedback about our pre-visualisation tool.

Both groups suggested that the system's setup promoted collaboration and communication, which is important early on in the planning phase of film creation. However, both groups agreed that such a tool is only useful for low-fidelity pre-visualisation, as it might become "cumbersome" to perform detailed animations using multitouch input. Furthermore, the system was often too dependent on the viewpoint, which was a single user task, effectively minimizing the amount of work that could actually be done by collaborative users simultaneously. This study highlights the potential of a multitouch, collaborative pre-visualisation tool.

Glossary

2d: two dimensional

3d: three dimensional

collaboration: a process in which people work together to accomplish a certain task

CSCW: computer supported cooperative work

DOF: degrees of freedom

high-fidelity: content of high quality, generally requiring a large amount of effort and taking long time to create

low-fidelity: content of low quality, generally requiring a less effort and taking less time to create than high-fidelity content

MR: mixed reality - a combination of mixed and virtual reality

pre-visualisation: a process in which people work together to accomplish a certain task

qualitative: relating to, measuring, or measured by the quality of something rather than its quantity

quantitative: relating to, measuring, or measured by the quantity of something rather than its quality

SUS: system usability scale - a standardised method of assessing the usability of an application or a web site

UCSD: user centred system's design

view-point: perspective from which a virtual environment is displayed on screen

viewport: a framed area on a display screen for viewing information

VE: virtual environment - a digital, 3d representation of a real or imaginary setting/surrounding/environment

VR: virtual reality - a technique of using computers to model real (or imaginary) environments in three dimensions that allows people to interact with the environment in a fashion that is both natural and intuitive[52].

WYSIWYG: what-you-see-is-what-you-get.

WYSIWIS: what-you-see-is-what-I-see.

Contents

List of Figures

Chapter 1

Introduction

1.1 Previsualization

Computer animation is a popular and wide-spread form of entertainment. It is prevalent in film, but also extends to other media, such as video games, television series, and advertising. Over time this grounded industry has established concrete processes for completing such undertakings. Films generally go through an entire production lifecycle, from the initial concept to the final cut. This life-cycle consists of three phases, namely pre-production, production and post-production[65]. Pre-production is the phase in which the story is created and all planning and budgeting takes place. The actual film is created in high-quality in the production phase. Post-production is where marketing and advertising of the product, as well as other related tasks take place to launch the film.

This research focuses on a specific part of the pre-production phase, known as previsualization (previz). In this phase stakeholders visualise their script prior to the actual filming, hence the term *pre*visualization. Storyboarding is one of many common previz methods used early in the pre-production phase. Storyboards are sequences of annotated sketches that illustrate key camera angles and the scene composition to communicate the flow of scenes. This is occasionally followed by animated previsualization, which gives stakeholders a feel for the composition and timing of scenes. In many cases animated previz is low-budget and allows for early course correction, which is why a method of rapid scene prototyping can be critical in a film's production lifecycle.

1.2 Problem Definition

One characteristic of previz is its collaborative nature, in which the director, cinematographers, digital artists, producers, and others participate. In cases where very specialised software is used for previz, specialists with corresponding skills are required to construct the scenes. This research presents a low-fidelity previz system that removes the need for the skill and experience in using animation tools, by making the interaction simple and intuitive. One method of interaction that is maturing rapidly is touch, which is present in cellphones, cars, ticket-booking devices, ATMs, etc.. Multitouch technology allows multiple users to collaborate on a single device naturally, allowing the interactions from different users to occur simultaneously. This research aims to create a multitouch-based tool that allows collaborative previsualization of animated scenes. There are various complications that arise when considering a multi-user, collaborative interface. Complications include conflict resolution between users, and determining ownership of specific user interface controls and scene-based objects. By considering these complications and designing a multi-user interface, this research assesses whether multitouch is a suitable medium for collaborative previsualization.

Research question: How suitable is multitouch for same-time, same-place collaboration on a single device for low-fidelity previsualisation of computer animated scenes?

1.3 Proposed Solution

In creating a system to address the research question posed above, various research areas have to be considered. The previsualization we are dealing with is 3D (three-dimensional) animated. As shown in Figure 1.1, this project touches on 3D virtual environment navigation and object manipulation, previsualization, computer supported cooperative work, multitouch-based interaction, and user interface and usability design. Part I of this report consists of three chapters, each of which is dedicated to one of the spheres in Figure 1.1.

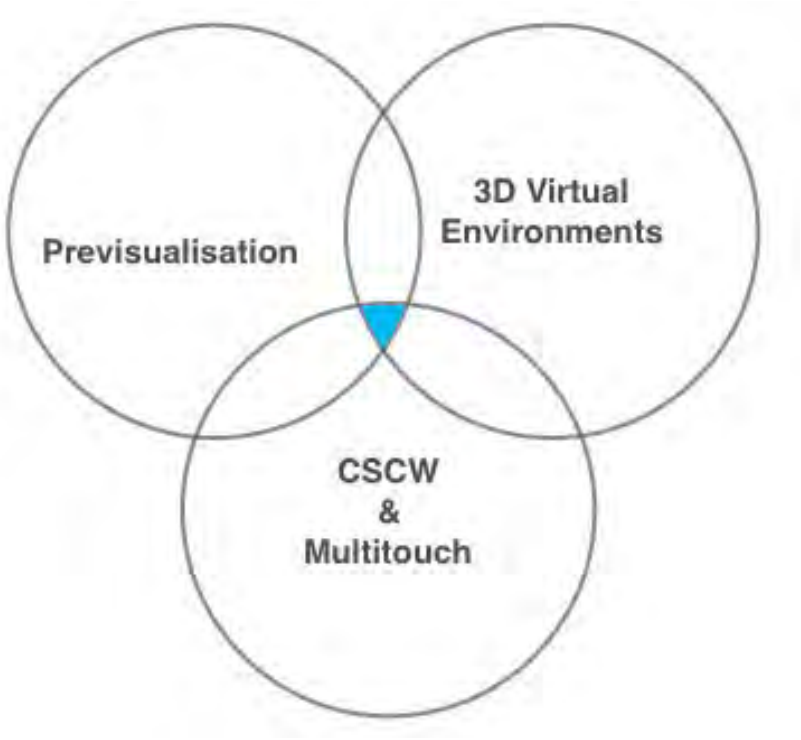


Figure 1.1: This research is the intersection of multiple research fields, including 3D virtual environment navigation, 3D object manipulation, previsualisation, computer supported cooperative work, multitouch-based interaction, and user interface and usability design.

In Section 5.4 of the Collaboration Design chapter we detail all rules and measures taken to cater for collaboration. The proposed solution provides a set of controls for every participating user, i.e. a user widget. Each of these user widgets allows some fundamental tasks to be performed when constructing a scene. Three of these user widgets can be seen under the hands of participants in the left of Figure 1.2. The right of Figure 1.2 shows different types of widgets: the left one used to place objects in the scene, the bottom right control is the user widget, the top right control is the camera widget, and the most central widget is to control one of several keyframes of an animated object. In several cases, constraints have been integrated to isolate control over certain objects and interface elements for individual users. Visual feedback has also been integrated for a more comprehensive user experience. The one core feature is that most widgets have a circular shape, and can be reoriented to suit any particular user's perspective. Furthermore, the round interfaces indicate the interaction radii of these user widgets, to further integrate collaboration by separating simultaneous interactions on different user widgets. The user interface and interaction design is expanded on throughout the System Design chapter.

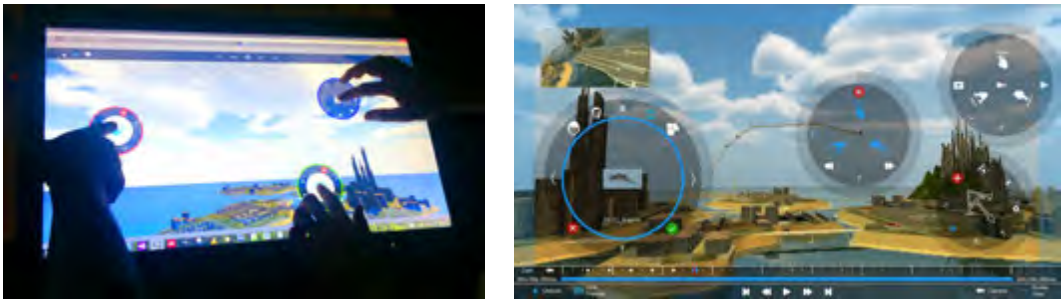


Figure 1.2: **Left:** A photograph of multiple users interacting with the system. **Right:** a screen shot of how the system is used to edit an animated object, while simultaneously placing an object into the scene independently.

Hardware A fairly large display, available at consumer prices is required to conduct this research, as it provides sufficient physical space for multiple people to use it simultaneously. The device used for this research is a Lenovo IdeaCentre Horizon A720, a 27 inch, ten-point multitouch-enabled computer [82]. Furthermore, its price range is within that of general consumer goods, making such devices available for smaller, indie studios as well. One advantage of the A720 is that it is a regular PC, which has the additional multitouch functionality. This makes it even more feasible for small companies to purchase such a device, as it would serve purposes other than just multitouch-based previz. The device can be used as a regular PC, yet the angle of the display can also be adjusted to be parallel to a tabletop as shown in figure 1.3.

Software The software used to create the system is the Unity 3D game engine [122], using C# as the programming language. A game engine is a more suitable choice than a graphics API such as DirectX or OpenGL, as this problem deals with interaction rather than rendering. Additionally, Unity provides support for targeting multiple different platforms, including multitouch-based operating systems such as iOS, Android and Windows 8. At the time of writing the software, Unity did not support multitouch for Windows directly[72], which is why the multitouch library TouchScript [115] was a necessity for this system.

Figure 1.3: The Lenovo IdeaCentre Horizon A720, is a regular PC with multitouch capability. It also has the capability to fold down, effectively acting as a touch table.

1.4 Design & Methodology

The research area of computer-supported cooperative work (CSCW) has stated the importance of evaluating systems [90, 9, 64], such as this collaborative previsualization tool. It is extremely difficult to conduct evaluation of collaborative systems [64] and there have been several cases of poor evaluation procedures [125], which is why this research used various different methods of data capturing. During the development of the system, a software development methodology was followed to aim for a good design of the system. Abiding by the design philosophy of user-centred systems design (UCSD) [60, 95], this system went through several design iterations, with each step adding consistencies and simplifying the experience to improve its usability.

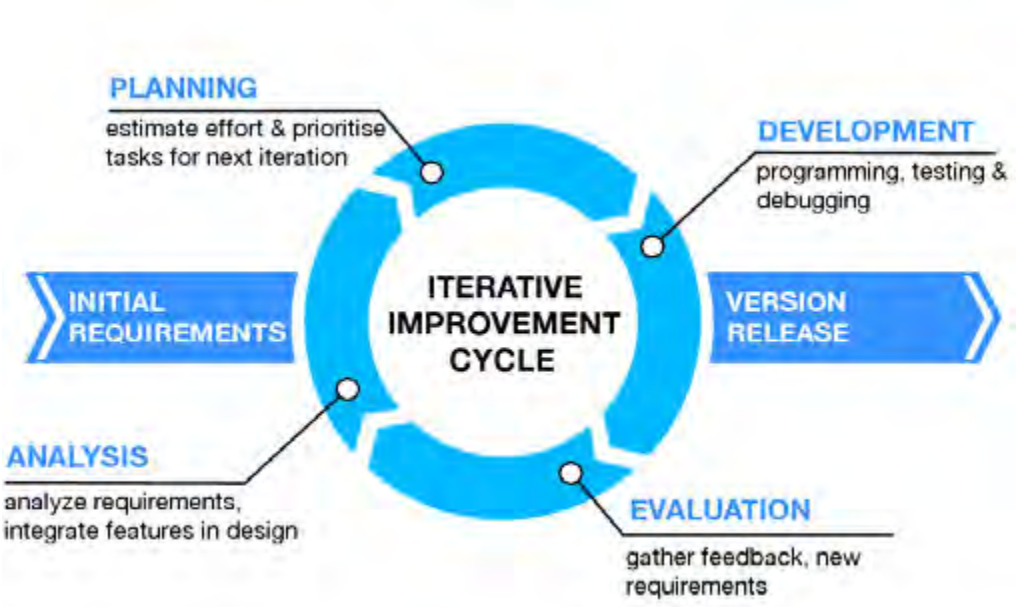


Figure 1.4: This is a diagram illustrating the iterative improvement cycle. This is a process going through stages of planning, development, evaluation and analysis in several iterations, where each one aims to improve the system being developed.

The collaboration and user interface design of this system, incorporated circular widgets for all scene objects and a toolbar that can be moved to any edge of the screen. The circular design is to support reorienting the a widget to suit a user's perspective regardless of which side of the screen they are standing/sitting. Regardless of the widgets orientation, it will not change the real estate it uses up on the screen, since it is circular. Depending on the type of component being dealt with, the buttons on the circular widget are different. This allows for a consistent and context-sensitive design for all widgets. Widgets usually fade out (become transparent) when not being interacted with and are generally in an idle state. To begin interacting with an object, it needs to be selected with a single tap, which brings up it's context-sensitive circular widget, that gives the user options for further interaction. An example of circular widgets are shown in Figure 1.2.

1.5 Thesis Structure

The remainder of this thesis covers the aforementioned content in more detail, which has been divided into three parts. Part I covers the fundamentals and previous work, which consist of three chapters. Chapter 2 covers the theory, literature and tools of previsualization, detailing the steps to create animatics (animated previsualization), and show different kinds of existing previsualization methods. Chapter 3 covers the fundamentals of virtual environments and existing work in this area. Chapter 4 presents literature of previous work in CSCW and multitouch interaction. In each of these chapters the pros and cons are considered for each of these systems in relation to this specific research, to conclude the fundamentals presented in Part I.

Part II covers the design and implementation of the previz system created in this research, and consists of two chapters. Chapter 5 presents the collaboration design of this system, considering various methods to support collaboration, before presenting the chosen one. Thereafter Chapter 6 presents the system design, includes the how the collaboration design was applied to various system components. The system design also covers usability integration, and multitouch interaction techniques, providing motivations for all decisions made in the process.

Part III concludes the research with an evaluation of the system. In Chapter 7 the chosen evaluation method is laid out, followed by the design of the user experiments and concluded with the results. Chapter 8 is a retrospective of the system, together with future work, concluding the thesis.

Part I

Fundamentals & Previous Work

Chapter 2

Previsualisation

In this chapter the animation production pipeline is introduced, with details of tasks tackled in the various stages of the pre-production process. Tasks include designing and creating characters and assets, planning shots and creating low budget animations. Thereafter, several different types of previsualisation (previz) are presented, in order to place this research in context. The chapter concludes with examples of tools used to create animated previz or even full films in some cases.

2.1 Animation Production Pipeline

An animated short or film's production life-cycle can be split into three distinct stages. In the film and animation industry these stages are referred to as pre-production, production and post-production. While the film and animation industries are overly related, there remain some differences between creating a purely live-action film compared to a completely animated film. The remainder of this section breaks down the production pipeline specifically for animation [65, 81].

2.1.1 Pre-production

The first stage in the production life-cycle of an animated piece is the concept sign-off, in which the initial vision of the film are prototyped and approved. Pre-production is considered crucial, because it paints a good picture of what is expected and it can foreshadow infeasible parts of the planned animated piece. The team usually starts off with concepts, which are then pieced together to create a full story, subsequent to which the script is written. Generally, after the script has been finalised other essentials, such as the story boards, layouts, model sheets and animations are worked on.

Story Boarding

One purpose of story boarding is to help complete the storyline's development. However, the more important and practical use of storyboards is as a guideline for the animation process. Story boards show key frames of events, including the major focus of actors, desired camera angles and potentially the background scenery. All this is communicated through comic-like illustrations, often accompanied by textual annotations for additional information that cannot be directly communicated with a static image (e.g. camera behaviour). Later on in the production life cycle the initial storyboards can also be used as a reference or a reminder of the original concept (see Figure 2.2).

Layouts

Layouts are created once the storyboards have been signed off. The job of the layout artist(s) is to take the storyboards and translate them into a more detailed and usable set of drawings for the

animators[25]. The layout department meets with the director to design character appearances and scenery. Once this is complete, the scenes are staged, showing the different character's positions throughout each shot.

Model Sheets

The modelling team needs precise specifications in order to model all key characters and objects. Model sheets accomplish this using groups of drawn pictures, illustrating various scenarios for character, such as poses and expressions. This starts off with an illustration from every side, and goes all the way to including every expression the characters could make, as well as every pose they could adopt. These sheets help keeping the character designs consistent and detailed, while animators are creating the first visualisations of the characters. This is also the stage where character designs are finalised in time for production, serving as blueprints for the modelling department.

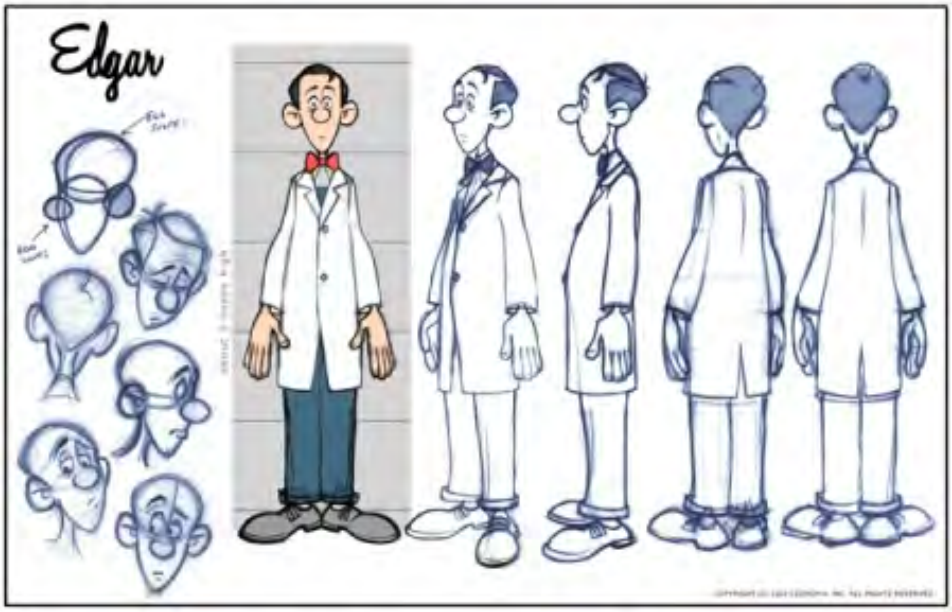


Figure 2.1: *Example of a model sheet for a character.[24]*

Animatics

Once the storyboards have been signed off and there is a basic design of the characters used in animatics. These are the first animated version of characters and scenes, created by the previz department. Animatics provide a better idea of the motion, composition and timing of animated sequences. Furthermore, they help the Director to plan how visual effects will be used in the final product, as well as how to stage the provided sequences.

2.1.2 Production and Post-Production

Essentially, pre-production centres on concept sign off, after which the production phase begins. This consists of several stages, which together produce the final outcome. Some of the tasks include **layout**, **modelling**, **texturing**, **lighting**, **rigging** and **animation**. These tasks basically involve creating the high-detail version of the 3d model geometry, their surface detail, and more technical aspects such as lighting, applying an animatable skeleton for the models and then animating the characters.

Post-Production is the final stage of the creation process for animated shorts and films. To summarise this stage, the individual animated scenes are put together into a single video. The three significant phases are **compositing**, **sound editing** and **video editing**. With these tasks the final animated film is created as a seamless film with full detail graphics, sound and lip sync.

There will be no further mention of the details in the production and post-production stages, as the subject of this research is previz, which is a part of pre-production.

2.2 Types of Previz

There are numerous different types of previz, each with it's own purpose. Several types of storyboarding and animated previz will be covered in this section.

2.2.1 Traditional Storyboarding

One common type of previz is storyboarding, which is often performed during and after a script has been written. This technique can be very effective in visually telling a story. In some media, such as comics and graphic novels, a very polished storyboard is the final product and tells the entire story. Storyboards are a sequence of images showing progression of scenes and key events from specific camera angles. Occasionally, storyboards are guided by annotations to better describe the respective shots. Storyboards are also used in live-action film, as shown in Figure 2.2, a storyboard from the movie American Beauty (1999).

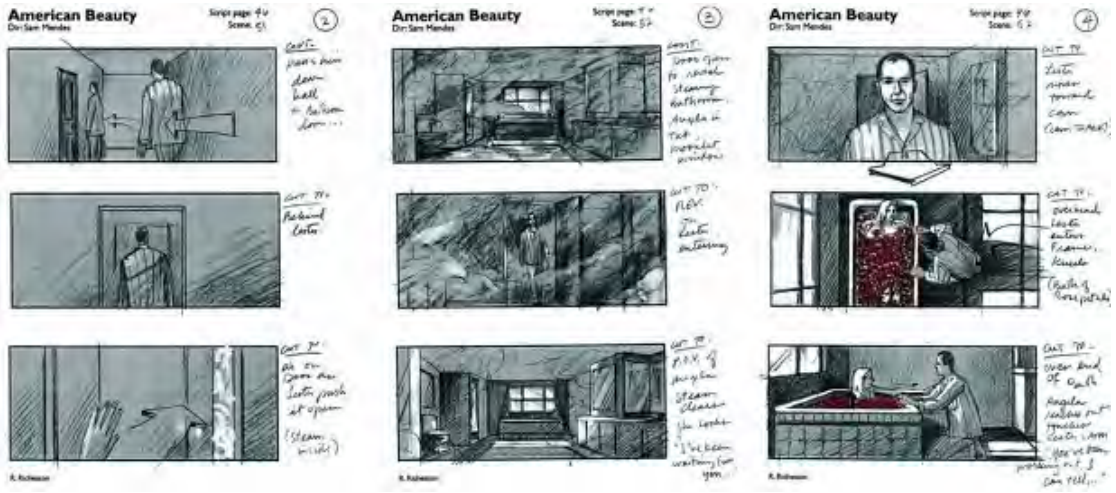


Figure 2.2: *Storyboards are used as a very early, non-animated previz technique. This is an example of an annotated storyboard, containing information for each key frame, from the movie American Beauty (1999) [3].*

Usually the storyboards are the initial visualisation of scenes, but are subject to change in the final release, based on how drastically animatics are adjusted. In some cases scenes are scrapped entirely, whereas others are can be traced back all the way to the storyboarding phase, as shown in Figure 2.3 which compares the storyboard to the final motion picture of Tangled(2010).

Figure 2.3: *A comparison of storyboard frames (left) and their counterparts (right) in final footage of Tangled (2010) [4].*

2.2.2 3D Story Boarding

In 3D storyboards, artists interact with a 3D environment to place characters and objects within scenes in the required poses. The depicted characters occasionally also show the required facial expressions. One advantage of this approach is rapid editing of key frames, as one manipulates a 3d virtual environment to take snap shots, rather than having to draw each key frame from scratch. The disadvantage with 3d storyboarding is that one may have to first create all required 3d assets, whereas storyboard artists drawing would be able to draw keyframes from scratch immediately.

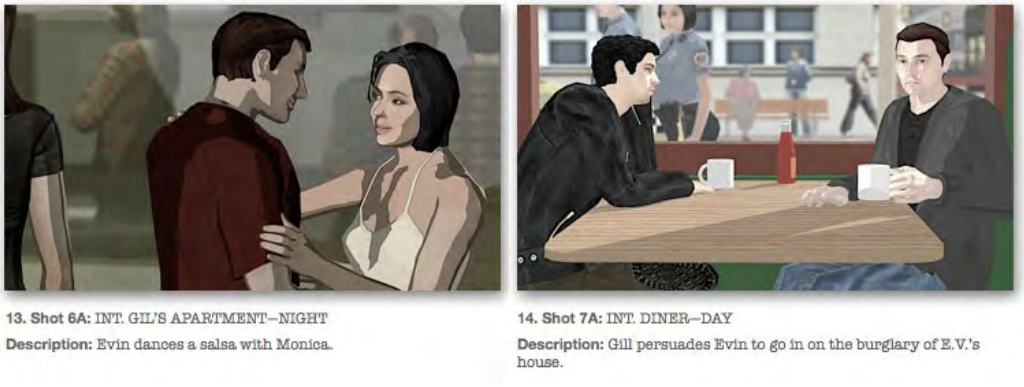


Figure 2.4: *An example of 3D storyboards using Toon Shading as the rendering technique [31].*

2.2.3 3D Animated Previz

Animated previzs, or animatics, consist of low fidelity models animated to show the composition of the scene with the timing and key camera angles. In these basic versions scenes are created from the storyboards to communicate the composition and timing of scenes, which allows for low budget changes before going into full production. The only disadvantage is the quality of the animatics, which lacks the detail of the final cut.

Figure 2.5: *Two frames of the animated previz (left) vs the final outcome (right) of the movie The Golden Compass (2007)[26, 15].*

2.2.4 Low vs High Fidelity Previz

Low fidelity, Animated Previz

Animated previz of films can be considered as prototyping of the final product. Low-fidelity (lo-fi) prototypes are constructed to depict concepts, design alternatives, and layouts rather than products with full feature sets [105]. A lo-fi animated previz would therefore only focus on depicting the core aspects of previz, which are composition and timing. All other aspects of the animated piece, such as the visual effects and sound, are generally quite incomplete, or may not even be existent at this stage. The purpose of lo-fi previz is to get a rough representation of the final outcome, in a time-efficient and cost-effective manner, such that drastic changes can be made very early in the production life-cycle.

Figure 2.6: *Low fidelity previz (left) vs final scene (right) from the movie Iron Man 3 (2013) [21].*

High fidelity, Animated Previz

Unlike the former, high fidelity previz is more sophisticated and with higher detail, but takes more time and effort to create. In animated previz, this can include the full 3d geometry as well as animation, composition and timing, whilst omitting texture details and other visual effects. This is more likely to occur when the animated short or film is already in full production and minor details are being adjusted.

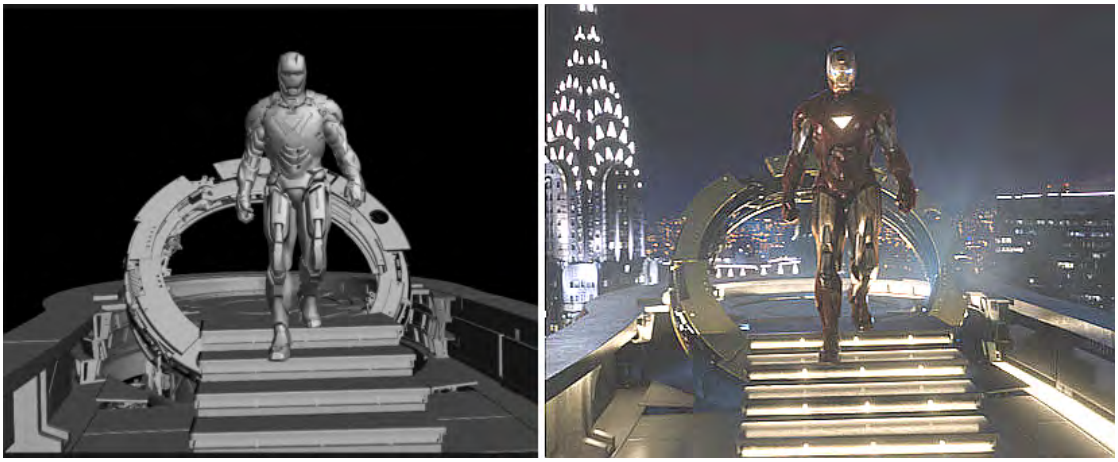


Figure 2.7: *High fidelity previz (left) vs final scene (right) of Iron Man 2 (2010), shows that in the previz version most of the geometry is already complete and what is left is texturing and other visual effects, which usually take place in the post production phase [20].*

2.3 Animation

2.3.1 Animation Frames

One further aspect to consider is how the final, animated video is constructed. In basic terms it is not too different from film. Frames per second (fps) are the number of frames, or still images, shown in one second. When a live-action scene is being recorded, the camera samples images at a high enough rate (29 fps in the US and 25 fps in Europe), and stores these in a digital medium or, in traditional cases, as film. When the images are shown in succession at a sufficiently high frame rate, the human eye usually perceives this as a continuous motion of graphical elements, rather than a sequence of images being shown in rapid succession. Conceptually, this functions like a really fast slideshow. In fact, in 1824, Peter Roget created and published a book on the theory of "The persistence of vision with regard to moving objects." [7]. This theory states that the appearance of motion could be created by a rapid succession of images. The same concept is used in 2-dimensional, drawn animated films[2, 57] and claymation[8]. In computer animation the rapidly captured images are snapshots of a virtual scene, for which each frame has been completely rendered.

Rendering is the process that takes all objects in the 3D virtual scene, calculates their visibility and applies visual effects, among many other tasks, and eventually produces rasterised images, which are projected onto a typically flat display[76]. This process can be computationally expensive, especially in computer animation. To speed up this process, animation studios usually have render farms, which are clusters that contribute their processing power to a render task. While rendering can be discussed in much more detail, in this thesis we are not dealing with a computer graphics problem. This research focuses on HCI (human computer interaction) and CSCW (computer supported cooperative work), so rendering will not be discussed further.

2.3.2 Animation Timeline

A graphical, interactive timeline has been used to control and edit animated scenes in various animation packages and even game engines [122, 56, 6, 123, 41, 40]. In its most common form, a timeline is docked at the bottom of the application window and scrolls horizontally, where the starting point is on the far left of the scrollable region. Various controls exist to play and pause the created animations. The time position of the current point in the playback is usually indicated by a thin vertical line, running from the top to the bottom of the timeline widget. Additionally, at the top (or bottom) of the scrollable region a scale (or ruler) is provided, labelling the time of each point on the timeline. Lastly, as the playback occurs, the animation takes place and this is visible on the timeline as keyframes or clips.

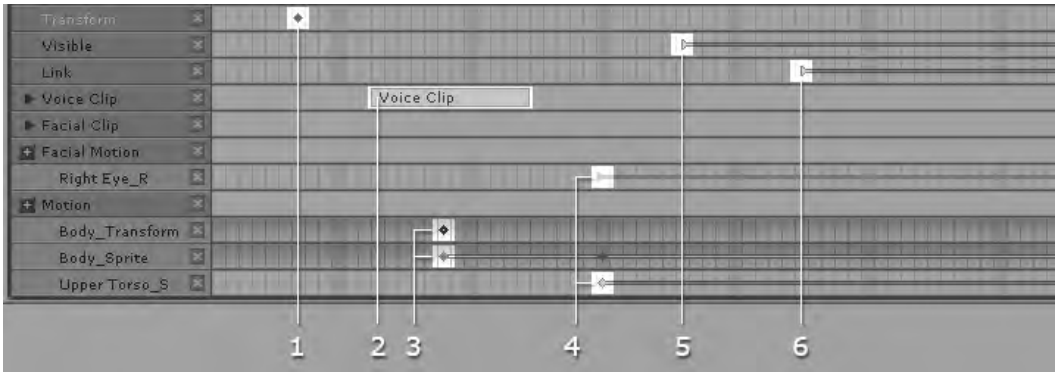


Figure 2.8: *On this animation timeline, the different rows (or bars) indicate channels for various aspects of the clip, such as the position, voice clip and facial motion . The numbers indicate the beginning the clips on the various channels[34].*

2.3.3 Keyframes

Let us consider a simplified version of the animation involving object transformations, specifically their position and orientation. A basic form of animation requires at least two transforms (states) of an object, a start and end frame. These frames are specified by a user, and the system performs the animation by interpolating the object's transform from start to end, i.e. by automatically producing all intermediate frames. Interpolations can be done in a number of ways, including linear, slow-in slow-out, and several other ways. Furthermore, one can use graphical tools such as curve editors to create custom interpolation speeds. Using this approach, a desired animation can be created via a sequence of transitions, each connected by a set of so called keyframes. On timelines, these are usually shown as rectangular GUI elements and are positioned at the corresponding points in time.

Figure 2.9: *QuickTime (top) shows regular snapshot of the entire footage, whereas YouTube (bottom) shows frames of footage neighbouring the selected point in time. The movie shown in the clips is Final Fantasy 7: Advent Children (2005). (Images generated from screenshots, source of video: [11])*

2.3.4 Clips

Another way of showing data on a playback timeline is by using clips. That is, to show a sequence of snapshots of the rendered footage on the timeline. This approach is used in QuickTime[5] for clip editing purposes. In this case a limited number of snapshots are shown from the start to the end of the video file and the respective clips are spaced at regular intervals, depending on the length of the video and how wide the timeline is. A slightly modified version is used in YouTube[32], where only the snapshots near the selected frame are shown, with the current one being larger than the neighbouring frames. As users scroll along the playback timeline in the YouTube video viewer, the shown frames update to reflect the appropriate footage. With both these approaches the video footage needs to be available such that the respective frames can be extracted. Both methods of using clips are shown in figure 2.9. Clips can also be used for audio clips on animation timelines that use keyframes.

2.4 Previz Literature

2.4.1 Systems for Previz

The Virtual Cinematographer is a system created for automated camera movement to portray actors and content in scenes. As shown in the left of Figure 2.10, a real-time application contains all events and geometric information, which the virtual cinematographer uses to automate the camera's behaviour. The outputs from the virtual cinematographer and the entire real-time application are displayed on screen by the renderer. The automated camera movement to to portray actors in scenes was implemented in the form of film *idioms* [69]. The purpose of each idiom is to capture a specific kind of scene, e.g. a conversation between three virtual actors, as shown on the right of Figure 2.10. These idioms are encoded as a set of small hierarchically organised finite state machines. This kind of system is not suitable for this research as the camera movement is automated, whereas a director would likely require control over shots and sequences. Furthermore, directing styles differ between directors to a greater or lesser extent, which no automated system could fully support. Nevertheless, this research presents some fundamental concepts of cinematography and camera control that influenced the development of our system.

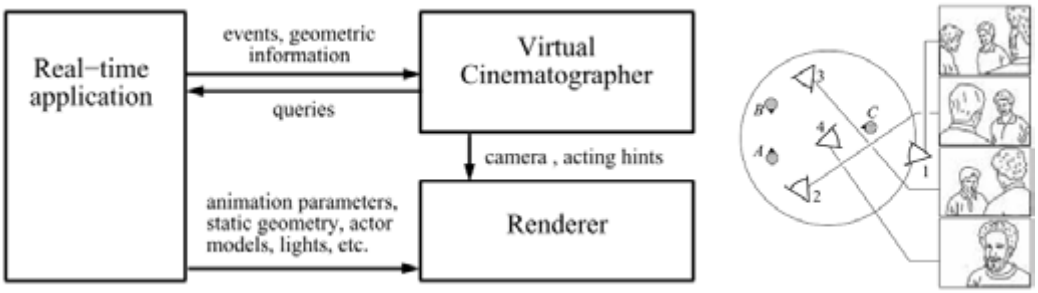


Figure 2.10: ***Left:*** *The virtual cinematographer is a module that automates camera movement for specific kinds of scenes in a real-time appliaction. The camera's perspective and movement through the virtual environment is displayed on screen with a renderer.* ***Right:*** *A four-shot formula to depict a conversation between three actors.*

Shapiro presents the Novice User's Camera Control Interface (NUCCI) [110], which is a real-time cinematic solution to previz. This project re-staged parts of Fincher's *Panic Room (2002)* in a 3d real-time game environment. Users could either watch the scene with the original camera angles, or create their own camera path.

One way in which previz can be sped up is by using mixed-reality (a combination of virtual and actual reality), e.g. by recording a real person's movements with motion capture (see bottom of Figure 2.11) and applying these recordings to a virtual character. Such an approach is quicker to generate animations, than having a digital artist create animations manually from scratch. MR-PreViz (Mixed Reality Pre-visualisation) makes use of mixed reality (MR) technology for previz [119], allowing the merging of real background and the computer generated characters and creatures in an open set or outdoor location. The previz technology cultivated by MR-PreViz is applicable to motion picture, beyond just live-action film, making it suitable for animated previz. The drawback of such a system is that it is more suited to the animation of humanoid creatures. While it is not impossible, it may require more effort to capture the motion of other anatomical structures such as four-legged animals or birds, as it may be difficult to capture the desired motion of an arbitrary animal at will. A human could try to mimic the motion of the desired animal, but this may not seem realistic enough, due to the differences in the anatomy of the human and the desired animal. This is relevant, because animal could very well be the main characters in animated movies, such as *Zambezia (2012)* and *Khumba (2013)*, both created by Triggerfish Animation Studios.

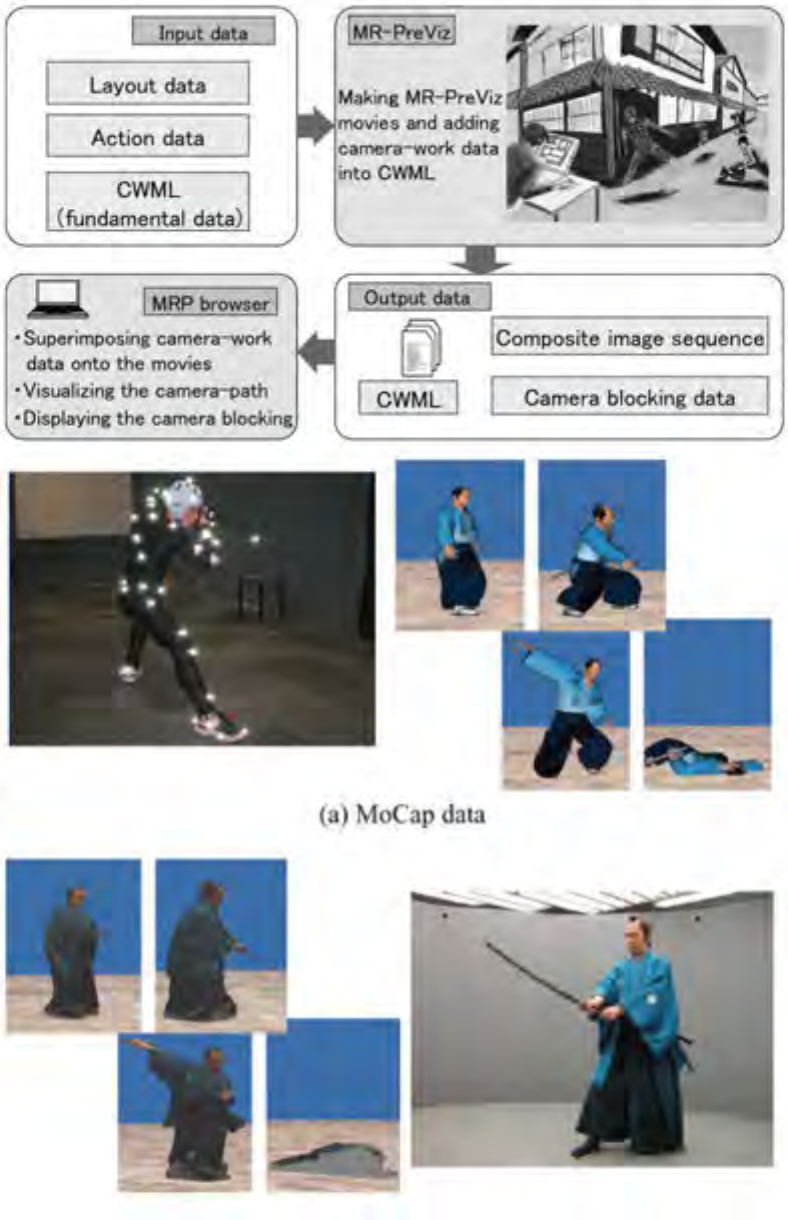


Figure 2.11: MRPreViz: A motion capture tool allowing a combination of virtual and actual reality. **Top:** An overview of how the different modules fit together and how the flow of data occurs to generate the final animated scene. **Bottom:** An example of how motion is captured from an actor and applied to a virtual actor.

As stated in Section 2.1.1, one part of previz is creating layouts and storyboards, which help to conceptualise the scene setting and composition, as well as shot sequences and key camera angles. Since the aim of this research is to pre-visualise 3d scenes, it can be useful to consider using 3d storyboards and layouts. Zeleznik et al. present SKETCH, an application that combines the advantages of 3D computer modelling systems with the sketching to rapidly conceptualise and edit approximate 3D scenes[128]. This is achieved by using simple, non-photorealistic rendering and an entirely gestural interface, based on simplified line drawings of primitives that allow all operations to be specified within

a 3D world.

2.4.2 Game Technology for Previz

Manovich discusses specific topics of the digital visual media and defines principles (modularity, variability, automation, numerical transcoding and cultural transcoding), which apply to video games and computer generated film [84]. In concept, both digital films and games are based on the idea of the computer as a "media processor". Technically speaking, the graphics hardware between CGI render farms and that of gaming consoles and PCs is not too different, making it possible to achieve even sophisticated film renders using a game engine on a home PC.

Nitsche outlines the value of real-time 3D engines for previz of modern films [94]. For over a decade now, game engines have been used for previz, for e.g. a modification of the Unreal engine for previsualising Steven Spielberg's A.I. (2001). Nitsche discusses the parallels between the two increasingly digitised technologies, and outlines the problems and requirements posed by previz. According to Nitsche's research, animation control and camera control are the two main areas that need to be addressed. Several prototypes were created and presented by machinima artists and professional film/TV producers, who gave very positive feedback. There seems to be a great interest in using game technology for previsualization, which is improving over time as prototypes and full applications evolve with different feature sets. While some components, such as the camera control in a game world, are very versatile, it can be more difficult to find the right interface for traditional film and TV production methods [94]. Furthermore, with real-time environments such as game engines, it can be difficult to create new editing controls that embrace the most powerful features of a game engine, such as Global Illumination[53], Volumetric Lighting[98, 104, 121], and Particle Effects [89, 101, 116, 78].

Northam et al.. present a collaborative real-time previz tool for video games and film, which uses helps to create still images, walk-throughs and even preliminary animations and of scenes, levels and other assets[97]. Their solution RTFX (real-time special effects) is a chat-like client-server architecture that connects different previz tools to improve collaboration. An example of the combining different software can be seen in Figure 2.12, where real-time motion capture data is combined with light assets from Houdini[114] and imported in UDK[56, 55]. Each previz tool uses an RTFXClient plug-in that sends messages containing assets and scene data, such as meshes and the camera position, the RTFXServer. When the server updates other clients in the network with the assets and data. What makes this framework appealing, is that RTFXClient plugin-ins can be written for any animation application or game engine, making this work generic.

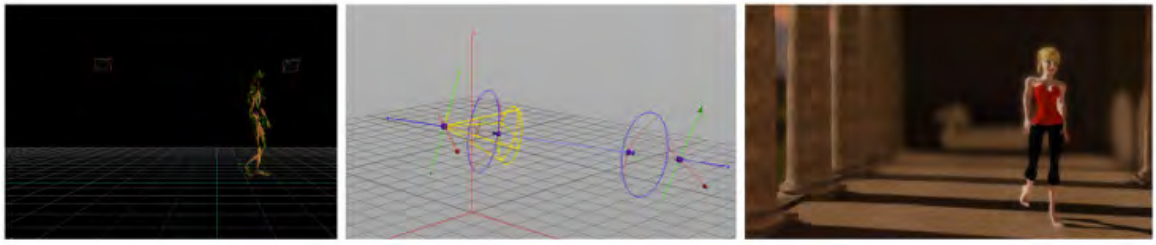


Figure 2.12: *RTFX previz scenario combining real time motion capture data* ***(left)*** *with light assets from Houdini* ***(middle)*** *into a UDK set* ***(right)***.

2.5 Previz Software

The type of systems covered here are focused on previz and film generation. These are systems that help in communicating a prototype version of the final, animated product. The main focus of previz systems is the composition and timing of scenes. It should be mentioned that the previz software used by animation studios is often the same software they use to create the final product. One reason for this is the possibility of reusing the previz data generated in the pre-production phase. Popular software for 3d modelling and animation includes XSI SoftImage, Maya, 3DS Max, AnimeStudio and Blender [40, 6, 41]. The aforementioned systems usually require significant learning in order to produce something reasonable, without even utilising the full capability of the system. Instead, we focus on software with less of a learning curve.

2.5.1 iClone

One feature rich previz system for creating animations is iClone [17]. This tool simplifies the animation aspect substantially, as it supports Microsoft's XBox Kinect, which can be used for motion capture. Users can move as they desire and the system records their motions, turning them into biped animations in the virtual scene. This take on recording actual motion of users substantially reduces the time required to create realistic body animation. Not only does iClone cater for body animations, but it also provides the functionality to create facial animations. Furthermore, one can save the created animations and even transfer them from one character to another easily. Animations can also be exported in common data formats, e.g. FBX, which are suitable for various game engines, such as Unity.

Figure 2.13: *iClone allows users to record their movement with a motion capture device to animate 3D models.*

2.5.2 Movie Storm

Movie Storm [33] is a desktop-based film creation tool with a versatile set of features, useful for animated film creation. One thing to note is that this software is less like traditional animation tools, and more like being on a film set, in terms of camera setup and "physical" limitations in the virtual world. Furthermore, it is said to be similar to computer games in some aspects. In Movie Storm creating a film consists of several basic steps: set building, character creation, directing, filming, editing and rendering.

A user can either create a new set from scratch or choose a predefined set from a collection of interiors and exteriors. Once the setting has been chosen, one can place props and lights into the scene and customise and scale these arbitrarily. Lastly, one can also create custom assets and import them into the assets library. Characters can also be created with the aid of predefined assets, and their physical appearances can be customised with the aid of sliders. This applies to the body as well as the face and hairstyle of characters. To conclude, the character creation tool, Movie Storm also caters for outfitting characters with different clothes and accessories.

Directing has been implemented in a very powerful and intuitive way, in that the director merely needs to specify basic tasks and the system deals with the rest. For example, when making a user walk from one point to another, one merely has to specify the character's destination. Walking is automatically animated, and should there be an obstacle such as a closed door along the path, then the character will automatically open it. A variety of common animations animations, such as walking, opening doors, etc., are included by default, but additional work is required for such functionality with custom objects.

While Movie Storm is very powerful and intuitive for previz, it is very specific and thus limited in the things that can be done. Nevertheless, the content is extensible with custom assets and animations, providing users with the option of easily.

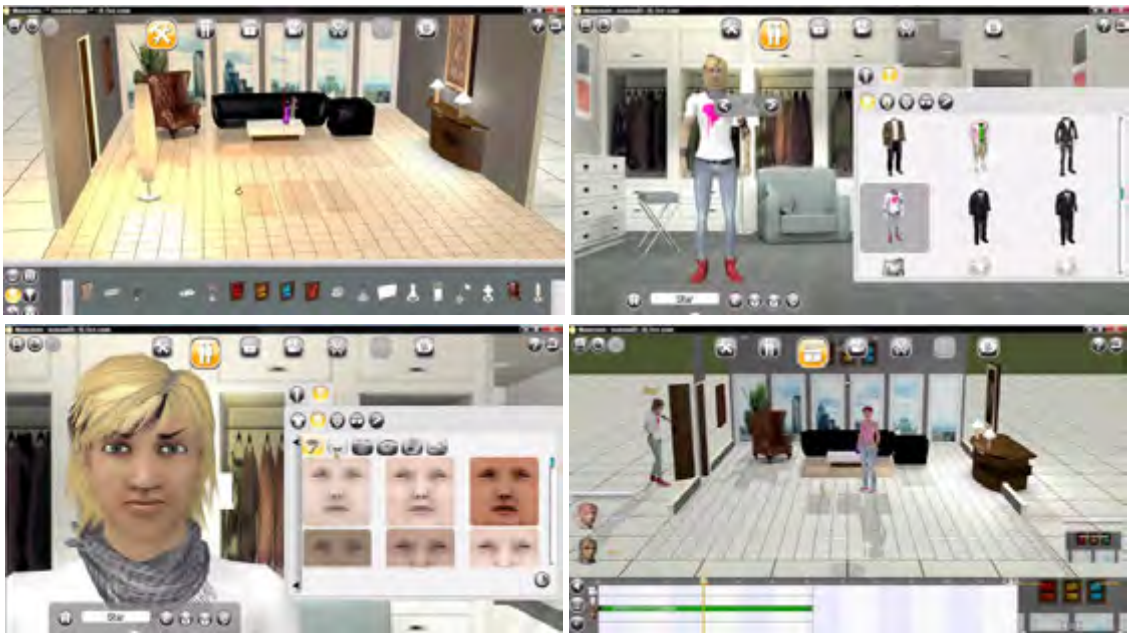


Figure 2.14: *Some of the core building blocks of creating a movie using Movie Storm.* ***Top left:*** *set building interface.* ***Top right:*** *setting the character's body type and clothing.* ***Bottom left:*** *customising the character's face and hairstyle.* ***Bottom right:*** *specifying the character on the left to walk to the other character, where the walking and door opening animations are automated.*

2.5.3 Source Filmmaker

Another powerful film creation tool that supports collaboration is the Source Filmmaker (SFM) by Valve [30], the creators of Steam [27]. The SFM repurposes the video game world into a virtual movie studio[123]. With this implementation the entire pipeline of an animation studio has been condensed onto a single gaming PC. The SFM combines the art of cinematography, film editing and animation to immediately reflect late project story changes. The character animation tools in the SFM allow one can to capture motion, insert clip motion and even create new animations from scratch. Regardless of how the animation was created, powerful motion editors can be used to modify the existing animations arbitrarily. It features GPU powered facial animation that allows for multiple characters to deliver subtle nuanced facial performances with feature film quality lip sync.

It should be explained how one would go about using such an unconventional approach to create an animated scene. Whereas on a typical film set the director would say *"lights ... camera ... action!"* with SFM it is actually in reverse: *"action ... camera ... light"*. The users first create the actor behaviour, then adjust the camera motion and finally fine-tune visual effects, such as lighting.

Figure 2.15 shows some of the steps required to perform a cutscene. One or more users can mock up an idea by playing through a scenery as actors in the virtual environment, and modify the recorded scene later on. This is the multi-user aspect, as all "players" are essentially the actors, where each one has to play their role in the virtual scene. This includes the steps of capturing motion, placing cameras, adding assets to the game world. Furthermore, one can re-perform a take, tune lighting, adjust sound, and perform GPU-powered facial animation for subtle performances with feature film quality lip sync, all without leaving the context of the game world.

While this tool contains many advanced features and sophisticated tools for editing scenes, it is a fairly simple process to create an initial scene. SFM's underlying use of gameplay to construct scenes makes it quite accessible, since no complicated user input is required to get a basic animation going. However, such an approach may quickly become confusing and less synchronised when multiple people are controlling characters in the same screen. This can become particularly bothersome when actors' paths cross, which could result in user's intertwining their hands and arms on a multitouch interface. However, one benefit of same-time, same-place CSCW (see Section 4.2) is that social cues can be used to prevent this, as each participant can see if a user is interacting with the scene. Naturally one would wait for a user to complete their tasks, before taking over an occluded or occupied part of the user interface.

Figure 2.15: *The SFM allows one or more users to capture gameplay* ***(top left)*** *and editing the recorded world in many was to create an animated scene. Features include GPU powered facial animation* ***(bottom left)*** *and editing the animation with powerful motion editors* ***(top and bottom right)***.

2.5.4 Frameforge

One previz system, Frameforge [12], is a tool that allows for very detailed planning of sets. The Control Room is a feature used to build sets, place cameras and capture storyboards. Another vital component is the Shot Manager, with which one can view and rearrange stored shots, add arrows to indicate movement and flow, and play the shot sequences as slideshows with full control over the duration of individual shots. It is renowned for its custom shot creation, which even caters for real-world film set conditions such as the spatial limitations of cinematographers and zoom thresholds of lenses. However, as the name suggests, Frameforge is more suitable for creating stills (non-animated frames) and story boards, rather than animated sequences. Frameforge's features suggest that it is more appropriate for live action film (involving human actors and real-world sets) rather than pure 3d animation. Such features include: specifying the camera's film/video frame size and aperture, and the control room(see Figure 2.16), which is laid out very similar to a real, multi-camera TV control room.

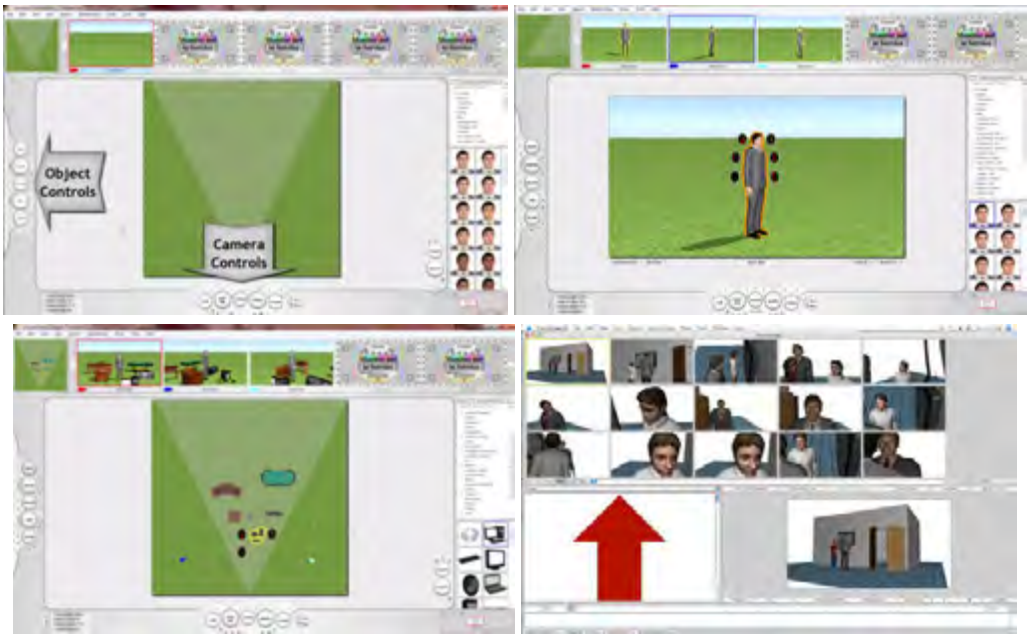


Figure 2.16: *Frame Forge's Control Room enables users to manage many aspects of the scene being previsualised.* ***left:*** *The controls for the cameras and objects reside here reside on the bottom and side panels, and the central region of the interface is the live preview that shows users the entire set they are filming in.* ***top right:*** *Object's can be placed from the object library on the right into the scene, using drag and drop.* ***bottom right:*** *The Shot Manager provides the ability to rearrange, delete and manage all shots taken for a particular scene.*

Chapter Summary

In this chapter we have presented the production life-cycle, focusing on the pre-production phase. This included a step by step procedure to pre-visualise a scene from the initial storyboarding all the way through to a digital animation. Previz is used to sign-off the concept, and is useful for quick and affordable changes before the full production phase begins. The following chapter covers more technical aspects of the underlying research and technology.

Chapter 3

Virtual Environments

This chapter provides a fundamental, technical understanding of the relevant research fields, which is essential for subsequent chapters. Although this research focuses on human computer interfaces (HCI), it is an intersection of several areas in computer science. A considerable amount of work exists in these fields, as shown in the following chapter. These research areas include navigation mechanisms in virtual environments, manipulating objects in these environments, and multitouch-based interaction. Other topics such as user-centred system's design (UCSD), underlying rules and guidelines for user interface design, and the supporting collaborative interaction are also presented.

3.1 Virtual Environments

Three-dimensional (3d), animated scenes take place in virtual environments (VEs). To elaborate, this involves a virtual camera imaging 3d actors and objects residing in a digital scenery (see Figure 3.1). This section explains fundamental concepts required to create and manipulate a 3d setting in a digital realm. The following subsections describe key components of Figure 3.1. The following subsections describe the key components of this figure.

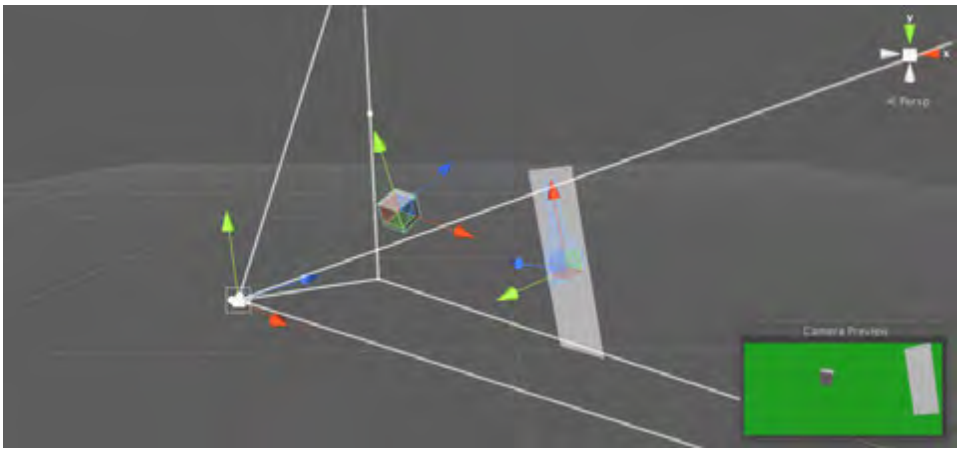


Figure 3.1: ***Coordinate systems*** *Depending on the purpose, a local or global coordinate system is more suitable.* ***a) Global:*** *The lower left intersection of the x, y & z axes is the origin of the global coordinate system. It is used for tasks like placing objects in scenes.* ***b & c) Local:*** *These coordinate systems are the local to the respective objects.*

3.1.1 Coordinate Spaces and Virtual Objects

A cartesian coordinate system is characteristic of 3d virtual environments. This coordinate system allows an object's position to be defined within a scene. The x, y and z components of an object specify its position in the global coordinate space relative to the environments origin, a fixed zero point. Additionally, the objects residing in the scene also have their own, local coordinate systems, which are always relative to their transform. A transform is used to describe an object's basic size and placement within an environment. Characteristic properties of a transform are its position, rotation and scale, which describe an object's placement, orientation and size respectively.

3.1.2 Viewpoint Manipulation

As previously stated, the environment is displayed on screen via a viewpoint. Reasonably unrestricted exploration of VEs requires arbitrary viewpoint manipulation. Several navigation mechanisms have become common place and some are more suitable than others, depending on the application in question. For instance, some applications [39, 16] use a double tap/click to bring an arbitrary point in the environment to the viewpoint's focus. Other applications, such as first-person perspective games, would instead require users to hold down an input key or tilt a joystick in a direction, continuously moving the viewpoint until the user has reached their desired location. Many desktop based applications (such as the UnrealEngine [56] and Unity3d [122]) use a combination of the keyboard and mouse for movement and orientation respectively.

While input mechanisms vary between devices, there are two typical methods of navigating a virtual environment, by mapping 2D input into one of two (possibly 3D) coordinate systems. With the one method, a user controls the viewport, i.e. the window displaying the virtual world on the screen from the perspective of the viewpoint. With the other method a user transforms the virtual world, while the viewpoint remains fixed.

3.1.3 Virtual Camera

In the context of virtual environments and films a camera is perhaps the most important component, because without it, one would not be able to display the scenery (see Figure 3.2). While there are many important aspects with regards to configurations in film cameras, one of the most important is the type of lens used. This subsection describes a part of a film camera's physical configuration, and how that relates to a virtual camera. Unlike a real film set's cameras, those used for virtual environments rely on slightly different properties which define the appearance of an observed scene. The configuration and parameters required to specify a 3d camera are illustrated in the figure below.

As discussed previously, all objects residing in 3D space have a transform associated with them, specifying their position, orientation and scale. The virtual camera is no different, as the resulting, rendered image depends on the camera's position and orientation. The scale is negligible, because the camera itself has no actually visible 3d data, such as a mesh, associated with it. Other properties that define the viewing space are the near and far clipping planes. These constrain the visibility of objects along the depth axis, allowing only objects/parts of objects between these planes to be processed by the camera. Then there is the field of view (FOV), which specifies how wide the viewing angle is. Using both the near and far distance in combination with the FOV one can construct the view frustum. The frustum, i.e. viewing volume, can be described as a rectangular prism. Everything within the view frustum is potentially visible by the camera, unless occluded by another object.

Additional properties, relative to the camera's orientation, are the forward, up and right vectors. These can be used for camera relative transformations, such as panning in a direction that is parallel to the camera plane.

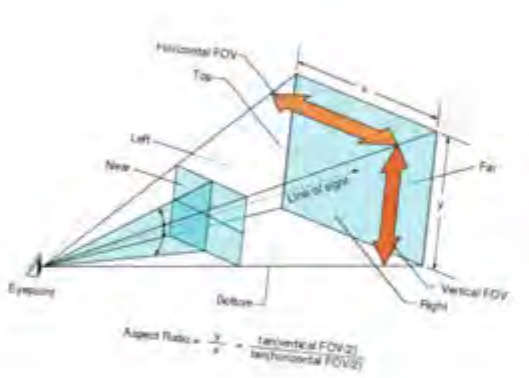


Figure 3.2: *Configuration and parameters required to specify a 3D camera*

3.2 Navigation of 3D Virtual Environments

In this section we cover systems with different methods of navigation through virtual environments. The target device of this study is a touch-table, which supports only two dimensional (2d) input on the touch display. For this reason we are going to consider systems that use 2d input to navigate through three dimensional (3d) virtual environments.

The initial usability principles were created for traditional computing devices, which is why they are not focused on characteristics unique to systems centred around virtual environments. This includes the design of navigation mechanisms and way finding, as well as how to select and manipulate objects, and integrating system feedback for a user's visual, auditory and haptic senses. Stanney et al. present a technique to better evaluate VEs, the Multi-criteria Assessment of Usability for Virtual Environments (MAUVE) system [117]. For a better user experience in VEs, designers need to enhance presence, immersion and system comfort, while taking care to minimise sickness and adverse aftereffects. The (MAUVE) technique categorises and integrates these VE attributes into a systematic approach for VE usability design and evaluation.

Davies presents the use of virtual reality adapted for participatory design of work environments [52]. This study shows that such systems work for small groups but require a VR/VE expert to drive the process in larger groups. Chow et al. evaluate navigation in a 3d VE using multitouch for the general public in participatory planning [51]. This research shows that multitouch may better engage participants, thereby improving their understanding of planning policies and proposed projects.

Benzina et al. introduced a one-handed navigation technique for VEs they call Phone-Based Motion Control [46]. This system uses a touch enabled mobile phone with integrated motion sensors as a 3d spatial input device. The input on the touch screen enables translation and using the orientation sensors allows for 3d reorientation. Each degree of freedom maps to a separate interaction technique, and this work examines how many degrees of freedom (DOF) are required for navigation through a virtual environment as easily as possible. Furthermore, Benzina et al. investigate different mapping functions between the user's actions and the viewpoint reactions in the virtual reality (VR). In this study four metaphors are developed for the steer-based rotation control technique. Each of them uses *Touch Translation* combined with the following:

Rotate by Heading

This approach uses the metaphor of walking or riding a bicycle. 4 DOF in the mobile device for translation in x and y and rotation using roll and pitch are mapped to 5 DOF in the Virtual environment: translation (X, -Z) and rotation (heading, roll, pitch) are shown in Figure 3.3(a).

Rotate by Roll

This approach uses the metaphor of an airplane. Here 4 DOF in translation (X, Y) and rotation

(roll, pitch) are mapped to the same 5 DOF as with *Rotate by Heading*, as depicted in Figure 3.3 (b)

Rotate by Roll with Fixed Horizon
This method differs from *Rotate by Roll* in that the horizon remains aligned horizontally. The same 4 DOF for rotation and translation are mapped to 4 DOF in the VE: translation (X, -Z) and rotation(heading and pitch), as shown in Figure 3.3 (c).

Merged Rotation
This technique combines Rotate by Heading and Rotate by Roll. Here 5 DOF (translation {X ,Y}, and rotation {heading, roll, pitch}) are mapped to 4 DOF in the VE: translation (X, -Z) and rotation (heading, roll, pitch), as depicted in Figure 3.3 (d).

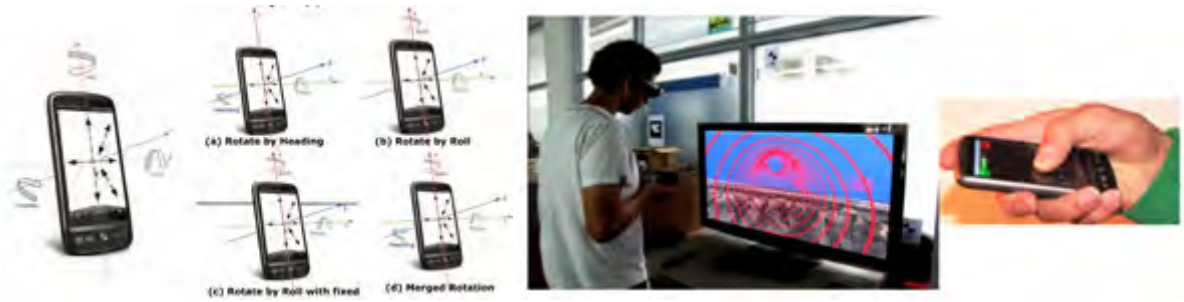


Figure 3.3: ***(left:)*** *An illustration of the four metaphors for navigating a virtual environment using a mobile device.* ***(right:)*** *A photo of someone navigating the virtual environment using this system.*

The only drawback with this approach, compared to the system we are developing, is that it requires more than one device for input. With n number of users, one will need n number of mobile devices for a collaborative setup, to avoid passing around the navigation device for a more efficient workflow.

3.3 Object Manipulation in 3D Environments

In order to construct a 3d animated scene, users will be required to manipulate objects in 3d space. There has to be sufficient freedom for arbitrary object manipulation, in order to construct scenes as desired by the director. The object manipulation is, however, not completely arbitrary, as characters' geometry cannot be deformed, for instance.

Object manipulation in 3d environments is a very matured research field. Shoemake presented the effectiveness of the Arcball, which uses quaternions for virtual trackballs [113]. Bade et al. evaluated four different methods to achieve 3d rotation using a mouse [42]. This included the virtual trackball mechanism of Shoemake [113], and two adaptations of the Two Axis Valuator [50]. The Two Axis Valuator was superior for rotation according to their user study. Both studies only required 2DOF rotation control. Kratz et el. [79] used a trackball for 3D rotations and how this can be applied to touch input. Zhao et al. [129] presented tasks requiring full 3d rotation control. Hinckley et al. [71] compared 2d trackball rotation and full 3d input. This user study presented the issue of 3d input for rotation, due to the lack of tactile feedback, making users less aware of the orientation of the object. Scheurich and Stuerzlinger [107] presented a method to rotate objects in 3d, designed specifically for one-handed interactions on tablets and touch screens. Their results suggested that a one-handed rotation technique improves both the accuracy and speed of 3d rotation techniques.

Kruger et al. [80] designed an interaction mechanism that allows integrated rotation and translation. They term this mechanism "Rotate'N Translate" (RNT) as it provides combined control of translation and rotation, using only one finger. One motivation of their approach is that rotation

and translation are inseparable in the real world. The metaphor used to describe RNT is that of a stream against which an object is moved, but the behaviour can also be described with a sheet of paper being moved across a table with friction, and is illustrated in Figure 3.4. When commencing a pull gesture from the central region of the sheet, the object will follow the finger and translate to the updated position continuously, without being reoriented. The central region is termed the "translation only region". Initiating the same interaction from a point further away from the centre and not in line with the direction of motion, will cause rotation. What makes this particularly useful for this research is that only a single finger is required to perform rotation and translation, without the need to explicitly switch interaction modes. The empirical results obtained in the testing showed that the integrated rotation and translation is more efficient (as quick to do) than a separation of the two, which is was termed "traditional moded" or TM. The system was evaluated with three tasks: precision targeting, document passing, and collaborative document passing. Users found RNT more learnable and preferred using RNT over TM for the second and third tasks.

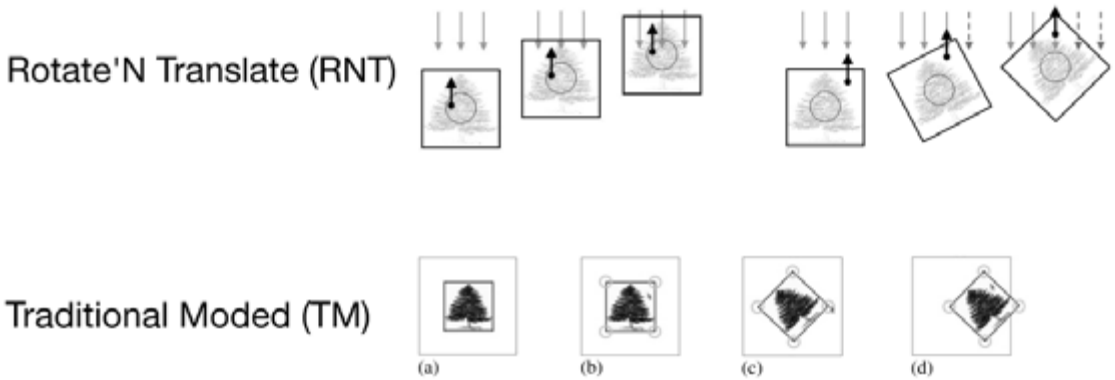


Figure 3.4: ***Rotate'N Translate (RNT)****: When an object is manipulated it can either only be translated(left), only rotated or both rotated and translated (right), depending on the initial contact point and the subsequent direction of motion.* ***Traditional-Moded (TM)****: An object can be rotated on the spot when grabbed from the corners and it can be moved if the contact point begins anywhere within the object's area.* ***(a)*** *object is selected through a contact point* ***(b)*** *object is rotated from corner* ***(c)*** *interaction is released* ***(d)*** *object is translated*

Another, more relevant approach for touch-based, 3d object manipulation is the *Z-technique* [86]. This is in contrast to the common technique of utilising multiple viewports (see Section 5.2.2), each viewing the scene along one of the global x, y or z axes, where the fourth viewport presents an arbitrary viewpoint. The other three fixed perspectives are from the front, side-on and top-down perspectives, which are often orthographicBeing able to move objects in any of these 3 fixed perspective viewports ultimately allows for precise, arbitrary object placement. On the other hand, the *Z-technique* uses a single viewport, where two fingers allow for three degrees of freedom. Touching and dragging an object moves it parallel to the screen in the x-y plane. Adding an additional finger and dragging it up or down moves the object along the depth, (i.e. z-axis) as depicted in Figure 3.5.

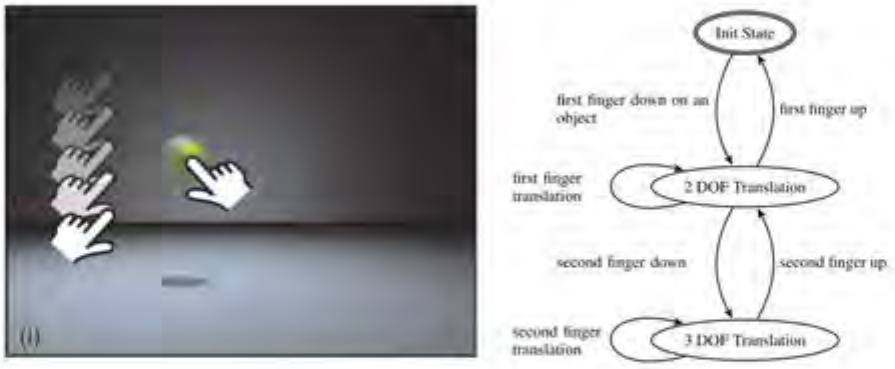


Figure 3.5: *The Z-technique allows for 3 degrees of freedom with a combination of either one or two fingers.* ***(left)*** *An object can be moved along the x-y-plane with one finger, and adding a second finger allows displacement of the object along the z-axis.* ***(right)*** *This state machine illustrates how the number of touch points affect the degrees of freedom.*

Hancock et al. present tabletop interaction techniques to provide control of all kinds of 3d rotation coupled with translation [66]. Their research involves creating three different touch techniques, each created with a tradeoff of simplicity against accuracy.

One-touch input achieves 5DOF by extending the RNT algorithm [80] into the third dimension. Rather than rotating explicitly about the axes (roll, pitch and yaw), the point of contact determines the axis of rotation. Although this technique allows rotating and translating a 3D object to any position and orientation, users often want to perform more restricted rotation and translation. Such constrained transformation can be achieved through dedicated areas for a polygonal object, and for non-polygonal objects a more central area on the surface of an object can be chosen.

Two-touch input can achieve 5DOF or 6DOF, where the first touch point can use the RNT algorithm [80] for translation in x and y, as well as yaw. The second touch point is used to specify pitch and yaw. To move along the z-axis, one can perform a pinch gesture.

Three-touch input can achieve 5DOF or 6DOF, using 6DOF as input. Here the first touch point is used for translation, the second point to specify yaw about the first point, and the third point specifies pitch and yaw about the first point. The order of the touch points is determined either by the order in which they make contact with the device, or in a predefined order, should the source of each touch point be identifiable.

Often systems provide elegant solutions, which are quite robust. The drawback is that these do not always cater for multiple users. In particular, bimanual (two handed) interaction techniques [44] and interaction using 3d input are not suitable for this study, as the aim here is to use a single device and cater for input from as many users as possible. The next section covers applications using touch-tables for multi-user participation.

Chapter Summary

This chapter covered the fundamentals of object manipulation and navigation in 3D virtual environments. This included the basic theory, terminology and common practices for interacting in these environments. The chapter was concluded with previous work, specifically covering multitouch based interaction to interact with virtual environments. The following chapter covers previous work and literature in CSCW and collaboration using multitouch.

Chapter 4

CSCW & Multitouch

This chapter provides a fundamental, technical understanding of the relevant research fields, which is essential for subsequent chapters. Although this research focuses on human computer interfaces (HCI), it is an intersection of several areas in computer science. A considerable amount of work exists in these fields, as shown in the following chapter. These research areas include navigation mechanisms in virtual environments, manipulating objects in these environments, and multitouch-based interaction. Other topics such as user-centred system design (UCSD), underlying rules and guidelines for user interface design, and the supporting collaborative interaction are also presented.

4.1 Touch-based, Collaborative Systems

4.1.1 Large, Multitouch-enabled Devices

It is important to note that no user interface is "natural" [96] as the interaction techniques to use interfaces have to be learned. Some interaction methods might be more intuitive to some users than others, depending on their cultural background. One of the implications of a "natural" interface is one that is easy to learn and use. Occasionally when interaction designers create new systems, familiar gestures from similar systems are used in order to reduce the learning curve of the new system, and effectively increasing it's learnability.

In recent years touch-based devices, such as smartphones and tablets, have become very common consumer goods. It is very common to see people using touch-based devices for personal or professional purposes on most parts of the world. Even touch tables have been around for several years (e.g. as early as in1982 at the University of Toronto [88]), although these were not in the commercial price range until recently. At this point they are becoming more readily available in the consumer market. The steadily growing familiarity of touch-based devices may suggest this type of interaction is becoming more intuitive. This project aims to create a system to support collaborative work, which is discussed more in the following section. A touch table is a suitable device for collaborative work, because the device can physically cater for multiple users interacting with it simultaneously, in particular because of the device's size and multitouch support.

The proliferation of touch-based interaction gave birth to a variety of multi-touch enabled devices. Multitouch Walls [73] are vertically oriented, multitouch-enabled displays, as those shown on the left of Figure 4.1. An advanced variant of touch walls is through-window touch [124], enabling users positioned on one side of a glass window to interact with a touch wall positioned on the other side of the glass.

Figure 4.1: *Varieties of multitouch devices are touch walls and touch scrolls*

4.1.2 Multitouch-based Interaction Design

One issue with multitouch based interaction, particularly on large table-top touch surfaces, is the lack of standardisation for touch techniques[85, 74]. It is true that this area of computer science is rapidly evolving, to the point that major competitors in the mobile smart phone market have arrived at similar solutions for touch based interaction. Nevertheless, most day-to-day applications present on smart phones deal with similar interactions, mostly to do with 2-dimensional content, such as email, web browsing and document management.

3D Tabletop Interaction Design Guidelines As tabletop studies continue to progress, corresponding interaction design guidelines become more precise. Below are some of the existing design guidelines, presented in another study by Hancock et al.[66] (pp. 1148–1149):

- **provide more degrees of freedom:** various tasks users would like to accomplish require full rotation and translation in three dimensions.
  - *simultaneity of rotation and translation:* in the real world people are capable of rotating and translating objects simultaneously. To make the experience on a tabletop application as natural as possible, the system should allow for simultaneous rotation and translation.
  - *independence of rotation and translation:* while the above holds true, users may also want to perform rotations or translation separately from each other
- **provide connected manipulation:** this is a form of direct manipulation in which the user maintains a visual and physical connection with the object in question throughout the lifetime of the interaction. To elaborate, the object must not only be visible to the user but it should also be in direct contact with the interaction point throughout the interaction.
- **prevent cognitive disconnect:** actions that do not conform to what users expect, should be avoided. If a user expects to pick an 3D object from a scene that is clearly further away along the depth axis than other objects, and most certainly the GUI (which is never occluded by any 3D objects), than the user should be able to do so.
- **provide appropriate 3D visual feedback**: 3D visual cues can make interactions more intuitive, as they relate to the physical world quite closely and are therefore more familiar.
  - *provide appropriate shading:* since tabletops allow interaction from any side of the device, there is potential for misinterpretation of shapes and objects. This can be caused by users recognising convexity and concavity incorrectly, as they may expect the lighting to come from above, behind their shoulder.

– *consider parallax:* with changing depth along the z-axis a single perspective projection of the viewpoint can appear differently to users at different locations of the touch table. This is also true for a single user, where parallax can occur horizontally across the display due to its size and the user's location.

4.2 Computer Supported Cooperative Work

The collaborative focus of this system places it in the research domain of Computer Supported Cooperative Work (CSCW). This is a field which has been around for decades, the first workshop having taken place in 1984 by Irene Greif and Paul Cashman [108], and after several conferences the CSCW Journal appeared in 1992. As indicated by the name of the journal, CSCW is all about collaboration, encompassing many different areas of research. It follows that best practices and guidelines have been formulated over the years of this journal. However, these guidelines vary depending on the problem being tackled. A focused set of design guidelines for this particular research is covered in Section 5.4 of the collaboration design chapter.

4.2.1 Concepts of CSCW

CSCW systems relate functional features with the social aspects of teamwork [102]. The functionality impacts work behaviour and influences the behaviour of individual group members, as well as the efficiency of the entire group using the system. The success factor of a CSCW system can be determined by the *interaction, coordination, distribution, user-specific reactions, visualisation, and data hiding.* Depending on the CSCW environment, the interaction requires different response times and needs to proceed either synchronously or asynchronously. Joint editing of the same data by multiple users is done synchronously, whereas sending a message is an asynchronous task. It all depends on how dependent one part of the data is from another. *Coordination* deals with the group members communication, which varies with the size of the group and how the individual members prefer to interact. Larger groups require more coordination than smaller ones. With a brainstorming or design system, users are more likely to communicate spontaneously, whereas a conference setting requires one user to take the lead. *Distribution* deals with the (geographic) location of the involved users, and how the a system has to be adjusted to cater for this. A distributed working environment requires additional transfer channels for explicit interaction involving gestures and speech, which may differ when users from different continents and cultures are involved. Apart from technical challenges, global distribution of cooperating partners not only crosses time zones, but also suffers from differences in social, cultural and politics (e.g. language, negotiation strategies, behaviours, etc.). CSCW systems without *user-specific reactions* are *collaboration transparent*; for joint work this type of system lets users work with a familiar user interface and access existing application programs. *Collaboration aware* CSCW systems "know" the number of users and their individual roles, and are developed as multiuser applications to support collaboration. *Visualisation,* the what-you-see-is-what-I-see (WYSIWIS) paradigm [118], determines how data is visualised and used in a collaborative setting. Collaboration transparent systems are strongly related to WYSIWIS, but there are several degrees of this paradigm. A more relaxed version of WYSIWIS in a collaborative context is where users can see "different sides of the same cube" at the same time, which would be catered for by the CSCW system, e.g. with user-specific viewports. Data hiding deals with the separation of private and public data, where public data should be accessible to certain people, groups or the general public.

4.2.2 CSCW System Classifications

Depending on the requirements of a collaborative tool, there are distinctions in the CSCW system design: the form of interaction (synchronous and asynchronous), and the geographical nature of the users (remote versus co-located) [103]. Different combinations of these usage setups give rise to four

CSCW classifications of CSCW systems: *Message Systems, Computer Conferencing, Meeting Rooms,* and *Co-Authoring and Argumentation Systems,* as depicted in Figure *4.2.*

Message Systems enable cooperation by exchanging messages, which is required for asynchronous and remote collaboration. A client-server architecture is applicable for Message Systems.

Computer Conferencing was first envisaged in the early 1970s and developed independently from Message Systems. With this classification of CSCW system interaction with the shared information space can be synchronous in real-time or asynchronous over long time periods. While in the early 70s textual information was mostly used, Computer Conferencing is not limited to this type of information. One can also share digital design work, multimedia such as video and music, as well as computer graphics work with this system classification.

Typical automated meeting rooms are composed of a conference room that contains a large screen video projector, a computer (or a network of computers), video terminals, a number of individual input/voting terminals, and a control terminal. The devices managing the meeting and overall collaboration space, usually requires multi-user software based on a form of analytical decision technique. Software for graphics, vote tally and display also are also required to facilitate the meeting.

Co-Authoring and Argumentation systems generally aim to support and represent the negotiation and argumentation involved when working in a group. Cooperative authoring of documents is indicative of this type of collaboration, in which the final iteration of documents represents the outcome of a process of negotiation between authors.

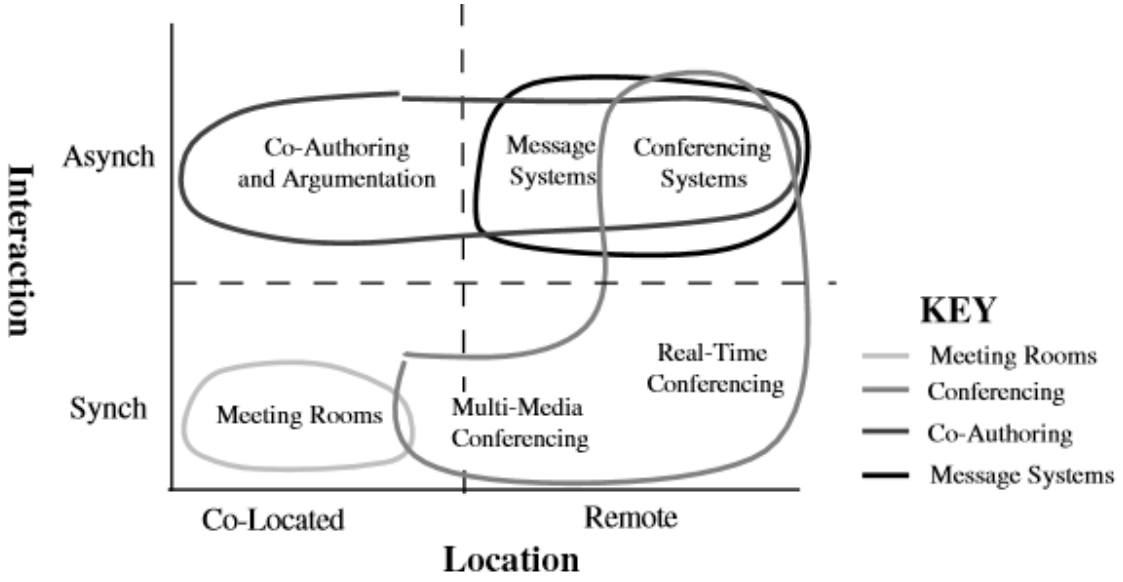


Figure 4.2: *On a high level, CSCW systems can be defined by the* ***form of interaction*** *(asynchronous and synchronous) and the* ***geographical nature of the users*** *(remote versus co-located). This gives rise to four classifications for CSCW systems:* ***Message Systems, Computer Conferencing, Meeting Rooms, and Co-Authoring and Argumentation Systems.***

4.2.3 Previous Work in CSCW

A touch table can be a suitable device for collaboration, as it is large enough to accommodate and process the input of multiple users. One form of collaborative interaction involves the aid of user tokens. These are tracked by the touchscreen and often used to identify individual users. Nevertheless, there are still several measures required, in order to cater for collaboration of multiple users. There are multipole applications supporting collaboration using multitouch, such as Futura [38], which drives

collaborative learning and game play on a touch table, and CRISTAL [109], a home media and device controller. This sections takes touch table based systems into consideration, which are designed for multi-user inputs.

DiamondSpin [111] is a toolkit for multi-user interaction that has been around since the early 2000s. This toolkit provides an API for developers to accomplish several tasks required for tabletop collaboration. Such tasks include visual document management, document control and interaction, manipulation, rotational UI, digital tabletop layout, and multi-user support. Several multi-user applications have been created with the aid of DiamondSpin, proving it's effectiveness. Nevertheless, one drawback of the framework is that it is not very suited to 3d interaction, but rather 2d document management, etc..

The Innovim model is a tabletop-based document management application designed for multiuser interaction as a means of collaboration [83]. Some of the key interface components included are cards, a fan menu and a toolbar. Cards are workspaces used for organising knowledge. A card can contain a variety of objects, including text, images and even free hand drawings. Cards also support hierarchy, in that any object can be the ancestor of another object, via a link. Furthermore, one can write plugins to extend the supported type of objects. Fan menus, as shown in Figure 4.3 a), can be brought up by touching an empty space of a card, although some contiguous space is ignored to avoid overlap with existent menus. Depending on the chosen action, the fan menu will either show a sub menu(e.g. to select a picture) or disappear (for freehand drawing). The toolbar is used to for additional functions relating to the selected card, and it is positioned at the bottom right of every object. There are generic functions such as undo, redo, and deleting the card. Context sensitive actions also exist for the type of object. One context sensitive action is for the freehand drawing object, as the drawing gesture would otherwise move the card around. Lastly, certain gestures exist to perform some of the actions on the toolbar, as depicted in Figure 4.3. One feature particularly worth noting is how the undo functionality is integrated. With a user-blind system, i.e. one that is unaware of which tasks are performed by which user, an undo stack is kept for each object. This implementation of the undo stack, as opposed to a global stack, is a necessity in a multiuser context, as it that prevents users from interfering with each other's interactions. However, Innovim is more useful for document management, than VE creation. Despite this, concepts like the context specific controls for each card type can be applied in a collaborative previsualization tool.

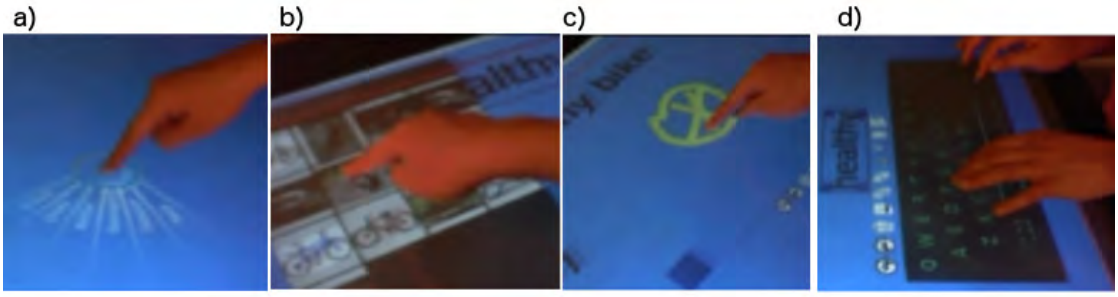


Figure 4.3: *The Innovim model, uses cards to host information (objects) such a text, images and freehand drawings. The multitouch interface supports multiple users and combines gestures with interface elements to enhance collaboration, which include:* ***a)*** *fan menu,* ***b)*** *image selection,* ***c)*** *freehand drawing,* ***d)*** *text input.*

Bowers did a field study of procurement, implementation and use of a local area network (LAN), which was purposed to run CSCW applications in an organisation (the U.K.'s central government) [48]. During the study, the network ran into a number of difficulties: it was resisted by its potential users for a variety of reasons, it was faced with being withdrawn from service on a number of occasions, and it remained only partly used at the time of writing. Consequently, the study had several findings:

the kinds of problems a project, purposed to introduce CSCW to a real-world organisation, is likely to face, a series of concepts to help manage the complexity of these problems, and effectively adding to and extending previous studies in this field.

Despite all the work in CSCW [108], only very few such systems are widely used. This is particularly true for working groups across multiple organisations, where the deployment of CSCW technologies is hampered by problems of heterogeneity in computing hardware and software [45]. Bentley et al. present the BSCW (Basic Support for Cooperative Work) shared workspace system, which is an extension to the Web architecture, providing basic facilities for collaborative information sharing from unmodified web browsers. The conclusion of this work is that building in the strengths of the Web can give significantly benefit developers, making it easier to develop and deploy CSCW applications. Unfortunately, the variety of applications that can be built on the Web is limited, particularly, for applications that deal with creating and receiving large assets, such as 3D models.

All the mentioned considerations, configurations and suggestions for developing a CSCW are considered for the system created in this research. The decision for creating the final collaboration setup are motivated in the collaboration design chapter.

4.3 User-Centred System Design

A commonly used design philosophy for developing user-based systems is user-centred system design (UCSD) [60, 95]. It is a process consisting of iterative improvement cycles, where the developer(s) analyse a new set of requirements from the users/clients, upon which they implement (at the very least a subset of) the required features and functionality into the system. Such an iteration is usually concluded with a presentation of the new features to the client, followed by an updated list of requirements that is formulated by the client(s). At this point the process is repeated, as indicated in Figure 4.4, where the developers keep collaborating with the clients to improve the system, until the client is satisfied with the result, or one of many other possible terminating conditions has been met, e.g. the exhaustion of funds or cancellation of the project.

For a user-based system it is absolutely necessary to perform user testing in order to assess the system's usability. The usability of a system is defined by five quality components: the ease of use, learnability, satisfaction, memorability, and recovery from errors [93]. This relates to the layout and graphical elements, as well as interaction techniques and system flow, which create the overall user experience. User testing is also important, because it can easily and quickly reveal bugs in systems, which may not have been apparent to the developers. A commonly practiced set of usability design principles guidelines is listed in Appendix A and summarised in the section below.

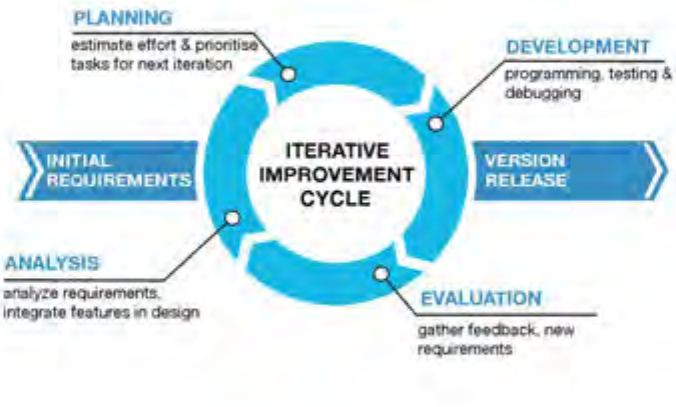


Figure 4.4: *This is a diagram illustrating the iterative improvement cycle. This is a process going through stages of planning, development, evaluation and analysis in several iterations, where each one aims to improve the system being developed.*

4.3.1 Usability Design Principles & Guidelines

Usability is the ease of use and learnability of a man made object. A heuristic is a rule (or a set of rules) intended to increase the probability of solving a problem. Usability heuristics can be understood as guidelines that help to create an interface with good usability features. This section elaborates the importance of usability in designing interfaces and provides guidelines for best practices. A user interface is called an interface, because it is the intermediary between the user and the content being interacted with. It follows that a good user interface makes it easy to accomplish any possible task within the given system. Nielsen's ten usability heuristics [91] are conventional guidelines that help to create a sensible, user-centred system.

Nielsen's Ten Usability Heuristics

A widely accepted and good set of design principles for user interfaces are Jacob Nielsen's Ten Usability Heuristics [1], outlined below. The actual heuristics are in bold, and each is explained with an annotation below it.

- ***Visibility of System Status***

When interacting with a system, users should always be aware of what is going on. This is achieved with suitable feedback within reasonable time.

- ***Match between system and the real world***

The system's use of language should be comprehensive to the user, rather than being system-oriented. This can be achieved by making information appear in a logical and natural order by following real-world conventions.

- ***User control and freedom***

It is common for user's to accidentally choose undesired system functions. In such cases, users should be able to recover quickly from an unwanted state, without having to go through extended dialogs. Undo and redo functionality should be supported.

- ***Consistency and standards***

It should be clear to the user what a particular phrase, word or situation means. By following platform conventions, users would not have to wonder if different terms mean the same thing.

- ***Error prevention***

While it is good to inform users when an error has occurred, it is better to prevent a problem from occurring at all. Error-prone conditions should either be eliminated or checked for, in order to provide users with a confirmation option before they perform the action.

- ***Recognition rather than recall***

Users should not have to remember much information when using a system. Instead, usage instructions should either be clearly visible or easily retrievable whenever appropriate.

- ***Flexibility and efficiency of use***

Novice users often take the obvious, longer route to solve a problem, whereas experienced users may want to use shortcuts to accelerate the interaction.

- ***Aesthetics and minimalist design***

Information that is irrelevant or rarely needed, should not be present in dialogues. Any extra piece of information can distract users and may reduce their attention to the significant bits information. Additionally, aesthetics tend to improve results in usability testing. If the interface looks nice, people become more creative and solve more problems.

- ***Help users recognize, diagnose and recover from errors***

When an error occurs, it should be stated in a manner that is understandable to the user, indicating the problem explicitly and suggesting a solution.

- ***Help and documentation***

A good user interface would not have to be accompanied by a documentation of its usage. Nonetheless, in case of uncertainty a user should always be able to quickly look up a specific task, without having to do much reading (unless the task is not a simple one to solve, of course).

Design Criteria

In addition to the aforementioned usability heuristics, the following are a list of criteria which aid in designing and implementing efficient, user-friendly interface. Some of key points for designing usable interfaces are discussed in Jones and Marsden's Book "Mobile Interaction Design" [75], and are summarised below.

- ***Affordances*** of an object are properties of the object which give users clues as to how the device is used.
- ***Mapping*** is concerned with ensuring that there is a natural correlation between objects and the interface controlling them.
- ***Constraints*** on a design are made ensure that it can only be used the correct way.
- ***Visualizing*** ensures that features are made visible to the user. Bad examples would be command line interfaces, whereas good examples are graphical menu system, presenting all possible actions.
- ***Simplicity*** is often said to be good. While simpler is always better, things should not be made too simple though.
- ***Consistency*** is vital in interface design. To increase the usability of an interface, it is best to use interface elements that users are already familiar with from other interfaces.

4.3.2 Evaluation Methods

As with all systems, CSCW needs to be evaluated in order to determine the success of the created system. Common evaluation methods in the CSCW literature in the context of computer science, including heuristic evaluations, user testing, lab experiments, interviews and questionnaires, focus groups and customer feedback, longitudinal trials and semi-realistic ethnography [9]. Depending on the circumstances in conducting the study and nature of the research, a variety of methods may be applicable. The chosen evaluation method is detailed and motivated in Section 7.2.

Chapter Summary

In this chapter we have presented the fundamentals of the research fields relevant to this study. This included the concept and applications of previsualisation to film and animation, virtual environment navigation and object manipulation and multitouch interaction. The process of user-centred system's design, as well as underlying rules and guidelines for user interface design, and the support for collaboration have also been covered. In the following chapter we take a closer look at previous work in these fields to gain an understanding of which approaches and techniques were effective, and how they are, or are not, applicable to this research.

Part II

Design & Implementation

Chapter 5

Collaboration Design

While the previous chapter covered previous work, related to this study, this chapter presents different approaches to support a collaborative work environment. When presenting the various approaches, it is also discussed how they are or are not suitable for the system we created.

5.1 Design Goals

To successfully support multitouch-based collaboration, a number of goals have to be met. This section lists the goals and presents a design aimed at accomplishing them. In this chapter an overarching collaboration design is presented. Thereafter, the user interface and interaction design chapter provide a more detailed design of the individual system components. The extent to which these design goals have been met is discussed in the evaluation chapter. The design goals are as follows:

A) The system's functionality and interaction strategies are quick to learn: Once an interface's functionality is understood, it becomes transparent, making it easier for a user to accomplish a desired task.

***B) There is little to no confusion about which user is controlling which object*(s):** In a multi-user environment, reduced confusion between users and their actions allows for more successful collaboration.

C) Viewpoint manipulation is easy and intuitive: Arbitrary virtual environment navigation needs to be easy, such that desired tasks can be accomplished quickly.

D) Render camera manipulation is easy and intuitive: Controlling the camera that shows the final animation needs to be easy in order to produce the desired shot sequences effortlessly.

E) Object manipulation is easy and intuitive: Objects in the scene need to be manipulated as desired by the users, as the director wants to use the time as wisely as possible.

***F) The feedback for all interactions is sufficient and unambiguous*:** Users should never have to question whether any interaction was registered by the system.

***G) The overall system is easy to use*:** This will make it more accessible for groups of users collaborating in animation, and possible in other, related professions.

This research focuses on the design of a previz tool that supports multi-user interaction on a single device. The idea of this setup is to promote collaboration, as the users are physically gathered around the same device. Several factors need to be considered for a successful multi-user interface. One of these is the number of users able to interact with the scene simultaneously. Due to the limitations of a 27 inch display[82], this system only caters for small groups of three to four users at any given time. While the device can only track up to 10 simultaneous touch points, this is less of a limiting factor, as this number is likely to increase with advancing technology. Apart from that limitation, larger groups

could experience a bottleneck with the input device, requiring an expert user to perform their tasks For instance, if a large amount of users are interacting at the same time, the amount of widgets would occlude a large portion of the scene. In the case of a single user interacting with the system, a desktop environment is suitable, for a variety of previz software already exist, as discussed in previous chapters. However, this type of setup could support collaborative aspects of pre-visualisation.

5.2 Alternatives for Cooperative Support

Before proceeding with the general guidelines used for the collaboration design, a few alternatives are listed with motivations for why these approaches were not chosen. This includes networked collaboration, which can be impacted negatively by networked collaboration. Multiple mice on a single device could work, but may cause confusion regarding which user is performing which interaction on the screen, since object manipulation is done indirectly via a mouse. With multiple tablets, that are synchronised in a local network, users would each have their own view of the environment, and their own set of controls.

5.2.1 Networked Collaboration

One possible option for a collaborative system, incorporating a shared environment, is to include networking as a means of collaboration. With this approach each user can have their own set of controls and again it needs to be made clear which user is engaging with which scene object(s). One major advantage of sharing a virtual scene over a network is that users need not be in the same room or even on the same continent to work together, effectively nullifying any geographical limitations. Furthermore, with such an approach, the number of simultaneous users is limited by network quality rather than the hardware. Users should also be able to communicate verbally rather than in writing, as this is a quicker and more natural form of communication. Verbal communication is also to be preferred, because users' hands may be occupied with the user interface or interacting with scene objects. Nevertheless, the networked approach would require each participant to have a network connection fast enough to allow negligible latency for fundamental tasks. These tasks include scene interactions and, ideally, verbal communication between participants for a more responsive interaction experience. A large amount of 3d geometry files is not really an issue regarding data transmission, since the data could be cached locally. Ultimately, the face-to-face interaction is better suited to collaboration and is unaffected by network limitations.

Figure 5.1: *Two variations of networking as a means of collaboration.* ***left)*** *a peer-to-peer configuration,* ***right)*** *a client-server network architecture for collaboration.*

5.2.2 Viewports

One frequently considered option for collaboration in this context is user-specific viewports. Modelling and animation packages[41, 40], as well as game engines[56], often make use of viewports for fixed axes of the environment. In utilising viewports for collaboration, each participant is assigned a sub-portion of the screen that encompasses their own perspective of the scenery, accompanied by a set of controls to accomplish all required tasks.

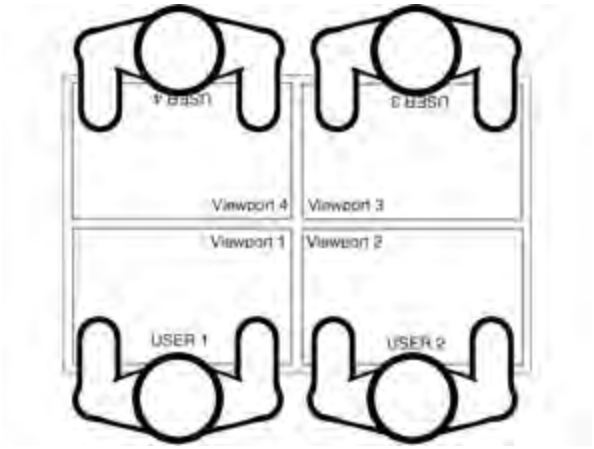


Figure 5.2: *With a shared multitouch-based device, users each have their own viewport depicting a representation of the scene. Each viewport also contains all controls a user requires.*

The advantage here is the set of controls localised for each user, oriented to suit the particular user. Thus one can then interact with any object in the scene as required. However, unless clearly indicated by the interface, it may not be obvious which scene objects are available for interaction. In the case where objects are engaged by users feedback as to which user is interacting with which object would be required. We aim to provide a collaborative experience not only to support multi-user interaction, but also to ensure that all users are in agreement about occurrences in the scene, minimising any confusion. In contrast, with a single viewport shared by multiple users it is immediately clear which user is interacting with which object, because they are directly interacting with it. Therefore it was decided to use a single, shared viewport instead of user-specific viewports.

5.2.3 Multi-mouse Collaboration

These considerations led to the choice of a single device, without networking, using a shared perspective of the virtual environment. It is true that large, multitouch-enabled devices are not very common in the animation industry at the time of this research. Hansen et. al did a study that compared of multitouch and multi-mouse for visual tasks that required coordination and collaboration [68]. They found that participants were more efficient with the multi-mouse setup, although they preferred the multitouch tabletop. Hansen's et al. used their study as a platform for discussing how to interpret results from studies that compare an exciting technology to one that is not. PC's are much more common in many industries. Therefore a low cost option would be to use a single PC with multi-mouse input, where each user is assigned a mouse. However, certain issues arise: Firstly, it can be difficult to determine which mouse pointer is controlled by which user. Even if the pointers were colour coded, this would require users to remember the colour associated with each user, detracting from focus required for accomplishing the previz task. Secondly, a more direct method of engaging with 3d objects may be

more intuitive, when considering fundamental tasks such as object rotation and placement. With touch-based interactions, users are physically in the same room and directly interacting with objects on the display, therefore there is no doubt about which user is engaging with any object.

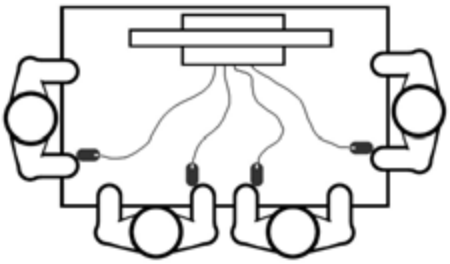


Figure 5.3: *A single-computer-multiple-mouse setup, where each user is assigned their own mouse and all mice share the same desktop environment for collaboration.*

5.2.4 Touch-based Collaboration

Multitouch-based interaction has been argued as the preferred choice for collaboration in this research. There are several options to consider, such as touch tables, tablets and touch walls. A single, average-sized with a roughly ten inch display is not suitable for multi-user input, as there is only a small amount of screen real estate to be shared amongst multiple users. However, each user could have their own tablet in and collaboration could occur in a virtual environment synced for on all participants' tablets, via a local network. This is effectively a similar setup to the iPad's Garage band app [13]. In this approach each participant in a group of users plays a single instrument on their own iPad, and the combined effort produces a song containing all instruments [19]. Nevertheless, this would re-introduce the multi-viewport approach already discarded previously. Furthermore, for the multi-tablet approach with a synchronised virtual environment, a networking component would also have to be constructed, as the devices would have to communicate with each other. Thus a large multitouch enabled device is the desired choice.

Figure 5.4: *Multitouch-based collaboration alternatives. Formula D Interactive's Touch wall* ***(left) [58]*** *for interacting with others on the same device, and tablets* ***(right)*** *syncing with each other via a network, in this case using the iOS GarageBand app [14].*

Smaller animation studios may have very low budgets, making a cost effective solution desirable. Catering for four users requires four tablets, such as iPads, which can be more expensive than a single device, such as the Lenovo IdeaCentre A720. An iPad is priced at around $500, and the A720 costs $1449, all prices being at the time of writing. Supporting three users in either setup would cost around the same, however, with the A720 there would be much more screen space available. The table PC

could even accommodate a fourth user, comfortably, which is true for the multi iPad setup as well, but requires and additional $500. For small groups the touch table is suitable, whereas the iPad solution may be more applicable for larger groups as it can scale with the number of users.

Another option is touch wall as depicted in figure 5.4. Touch walls have been used for various applications, but suffer from certain drawbacks. The major flaw is the gorilla arm effect [70], in which a device cannot be used for long periods of time, as one would need the arm strength of a gorilla in order to keep their arm extended out for lengthy durations. In addition to that, a touch wall of a certain size could accommodate fewer users than a touch table of the same size. Users can sit around a touch table on all edges (i.e. along the perimeter) of the device, whereas with a touch wall, there is only about enough space for people along the one width (one side) of the touch wall.

Scott et. al present system guidelines for co-located, collaborative work on a Tabletop Display [28]. It is stated in their that the technology must support: (1) natural interpersonal interaction, (2) transitions between activities, (3) transitions between personal and group work, (4) transitions between tabletop collaboration and external work, (5) the use of physical objects, (6) accessing shared physical and digital objects, (7) flexible user arrangements, and (8) simultaneous user interactions. The integration of these guidelines is explained throughout the rest of the design, beginning in this chapter with generic examples of our collaboration design, and more concrete examples are present in Chapter 6.

5.3 Task Analysis

Meetings took place with several different studios, specialising in digital art, film and animation, to identify requirements. Once the requirements had been analysed, a set of required system capabilities (or use cases) was formulated. From here it became necessary to define how interactions could be performed without obstructing other users. This gave rise to an interaction design guideline for this system: "*Any task that affects the entire scene is a single user task. Any task that affects only a subcomponent of the scene is a multi-user task.*" Below is a list of the fundamental use cases, followed by a categorisation of user tasks.

5.3.1 SYSTEM CAPABILITIES

The system needs to satisfy requirements suggested by film and animation specialists, which are detailed below.

- **Add an object to the scene:** One should be able to choose an object from a library to add to the system.
- **Manipulate an object in the scene:** In order to animate an object one has to be able to translate and rotate it in the virtual environment.
- **Capture a keyframe of an object, appending its animation path:** One should be able to store the current position and orientation of an object to as a keyframe to begin building an animation.
- **Edit a keyframe's transform:** One should always be able to edit previous animations, should changes be required.
- **Delete a keyframe from an object, altering it's animation path:** In the case of a change or a simply mistakenly having added a keyframe, one should be able to remove it just as easily.
- **Play back the scene (i.e. viewing all animations of the scene):** One should be able to view the entire animatic that was created.

- **Play back the animation of individual objects:** A user should be able to preview the animation they are working on, without playing back the entire scene, which could disrupt users that are manipulating other objects.
- **View keyframes of all objects on a single timeline:**This is to help with synchronisation of animations between objects.
- **The animation timeline should be reachable for any user:** When dealing with a large multitouch table, it may be difficult to reach some UI components, based on one's position (e.g. having to lean over the device to reach for something on the other device. This requirement states that a user should always be able to reach the animation timeline, regardless of where the are positioned around the device.
- **Manipulate the viewpoint:** Users should be able to change the perspective from which the scene is being shown, such that the viewpoint can be adjusted to the transform most suitable for creating the desired scene.
- **Manipulate the render camera:** The render camera captures the scene, rendering the motions of objects in the virtual environment on the screen. One should be able to animate the render camera independently from the scene view, in case one user animates an object, while another animates the render camera.

5.3.2 MULTI-USER TASKS

These tasks can be performed simultaneously, as they only affect a sub region of the display.

User Widget: Each user has access to their own UI controller, granting the users control of core system functionality. This widget allows users to add content to the scene and begin viewpoint manipulation (which is a single-user task).

Menu Interactions: Since each user can navigate through a menu on their own user widget, such interactions do not necessarily interfere with other users actions.

Previz Object: Objects that have been added to the scene can be transformed, animated duplicated and deleted. One particular previz object is the camera, which can also be manipulated like other objects, but it cannot be deleted, as there would be no rendered image without it.

Object Keyframes: One method to animate an object is by saving states, specifically the position and orientation of the object, at different points in time, where the interpolation between these states produces the animation. One can alter these stored states in 3D, effectively editing the keyframe.

Render Camera: While the render camera (not the viewpoint) is being manipulated, users can continue to interact with the scene objects. The render camera's projection should be viewable in a separate preview window, so as to not disrupt users who are simultaneously interacting with scene objects. For example, as one user animates an object moving across the floor, another user can to create a top-down, panning shot that follows this object, all while the scene view is positioned to show the scene from a side-on perspective.

5.3.3 SINGLE USER TASKS

When these tasks are enabled most multi-user tasks are disabled by design, as single user tasks can easily disrupt other users.

View Point Manipulation: This is classified as a single user task, because changing the viewpoint would almost certainly disrupt other users busy with the scene. As a result only one user should be able to control the view point at a time by unlocking it. While one user controls the viewpoint, no other user can take the view point control away, so as to not disrupt the user navigating the scene. Other users can only gain control of the scene view if it is locked (i.e. multi-user state).

Animation Timeline: It would be troublesome to have two users interacting with the timeline simultaneously, as different points in time might be selected by different users, causing confusion. Furthermore, the timeline will affects the entire scene, which is our the definition of a single user task. Any user should be able to take control of the timeline, without having to go to the side of the device on which the animation timeline is currently positioned. This gives rise to the following design: the animation timeline can be made visible from any of the four screen edges and will be oriented correspondingly, and it can be locked to any edge, to prevent disruptive interactions, and it needs to be unlocked by the controlling user before it can be activated elsewhere on the screen.

5.4 Collaboration Design

This section summarises the approach taken to build a system that supports collaboration on a touch table. The collaboration design covers high-level CSCW system classifications and concepts, through to lower level details, such as the management and tracking of multi-user touch points using state machines. Furthermore, we also present a template for a core user interface component, designed to support collaboration.

CSCW Classification

The classification of this CSCW system is shown in Figure 5.5 and can be described as a co-located system with synchronous and asynchronous interaction. CSCW classifications are outlined in Section 4.2, presented by Reinhard et al.[102]. According to the task analysis above, the interaction is mostly *asynchronous*, but at the very least the view point manipulation is *synchronous*. The system is designed for smaller groups, meaning that no leading voice is required for *coordination*. While users may speak equally and spontaneously, there can still be a leading voice, e.g. a director, depending on how the team members are accustomed to working with each other. Since all users are working on the same physical device, there is no need to cater for *distribution* over the internet or even a local network. However, since some users will likely be on different sides of the display, the interface and VE orientation needs to accommodate how users are distributed around the device. Furthermore, due to the physical setup of the system, all users *visualise* the same content, abiding by the WYSIWIS (what-you-see-is-what-I-see) paradigm. The system is designed to cater for multiple users, but is not *collaboration aware*, in that it does not know how many users are interacting. Each user is given the exact same set of controls, and these are distributed by one or more users.

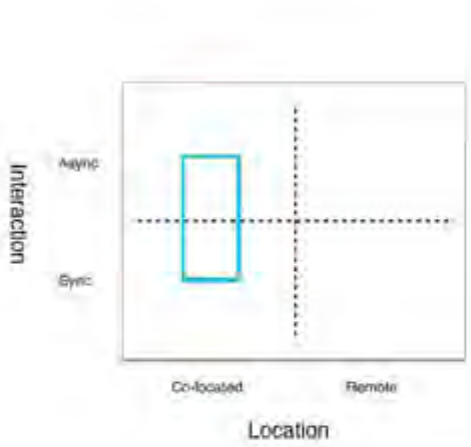


Figure 5.5: *This system is designed to be co-located, as users interact on the same device, rather than over a network, which is classified as remote. The interaction is both synchronous (for single user tasks such as viewpoint manipulation) and asynchronous (where multiple users manipulate different previz objects at the same time).*

Modes of Interaction

Some of the fundamental tasks include manipulating scene objects and changing the viewpoint. One issue in allowing these actions to occur simultaneously is when scene objects are being interacted with by users. These objects can be hidden if another user manipulates the view point. This gave rise to interaction modes, certain states in which the system can be used, each of which provides specific functionality. The states are Object Manipulation and Viewpoint Manipulation, and they are global states, meaning that the entire system is in either of the two states at any given instant. When Object Manipulation is enabled, multiple users can interact with the system simultaneously. In this state users can place and orient 3D objects arbitrarily, while the viewpoint is locked. To switch to Viewpoint Manipulation, any one user has to unlock the viewpoint. When the system is in the Viewpoint Manipulation state, only one user can manipulate the view point at a time, and no users can interact with scene objects. The design decision, to use global interaction states, was made to prevent actions from any user disrupting the workflow of others, making the respective user actions completely disjoint. One can toggle between these two states by locking and unlocking the viewpoint. It is very important for mode-based systems to provide feedback and clearly indicate which mode is active. Otherwise users may perform actions that yield undesired outcomes, due to the wrong mode being active. This is a basic description of how this mechanism caters for collaboration; the integrated usability of this behaviour is explained in more detail throughout the system design chapter, as each component of the system is explained.

The Ring Interface

After considering the use cases and requirements, a base widget was created that would help users accomplish most tasks. Due to the circular design we refer to it as a "ring interface", and it solves certain problems with collaboration and orientation issues, similar to the circular widgets in Sams' multitouch-based widget interface[127, 106]. Depending on which side of the table users are positioned at, they can arbitrarily rotate the circular widget. This can be done without changing the screen real estate changing, as would be the case when rotating a rectangular UI, for example. There are a variety of similar, circular widgets (also known as "pie menus") being used for multitouch [49, 126]. The advantage of using this was used as the basic foundation, from which the rest of the circular design was designed and developed in this research. Its circular design allows it to use up the exact same screen real-estate, regardless of its orientation. Some widgets inherit the behaviour of this ring interface and add more specific functionality depending on the task they are designed to accomplish.

These widgets are detailed in the following chapter, which covers the entire system's user interface and interaction design.

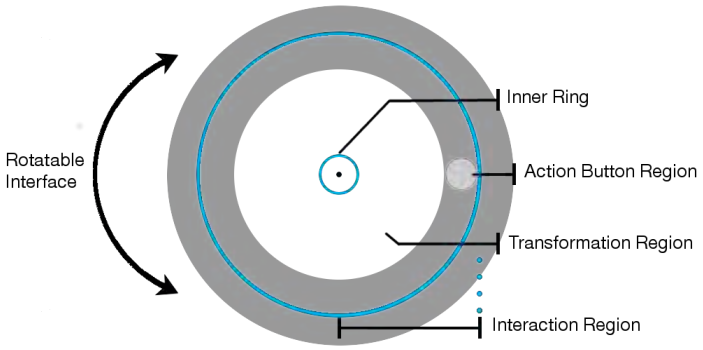


Figure 5.6: *This is the base model of the ring interface, used for multiple most components of the system. Each component adopts this basic functionality and adds context specific and functionality in the button layer, as well as object specific interactions (such as rotation and translation).*

The ring interface, illustrated in figure 5.6, serves multiple purposes in supporting collaboration. Such an interface is generally tied to an object and should only be engaged with by a single user at a time. Interactions only affect an object if they occur or begin within the interaction region. Such visual feedback communicates to users which objects are being interacted with and where on the screen. Consequently, this aims to prevent users from performing actions that conflict with other users. While the respective objects are visible within the transparent inner portion of the ring interface, buttons are located on the darker edge. These buttons perform object specific actions, aiding in scene animation and editing. Lastly, users can rotate the ring arbitrarily to a more suitable orientation for different sides of the touch table. A circular shape is useful, because as the ring is rotated, the screen real estate it takes up along the two dimensions of the display does not change.

Touch Point Processing

As already stated, the ring interface provides localised controls for a particular component. Everything within the outermost ring, or boundary, is referred to as the interaction region. While touch points outside the interaction region of a particular ring do not affect it, touch points within the interaction region are processed in a particular way as indicated by the state machine in figure5.7. This is almost *analogous* to windows and widgets on most operating systems and mobile platforms. If a user clicks outside of an applications window on a desktop app, they are usually not affecting that particular app in any way. This conventional approach was applied to the ring interface too.

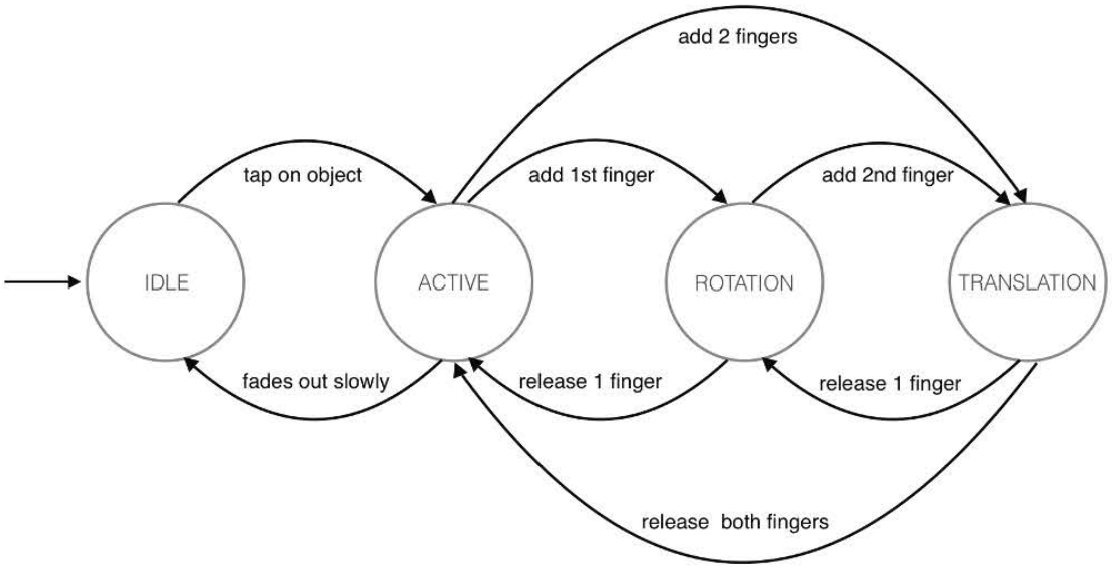


Figure 5.7: *This state machine illustrates how touch points are processed to accomplish different transformations, using the ring interface.* ***Idle:*** *Initially the object is in a rest state, with no UI or interactions available, apart from tapping the object with a finger to activate it.* ***Active:*** *The object's control ring becomes visible, with the object specific functions.* ***Rotation:*** *By dragging a single finger around within the control ring reorients the object using arcball rotation.* ***Translation:*** *Translation is achieved with two fingers, in the x-y plane by moving both fingers separately, or along the z-axis by moving the fingers together or apart.*

As the number of touch points, attached to a ring interface, changes, the interface's state machine transitions between different states. Each state determines the behaviour of the interface, i.e. the behaviour of the underlying object, in response to instantaneous or continuous touch gestures. While there is no touch point attached, the ring interface is in an idle, or listening, state. Most interfaces are attached to virtual objects, thus the interfaces can fade away to avoid unnecessarily occupying screen real estate. To activate a ring interface, one taps the underlying virtual 3d object, making the interface visible. When placing a single finger within the interaction region, the object rotation state is active. An object can then be reoriented using arcball rotation[112], which uses the delta position of the touch point as input. Once a second finger is added, the interface transitions to the translation state. Moving both fingers in the same direction on the (conceptually) "2-dimensional" screen moves the underlying object parallel to the camera plane, following the fingers. However, moving the fingers together or apart, i.e. performing a pinch gesture, moves the object further from or closer to the camera plane. In this way one can translate an object along all three spatial dimensions using two fingers. Lastly, one can switch between states that are not adjacent, by simply applying the required number of fingers for the desired action.

This approach makes it possible, in theory, to translate a 3D object along all three spatial dimensions using just two fingers. Furthermore, an object can be rotated around all three axes with just one finger and two input dimensions, using arcball rotation[112].

Chapter Summary

This chapter focused on the collaboration aspect of the system. We considered various alternatives to support collaboration and discussed why the chosen variation was the most suitable. After choosing

the device and setup, a user interface template (the Control Ring) was presented, followed by a state machine that tracks touch points associated with the Control Rings. This needed to be done, since the system is unaware of how many users interact with it. In Chapter 6 covers the entire system and details each component thereof, applying the design present at the end of this chapter.

Chapter 6

System Design

This chapter covers the interaction design of the system, and how the user interface was designed to help communicate and guide the user to the correct interactions for accomplishing pre-vis tasks. The first step was to gather the system's core requirements, obtained from digital artists, and animators [22, 29] and independent filmmakers [10]. This was a necessary step to determine the fundamental purposes of animated previsualisation, which are 1) the *composition of scenes* and 2) the *timing of events*. As previously stated, animated pre-vis is typically the first playable iteration to follow the storyboarding phase. Therefore, it is more important to focus on the aforementioned, fundamental aspects of a scene, rather than high fidelity elements such as detailed environments, precise character appearance and visual effects. As mentioned in chapter 3, existing previsualisation packages were also examined to extract the essentials for constructing a minimum viable product that is feasible for this study. The individual components that comprise the multi-user previz tool are described in a sequence that conveys their purpose more comprehensively.

6.1 Evolution of the System Design

As stated in the collaboration design chapter, several alternatives for supporting collaboration were considered and discarded for various reasons. Eventually it was decided to support collaboration in which users construct an animated scene together, using a single shared perspective of a virtual environment on a single device. While taking this criteria into consideration, the interface and interaction design still underwent two iterations of improvement. The evaluation of most initial improvement cycles was conducted by usability specialists, in the form of heuristic evaluations [92, 91]. Once a full, well-rounded design was established, target users volunteered to provide input, where the entire experiment procedure is and results are detailed throughout the evaluation chapter.

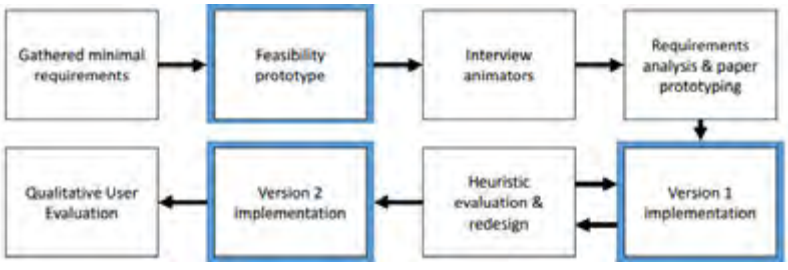


Figure 6.1: *These are the significant steps taken to create the final system, feasibility prototyping, requirements analysis, and iterations of design, implementation and heuristic evaluation. The blue blocks are programming phases.*

System Overview

The work flow of the system can be considered as a cycle of iterative improvement on the scene until it has satisfied all stakeholders' needs. This cycle consists of users creating the scene, viewing it, assessing it and making changes, until the scene's basic flow is finalised. The previz tool is unaware of the amount of users, and cannot distinguish users. Instead, the tool was designed such that each interactive object is assigned a context sensitive interaction ring, as presented in Chapter 5, Section 5.4. These widgets have an interaction region and limit the amount of simultaneous touch points interacting with them. In this way single user interaction is enforced per widget, and multiple widgets can exist on the screen at the same time, thereby catering for overall multi-user interaction.

One constraint was integrated into the design from the start, which is the following: navigating the scene requires all other object manipulation tasks to be disabled, to minimise the amount of confusion against users who are interacting with objects. Furthermore, it had also been decided that a widget (see Figure 6.4) should be used for manipulating the viewpoint, such that the control over the viewpoint can easily be transferred between users. Manipulating and animating 3d objects was enabled with a similar user interface, that are attached to the objects being manipulated. In this way one of the users can manipulate the viewpoint to a suitable perspective for the current part of a scene being created. Once this perspective is achieved, the viewpoint is locked and all users can animate the objects as required. Every time a new perspective is required, the viewpoint is unlocked and transformed to obtain the desired perspective, upon which users can continue manipulating objects. Editing the keyframes of animated objects can be done directly on the objects or achieved with a timeline. The timeline that can be docked to one of the four screen edges, and is oriented to suit a user positioned at the respective screen edge (see Figure 6.14). The components of the system are illustrated in Figure 6.2.

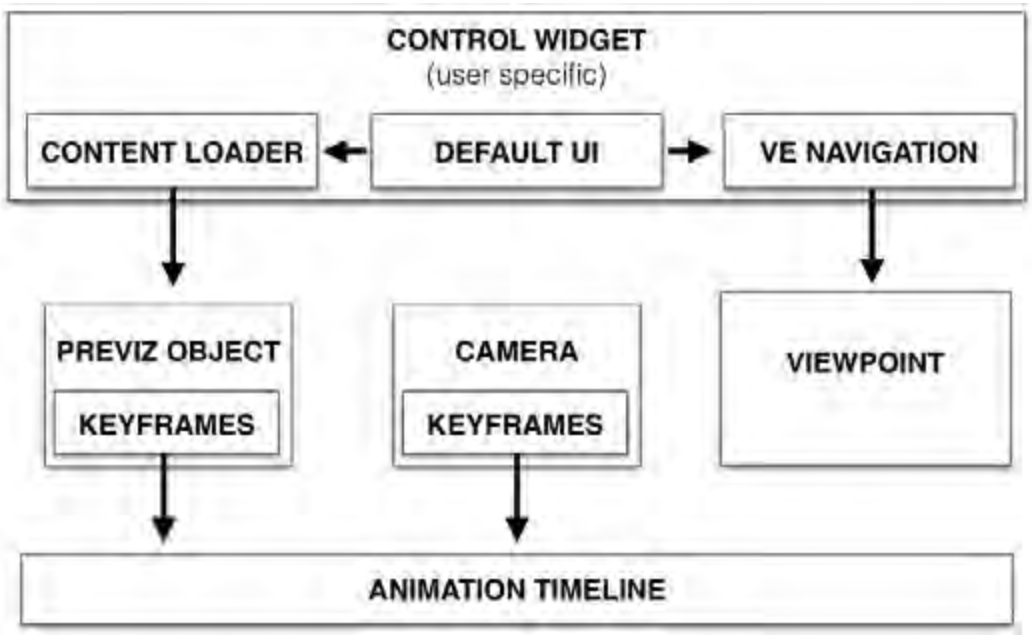


Figure 6.2: *Each user obtains a control widget, with which they can place objects asynchronously or navigate the scene in a synchronous way. The viewpoint can unlocked and controlled by any one user widget at a time. Placed objects, i.e. previz objects, can be manipulated and animated by creating keyframes for it. The same goes for the camera, which shows the final animated sequence. All the animation data can be viewed on the animation timeline.*

6.2 The User Widget

The User Widget (see Figure 6.3) enables a user to accomplish various tasks independently. At least one User Widget exist on-screen at all times, and to additional User Widgets can be created with the use of the Copy Button on the UI. The User Widget is also used as an entry point to several key areas of the interface, including navigation in the virtual environment, discussed in Section 6.3, and to add content to the scene (see Section 6.4.1). Users can rotate widgets to a suitable orientation, depending on which side of the table they are on. Note that the user widget has a graphic of an arrow pointing upwards as shown in Figure 6.3. This is a merely a guide to indicate which is the correct orientation of the user widget.

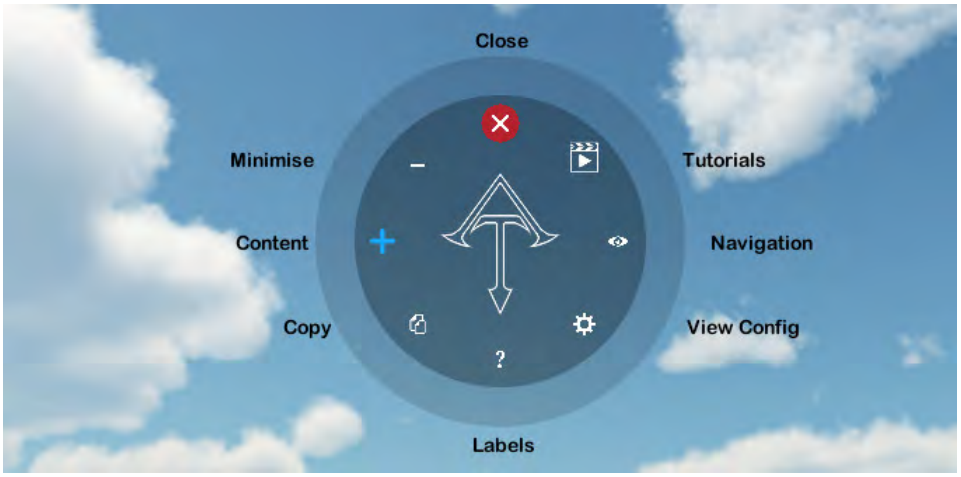


Figure 6.3: *A User Widget (default state) is created for each user, allowing them to accomplish various fundamental tasks, such as navigation within the virtual environment and adding content to the scene.*

Some important points about this interface:

- There is always at least one User Widget visible/available. Without a User Widget, users will neither be able to add content to the scene, nor navigate the environment.
- Users can create multiple User Widgets using the duplicate button, should other users want to join the collaborative process.
- An entire User Widget can be flicked across the display (e.g., to another user) and will continue to move with decaying momentum. This way a user can create and pass a User Widget to a user on the opposite side of the display, without having to reach across the table and possibly interrupting other users.
- Users can configure the scene navigation preferences with the User Widget. This way, each User Widget will also have a customised navigation mechanism and parameters, suiting them.
- The question mark button at the bottom toggles labels that show the meaning of all buttons on the respective User Widget. This is quick and easy, and one does not have to look up the meaning of features in a separate document.

Consider user A controlling the viewpoint, as reflected by their control interface, as shown in Figure 6.4. While this occurs, no other user will be able to manipulate the viewpoint, unless they physically take over the control widget of user A or begin using the Navigation Touch mechanism, more discussed in Section 6.3.3.

6.3 Scene Navigation

Since multiple User Widgets can exist simultaneously, it is important to indicate from which ring a user took control of the scene view. Furthermore, all the other users are not able to interfere with the user navigating the scene, the temporary inability to perform certain (single-user) tasks also needs to be communicated visually.

A user can navigate a scene using the Navigation Token or direct manipulation. In both cases the User Widget at the bottom depicts the state of a ring that did not take control of the scene view. Notice that this ring can also not lock the scene view, which would interfere with the user navigating the scene.

When using the Navigation Token to explore the scene, the User Widget morphs into the Navigation Token, as seen in Figure 6.4. During this state the respective user cannot close (i.e. delete) their User Widget, as it is controlling the screen. Including such functionality is possible in theory, with the scene view being locked, upon deleting the ring in control of the scene view. This would subsequently allow any remaining users to take on control of the scene view if desired. However, it was decided to make the user explicitly lock the scene view before deleting their token for collaborative purposes. With this approach, other users would be aware of these events, as they are performed by the user in control of the scene, rather being performed implicitly, in an automated fashion.

6.3.1 Navigation Token

As noted above, the Control Ring (Figure 6.3) can be transformed into the Navigation Token (Figure6.4) by the user pressing the eye button on the control ring. On the Navigation Token, the same button is used to lock the view, which morphs the Navigation Token back to the Control Ring.

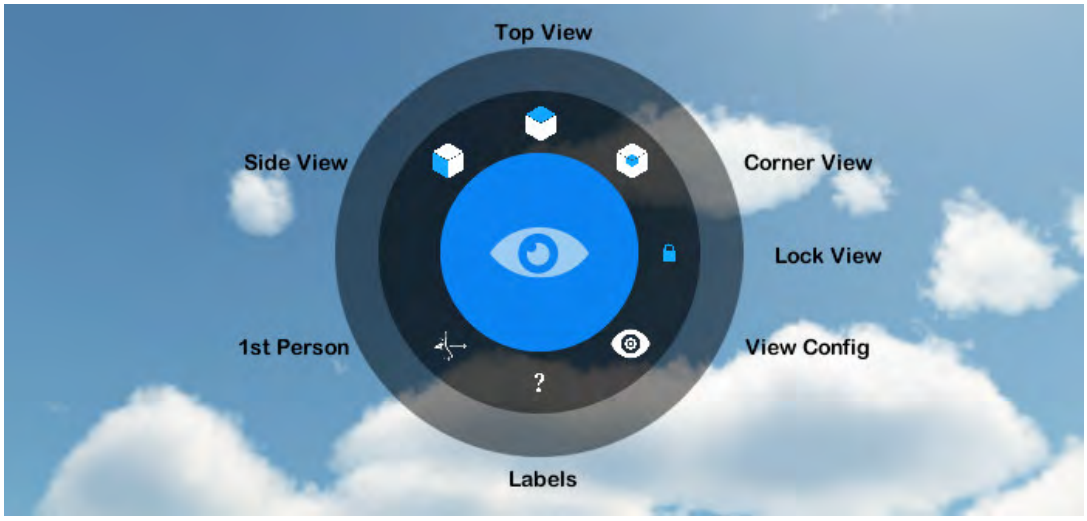


Figure 6.4: *The Navigation Token is a mode of the User Widget, with which one can navigate the virtual environment. It also provides various predefined perspectives (top-down, side-on and corner view).*

Reorienting the viewpoint, using the arcball mechanism, is achieved by moving a single finger in the central region of the token (see Figure 6.4). As a user changes the viewpoint, the token follows the finger that is performing the reorientation, or the centroid of the two fingers in the case of translation. This

is important when large changes are required in the viewpoint's transformation. It is less troublesome to perform fewer large gestures than many small ones to achieve the same outcome.

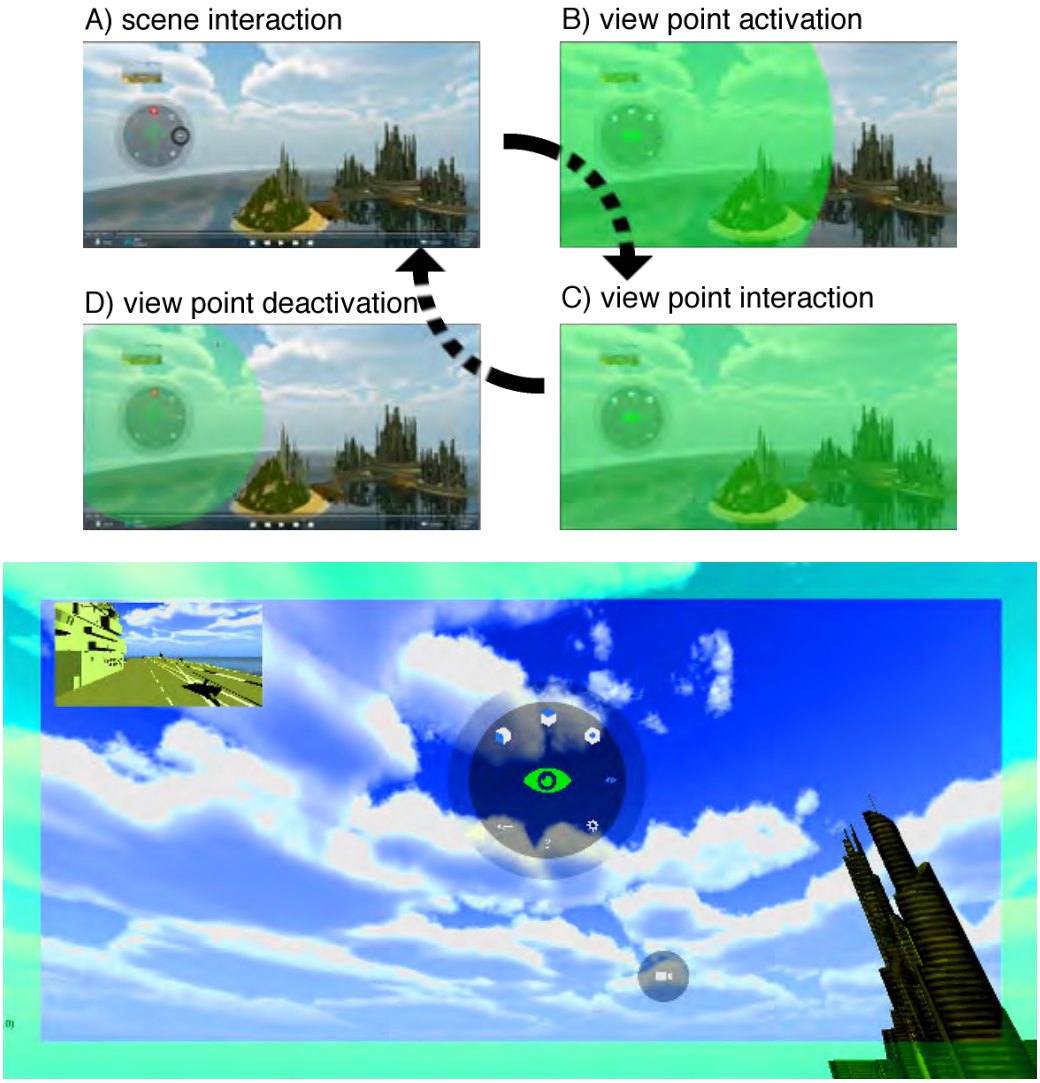


Figure 6.5: *View point manipulation activation and deactivation transitions and feedback. This notifies the user when the system is in navigation mode. In a) the view point manipulation is activated, with the respective user's widget colour expanding from their widget as illustrated in b). In c) view point manipulation is active, and when deactivating it, the reverse feedback animation happens, in which the screen overlay colour collapses to the controlling widget as shown in d). With viewpoint manipulation enabled, overlaying the entire screen, as in c), makes the environment appear differently, which is why only the border of the screen is highlighted with the colour of the user widget, that is in control of the view point, as shown in the bottom most illustration in this figure.*

6.3.2 Predefined Perspectives

Common predefined perspectives (see Figure 6.6) can be activated using the three top buttons of the navigation token, with these cube-like icons. One can change the perspective of the scene view to be a top-down, side-on or corner view, which is similar to an isometric perspective. Each time the a new predefined perspective is activated, the camera glides from it's current transform to the specified predefined perspective. This helps users understand the change of the perspective, whereas an instant change in position (a teleportation of the viewpoint) may be confusing, because the user may not know where the newly acquired position is in relation to the previous one.

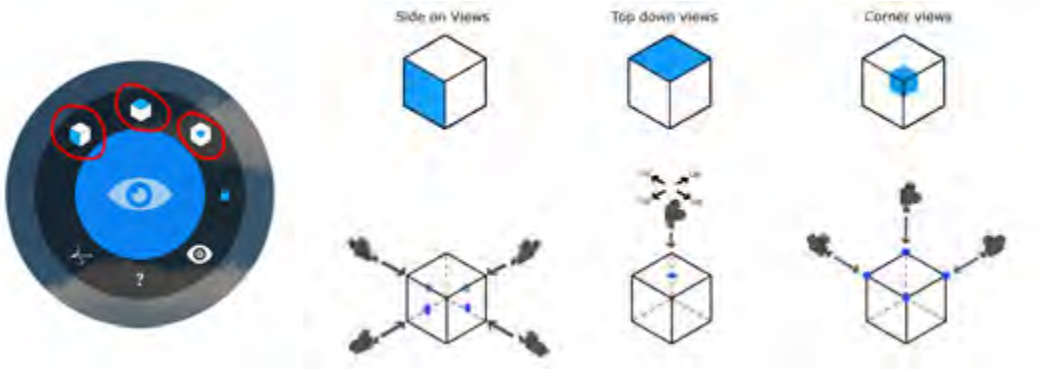


Figure 6.6: *Predefined perspectives enable users to quickly move the viewpoint to preconfigured views, namely top-down, side-on, and corner-view, with the push of a button.* ***Left:*** *The position of these predefined perspective buttons on the Navigation Token, circled in red.* ***Right:*** *Metaphor for use of icons, relating to the respective perspectives. The top row illustrates the icons for the buttons to cycle through top-down, side-on and corner views. The top row indicates from what perspective the view point would be looking at a particular object or the scene, which would be in the centre the cube in this illustration.*

Furthermore, pressing a single one of these buttons repeatedly will cycle the SceneView through four different orientations of the respective view. For example, cycling through the side-on views could be described as viewing the scene from one of the four cardinal points at any given time.

6.3.3 Navigation Touch

The alternative to the Navigation Token is what this research refers to as Direct Touch Navigation. This employs a very similar mechanism to the Navigation Token, but uses slightly different feedback. The one major distinction is that the user can place either one or two fingers anywhere on the screen, upon which one or two small tokens seek out the finger(s). This design decision was to allow any user to take over the control of the view, without having to use their own control ring. As shown in Figure6.7, no navigation takes place until a touch token has locked onto a finger. The tokens will then follow the view controller's fingers and perform the respective scene view manipulations, until all controlling fingers are released from the multi-touch surface. The last point to note is that the tokens are completely transparent when inactive and fully opaque when in use, where the transparency transitions between these two states when users begin or end their scene view manipulation. Transitioning the transparency was implemented to provide the users with additional visual feedback about the state of the scene view

and its controller. There are two way in which the view point can be manipulated, namely through translation and reorientation, and the paragraphs below describe how this can be achieved using Direct Touch Navigation.

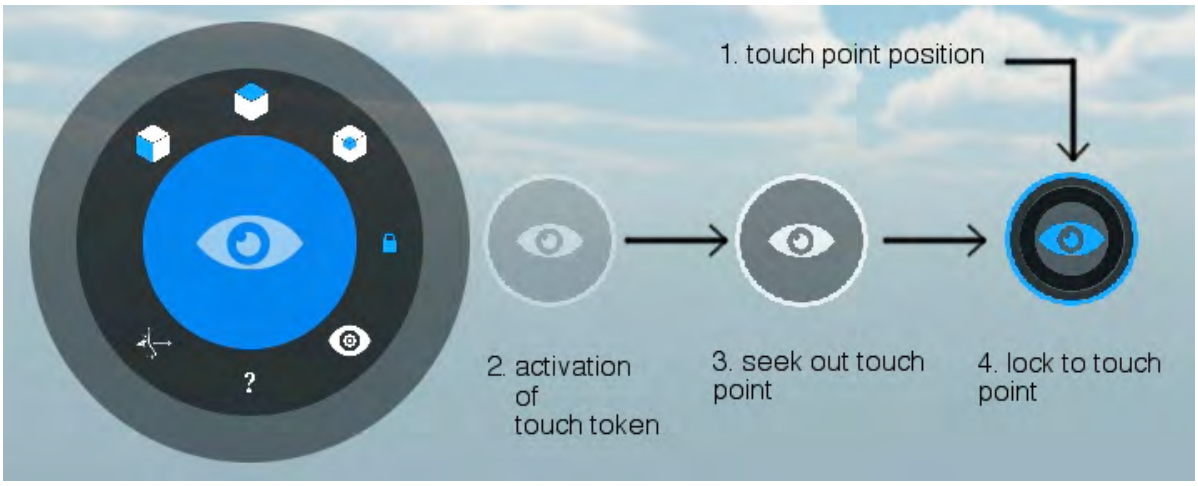


Figure 6.7: *Direct Touch Navigation enables users to navigate the environment without directly using the Navigation Token. When a finger is placed anywhere on the screen, a touch token seeks out the touch point (starting from the Navigation Token). Once it has reached the touch point, it obtains the colour of the respective user widget and can be used to navigate the environment. One finger is used for rotation and two fingers are used for translation (panning and z-axis translation, using a pinch gesture).*

To maintain consistency with object manipulation and the Navigation Token, the scene can be reoriented (i.e. rotated) using one finger. This can occur using either the first-person or arcball mechanism, depending on the user's preferences. Each user can change these preferences with their control ring, as it is still visible even when it cannot manipulate the scene view.

For the purpose of regularity within the system, translation of the viewpoint is performed with two fingers. Moving in the x-y plane is performed by moving both fingers together along the touch surface. Users can invert the direction of both these axes separately, within the view configuration menu, based on their preference. Translation along the z-axis is performed using a pinch gesture. Moving towards an arbitrary location is achieved by moving the fingers apart, and vice versa for moving away from arbitrary locations. This gesture is inspired by the pinch zoom gesture, which yields a similar result when performed on a document or map application, such as Google Maps [16]. A centroid of the two fingers is shown using a crosshair (i.e. a target) to help guide users by indicating the actual point they are moving towards or away from. As shown in the touchpoint processing state machine in Figure 5.7, one touch point is for reorienting the viewpoint, and two touch points are used for translation.

6.3.4 Scene Navigation Preferences

The user-specific scene view navigation preferences allow each user to fine tune their method of navigating the virtual environment. Whenever a user unlocks, i.e. takes control of the scene view, their Control Ring's navigation preferences are sent to the "SceneViewer" object, which adjusts itself for the user, based on the provided settings. The navigation preferences, as shown in Figure 6.8, allow users to toggle between desktop-based input and multitouch, choose arcball viewing or a first-person perspective for the flying camera, and adjust the sensitivity of rotation, translation and zooming.

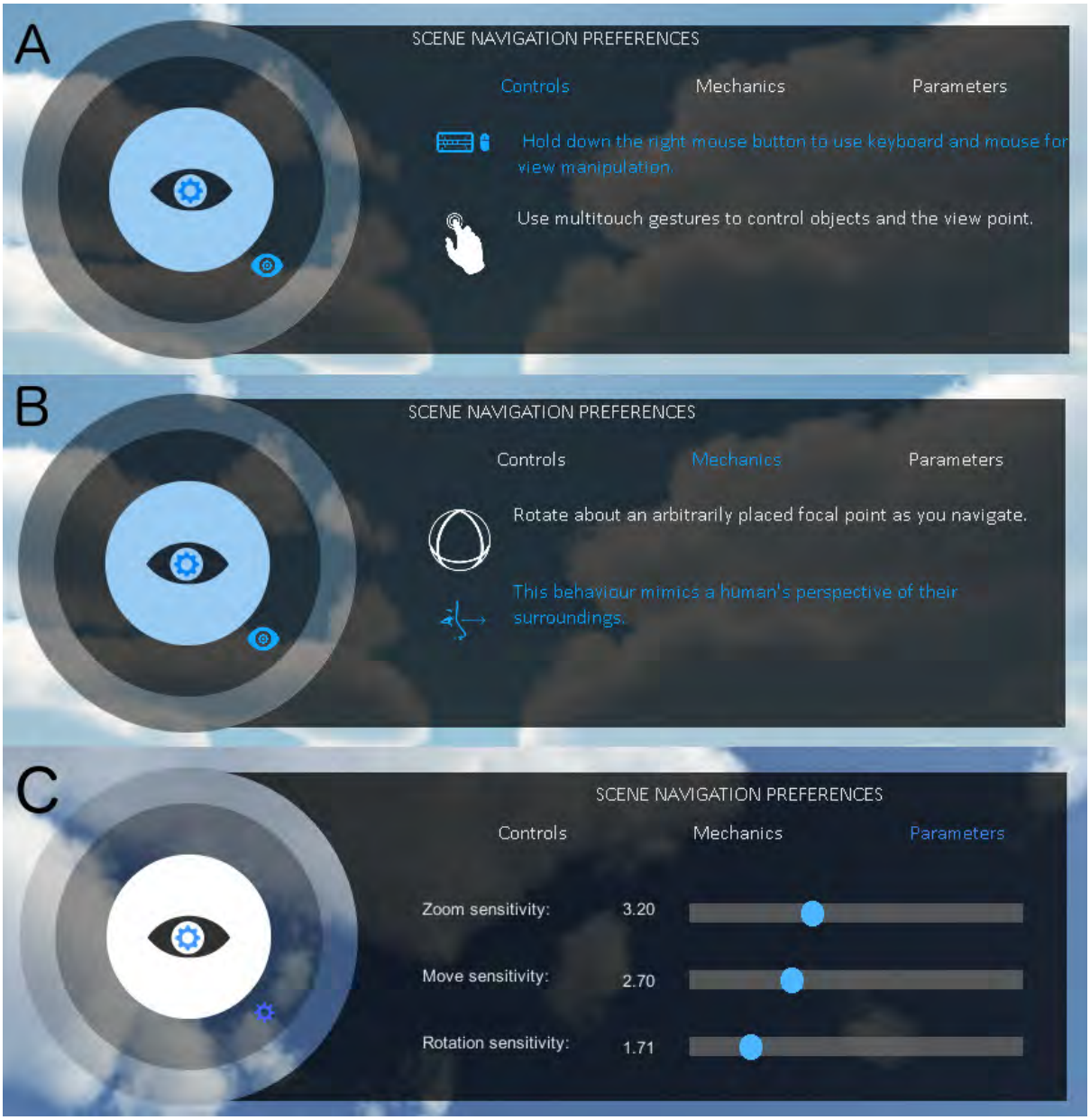


Figure 6.8: *View Point Manipulation Preferences allow each user to have their ideal configuration when controlling the view point.*

6.4 Previz Objects

6.4.1 Content Loader

A vital requirement for a previz tool is the ability to add and remove content in a scene. Such tasks should take place easily without disrupting the workflow of other users. Furthermore, each user may want to add content from wherever they are positioned around the table. For these reasons the content loader has also been integrated as an extension of the scene Control Ring. Since the Control Ring can be reoriented and moved around arbitrarily whilst other users are undisturbed, this characteristic is also a part of the content loader.

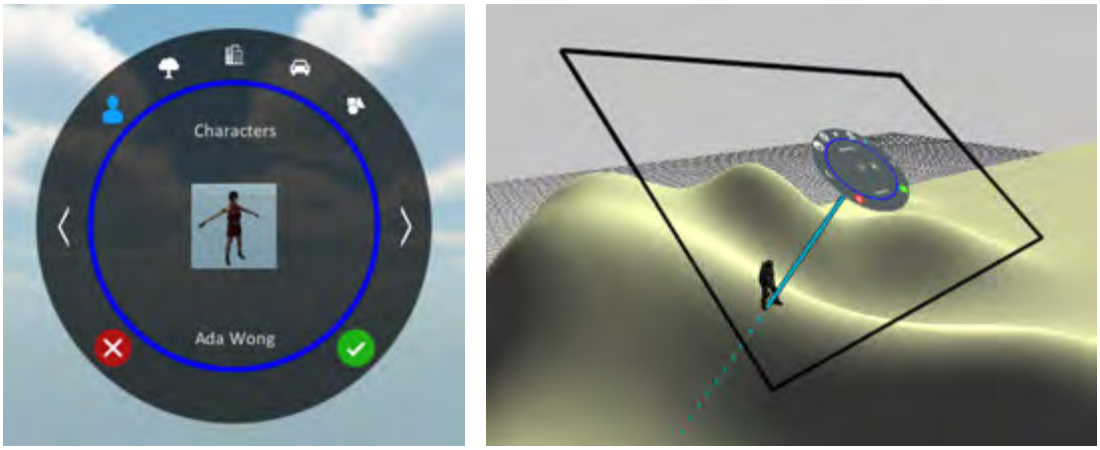


Figure 6.9: *The Content Loader allows users to select an object from a collection, consisting of various categories* ***(left)****, and then place the object in the scene* ***(right)****. This is done by casting a ray from the centre of the Content Loader, and placing the object wherever this ray intersects with virtual environment.*

Once the content loader is activated one can browse through a collection of objects, each residing in a particular category. Categories have been chosen for specific scenarios in the user experiments, and include the following types: people (heroes and villains), vehicles, structures and primitive 3D objects. Once an object has been chosen it can be added to the scene with the green tick button on the interface. A ray is cast from the centre of the content loader into the scene to determine where the chosen object is spawned. The 3D point where this ray intersects with a surface in the environment is used to place the object in the scene. Should there be no intersection, then the object is placed a specific distance into the scene, originating from where the ray was cast.

6.4.2 Object Manipulation

To enforce consistency, user interactions only affect the selected object when they occur within a certain radius. This research refers to this radius as the interaction radius or interaction region, and it is indicated by the inner ring. Within this region the respective object can be translated and rotated using certain gestures. Translation occurs in a plane parallel to the scene view's projection plane, and is performed by dragging two fingers starting within the interaction region. An object can be rotated with one finger from inside the interaction region, using the arcball mechanism. Depending on what the user is doing, the interaction ring will show appropriate feedback, as illustrated in Figure 6.10. The default feedback image, consisting of the 3 blue hand gestures and associated tasks, is displayed when no gesture is activated. This image hints to the user how to manipulate the object's position and orientation using a single word and icon for each case. The design employs the recognition over

recall heuristic, as users can see the required gesture for each possible transformation directly on the object. To elaborate, icons have been placed within the transformation region (the inner circle of the interface) to indicate the required gesture for a particular transformation of previz object (translation, rotation, scaling). With this design, users do not have to remember which gesture performs which transformation, but can read it directly on the user interface. The reason for adding these gesture images is to employ the usability principle of *recognition over recall* 4.3.1, where the user can directly see which effect a gesture.

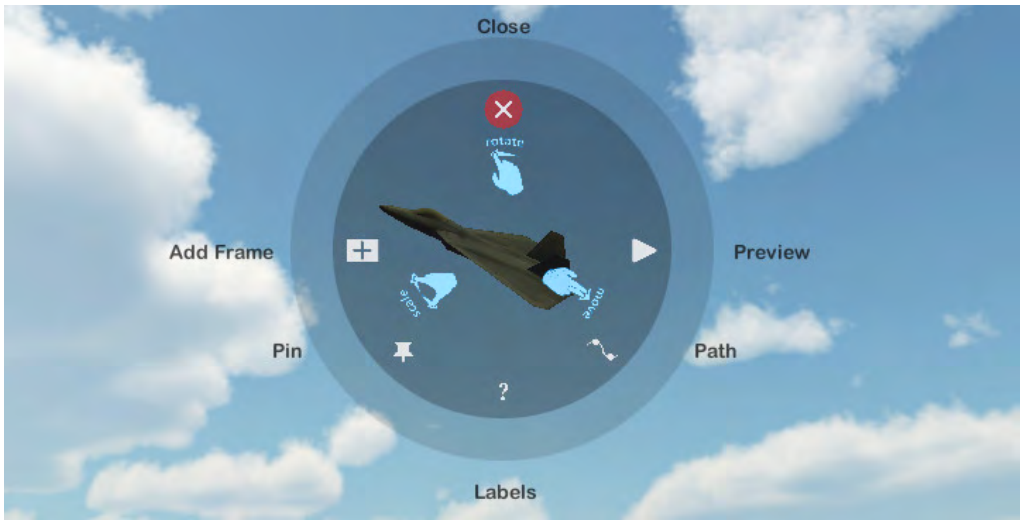


Figure 6.10: *The Previz Object Widget allows users to transform objects and animate them.*

Additional functionality is included on the outside edge of the Control Ring. These include the ability to delete the object, capture keyframes, toggle the animation path's visibility and preview the specific object's animation. In order to construct an animation path a user can place an object's state in a series of locations, with arbitrary orientations for each of these. This is known as a keyframe, where all intermediate frames lie between keyframes. Interpolating these keyframes produces an animation. Each time the user places a keyframe, this is added to the end of an object's animation sequence. A simple animated preview of this object can be performed using the "play" button on the ring interface. In this preview only the selected object is being animated, leaving all other objects unchanged. To view every keyframe of a particular object one can press the "path" button on the interaction ring. This displays a 3D representation of all keyframes, which is described further in the next section.

6.4.3 Keyframe Manipulation

Users will rarely get an animation right on the first attempt. Even if that is the case, users may often want to adjust some parts of the animation. The keyframe subsystem allows path editing of animated objects using 3D keyframe representations of the respective object. To toggle this animation path one uses the path button on the previz object's interaction ring. When path editing is enabled, a line becomes visible, connecting a set of nodes placed at keyframes, positioned using the mesh of the object in question.

The respective object's 3D orientation and position at each frame is depicted using a slightly smaller, transparent copy of the original objects mesh. One can manipulate the keyframe's transform in the same way as with the previz objects, to keep things consistent. It is important to think of this collection of keyframes as a list or array that is contained within each previz object. On the left and right sides of the ring are buttons to add new keyframes either immediately before or after the current

frame in the sequence. Deleting a keyframe is equivalent to removing an object from the sequence. While the keyframes of different previz objects can be edited simultaneously, for each individual object only a single keyframe can be edited at a time. This constraint reduces clutter on the display, as the interaction rings of neighbouring keyframes could otherwise overlap.

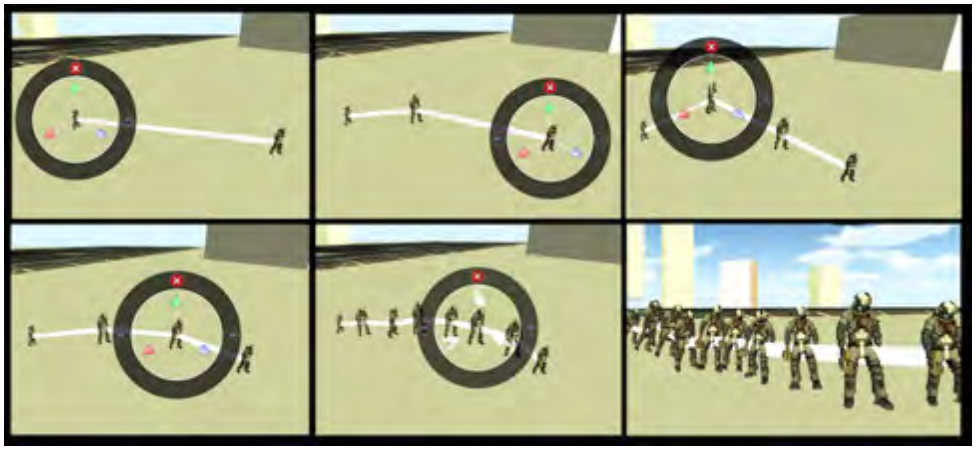


Figure 6.11: *Keyframes can be manipulated on a 3d path, with the same interactions as used for their parent Previz Object.*

6.5 Scene Camera

All components mentioned above are vital for constructing a scene. It still has to be described how the final video is constructed from all the captured frames. The scene camera fulfils this purpose by using already existing concepts. Just like the previz objects, there is a 3D representation of the scene camera in the virtual environment. When selected, the scene camera projects an interaction ring onto the display, similar to the previz object. One major distinction from the previz objects is that the render camera projects a corresponding icon onto the screen when it is visible. The icon is actually a button that displays the full interaction ring when pressed. With this approach the camera can be selected with ease from any distance, so long as it is inside the scene view's frustum.

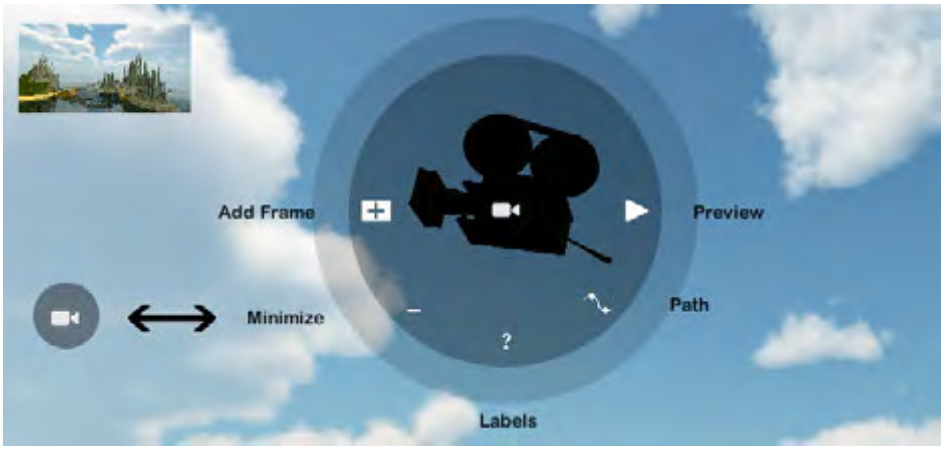


Figure 6.12: ***left**: Scene camera's minimised interface, **right**: scene camera's full interface.*

The render camera is animated in the same way as previz objects, with the use of 3D keyframes. Since it is animated over time, the camera also has an animation path *constructed* in the 3d environment. The visibility of this path can also be toggled using the path button on the scene camera's interaction ring.

6.6 Animation Timeline

The last component, the animation timeline (see Figure 6.13), is the interface that brings all the scene content together. It displays the keyframes of the render camera and all selected objects in relation to each other on a timeline. Other important elements contained within this interface are the playback controls in the central region, as well as the duration of the entire clip and the the current point in time on the left and right ends of the interface.

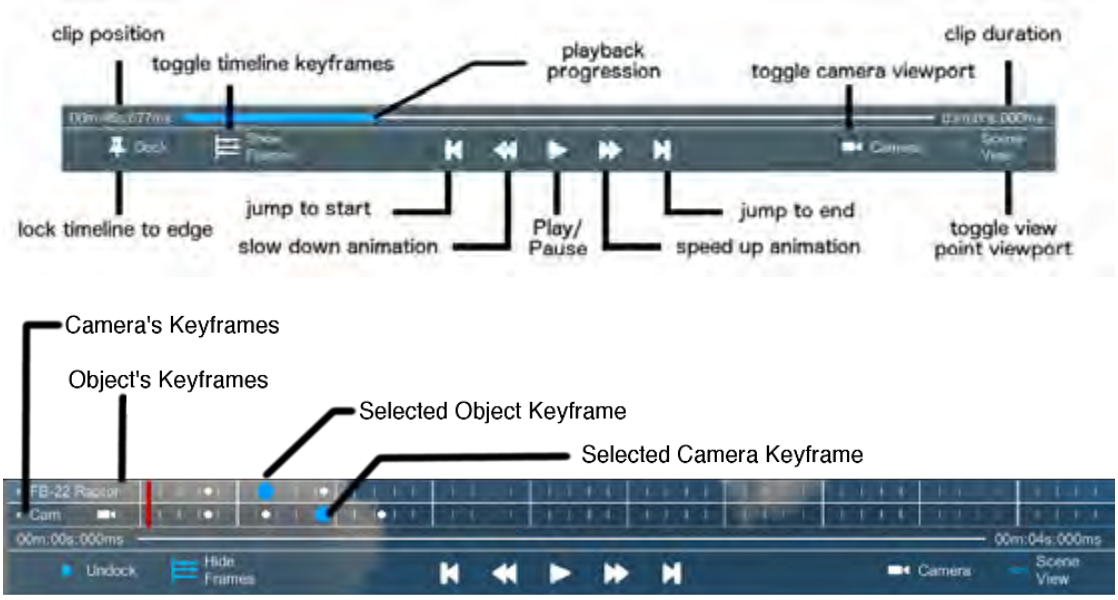


Figure 6.13: *The animation toolbar is used to control playback of the scene, as well as toggling visibility of the render camera and viewpoint viewports, and the keyframes. Furthermore, one can also choose to dock the toolbar to a specific edge, using the "Dock" button.*

6.6.1 Layout & Collaboration

Unlike all previous interfaces, the animation timeline is not circular in shape. The timeline is a rectangular widget running along the bottom edge of the display. It is the convention to display timelines in such a manner, where time progresses from left to right, in order to show as much information as possible. With such a necessarily large interface component the ring interface approach does not make sense for collaboration. In the previous chapter the timeline was defined as a cross-user interface, i.e. it should only be used by one person at a time, whilst other multi-user tasks continue to take place with minimal interference. By swiping up from the any edge of the screen, users can make the timeline appear on the respective edge. This gesture is inspired by iOS and Android's notification menu, which is brought up in a similar manner. Initially a there was one long button going along the edge of each side of the screen, and pressing it would bring the timeline to the respective side. However, it became troublesome when part of an object's control ring was on the edge of the screen and underneath this

timeline button. When the user presses the control ring button, their input is ambiguous to the system, since they would also be taking over the timeline. It should be noted that the timeline can only be available on one of the four screen edges at a time. Furthermore, the timeline is locked to the edge it is on, by pressing the leftmost "dock" toggle button.

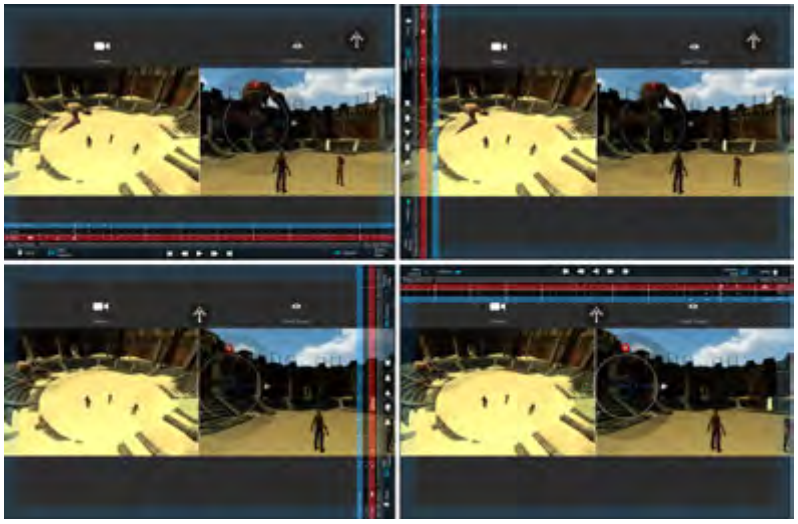


Figure 6.14: The timeline can be moved to any edge of the screen, reorienting all its contents to suit users positioned at that edge of the screen. In the top left and top right images the timeline is on the bottom and left edge, respectively. On the bottom left and bottom right images the timeline is on the right and at the top edge, respectively.

6.6.2 Viewports

One useful feature is to view the display from the perspective of the scene viewpoint and the scene camera side by side in separate viewports. Two toggle buttons on the right-hand side enable toggling the perspective of the view point, both the view point and scene camera's or only the scene camera. Toggling these viewports animates their respective size and position, to indicate to users how the interface is changing. This can be useful as different perspectives may be desired at different times in the editing process. For instance, when initially adding objects to the scene, the view point would project its perspective on the entire display. At some later stages, users may want to see the scene camera's projection, while editing the scene at the same time. Finally, when previewing a work in progress or, especially the final previz scene, the scene camera would project the scene on the entire display.

Figure 6.15: *Three ways in which the viewports of the camera and scene view can be arranged.* ***left:*** *large scene view with small camera preview in top left corner;* ***middle:*** *camera on left and scene view on right;* ***right:*** *fullscreen camera viewport.*

6.6.3 Keyframes

As users manipulate objects and add keyframes for the animation, these begin to appear on the timeline (see at the bottom of Figure 6.16). Each selected object displays its keyframes above the playback bar, on its own line, preceded by the name of the selected object. Since multiple objects can be selected at the same time, the respective timelines would stack up above the playback layer. Since the interface can get cluttered quickly. When many objects are selected a measure has been taken to reduce this: The objects' keyframe channels fade out along with the object's interaction ring as the object slowly returns to idle. Nevertheless, with the aid of the "pin" button on the far left of an object's channel, the keyframes of that respective object can be visible at all times. Whilst editing a path of an object the selected keyframe in the 3d world is represented with a selectable circular button on the 2d timeline interface, that is larger than all the other keyframe buttons.

Figure 6.16: *Keyframes appear on the timeline for each object, as the keyframes are created.*

6.7 Example

Now that all individual components have been described and illustrated, this chapter is concluded with one example of a basic scene creation process, to show how these components fit together. Consider a storyboard with a scene where a car and a motorcycle are driving towards each other and each of them turns right in the last minute, evading each other by inches, as part of a driving stunt for an action scene, for example.

This last section describes three users creating an animated scene. Initially there would only be one User Widget in the virtual environment. One user could change the viewpoint to a top-down perspective. This would make it easier to move the vehicles simultaneously, as the floor plain would be parallel to the display, and all translation is always parallel to the screen. Therefore objects would move along the floor plain. First, objects need to be added though. To achieve this, the user would have to unlock the viewpoint, and press the "top-down" perspective button, as shown in Figure 6.6. Now that a suitable perspective of the scene has been acquired, the user can lock the viewpoint again, which would morph the Navigation Token (Figure 6.3.1) back to the User Widget (Figure 6.3).

Now content needs to be added to the environment, specifically the motorbike and car. At this point the one user duplicate his User Widget and flick the new Widget to another user standing on the other side of the screen. From here both users could morph their User Widgets into Content Loaders (Figure 6.9), each adding one of the vehicles to a suitable location on the floor plain, abiding by the storyboard.

Now that all objects (actors) are in the scene, animation can begin. The scene aims to show a near miss between the two vehicles, which basically consists of 3 key locations for each vehicle (see top of Figure 6.17): A) the initial position of each vehicle; B) the point where they almost collide with each other; C) where each vehicle ends up after the near miss. If the objects were not added to the scene in suitable starting positions, they could be moved there by the users. The bottom of Figure 6.17 illustrates the user interface as the users add each keyframe for the keyframes labeled as A), B), and C). Once at the starting position, each user should store the transform, adding a keyframe to the respective object's animation path. From here, each user could move and rotate their vehicle to the point where they evade each other, storing the second keyframe of each vehicle. Finally, each user could move each vehicle to the final position and store the third keyframe. At this point a basic animation has been created. The animation of each object can be adjusted, by pressing the animation path button on the object's UI, which would display the editable keyframes of the object, with a line showing the animation path. Furthermore, new keyframes can then be added between existing ones or by appending the animation path, and undesired keyframes can be deleted.

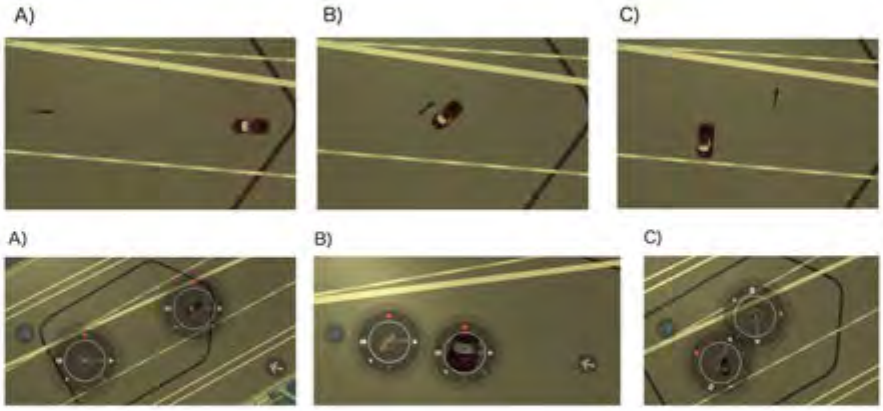


Figure 6.17: *An example of how two users would construct a stunt driving scene, where there is a near-miss occurring between a car and a motorcycle.* ***Top:*** *the three key frames for the bike and motorcycle.* ***Bottom****: The user interface used to add each of those keyframes. In both cases one user is positioned on the left edge of the screen and the other on the right edge.*

One render camera always exists in the scene, which needs to be animated according to the storyboards. This would be achievable in the same way that objects are animated, to keep things consistent. Either one of the two users could animate the camera after the objects have been animated, or a third user could animate the camera, while the other two are animating the objects. Alternatively, one user could be a director, while the other users follow the director's instructions. In case the directors instructions are misinterpreted by the other users, the director could simply take control of the scene to illustrate what they mean.

Summary

This chapter details the design of the previz tool, explaining each component's user interface and functionality, along with interaction mechanisms. The last section is an example usage scenario of the system, where a storyboard of a basic scene is described and one possible procedure is outlined for how a group of three users could use the system to generate a low fidelity, animated version of the storyboard. The following chapter goes over the evaluation of the previz tool. Included in the evaluation is an overview of several evaluation techniques, followed by the design and execution of an experiment with two groups, that are to create an animatic from the same storyboard as in example usage scenario above. The evaluation is concluded with findings of the experiments and a discussion on the implication of these findings.

Part III

Evaluation

Chapter 7

Evaluation

This chapter details the evaluation of the system, beginning with a design of our evaluation procedures, in which the experiment setup is described, followed by a breakdown of the full experimental procedure. This includes ways in which data was captured, as well as the tasks performed by the participants before, during and after the experiment. The chapter is concluded with an analysis of the results.

7.1 Evaluation Methods

Before presenting the approach used to evaluate this system, one must consider existing evaluation techniques involving human participants as test subjects. A major determining factor when selecting an approach is the data the researcher is seeking, which is usually numerical data that can be turned into statistics, or users' experience with regard to a system's characteristics. The former is referred to as quantitative evaluation, whereas the latter is known as qualitative evaluation. Quantitative and Qualitative evaluation differ significantly in their purpose, the type of research questions asked, the sampling of data, and finally the interpretation and presentation of the results. Another approach is to use mixed methods, whereby a combination of quantitative and qualitative evaluation techniques are used to gather and analyse data. Each of these fields are very large research areas that can be discussed in further depth. However, a basic overview of the three evaluation techniques is provided and the chosen technique is described in further detail along with the undertaken experiment procedure. The following should be considered before choosing a particular approach [62, 87, 77]:

7.1.1 Quantitative Evaluation

Quantitative evaluation typically generates quantifiable or numerical data, which can be processed into relevant statistics [63, 35], in order to quantify the problem at hand. As such the research question poses a hypothesis, to determine the influence of an independent variable on one or more dependent variables. One example would be to consider the usability of a previz tool on multitouch, and compare it to a the equivalent software on a desktop (keyboard and mouse) setup. In this case the usability of the previz tool may depend on the interaction required by the device.

To guide the overview of quantitative research we will use the following example hypothesis for the rest of this paragraph: *"How much more efficient is multitouch than a desktop (keyboard and mouse) setup for a single user?"*. In this case one would have to define the hypothesis more and explain what exactly is meant by "efficient", effectively operationalising it in a way that can be measured. Data can be sampled in numerous ways, so long as it produces numerical data, or can be transformed into useable statistics. With our running example, one could take two (preferably large) evenly sized groups of users, and alternate which interface is used first by each group. One could measure the duration of interaction to calculate an average task completion time for each user on both interfaces and use the entire sample to determine the more usable interface, if any. In this case the task completion time

would be an operationalisation of the efficiency of use of both interfaces. Another way to sample data would be with a post-experiment, user questionnaire, such as the SUS, to measure the usability of both interfaces. Provided the number of samples is large enough for analysis, ANOVA , MANOVA and ANCOVA tests are commonly used to determine statistical significance in the data [54, 59]. In any case the analysis of the sampled data is usually numerical or quantifiable, and can be represented with tables, graphs, charts, and other common data visualisation techniques. The analysis of the quantitative data will then help to determine wether the initially stated hypothesis is true or false.

7.1.2 Qualitative Evaluation

Qualitative evaluation is used to gain insight into underlying opinions, reasons and motivations, something that cannot be determined by numerical data alone [77, 35]. It aims to provide an understanding of the underlying problem in the research, and can often influence new ideas or suggest hypotheses for potential quantitative research. As such the research question is different to quantitative evaluation, in that it aims to reveal a characteristics of a particular system. An example of a qualitative research question would be: *"How suitable is multitouch for collaboration with multiple users on a single device?"*. Qualitative evaluation deals with the experience users have whilst interacting with a system. In this case, the researcher usually samples all data, with the aid of video footage, interviews, etc.. Furthermore, the principle researcher can find assistance in sampling, with the aid of research assistants and with data collection instruments. An instrument simplifies the process of capturing the desired data, and can be anything from a physical object to software. An example of a physical instrument is a voice recorder used for recording interviews, rather than trying to write everything down by hand. A software-based instrument would record desired information, such as keeping track of how often particular gestures are performed incorrectly, for example. Qualitative sampling is discussed further in the following section. Unlike quantitative research, qualitative data analysis can give rise to categories or themes prevalent amongst the participants' experience of the system. These are typically described with potentially long write-ups and can even include direct quotes from the participants. With qualitative evaluation, a common approach to analyse the data is with grounded theory (GT). Grounded theory is *"the discovery of theory from data systematically obtained from social research"*(Glaser and Strauss) [61].

7.1.3 Mixed Methods Evaluation

Mixed methods evaluation collects a mix of quantitative and qualitative data in a single study, and analyses the data to understand an evaluation problem [120]. Quantitative data captures a potentially large amount of data from suitably large sample sets. Sizes of large sample sets vary, depending on the research question, but usually the more samples, the better. The qualitative data provides contextual information of the captured quantitative data. With mixed methods evaluation one could for instance determine the more usable one of two interfaces using quantitative methods, and then gain an understanding of why that interface may be superior. Mixed methods evaluation is advantageous in *"that it balances efficient data collection and analysis with data that provides context" [35].* Mixed methods are also useful with verifying the same research question from different angles, increasing the confidence in the validity of findings.

7.1.4 Qualitative Sampling

Now that a variety of common evaluation methods have been covered, the next question is who to recruit as participants for the study. This is referred to as sampling. There are three categories for samples, namely convenience samples, purposeful samples, and theoretical samples [62].

Convenience samples are, as the term suggests, easy for the researcher to recruit, but do not fit the position of ideal test subjects that may have a certain skill set a particular area of expertise. Such samples are considered the least scientific, yielding experiment results that lack intellectual credibility

and are generally frowned upon in academia. One example would be to use people in one's neighbourhood, because they are close by. However, such test subjects may not have the necessary skill or background knowledge required for the experiments.

Purposeful samples are a much better alternative for choosing participants. These are participants who one recruited thoughtfully and purposefully, in order to fully answer the research questions. These can be further subdivided into several groups. One of these is a "Snowball Sample", where the participant informs the researcher about more possible candidates. Then there are the opportunistic or emergent samples, which are participants that are recruited during the process of the experimental data gathering. A maximum variation sample allows the researcher to maximise the diversity of information relevant to the research question. There is also the extreme or deviant case, which is very unusual and more commonly used for a case study. An example of a deviant case would be someone whose skill set is completely unrelated to what is sought out by the research. Lastly, there is the typical case, which is essentially the opposite of the deviant case.

Theoretical samples are participants who have experienced the the process of interest and can help to develop a well rounded-theory. Such samples typically used for grounded theory studies. Within theoretical samples are what is referred to as discriminant samples. This is a new group of participants recruited to see if their experience falls in line with the theory that was created in the grounded theory study. Discriminant samples can help to verify if the theory, created in the grounded theory study, is accurate or not.

7.2 Experiment Design

7.2.1 Mixed Samples

This research aims to clarify what actually helps participants to collaborate better, and how this can be achieved with a system that promotes CSCW. In order to obtain such answers, a qualitative evaluation is much more suitable, as was described in Section 7.1.2. Furthermore, a qualitative evaluation was chosen for this research, due to the limited contacts available in this field. Now that the preferred type of evaluation has been motivated, the different types of participants need to be explained, along with their suitability for the research.

The two groups each consisted of 3 participants, and more importantly, colleagues that work together on a regular basis. The first group, Group A, consisted of purposeful samples as participants from relevant backgrounds were recruited. This group consisted of filmmakers, and their feedback was recorded as the principle feedback in this research. The second group was a theoretical sample, as explained in Section 7.1.4. The purpose of this group was to determine verify the prevalence of the collaboration experienced by the first group when using the same-time, same place environment.

7.2.2 Test Subjects

Evaluating the collaborative system presented in this thesis requires that multiple users interact with the system simultaneously, i.e. groupware evaluation [43]. Groupware development is not easy, as it can be costly and evaluating basic support for teamwork activities is difficult. Moreover, this research requires purposeful samples, meaning that participants need to be experienced with computer animation or at the very least film pre-visualisation. Test subjects who are likely to conduct previz in their daily work are more suited to the study.

In evaluating the system, groups of slightly different backgrounds were chosen, such that their overlap would be mostly about collaboration, which is what the system aims to enforce. This evaluation gathers data from different groups and considers the overlap, as the intersection is relevant data resulting from the evaluation. If the feedback is (more or less) consistent between the two groups, despite their different backgrounds, it is a strong indication that the raised points are relevant. The first group is from the film industry, i.e. accustomed to pre-visualisation, and consists of one editor, one cinematographer and one director. The second group comes from a more technical background,

in particular computer science. In this group one member is very familiar with multitouch-based interaction, another member has a background in graphics, virtual reality and games programming, and the last member has a solid foundation in systems architecture.

7.2.3 Setup and Procedure

The evaluation is designed to assess the quality of collaboration provided by the system. However, since an entire user interface was designed to accompany the collaboration, a usability evaluation is also required. During the development iterations the system was subjected to several heuristic evaluations in order to improve the user interface. One standard and widely used way of evaluating usability is the system usability scale (SUS) [36, 23], which is a questionnaire, containing 10 questions that are handed out to users after having conducted the experiment. The questions ask the user about the general usability, such as how cumbersome the system is to use or whether the users would use the system frequently. Questions are answered on a 5 point Likert scale where 1 and 5 represent 'strongly disagree' and 'strongly agree' respectively, 3 being a neutral opinion. For the full set of questions see Appendix D.

The experiments were conducted in a quiet room, with the device on one side of a table, leaving enough room for the priming exercise of collaboratively constructing a jigsaw puzzle, as well as filling in the experiment forms and conducting the user interview. Nunez et al. have discussed the concept of presence in virtual environments, one conception thereof being cognitive presence [99, 100]. Cognitive presence focuses on the impact of the virtual environment on the user's overall state of mind. In this experiment, users were asked to construct a scene while working together. We decided to prime the teams into a collaborative mindset, using a jigsaw puzzle. Essentially, completing a jigsaw puzzle with other people is similar enough to creating a scene, because they aim to complete an image while collaborating. Furthermore, putting together a jigsaw puzzle on a flat table is similar to the final exercise, in which users interact on a touch table. One distinguishing factor between the two exercises is that in the priming exercise one moves physical pieces around, whereas the pre-visualisation exercises is limited to moving virtual objects around in an environment.

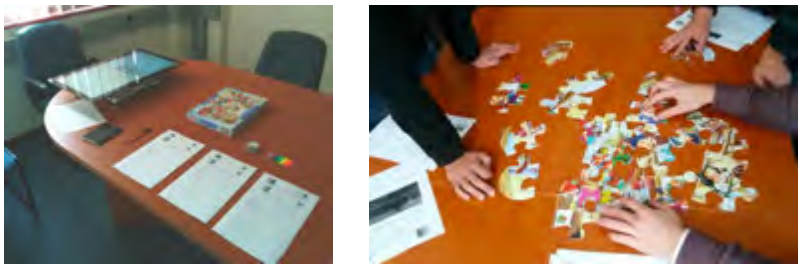


Figure 7.1: ***(left:)*** *An experiment room isolated the users from any distractions.* ***(right:)*** *Priming users into a collaborative mindset, by completing a jigsaw puzzle as a team, prior to proceeding with the actual experiment.*

In this section we outline the experimental procedure. The experiment form containing the process for the session, the questionnaires and interview questions can be found in Appendix D. It was clearly stated to the users that they were doing this entire session voluntarily, i.e. that they were not forced to stay and could halt whenever they wanted to. Some relevant subject demographics including exposure to previz packages, multitouch and virtual environments, were captured at the beginning of the session.

After capturing the demographics, the users were given their first task, which was to piece together a jigsaw puzzle as a team, with the intention of prompting them into a collaborative mindset. While

this may not have been necessary, as the participants work together in their profession anyway, it was considered a suitable precaution. The jigsaw puzzle only consisted of twenty pieces, in order to minimise the time taken to complete the puzzle. It was not supposed to be too trivial, e.g. a four piece puzzle, as that would be quick and easy to complete for an individual, and would not require collaboration. A puzzle with much more than twenty pieces could also have been used, but the choice for a reasonably low number was promote some collaboration, whilst keeping the duration of the experiment reasonable. The aim was to keep the users from feeling fatigue, and obtain as meaningful and thoughtful responses as possible in the interviews, which was the last stage of the experiments. The users still being willing to participate at this stage was important, because that is where a significant amount of the qualitative information was captured.

Once the users had completed the puzzle, they were shown a tutorial video of the system created for this study, illustrating how to generate a simple scene with multitouch. Thereafter, they began the task of creating a new scene, with a provided story board (Appendix D) as a guideline. Upon completion, each participant completed a usability questionnaire, namely the System Usability Scale (SUS). The experiment was concluded with an open, voice recorded interview designed to capture the users' experience with regard certain aspects. The recordings were transcribed and can be found in Appendix E. These included the suitability of multitouch for collaborative pre-visualisation, flaws with the interface degrading the user experience and possible ways to improve and extend the feature set.

7.3 Experiment Results

As already discussed, experimental data was gathered in various ways, including questionnaires, voice recorded interviews and video recording. In this section the information is analysed and categorised into themes to highlight where and why the system succeeds or fails. Each group had their own, characteristic views on the system, due to the nature of their skill sets. However, using groups from different, yet still relevant professional backgrounds is useful in triangulating the core features of the system, both good and bad.

7.3.1 Qualitative Interviewing

Certain measures have to be taken when conducting qualitative interviews, in order to minimise the bias an interviewer can impose on the interviewee(s). There are four key points that should be kept in mind, when designing and conducting qualitative interviews: use open-ended questions, avoid leading questions, probe issues in depth, and let the informant lead [18, 77].

A closed question is one for which the answer choicer are either given to the respondent or understood by the respondent, e.g. *"Would you prefer coffee, milk or hot chocolate?"*. An open question allows the respondent to answer without presented or implied choices, e.g. *"what would you like to drink?"*. Furthermore, open questions usually begin with the words *what, who, how, when, where or why*. The word "why" should be limited, however, because it suggests that there is a right answer.

Leading questions suggest a particular answer, or at the very least imply that one answer is more correct than another. This kind of question should be avoided, in order to allow people to answer in their own terms voicing their own views, values and experiences. A leading question is "*How good was the quality of your tea?*", whereas the non-leading equivalent is "*How was the quality of your tea?*". The questions prepared for this user interview are illustrated in Figure 7.2.

Probing issues effectively and in depth leads to more successful interviewing, as this stimulates an informant to produce more information, without injecting oneself into the interaction to the point that one only gets a reflection of oneself in the data [47]. One probing technique is the Silent Probe, in which the interviewer remains quiet, waiting for the informant to continues. This generally happens automatically, as the interviewer is often busy taking notes of what the informant has just finished saying. Another probing technique is the Echo Probe, in which the interviewer repeats the last thing an informant said and asks them to continue.

Open interviews are strongly recommended, as they allow the informant(s) to provide more data relevant around the question at hand. This flexibility can be extended to allowing informants to use their own words, as they may be more comfortable/less confused when using their own language/jargon. A crude measure of success is the volume of response, which should be around 80% their own words. Should the informant be unclear about anything, it is the interviewer's responsibility to follow up and get clarity, in order to not distort the data. In some cases one might need to use Translators, that provide literal translation, while keeping key terms in the local language. Fortunately, the interviewer all informants are fluent in english, and the line of work between these parties is not so different that a translator is required for the work jargon used in the different industries.

1. How intuitive and consistent did you find the system's UI and interaction?
2. If there was anything cumbersome, what was it and why?
3. How suitable do you think multitouch is ...
   a) ... for pre-visualisation of this kind?
   b) ... as a means of collaboration?
4. What held you back the most when using the system?
5. What do you think should be removed?
6. What do you think could be improved?
7. What other fundamental features would you desire in this system?
8. Were there any issues with simultaneous user interaction, and if so, please elaborate.
9. Was the user interface helpful or harmful for aiding with collaboration?
10. Were there any issues as to which user was in control of a particular component of the scene?
11. Did the interface help you to work together in a team, compared to past experiences?
12. Did the system improve your collaboration/team work in any way?
13. How do you feel about this attempt at creating a more involved collaborative experience for constructing scenes?
14. How effective do you think this type of system would be for previsualisation in your profession, whether within your company or generally in the industry?
15. Is there anything else you would like share about your experience using this system or enquire anything regarding this research?

Figure 7.2: *Interview questions for the participants.*

7.3.2 Differences Between Groups

Before jumping straight into the most essential gathered feedback, this short section covers the data which was quite unique to the two groups. The quotes in the following sections can be found in Appendix E, containing a transcribed version of the post-experiment user interviews. One thing to note with the developers is that the developer group had little prior exposure to pre-visualisation software. In the priming exercise, each developer worked on a portion of the jigsaw puzzle and eventually they combined the separate segments. The actual animation exercise was performed in a similar manner, where each user took on a responsibility, i.e. a role in creating the animatic. Although they took *"a role-based approach"* (UserB3) they all got *"very involved in [their] roles. So that forced [them] to know*

exactly what was going on in all the aspects." (UserB1).

Unlike the developers, the film makers are accustomed to pre-visualisation and related work, so each member already knew their role. During the priming exercise this group completed the jigsaw puzzle by building onto a single large piece, rather than completing segments and combining those at the end. For the most part one of the members put the pieces together, while the others directed or observed and stepped in occasionally. Interestingly, this behaviour also coincided with how they went about creating the animatic. One could argue that the priming exercise was successful. On the other hand, the groups were each picked from companies in which the members already worked together. In this case collaboration could very well be an everyday activity that does not require priming.

Lastly, some of the feedback from the groups was unique and characteristic to each. For the most part the developers focused on improving the system as it stands, from a much more technical perspective, rather than adding new features. The film makers were used to fully fledged pre-visualisation and film creation software, and their feedback was a list of features, including the following: an undo stack for each object; content extensibility, allowing artists to create more assets to use in the software; a collision system for objects in the virtual environment; a 3D frustum for the render camera, showing what is in its view volume ; the ability to add notes for additional information about any aspect of a scene.

7.3.3 Similarities Between Groups

Naturally, the most essential data lies in the feedback, which was largely consistent between the groups. In this section the overlapping points mentioned by the software developers and film makers are categorised and detailed, highlighting pros and cons. The analysis begins with the more general aspects of the system, such as its suitability on a touch table and the user interface design, and then proceeds to cover more specific matters, such as the mechanism used to navigate through virtual environments and create animations.

Multitouch

For low-fidelity animation using touch-based interaction worked, as one can move objects *"directly as opposed to sitting with the mouse trying to work through it"* (UserA2). However, for high fidelity animation using touch could become cumbersome, as *"it might be a bit finicky ... a bit painful"* (UserB3). It would be problematic with touch-based interaction and the *"accuracy of getting it down to the very detail of where you want to put an item"* (UserB3).

User Interface

The participants found the user interface to very minimalist, and the layout works well, because *"you have everything in one place and it doesn't get confusing"* (UserA3). This is likely due to the system being designed for low fidelity animation, which does not require a complex set of controls for a variety of functions, unlike high fidelity animation tools. However, while the interface and system are *"very simplistic"* (UserB3), the layout would have to be revised with increased functionality. As was intended by the design, the components had a *"fairly similar UI with mostly the same set of controls ... or functionality, in the context of the object"* (UserB3). One suggestion was a toolbar that provides additional functionality, such as creating user widgets. One further suggestion was to activate different interaction modes from the toolbar, e.g. translation or rotation, adapting the controls available on the ring interfaces of affected objects.

Another useful feature was the widget, which provided each user with a set of essential controls. One suggestion was to make this component personalisable, e.g. reserved slots for configurable shortcuts. Another suggestion was *"toolbar or something that you could create widgets from"* (UserB1), rather than duplicating an existing widget using its ring UI.

Both groups also suggested another layout for collaborating on the same device. The suggestion was a viewport for every user, providing each with their own set of controls and perspective. With a large

multitouch-enabled screen users felt *"like it's overwhelming how much space there was"* (UserA1), and would potentially find it more manageable with multiple, smaller viewports. Additionally, whenever the animation is played, it should be possible to preview it in fullscreen mode, despite the user specific viewports. A suggested alternative to having one's own viewport on the same device, is networked collaboration, where each user works on their own device and the environments are continuously synced. According to the film makers such software exists, where *"you can work off the same project file, but it's not like you're in a virtual room together"* (UserA1), effectively making the collaboration aspect less immersive. A major issue with this approach is internet speed and connectivity, as was realised by the participants. A point mentioned by one of the participants was that *"there are always going to be limitations, it doesn't matter how refined the software is ... there is always something that could be done better"* (UserA1). This user interface certainly achieved the minimalist design guideline, as users stated that *"there really wasn't that much to the interface, so it was quite easy to pick it up. Of course you need to learn how to use any interface in the beginning"* (UserB1). Lastly, the system should allow users to make annotations or comments. There is a multitude of ways in which to add comments: adding notes at certain parts of the playback on the timeline, general notes that apply to the scene as a whole, and attaching notes directly to static or animated objects in the 3d virtual scene. This feature can be accommodated in two ways: textual and auditory. For auditory notes, one could hold down a button to and the system would record the voice note until the button is released. Another implementation would be to press a button to start recording and pressing it again to end the recording. Of course in this case the other participating users should not interfere, unless they are contributing to the voice note. The alternative is to type out the notes, which requires a keyboard that can be rotatable and moveable to anywhere on the display, much like the other ring interfaces. The drawback with this is that more screen real estate is used than with a single button to start recording voice notes, ending the recording and playing/pausing the recorded voice note. Conversely, it one will be able to see multiple voice annotations at the same time, so long as they do not overlap, whereas one will only be able to listen to one voice note at a time, as it would likely get confusing otherwise.

Collaboration

Despite the suggestions for alternate configurations for collaboration, both groups found the single device setup helpful for team work, because it aided with communication. Furthermore, just "*the way the software is laid out it immediately at least promotes teamwork*" (UserA1). Each user having their own widgets and working on one screen means they "*have to work together, otherwise it's just three monkeys hitting buttons [...] it wouldn't make sense*" (UserA1). Users being in the same room using a single device is also a successful part of the system's collaboration design choice. The system was *"good for communication"* (UserB1) , which would be useful for preserving the director's vision from the planning phases through to full production. Conversely, for simultaneous user interaction the developers found the system *"harmful, because of that bottleneck due to the single user view point interaction"* (UserB1). This indicates a flaw in the design choice of only allowing one user to control the view point at a time; the finding is discussed in depth later in the chapter.

Naturally, the filmmakers focused much more on how this system would help them accomplish one subtask of their day job. While this group's director is used to storyboarding solo, using this system with his colleagues he *"needed their input"* and *"wanted to know what they think, because that's the feel for this engine"* (UserA2).

Under normal circumstances, this step of the preproduction phase can be quite lengthy, as approval needs to be granted by various parties, who may not always be present or available. Furthermore, should the concept for a scene be rejected at any stage, then it will have to be fixed and go through the entire approval cycle again. This system aims to reduce the time taken to iterate over scenes and it has proven to be effective in this respect, as working *"together on the previz as a team, every step of the way each of you will have a better understanding of what it is that the scene requires"* (UserA2).

While the system may be good for communication and consensus, it is "*not very collaborative in working asynchronously"* (UserB1). One of the users summarised what inhibited the collaboration in

stating that *"for simultaneous user interaction it was harmful, because of that bottleneck due to the single user view point interaction"* (UserB1).

The next section explains this statement in more depth, along with listing the pros and cons of the view point manipulation in general.

Viewpoint Navigation

Of all the components the users gave the most feedback about manipulating the viewpoint. As per design, navigating through the environment was a single user task, to prevent other users from interrupting. However, this introduced a number of issues. Despite the feedback about which user was in control of the view point, users often forgot to toggle navigation off when they wanted to interact with objects. Naturally, users would likely become more adept with increased exposure to the system. Nevertheless, this feature could have provided more feedback to indicate clearly when one can manipulate the view point or the objects.

Furthermore, since navigating the environment is a single user task, this imposed a bottleneck on the proportion of parallel tasks available to the users. *"The only issue is that, because it's so dependent on the view, only one person is really in control at a time"* (UserB1). Users found themselves changing the viewpoint for almost every subsequent keyframe they placed, which kept interrupting the multi-user interactions, since with certain scenes the system is highly dependent on the view. For scenes that use a fixed camera position and angle, this would be much less of an issue, as users only need to manipulate scene objects.

As both groups stated, it would have been better to restrict the view point manipulation in various ways. One suggestion was to use an isometric view, similar to many real time strategy (RTS) games, where you can interact with a fair amount of the environment without having to change the camera significantly. Even in this case, only panning and zooming would be required to adjust the view of the environment. Alternate suggestions were to limit view motion along a single arbitrary axis, and perform rough initial transformations that can be fine-tuned for more specific, smaller transformations (i.e. accurate control of the view point). As the director from the film group stated, *"there is too much choice in terms of movement"* (UserA2) and *"a bit easier would be to have less freedom, as much as that might contradict itself"* (UserA1). Another suggestion was for the camera to snap to certain orientations, such as top-down and side-on views, when rotating it.

Users also mentioned the desire for rotating around the normal vector of the camera plane, i.e. *"to rotate around the z axis like Google Maps or Google Earth*" (UserB3). Introducing such a feature would enable users to orient the viewpoint to suit themselves, regardless of which side of the display they are situated.

One feature mentioned repeatedly by both groups was the use of custom, configurable view points. To elaborate, users should be able to store arbitrary view points and be able to easily revisit them at a later stage, similar to how the predefined viewpoints worked. A camera that tracks viewpoints was another suggested change, as users said it would be*"interesting and potentially useful is if you could make your view the actual camera"* (UserB2). Thus a sequence of stored custom view points could determine the motion path of the render camera. Users reported that having to look for the camera was often troublesome, if it was outside the view point's frustum. Using the view point in place of the render camera would eliminate this issue.

Another desirable feature was *"if the camera could be locked to arbitrary objects"* (UserB1). With such a feature, one would only have to animate the objects and the camera follows them. This would simplify the process, as the camera's animation path would only have to be adjusted, rather than being created from scratch.

Some other feature improvements were mentioned. One of these was the ability to invert input for rotation and translation, since some users are more accustomed to interacting with the view point, as opposed to the environment. Such a feature could easily be included in the existing view point configuration, which is user specific in any event. Smoothing the movement of the view point was also preferred. Despite all the features the view point offered to the user, this component seemed to

introduce the most issues. As the director said, *"if I could move easily through this world, I would do crazy things with it"* (UserA2).

Animation and Timeline

The timeline and animation mechanisms were received far better by the users, than the view point controls. They could easily interact with these components, and only suggested minor improvements and changes. The system design catered for preserving as much real estate on the screen as possible, by only showing UI components of objects that were active at the time. However, users said it would be better*"if those objects were always on the timeline"* (UserB3) and *"if the timeline was always there"* (UserB1). This would incorporate a list of all selected objects somewhere on the interface. One should be able to show and hide arbitrary objects in the 3d scene, using this list. When selecting an object or keyframe on the list of objects, even if it is outside of the viewpoint's frustum, the view point should move directly to the selected object. Furthermore, users would have liked to interact with the 3d animation path (i.e. *"interacting with the actual line, using a handle on the curve"* (UserA1)) at any point, rather than only being able to manipulate the keyframes. Furthermore, one should be able to delete a subset, or even all keyframes of an object at once, rather than being limited to deleting each one individually. One idea from the feedback was to interact with the timeline using the mouse only. While this would make the system more than purely multitouch enabled, a mix of such input media may improve the system overall. Another thing some users had difficulty with was the arcball rotation to reorient objects during the animation. All these features would greatly reduce the number of interactions required to achieve the desired outcome, when compared to the way the system currently works. This is desirable, since fewer interactions to receive the same outcome requires users to have to think less about what to do next, ultimately making the system more usable.

Axis-specific rotation would have been preferred by both parties, using a transform manipulator attached to each object. These manipulators, used for rotation, translation and scaling, are common in game engines and animation packages, as mentioned in the background chapter of this report.

Some users also suggested the capability of drawing an animation path, which would take far less input than placing each keyframe to create the path. This feature was initially included, but it was found to be difficult to create a 3d path with a single stroke, using 2d input. The last suggestion was to automatically loop the animation playback, effectively playing it again from the start, once it finished, without additional user input.

7.3.4 Usability Evaluation

After completing the pre-visualisation task, the users evaluated the usability of the system with the System Usability Scale[23, 36], which is a ten point questionnaire. Each question is answered with a five-point Likert scale for each answer. It is important to note that there were too few samples (number of participants) for significant quantifiable data. Despite this fact, it is still worth using this data to get an idea of the system's usability, since the data was collected.

In order to interpret the scores, one has to subtract 1 from every odd question(e.g. question 1, 3, 5 etc.), and from every even question one needs to subtract the given score from 5. This yields answers from 0 to 4. These results are then added together and multiplied by 2.5, giving the scores a range of 0 to 100. It is important to note that this final score should not be considered as a percentage, but should only be considered for their percentile ranking. Previous research yielded 68 as the average SUS score, so anything with a score above that would imply a system's usability is above average. The SUS score in this study is 67,9, as can be seen in Appendix E. This is interesting, because the single-user scene navigation component seriously inhibited the multi-user aspect and had imposed many issues in the system's usability, as discussed in the previous section. Despite this, the system was still evaluated to have average usability. Had the scene navigation component been more usable, then this system would (most likely) obtained a higher SUS score, implying that system would have above average usability.

However, as already stated, these numbers cannot be considered, because such a small sample size (n=6) is not high enough for statistical significance.

7.3.5 Discussion

There was significant amount of feedback regarding the user experience. It was stated by both groups that adapting to such an unfamiliar kind of system takes time, but once the basics have been conceived, the interface was not very complex. Self documenting features like the gestures on the control ring for scene objects and the label toggle, which showed the name of each button on the respective interface, also proved to be useful. The fact that everything required by the user to keep going was on screen, minimised any disconnect users would otherwise experience if they had to look up controls in a separate manual, for example. Users mostly appreciated the entire minimalistic interface and enjoyed the simplicity of the controls. While a simple interface such as this works for low-fidelity prototyping, it would likely not scale with increased functionality, that would result in a more high-fidelity pre-visualisation tool. Furthermore, for more detailed animation (such as skeletal animation) and special effects, touch-based input may not be very practical. The reason for this is that accuracy of user input would decrease, compared to a desktop-based system, since using a finger as the input device would be a larger input point than a mouse pointer. Furthermore, using a keyboard and mouse could very well allow for more commands in a shorter time frame. Some functions would be specific/explicit to either input device, enabling users to combine mouse and keyboard inputs to easily accomplish complex tasks. Thus, multitouch based interaction is more suitable for rudimentary pre-visualisation, whereas desktop-based input is more appropriate for the high-fidelity equivalent.

In this study we posed the research question "Is multitouch suitable for multi-user pre-visualisation of computer animated scenes?". The previous sections present the data collected in the qualitative evaluation. According to both groups working on the same device, in the same room definitely helps participants communicate their thoughts. The director from the filmmakers emphasised this, in that normally he *"would have to do a lot of that by myself, but there's such a large component of it that's also collaborative"* (UserA2). In this regard, this research was a success, as communication (the underlying requirement for successful collaboration) is strongly enforced by this system. The purpose of pre-visualisation is to communicate the vision to the rest of the team, before continuing on to full production. Further feedback from the director was that, with this system he can *"talk through a scene in the way that it couldn't possibly be clearer unless you shoot the scene or the shot. I just think it's the best thing. I think it's incredible"* (UserA2). One could argue that users were excited to use a large, multitouch device, something that does not happen on a regular basis. On the contrary though, one of the participants found stated that working with a touch table is *"like it's overwhelming how much space there was"* (UserA1), which the other participants agreed with. Alternate suggestion for collaboration support included to work in a shared virtual environment over a network (local area network or over the internet). It is true that a lot of work in CSCW, particularly in the early days, focused on using networks for collaboration [45, 48, 97]. The limitations imposed when a network is used to share and synchronise virtual environments include latency and slow internet speed, which vary from country to country. One good compromise would be to work on a local area network (LAN), because that would not be dependent on internet speeds. Client server architectures like RTFX [97] enable digital assets to be easily transferable in real-time between modelling and animation packages, as well as game engines. Furthermore, these extensions can be written for any such package, according to Northam et al. As technology evolves, internet speed and quality are becoming less of an issue, but for now a LAN setup using something like RTFX would be a good alternative. Of course using a LAN requires participants to be in the same room, or at the very least in the same building. Communication might degrade, once participants are spatially separated from each other, which could in turn worsen the collaborative quality for low-fidelity pre-visualisation tasks. In fact, working on a single device with multiple users might be even quicker than being separated or having to go back and forth between all stakeholders, when trying to pre-visualise a scene. With this in mind, groupware is more appropriate for a multi-user planning tasks. In fact, stakeholders generally meet, when possible, to discuss scenes,

so the only required addition would be to add a multitouch device in the meeting room.

The one major drawback with the system's collaboration design was the viewpoint manipulation. This is particularly true for scenes with many camera angles, or where the view needs to be adjusted often. The collaboration design choice here is that only one user is allowed to change the view at any given time, and while doing so, the input of other users is ignored. The reason for this design choice was to separate view point manipulation from object manipulation and animation, using the respective modes. This was to prevent the case where one user is animating an object, and another user disrupting the animator by changing the view. This example of interference can easily hide objects that are being interacted with, thereby preventing further input affecting those objects. Unfortunately this design choice was over engineered, effectively degrading the full potential of the collaborative system. Both groups communicated verbally throughout the exercise, especially when the view point had to be adjusted. Therefore social collaboration, i.e. talking to each other while building the scene, was more than sufficient to prevent interference between users, as discussed above. This limitation severely reduced the efficiency with which a scene can be created. One option to fix this is to either have one control ring devoted solely to changing the viewpoint, while continuing to allow object manipulation, as was the case with previous iterations of this system. Another, quicker and more favourable solution is to simply omit the modal interaction, thus allowing each user to control the view with their own user widget. This solution is preferred, because each user can still navigate the virtual environment with their own, idealised view point manipulation preferences. It is reasonable to assume that a bottleneck affecting such a fundamental component of a system degrades the usability severely. Despite this drawback, the SUS score of 67.9 indicates that the system still has average usability, compared to the targeted "average usability" score of 68. This is not conclusive, though, as the small sample size ($n=6$) is insufficient for statistical significance. For future work this system would certainly need to undergo a quantitative usability evaluation. However, the focus of this work was to understand what is required to create a successful collaborative pre-visualisation system, and this motivated the use of qualitative evaluation.

7.3.6 Achievement of Design Goals

With the discussion above it can now be assessed whether the design goals, presented in Section 5.1, have been met. Goal A (the system's functionality and interaction strategies are quick to learn), was met as the user found that there was very little to learn with the system and it was a very minimalistic design. Goal B (there is little to no confusion about which user is controlling which object) was also met, as the users were constantly communicating and using social cues to determine who is in control of which part of the previsualisation tool. Goal C (viewpoint manipulation is very easy and intuitive) was not successfully met, as users suggested alternate, finer input controls for making small changes to the viewpoint, whereas more predefined perspectives would be useful for large changes to the viewpoint. There is a deeper discussion of this in the Viewpoint Navigation paragraph of Section 7.3.3 and Section 7.3.5. One could argue that goal D (render camera manipulation is easy and intuitive) was partially met, as the render camera manipulation is consistent with the object manipulation. However, users suggested that one should use the viewpoint as the camera too, which would effectively eliminate an unnecessary, additional component of the system, and transfer those to the viewpoint. As such one should conclude that goal D was not, since users thought this component was unnecessary. Goal E (object manipulation is easy and intuitive) was met, and, in particular, users found multitouch suitable for interacting with these objects to perform low fidelity animations. Something that could have been though is the configurability of the previz objects UI, to add additional functionality, etc., which is discussed more in Section 8.2. Goal F) (the feedback for all interactions is sufficient and unambiguous) was also met, since the users found the system quite easy to use. They did have problems with the scene navigation, which is did not meet this goal. Goal G) (the overall system is easy to use) was also met for the most part, since the users stated that they found the interface minimalistic and once they figured out how to use the various components. However, one major limitation was the viewpoint manipulation. This system would become cumbersome when it is used to construct a scene with a

large variety of different camera angles, as this would require the viewpoint to be adjusted frequently. Since the viewpoint manipulation was a drawback in the interface, and this can have an effect on the entire user experience, it can be said that the system was easy to use, apart from the viewpoint manipulation. Overall, the system met most of its design goals, with some exceptions, namely the lack of customisability and the viewpoint manipulation.

Chapter Summary

This chapter covered the essentials of qualitative evaluation and compared them to quantitative evaluation. Thereafter, the entire procedure for conducting the qualitative evaluation, was laid out, including priming tasks, user tasks and the interview, which revealed these key features and underlying issues of the system. The following chapter restates the key features and concludes this research with recommendations for future work.

Chapter 8

Conclusion

This study aimed to assess how suitable multitouch is for same-time, same-place collaboration on a single device for low-fidelity previsualisation of computer animation. Previsualisation is a collaborative process, bringing writers and directors together with artists and animators, to plan the composition and timing of their envisioned animation. To drive the support the collaboration, we created a previsualisation tool that utilises a large multitouch device on which participants work together to construct low-fidelity, animated scenes. The research began with implementing feasibility prototypes, followed by interviews of industry specialists to gather the fundamental requirements for a previsualisation tool. Thereafter, paper prototypes were created to incorporate the required features in a simple and consistent manner. Once these were approved, the development took place, repeatedly iterating between implementation and heuristic evaluations to improve the system. The final system was evaluated using two groups of specialists, one consisting of the software developers, and the other being from the filmmaking industry. Chapter 7 details the theory and procedure of the qualitative evaluation used to assess this system. Observations were made in the evaluation data, which indicated the most significant strengths and weaknesses of the system. This chapter concludes the research report, by stating the findings, followed by recommendations for future work.

8.1 Findings

This study has implications on low-fidelity animated previsualisation, using a single device to support collaboration. Overall, users found the general user interface and layout of all components straight forward and minimalistic. Apart from the animation timeline, all other user interface components were circular and consistent where possible. This made sense to the users, because they could reorient the circular interfaces without affecting the amount of screen real estate being used by the widgets. While this design choice worked for a low fidelity system, the users realised that such an interface may not be able to hold up with increasing functionality.

The evaluation also indicates that the single device setup strongly promotes communication between the involved parties. A common issue is communication in planning films where different stakeholders are involved at varying stages. Miscommunication during planning can easily lead to multiple cycles of performing the same previsualisations, until every stakeholder is satisfied with the result. Bringing all the users to the same room severely reduces the potential for miscommunication, thereby speeding up the previsualisation phase. One of the test subjects, a director, found with this system one can *"talk through a scene in the way that it couldn't possibly be clearer unless you shoot the scene or the shot"*.

That being said, this system is designed for low-fidelity previsualisation and not very suitable for highly detailed scenes. Multitouch is not very suitable for detailed animation, due to the vast amount of controls available, which would take up much more screen real-estate in order to by usable for touch input.

One drawback of the system was controlling the viewpoint, a single user task, which was a bottleneck for multi-user interaction, since no other user could interact with the scene objects in this circumstance. The other drawback was the amount of freedom a user had when manipulating the viewpoint. Users found that it could have been better to limit the viewpoint manipulation to a single axis at a time, rather than only using the full freedom of an arcball camera.

8.2 Future Work

A number of suggestions for features and improvements emerged from the evaluation. Some of these items should definitely be implemented in further research, to improve the system.

One notable request was for axis specific transformations of objects. This could be done in the conventional approach, which is to use manipulators attached to 3d objects, as shown in the background chapter.

User Interface

Another improvement would be to show a list of all scene objects on the GUI. This list would reduce provide a variety of functionality and can reduce the number of interactions required to accomplish a task, compared to the current implementation. Making annotations regarding certain aspects of the scene or general comments would also be useful to quickly communicate thoughts. These notes could be on the timeline, placed in the 3d environment or attached to specific objects.

Naturally, a feature useful in most software, and part of Nielsen's Ten Usability Design Principles is the ability to undo actions. This could be implemented such that each component has it's own undo stack, rather than having one undo stack for the entire system.

Grouping objects is also desirable in cases where a number of objects move together. This is certainly the case in scenes such as the one used in the system evaluation, in which a jet took off and the camera followed it (see the storyboard in Appendix D).

Viewpoint

While the GUI can be oriented to suit users on any side of a touch table, this is not true for the viewpoint, i.e. actual 3D scene. The only perspective from which the viewpoint displays the scene in a natural way from any angle is the top-down perspective. All other perspectives can potentially result in an upside down scene for some point of view around the table. To achieve this functionality one could implement a two finger rotation gesture to pivot the view point around it's z-axis, similar to the way it works on Google Maps[16].

Another way to improve the viewpoint manipulation, would be to snap the view to fully horizontal, top down, etc. whenever a user approaches one of these orientations. In this way the freedom of manipulating the viewpoint would be less of an issue.

Lastly, it is better to use the viewpoint as the render camera as well, rather than having to manipulate another virtual object to capture the scene. Firstly, the currently existing view point has a wide range of motion and more ways in which its perspective can be manipulated than the render camera object. Furthermore, with this approach one would not have to toggle between viewports to show what is within the render camera's view frustum, since the view point abides to the paradigm what-you-see-is-what-you-get (WYSIWYG[67]).

Animation

While the animation component accomplished the bare minimum, it can definitely be improved. Skeletal animation, or at the very least a collection of basic, configurable animations (e.g. walking, running, jumping, etc.) are recommended for a more complete previsualisation tool. Furthermore, these animations should be configurable, allowing users to create alternate interpolation types for the actual animation (e.g. slow-in slow-out). As suggested by participants in the evaluation, the animation path needs to be more curved and not only adjustable at the keyframes, but at any point on the curve.

This will create a more natural looking animation path and will provide users with more freedom in creating animations. Furthermore, a more sophisticated method of drawing animation paths should also be included. Finally, when users scroll through the created animation using the timeline, the objects and camera should be updated, to reflect the correct point in time on the animation, i.e. scrubbing. This interactive feature will provide users with immediate feedback about the state of the scene at any given instant.

Chapter Summary

The results of this research suggest that groupware can certainly improve collaboration, particularly in terms of communication, when the participants co-located. Even more so, working on the same device ensures they are aware of exactly what is happening in their workspace, as they can see each other's interactions. This is less true for physically separated workspaces. However, as this system was merely the outcome of iteratively improved prototypes, some high-fidelity features were desired during the evaluation, and should certainly be included in future work.

Bibliography

[1] "10 Heuristics for User Interface Design: Article by Jakob Nielsen." [Online]. Available: http://www.nngroup.com/articles/ten-usability-heuristics/

[2] "2D Traditional Animation." [Online]. Available: https://www.youtube.com/watch?v=lr-V5p7gc_Q

[3] "American Beauty Storyboard." [Online]. Available: http://nofilmschool.com/sites/default/files/uploads/2013/09/American-Beauty-storyboard.jpg

[4] "Animation Tidbits." [Online]. Available: http://animationtidbits.tumblr.com/post/92597087481/okami23-tangled-concept-art-vs-final-version

[5] "Apple - QuickTime." [Online]. Available: https://www.apple.com/za/quicktime/

[6] "blender.org - Home of the Blender project - Free and Open 3D Creation Software." [Online]. Available: http://www.blender.org/

[7] "Case Study of Animation Pioneers | amyrheadaca on WordPress.com." [Online]. Available: https://amyrheadaca.wordpress.com/2012/09/21/peter-mark-roget-persistence-of-vision/

[8] "Claymation Movies, Claymation videos, Software for Claymation." [Online]. Available: http://www.stopmotioncentral.com/clayanimation.html

[9] "Evaluation methods in the CSCW literature," Ph.D. dissertation. [Online]. Available: http://www.comp.lancs.ac.uk/computing/research/cseg/projects/evaluation/lit_methods.html

[10] "Figure One Films." [Online]. Available: http://figureonefilms.com/

[11] "Final Fantasy VII: Advent Children (2005) Trailer." [Online]. Available: https://www.youtube.com/watch?v=wut2am39z-c

[12] "Frameforge Previz Studio 3 - Redefining the Art of Preproduction." [Online]. Available: http://www.frameforge3d.com/Products/

[13] "GarageBand on the App Store on iTunes." [Online]. Available: https://itunes.apple.com/en/app/garageband/id408709785?mt=8

[14] "GarageBand updated with collaborative jam sessions | iMore." [Online]. Available: http://www.imore.com/garageband-updated-collaborative-jam-sessions

[15] "GoldenCompassBlimp.jpg." [Online]. Available: http://www.onehotpixel.com/images/previs_GC_03_M.jpg

[16] "Google Maps." [Online]. Available: https://www.google.com/maps/@-33.9121751,18.4098014,15z

[17] “iClone5 - Real-time 3D Animation Software.” [Online]. Available: http://www.reallusion.com/iclone/

[18] “Interviewing in Qualitative Research,” pp. 20–30.

[19] “iOS Collaboration With GarageBand.” [Online]. Available: http://forums.macrumors.com/showthread.php?t=1181697

[20] “Ironman.” [Online]. Available: https://www.fxguide.com/wp-content/uploads/2012/05/ironman.jpg

[21] “Ironman Lab.” [Online]. Available: http://www.cgw.com/images/media/2013-06/IronMan_03.jpg

[22] “Lung Animation | VFX | Animation Studio | Film | Motion Graphics | 3D.” [Online]. Available: http://lung.co.za/

[23] “Measuring Usability with the System Usability Scale (SUS): MeasuringU.” [Online]. Available: http://www.measuringu.com/sus.php

[24] “modelsheet.jpg.” [Online]. Available: https://cdn.tutsplus.com/cg/uploads/legacy/168_Article_StepByStepAnimatedMovie/modelsheet.jpg

[25] “Pre-Production Month: Layout by theanimationsource on DeviantArt.” [Online]. Available: http://theanimationsource.deviantart.com/journal/Pre-Production-Month-Layout-223403297

[26] “previs_golden_compass.jpg.” [Online]. Available: http://www.onehotpixel.com/images/previs_GC_05_M.jpg

[27] “Steam, The Ultimate Online Game Platform.” [Online]. Available: http://store.steampowered.com/about/

[28] “System Guidelines for Co-located, Collaborative Work on a Tabletop Display.” [Online]. Available: https://hci.stanford.edu/publications/2003/tabletopguidelines/tabletopguidelines.pdf

[29] “Triggerfish Animation Studios.” [Online]. Available: https://www.triggerfishstudios.com/en/

[30] “Valve.” [Online]. Available: http://www.valvesoftware.com/

[31] “What Is Previs 028.jpg.” [Online]. Available: https://library.creativecow.net/articles/cline_gare/Previs_intro/assets/What-Is-Previs_028.jpg

[32] “YouTube.” [Online]. Available: https://www.youtube.com/

[33] “Moviestorm,” 2013. [Online]. Available: http://www.moviestorm.co.uk/

[34] “Animation Timeline Editing,” 2015. [Online]. Available: http://www.reallusion.com/crazytalk/Animator/help/animator/Pro/09_Timeline/Animation_Timeline_Editing.htm

[35] Acet Inc., “Selecting an Evaluation : Qualitative , Quantitative , and Mixed Methods Approaches,” 2012. [Online]. Available: http://www.acetinc.com/assets/EvalTakeAways/ACET_QQMM_12-21-12.pdf

[36] A. S. f. P. Affairs, “System Usability Scale (SUS),” sep 2013. [Online]. Available: http://www.usability.gov/how-to-and-tools/methods/system-usability-scale.html

[37] H. Amoore and Writing Centre - University of Cape Town, “University of Cape Town - Avoiding Plagiarism: A Guide for Students,” Cape Town, pp. 1–3, 1999. [Online]. Available: http://www.uct.ac.za/downloads/uct.ac.za/about/policies/plagiarism_students.pdf

[38] A. N. Antle, A. Bevans, J. Tanenbaum, K. Seaborn, and S. Wang, "Futura: Design for Collaborative Learning and Game Play on a Multi-touch Digital Tabletop," School of Interactive Arts and Technology, Tech. Rep. [Online]. Available: http://www.antle.iat.sfu.ca/Futura/papers/Antle_TEI11_FuturaLearning.pdf

[39] Apple Inc., "Apple Maps." [Online]. Available: https://mapsconnect.apple.com/

[40] Autodesk, "3D Animation Software, Computer Animation Software | Maya | Autodesk." [Online]. Available: http://www.autodesk.com/products/maya/overview

[41] ——, "3d Modeling and Rendering Software | 3ds Max | Autodesk." [Online]. Available: http://www.autodesk.com/products/3ds-max/overview

[42] R. Bade, F. Ritter, and B. Preim, "Usability comparison of mouse-based interaction techniques for predictable 3d rotation," *Smart Graphics*, vol. 3638, pp. 138–150, 2005. [Online]. Available: http://www.springerlink.com/index/5egjuvehfq4lgat0.pdf

[43] K. Baker, S. Greenberg, and C. Gutwin, "Empirical development of a heuristic evaluation methodology for shared workspace groupware," *Proceedings of the 2002 ACM conference on Computer supported cooperative work - CSCW '02*, p. 96, 2002. [Online]. Available: http://portal.acm.org/citation.cfm?doid=587078.587093

[44] R. Balakrishnan and G. Kurtenbach, "Exploring Bimanual Camera Control and Object Manipulation in 3D Graphics Interfaces," *Proceedings of the SIGCHI conference on Human factors in computing systems: the CHI is the limit*, no. May, pp. 56–62, 1999.

[45] R. Bentley, T. Horstmann, and J. Trevor, "The World Wide Web as Enabling Technology for CSCW : The Case of BSCW," *Computer Supported Cooperative Work*, vol. 6, no. 2, pp. 111–134, 1997. [Online]. Available: http://www.springerlink.com/content/u4358784023m6857/

[46] A. Benzina, F. A. Reality, F. A. Reality, M. Ashry, N. C. City, M. Entrance, A. Tagamoa, G. Klinker, and F. A. Reality, "Phone-Based Motion Control in VR - Analysis of Degrees of Freedom," pp. 1519–1524, 2011.

[47] R. Bernard, *Handbook of Methods in Cultural Anthropology (2nd edition)*, vol. 5, no. 1.

[48] J. Bowers, "The work to make a network work: studying CSCW in action," *Proceedings of the 1994 ACM conference on Computer supported cooperative work - CSCW '94*, pp. 287–298, 1994. [Online]. Available: http://dl.acm.org/citation.cfm?id=193030$\delimiter"026E30F$nhttp://portal.acm.org/citation.cfm?doid=192844.193030

[49] P. Brandl, J. Leitner, T. Seifried, M. Haller, B. Doray, and P. To, "Occlusion-aware menu design for digital tabletops," *Proceedings of the 27th international conference extended abstracts on Human factors in computing systems - CHI EA '09*, p. 3223, 2009. [Online]. Available: http://dl.acm.org/citation.cfm?id=1520340.1520461

[50] M. Chen, S. J. Mountford, and A. Sellen, "A study in interactive 3-D rotation using 2-D control devices," *15th annual conference on Computer graphics and interactive techniques - SIGGRAPH '88*, vol. 22, no. 4, pp. 121–129, 1988. [Online]. Available: http://dl.acm.org/citation.cfm?id=378497$\delimiter"026E30F$nhttp://portal.acm.org/citation.cfm?doid=54852.378497

[51] E. Chow, A. Hammad, and P. Gauthier, "Multi-touch screens for navigating 3D virtual environments in participatory urban planning," Concordia University, Tech. Rep., 2011. [Online]. Available: http://portal.acm.org/citation.cfm?id=1979852

[52] R. Davies, "Adapting virtual reality for the participatory design of work environments," Tech. Rep. 1, mar 2004. [Online]. Available: http://link.springer.com/article/10.1023/B:COSU.0000014985.12045.9c

[53] P. Debevec, "Rendering Synthetic Objects into Real Scenes : Bridging Traditional and Image-based Graphics with Global Illumination and High Dynamic Range Photography."

[54] B. Donahue, "Data Analytic Practices 1."

[55] Epic Games, "UDK3 | WebHome," Cary, North Carolina. [Online]. Available: https://udn.epicgames.com/Three/WebHome.html

[56] ——, "UnrealEngine - Game Development Tools and Gaming Engine for Game Developers | Unreal Engine 4," 2014. [Online]. Available: https://www.unrealengine.com/what-is-unreal-engine-4

[57] J.-D. Fekete, É. Bizouarn, É. Cournarie, T. Galas, and F. Taillefer, "TicTacToon: A Paperless System for Professional 2-D Animation," *Proceedings of ACM SIGGRAPH 95 (Los Angeles, CA, August 6–11, 1995)*, pp. 79–90, 1995. [Online]. Available: http://dl.acm.org.ezproxy.uct.ac.za/citation.cfm?id=218417

[58] Formula D Interactive, "Design development of Multi-Touch Walls - Formula D interactive, South Africa." [Online]. Available: http://www.formula-d.com/multi-touch_wall_fdi100.html

[59] A. French, M. Macedo, J. Poulsen, T. Waterson, and A. Yu, "Multivariate Analysis of Variance (MANOVA)," *San Francisco State University*, pp. 1–8, 2002. [Online]. Available: http://online.sfsu.edu/{~}efc/classes/biol710/manova/manova.htm.

[60] J. Gabbard, D. Hix, and I. Swan, J.E., "User-centered design and evaluation of virtual environments," *IEEE Computer Graphics and Applications*, vol. 19, no. 6, pp. 51–59, 1999.

[61] B. G. Glaser, A. L. Strauss, A. T. Paul, S. Kaufmann, and B. Hildenbrand, *Grounded theory: strategien qualitativer forschung.* Huber, 2010.

[62] C. E. Glidden, "How to Design and Evaluate Research in Education (2nd ed.)." *PsycCRITIQUES*, vol. 38, no. 10, 1993.

[63] Government Accounting Office, "Quantitative Data Analysis: An Introduction," no. May, p. 134, 1992. [Online]. Available: http://www.gao.gov/special.pubs/pe10111.pdf

[64] J. Grudin, "Why Cscw Applications Fail : Problems in the Design and Evaluation of Organizational Interfaces," Tech. Rep., 1988. [Online]. Available: http://portal.acm.org/citation.cfm?id=62266.62273

[65] P. Gulati, "Step-by-Step : How to Make an Animated Movie - Tuts+ 3D & Motion Graphics Article," 2010. [Online]. Available: http://cgi.tutsplus.com/articles/step-by-step-how-to-make-an-animated-movie--cg-3257

[66] M. Hancock, S. Carpendale, and A. Cockburn, "Shallow-Depth 3D Interaction : Design and Evaluation of One-, Two- and Three-Touch Techniques," pp. 1147–1156, 2007.

[67] P. Hanrahan and P. Haeberli, "Direct WYSIWYG painting and texturing on 3D shapes," *ACM SIGGRAPH Computer Graphics*, vol. 24, no. 4, pp. 215–223, 1990.

[68] T. E. Hansen and J. P. Hourcade, "Comparing Multi-Touch Tabletops and Multi-Mouse SingleDisplay Groupware Setups," University of Iowa, Tech. Rep. [Online]. Available: http://homepage.cs.uiowa.edu/{~}hourcade/sdg-vs-mt-final.pdf

[69] L.-w. He, M. Cohen, and D. Salesin, "The virtual cinematographer: a paradigm for automatic real-time camera control and directing," Ph.D. dissertation, 1996. [Online]. Available: http://dl.acm.org/citation.cfm?id=237259

[70] J. Hincapié-Ramos and X. Guo, "Consumed Endurance: A metric to quantify arm fatigue of mid-air interactions," *Proceedings of the 32nd ...*, pp. 1063–1072, 2014. [Online]. Available: http://dl.acm.org/citation.cfm?id=2557130

[71] K. Hinckley, J. Tullio, R. Pausch, D. Proffitt, and N. Kassell, "Usability Analysis of 3D Rotation Techniques," *Proceedings of the 10th annual ACM symposium on User interface software and technology - UIST '97*, pp. 1–10, 1997. [Online]. Available: http://dl.acm.org/citation.cfm?id=263407.263408

[72] S. Hughes and Intel, "Adding Multi-Touch Support to Unity Games on Microsoft Windows 7 and Windows 8 Desktop," Ph.D. dissertation, 2013. [Online]. Available: https://software.intel.com/en-us/articles/adding-multi-touch-support-to-unity-games-on-microsoft-windows-7-and-windows-8-desktop

[73] Ideum, "Touch Walls | Products | Ideum." [Online]. Available: http://ideum.com/touch-walls/

[74] A. Ingram, X. Wang, and W. Ribarsky, "Towards the establishment of a framework for intuitive multi-touch interaction design," *Proceedings of the International Working Conference on Advanced Visual Interfaces - AVI '12*, p. 66, 2012. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2254556.2254571

[75] M. Jones and G. Marsden, "Mobile Interaction Design," Ph.D. dissertation, 2005.

[76] J. T. Kajiya, "The rendering equation," *ACM SIGGRAPH Computer Graphics*, vol. 20, no. 4, pp. 143–150, 1986. [Online]. Available: http://dl.acm.org.ezproxy.uct.ac.za/citation.cfm?id=15902

[77] B. Kaplan and J. Maxwell, "Qualitative Research Methods for Evaluating Computer Information Systems," pp. 30–56.

[78] P. Kipfer, M. Segal, and R. Westermann, "UberFlow : A GPU-Based Particle Engine," pp. 115–123, 2004.

[79] S. Kratz and M. Rohs, "Extending the virtual trackball metaphor to rear touch input," *3DUI 2010 - IEEE Symposium on 3D User Interfaces 2010, Proceedings*, pp. 111–114, 2010.

[80] R. Kruger, S. Carpendale, S. D. Scott, A. Tang, and C. Tn, "Fluid Integration of Rotation and Translation," Tech. Rep., 2005.

[81] N. Lemon, "Previsualization in Computer Animated Filmmaking," MA, Ohio State University, 2012.

[82] Lenovo, "Lenovo IdeaCentre A720 Datasheet," Ph.D. dissertation.

[83] Y. Magallanes, A. Molina, and J. A. Sánchez, "Combining gestures and graphical elements for collaboration using multi-touch surfaces," Tech. Rep., 2012.

[84] L. Manovich, "The Language of New Media Copyright," *MIT Press*, 2001.

[85] D. Marchal, C. Moerman, G. Casiez, and N. Roussel, "Designing Intuitive Multi-touch 3D Navigation Techniques," Tech. Rep., 2013.

[86] A. Martinet, G. Casiez, and L. Grisoni, "The design and evaluation of 3D positioning techniques for multi-touch displays," Tech. Rep., mar 2010. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5444709

[87] J. McDavid and L. Hawthorn, "Applying Qualitative Evaluation Methods," *Program evaluation and performance measurement: An introduction to practice*, pp. 165–200, 2006. [Online]. Available: http://books.google.com/books?hl=en{&}lr={&}id=ldYmClf6mGkC{&}oi=fnd{&}pg=PR15{&}dq=Program+Evaluation+and+Performance+Measurement:+An+Introduction+to+Practic{&}ots=ME_jI8soSP{&}sig=ym6FtTcRW46NTu2cWpem1yRG8CY

[88] N. Mehta, "A flexible machine interface," *MA Sc. Thesis, Department of Electrical Engineering, University of Toronto supervised by Professor KC Smith*, 1982.

[89] M. Müller, D. Charypar, and M. Gross, "Particle-Based Fluid Simulation for Interactive Applications," pp. 154–160, 2003.

[90] D. C. Neale, J. M. Carroll, and M. B. Rosson, "Evaluating Computer-Supported Cooperative Work : Models and Frameworks," Ph.D. dissertation, 2004.

[91] J. Nielsen, "10 Heuristics for User Interface Design." [Online]. Available: http://www.useit.com/papers/heuristic/heuristic_list.html

[92] ——, "Heuristic Evaluation," Ph.D. dissertation. [Online]. Available: http://www.useit.com/papers/heuristic/

[93] ——, "Usability 101: Definition and Fundamentals - What, Why, How (Jakob Nielsen's Alertbox)," Ph.D. dissertation. [Online]. Available: http://www.useit.com/alertbox/20030825.html

[94] M. Nitsche, "Experiments in the Use of Game Technology for Pre-Visualization," Georgia Institute of Technology, Tech. Rep., 2008.

[95] D. Norman and S. W. Draper, "USER CENTERED SYSTEM DESIGN New Perspectives on Human-Computer Interaction Edited by," pp. 87–124, 1986.

[96] D. A. Norman, "Natural User Interfaces Are Not Natural," Ph.D. dissertation, 2010.

[97] L. Northam, J. Istead, and C. S. Kaplan, "A collaborative real time previsualization tool for video games and film," New York, New York, USA, Tech. Rep., 2012. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2342896.2343036

[98] D. Nowrouzezahrai, J. Johnson, A. Selle, D. Lacewell, and M. Kaschalk, "A Programmable System for Artistic Volumetric Lighting," vol. 1, no. 212, pp. 1–8, 2011.

[99] D. Nunez and E. Blake, "Cognitive presence as a unified concept of virtual reality effectiveness," *ACM International Conference on Computer Graphics, Virtual Reality and Visualisation in Africa*, pp. 115–118, 2001. [Online]. Available: http://dl.acm.org/citation.cfm?id=513892$\delimiter"026E30F$nhttp://www.scopus.com/inward/record.url?eid=2-s2.0-1442328765{&}partnerID=40{&}md5=b68301b166032e3e97f14a271c51f2fd

[100] ——, "Conceptual Priming as a Determinant of Presence in Virtual Environments," Ph.D. dissertation, 2003.

[101] W. T. Reeves, "Particle Systems - A Technique for Modeling a Class of Fuzzy Objects," vol. 2, no. 2, pp. 91–108, 1983.

[102] W. Reinhard, J. Schweitzer, G. Volksen, and M. Weber, "CSCW tools: concepts and architectures," *Computer*, vol. 27, no. 5, pp. 28–36, 1994.

[103] T. Rodden, "A survey of CSCW systems," *Interacting with Computers*, vol. 3, no. 3, pp. 319–353, 1991.

[104] T. Ropinski and D. Christian, "Interactive Volumetric Lighting Simulating Scattering and Shadowing," pp. 169–176.

[105] J. Rudd, K. Stern, and S. Isensee, "Low vs. high-fidelity prototyping debate," *Interactions*, vol. 3, no. 1, pp. 76–85, 1996.

[106] I. Sams and J. Wesson, "Can Multi-touch Interaction Techniques be used to Support Collaborative Information Retrieval ?"

[107] D. Scheurich and W. Stuerzlinger, "A One-Handed Multi-Touch Method for 3D Rotations," Ph.D. dissertation.

[108] K. Schmidt and L. Bannon, "Constructing CSCW: The First Quarter Century," Ph.D. dissertation, aug 2013. [Online]. Available: http://link.springer.com/10.1007/s10606-013-9193-7

[109] T. Seifried, M. Haller, S. D. Scott, F. Perteneder, C. Rendl, D. Sakamoto, and M. Inami, "CRISTAL : A Collaborative Home Media and Device Controller Based on a Multi-touch Display," Ph.D. dissertation, 2009.

[110] M. Shapiro, "The Novice User's Camera Control Interface (NUCCI) A real-time cinematic solution to previsualization," Tech. Rep. [Online]. Available: https://www.researchgate.net/publication/238688293_The_Novice_User's_Camera_Control_Interface_NUCCI_A_real-time_cinematic_solution_to_previsualization

[111] C. Shen, "DiamondSpin : An Extensible Toolkit for Around-the-Table Interaction," Ph.D. dissertation, 2004.

[112] K. Shoemake, "ARCBALL : A User Interface for Specifying Three-Dimensional Orientation Using a Mouse," Ph.D. dissertation.

[113] ——, "ARCBALL: A user interface for specifying three-dimensinal orientation using a mouse," *Proceedings of Graphics Interface '92*, pp. 151–156, 1992.

[114] Side Effects Software Inc., "Houdini Apprentice." [Online]. Available: http://www.sidefx.com/index.php?option=com_download{&}task=apprentice{&}Itemid=208

[115] V. Simonov and Interactive Lab, "TouchScript," Ph.D. dissertation, 2014. [Online]. Available: http://touchscript.github.io/

[116] K. Sims and N. Highland, "Particle Animation and Rendering Using Data Parallel Computation," vol. 24, no. 4, 1990.

[117] K. M. Stanney, M. Mollaghasemi, L. Reeves, R. Breaux, and D. a. Graeber, "Usability engineering of virtual environments (VEs): Identifying multiple criteria that drive effective VE system design," Ph.D. dissertation, University of Central Florida, 2003.

[118] M. Stefik, D. G. Bobrow, G. Foster, S. Lanning, and D. Tatar, "WYSIWIS revised: early experiences with multiuser interfaces," *ACM Transactions on Information Systems*, vol. 5, no. 2, pp. 147–167, 1987.

[119] R. Tenmoku, "Design and Prototype Implementation of MR Pre-Visualization Workflow," pp. 1–7.

[120] G. This, I. The, O. Culture, C. The, and O. S. Orga, "Measuring Organiza- tional Cultures ; A Quali- tative and Quantitative Study across Twenty Cases Geert Hofstede Bram Neuijen Denise Davat Ohayv Geert Sanders," *Science*, vol. 35, no. 2, pp. 286–316, 1990. [Online]. Available: http://www.questia.com/PM.qst?a=o{&}amp$\delimiter"026E30F$nse=gglsc{&}amp$\delimiter"026E30F$nd=5000115551

[121] B. Tóth and T. Umenhoffer, "Real-time Volumetric Lighting in Participating Media," Budapest University of Technology and Economics, Budapest, Tech. Rep., 2009.

[122] Unity Technologies, "Unity - Game Engine." [Online]. Available: http://unity3d.com/

[123] Valve, "Source Filmmaker," Ph.D. dissertation, 2014. [Online]. Available: http://www.sourcefilmmaker.com/

[124] VisualPlanet, "Visualplanet touchfoil | Through Window Touch," Ph.D. dissertation. [Online]. Available: http://www.touchscreen-me.com/through-window-touch-vip.php

[125] J. Wainer and C. Barsottini, "Empirical research in CSCW - a review of the ACM/CSCW conferences from 1998 to 2004," Ph.D. dissertation, 2007. [Online]. Available: http://www.springerlink.com/index/F5857256NG420067.pdf$\delimiter"026E30F$nhttp://www.springerlink.com/index/10.1007/BF03192543

[126] M. Weiss, J. Wagner, R. Jennings, Y. Jansen, and R. Khoshabeh, "SLAPbook : Tangible Widgets on Multi-touch Tables in Groupware Environments," *Proceedings of the 3rd International Conference on Tangible and Embedded Interaction*, pp. 297–300, 2009.

[127] J. Wesson, D. Vogts, and I. Sams, "Exploring the Use of a Multi-touch Surface to support Collaborative Information Retrieval," in *SAICSIT 2012*, Port Elizabeth, 2012. [Online]. Available: http://dl.acm.org/citation.cfm?id=2389870

[128] R. C. Zeleznik, K. P. Herndon, and J. F. Hughes, "SKETCH : An Interface for Sketching 3D Scenes," Ph.D. dissertation, 1910.

[129] Y. J. Zhao, D. Shuralyov, and W. Stuerzlinger, "Comparison of multiple 3D rotation methods," *VECIMS 2011 - 2011 IEEE International Conference on Virtual Environments, Human-Computer Interfaces and Measurement Systems, Proceedings*, pp. 13–17, 2011.

Appendix A

Usability Design

This section lists best practices in usability design that one should abide by to increase the likelihood of creating a usable interface.

Nielsen's Ten Usability Heuristics

A widely accepted and good set of design principles for user interfaces are Jacob Nielsen's Ten Usability Heuristics, listed below.

- ***Visibility of System Status***

When interacting with a system, users should always be aware of what is going on. This is achieved with suitable feedback within reasonable time.

- ***Match between system and the real world***

The system's use of language should be comprehensive to the user, rather than being system-oriented. This can be achieved by making information appear in a logical and natural order by following real-world conventions.

- ***User control and freedom***

It is common for user's to accidentally choose undesired system functions. In such cases, users should be able to recover quickly from an unwanted state, without having to go through extended dialogs. Undo and redo functionality should be supported.

- ***Consistency and standards***

It should be clear to the user what a particular phrase, word or situation means. By following platform conventions, users would not have to wonder if different terms mean the same thing.

- ***Error prevention***

While it is good to inform users when an error has occurred, it is better to prevent a problem from occurring at all. Error-prone conditions should either be eliminated or checked for, in order to provide users with a confirmation option before they perform the action.

- ***Recognition rather than recall***

Users should not have to remember much information when using a system. Instead, usage instructions should either be clearly visible or easily retrievable whenever appropriate.

- ***Flexibility and efficiency of use***

Novice users often take the obvious, longer route to solve a problem, whereas experienced users may want to use shortcuts to accelerate the interaction.

- ***Aesthetics and minimalist design***

Information that is irrelevant or rarely needed, should not be present in dialogues. Any extra piece of information can distract users and may reduce their attention to the significant bits information. Additionally, aesthetics tend to improve results in usability testing. If the interface looks nice, people become more creative and solve more problems.

- ***Help users recognize, diagnose and recover from errors***

When an error occurs, it should be stated in a manner that is understandable to the user, indicating the problem explicitly and suggesting a solution.

- ***Help and documentation***

A good user interface would not have to be accompanied by a documentation of its usage. Nonetheless, in case of uncertainty a user should always be able to quickly look up a specific task, without having to do much reading (unless the task is not a simple one to solve, of course).

Design Criteria

In addition to the aforementioned usability heuristics, the following are a list of criteria [75], which aid in designing and implementing efficient, user-friendly interface.

- ***Affordances*** of an object are properties of the object which give users clues as to how the device is used.
- ***Mapping*** is concerned with ensuring that there is a natural correlation between objects and the interface controlling them.
- ***Constraints*** on a design are made ensure that it can only be used the correct way.
- ***Visualizing*** ensures that features are made visible to the user. Bad examples would be command line interfaces, whereas good examples are graphical menu system, presenting all possible actions.
- ***Simplicity*** is often said to be good. While simpler is always better, things should not be made too simple though.
- ***Consistency*** is vital in interface design. To increase the usability of an interface, it is best to use interface elements that users are already familiar with from other interfaces.

Appendix B

Touch Gesture Integration

The table below illustrates the integration touch gestures for the various user interfaces, to show consistency and indicate the lack of overlap and conflict between gestures.

Touch Gesture	3D Objects	Content Loader	Navigation
 single tap	select object (activates ring interface RI1)	N/A	navigate to an object
 double tap	N/A	spawn content loader (activates ring interface RI2)	navigate to an arbitrary point
 1 finger drag	rotate object arbitrarily using arcball	N/A	ArcBall: rotate around focal point 1stPerson: look around in environment
 2 finger drag	translate object parallel to the camera plane	move interface to reposition spawn point of objects	ArcBall: translate focal point 1stPerson: move viewpoint
 Pinch in/out	Scale object	N/A	ArcBall: zoom in/out of view point 1stPerson: translate viewpoint along Z-axis
			Perhaps: bring up a menu to choose viewing settings?

Figure B.1: Touch gesture integration into system's various UI components.

Appendix C

Ethics Clearance Letter

Faculty of Science
University of Cape Town
RONDEBOSCH 7701
South Africa

E-mail: richard.hill@uct.ac.za
Telephone: + 27 21 650 2786
Fax: + 27 21 650 3456

5 February 2015

Kwegyir (Bilo) Lwabona
Department of Computer Science

MULTITOUCH-BASED COLLABORATIVE PREVISUALISATION FOR COMPUTER ANIMATION

Dear Kwegyir (Bilo) Lwabona

I am pleased to inform you that the Faculty of Science Research Ethics Committee has approved the above-named application for research ethics clearance, subject to the conditions listed below. You are required to:

- implement the measures described in your application to ensure that the process of your research is ethically sound; and
- uphold ethical principles throughout all stages of the research, responding appropriately to unanticipated issues: please contact me if you need advice on ethical issues that arise.

Your approval code is: FSREC 01– 2015

I wish you success in your research.

Yours sincerely

Dr Richard Hill
Chair: Faculty of Science Research Ethics Committee

Cc: A/Prof James Gain, Supervisor

Figure C.1: Ethics clearance approval letter for conducting user experiments to evaluate the system.

Appendix D

Experiment Forms

Consent Form

- I agree to participate in this experiment.
- I agree to my responses being used for education and research.
- I understand that my personal information will only be used in aggregate form, so that I will not be personally identifiable.
- I understand that any material captured by this experiment is to be treated confidentially and none of it will be released to the public.
- I understand that I am under no obligation to participate in this project
- I understand that I have the right to withdraw from this experiment at any stage.
- I have read this consent form and the information it contains and had the opportunity to ask questions about them.

Name of Participant: _____

Signature of Participant: _____

Date: _____

Instruction Form

First and foremost, your agreement in participating in this study is fully appreciated. This form covers the experimental procedure, so please make sure to read it carefully.

Demographic Information Form

This information form may help us with the analysis of the results obtained in this experiment.

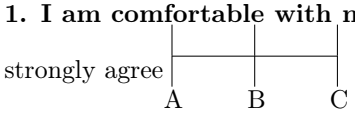
Age: _____

Gender: _____

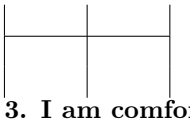
Expertise: _____

Exposure to previz software: _____

1. I am comfortable with multitouch interaction.



2. I am comfortable with 3D virtual environment navigation (e.g. video games).



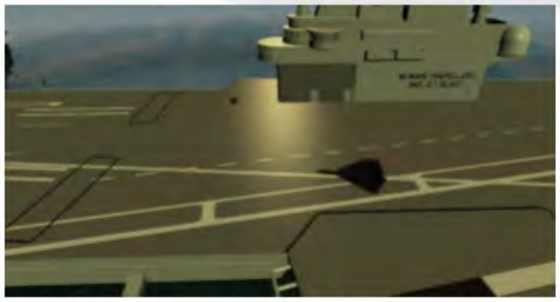
3. I am comfortable with 3D virtual object interaction and manipulation.

4. I am comfortable with video/film creation tools.

System Interaction Guide

Buttons on Ring
- Ring reorientation
- 1 finger interactions (rotation)
- 2 finger interactions (translation)

Storyboard Sheet



1. A fighter jet is on a hangar and busy speeding up for take-off.

2. As the jet gains enough speed it leaves the hangar's ramp and takes off. The camera follows it.

3. The jet is heading straight for the bridge and begins to turn for a barrel roll

4. As the jet approaches the bridge, the camera swings around from the back of the jet.

5. The jet is about to go underneath the bridge and continues with it's barrel roll

6. The jet comes out on the other side of the bridge completing it's barrel roll

7. As the jet is levelling out from the barrel roll, it starts pulling its nose up to ascend.

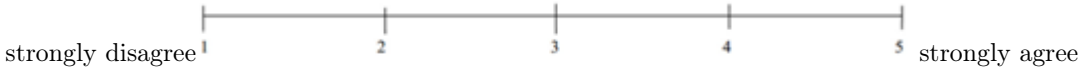
8. Once fully levelled out the jet ascends directly up. The camera views the jet's belly, with the city in the background

9. As the jet keeps ascending the camera pulls back, showing more of the city

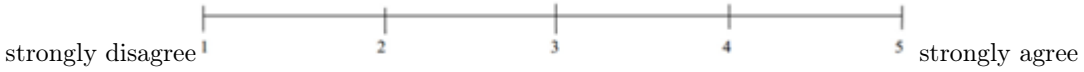
10. The jet has left the camera's view. One last shot of the city to end off this scene.

System Usability Scale Form

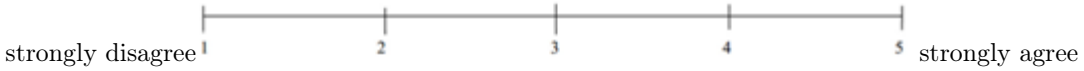
1. I think that I would like to use this system frequently.



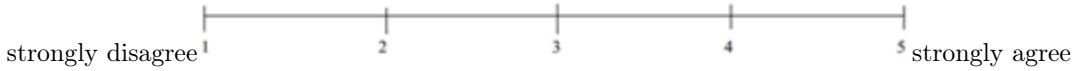
2. I found the system unnecessarily complex.



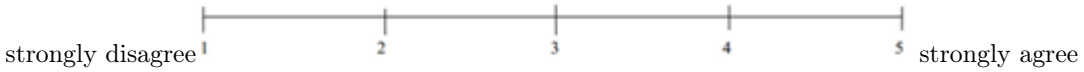
3. I thought the system was easy to use.



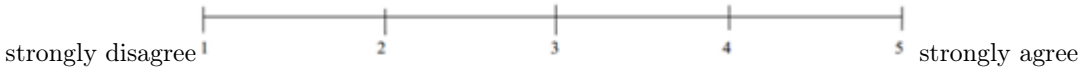
4. I think that I would need the support of a technical person to be able to use this system.



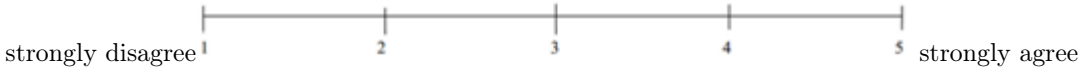
5. I found the various functions in this system were well integrated.



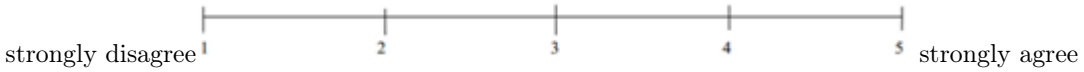
6. I thought there was too much inconsistency in this system.



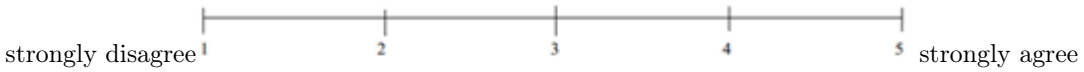
7. I would imagine that most people would learn to use this system very quickly.



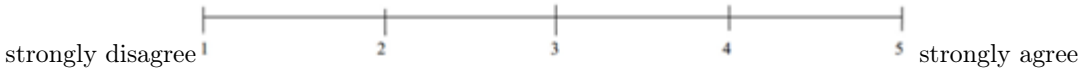
8. I found the system very cumbersome to use.



9. I felt very confident using the system.



10. I needed to learn a lot of things before I could get going with this system.



Interview Questions

1. How intuitive and consistent did you find the system's UI and interaction?
2. If there was anything cumbersome, what was it and why?
3. How suitable do you think multitouch is ...
   a) ... for pre-visualisation of this kind?
   b) ... as a means of collaboration?
4. What held you back the most when using the system?
5. What do you think should be removed?
6. What do you think could be improved?
7. What other fundamental features would you desire in this system?
8. Were there any issues with simultaneous user interaction, and if so, please elaborate.
9. Was the user interface helpful or harmful for aiding with collaboration?
10. Were there any issues as to which user was in control of a particular component of the scene?
11. Did the interface help you to work together in a team, compared to past experiences?
12. Did the system improve your collaboration/team work in any way?
13. How do you feel about this attempt at creating a more involved collaborative experience for constructing scenes?
14. How effective do you think this type of system would be for previsualisation in your profession, whether within your company or generally in the industry?
15. Is there anything else you would like share about your experience using this system or enquire anything regarding this research?

Appendix E

Experiment Data

Demographics

	Film Makers			Software Engineers		
	UserA1	UserA2	UserA3	UserB1	UserB2	UserB3
DEMOGRAPHICS						
Age	27	27	30	26	29	30
Occupation	Editor (Film)	Writer/ Film Director	Cinematographer, Film Editor	Software Developer	Software Developer	Software Developer
Previz Software	FrameForge	FrameForge	FrameForge, iClone	Unity (minimal)	None	None
comfort with multitouch interaction	1	1	1	1,5	2,2	1,5
comfort with VE navigation	3	1	1	1	1	2,5
comfort with 3d virtual object interaction	4	1	1	2	1,5	2
comfort with video creation tools	1	1	1	4,5	3	2,5

1. I am comfortable with multitouch interaction.

1 2 3 4 5

strongly agree | agree | neutral | disagree | strongly disagree

2. I am comfortable with 3D virtual environment navigation (e.g. video games).

1 2 3 4 5

strongly agree | agree | neutral | disagree | strongly disagree

3. I am comfortable with 3D virtual object interaction & manipulation.

1 2 3 4 5

strongly agree | agree | neutral | disagree | strongly disagree

4. I am comfortable with video/film creation tools.

1 2 3 4 5

strongly agree | agree | neutral | disagree | strongly disagree

System Usability Scale

		Film Makers			Software Engineers		
		UserA1	UserA2	UserA3	UserB1	UserB2	UserB3
	SYSTEM USABILITY SCALE						
1	I think that I would like to use this system frequently	1	1	1	3,5	2,6	1,5
2	I found the system unnecessarily complex	2,5	5	4	4	5	4
3	I thought the system was easy to use.	3	2	2	2,5	3	3,5
4	I think that I would need support of a technical person to be able to use this system	4	4	4	3,3	4,5	3,5
5	I found the various functions in this system were well integrated	2,5	2	2	3	2	2
6	I though there was too much inconsistency in this system	2,3	3	4	3,3	4,5	5
7	I would imagine that most people would learn to use this system very quickly.	1	2,3	3,5	2,3	3,5	2
8	I found the system very cumbersome to use	4	3	4	4	3	4
9	I felt very confident using the system	2	3	3	1,6	2,5	2
10	I needed to learn a lot of things before I could get going with this system	3	2,3	4	4	4	2,5

1. I think that I would like to use this system frequently.

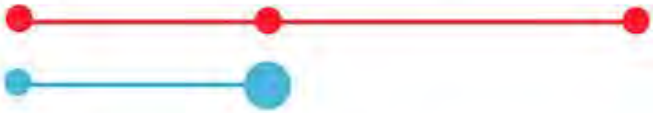
1 2 3 4 5

strongly disagree | disagree | neutral | agree | strongly agree

2. I found the system unnecessarily complex.

1 2 3 4 5

strongly disagree | disagree | neutral | agree | strongly agree



3. I thought the system was easy to use.

1 2 3 4 5

strongly disagree | disagree | neutral | agree | strongly agree

4. I think that I would need the support of a technical person to be able to use this system.

1 2 3 4 5

strongly disagree | disagree | neutral | agree | strongly agree

5. I found the various functions in this system were well integrated.

1 2 3 4 5

strongly disagree | disagree | neutral | agree | strongly agree

6. I thought there was too much inconsistency in this system.

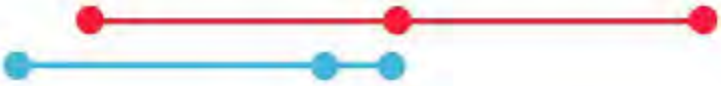
1 2 3 4 5

strongly disagree | disagree | neutral | agree | strongly agree

7. I would imagine that most people would learn to use this system very quickly.

1 2 3 4 5

strongly disagree | disagree | neutral | agree | strongly agree



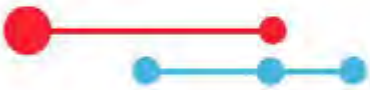
8. I found the system very cumbersome to use.

1 2 3 4 5

strongly disagree | disagree | neutral | agree | strongly agree

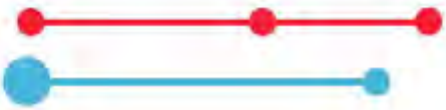
9. I felt very confident using the system.

1	2	3	4	5
strongly disagree	disagree	neutral	agree	strongly agree



10. I needed to learn a lot of things before I could get going with this system.

1	2	3	4	5
strongly disagree	disagree	neutral	agree	strongly agree



Interviews

User Testing Friday 12th June 2015

UserA2: It should not necessarily just be top view or side view. It should be whatever axis I'm on at the moment. I would like to be able to lock that in and, say, I want to move closer to that object or further away in.

QUESTION 1)
How intuitive and consistent did you find the system's UI and interaction?

UserA1: I found it intuitive, but it was inconsistent. So basic functions like moving objects around that's where I didn't find it very consistent but I found it intuitive in terms of once you explained to me the basic functions of how I can maneuver around on the map in the 3 dimensional space, I could do it myself and it wasn't a problem. There are certain things that you need to keep in mind at the beginning, but that's the same with any software where there are certain rules you have to follow to make it work.

UserA3: I agree I like the way you could get a widget and think it worked very well. The user specific widget and just in general.

UserA2: I think it would also be cool to maybe if you could ... make it personalize-able. E.g. You could have something like enlarge the object or whatever the case may be, e.g. undo. A customisable interface, e.g. two spaces that you could use as shortcuts and add anything else.

UserA1: I think to the layout of this .. the buttons and all the functionality that you can interact with I think it works well, because you have everything in one place and it doesn't get confusing. It's all in one space, so you can find it immediately. I don't know if that is going to work out later when you have more functionality, but for now it works really well.

UserA2: I wouldn't mind having something at the bottom of the screen. With photoshop as well I obviously have that thing on the side, so it does cut down your workspace a bit. I wouldn't mind something at the bottom so that I have options.

UserA1: If you can toggle between different, if you want to stick with a selected object what might work is if you have different function circles almost. If you in that toolbar at the bottom you can say "Now I want to transform the object, now I want to do something else". It brings up the same interface, but with different functions.

Interviewer: Interaction modes? Yes.

QUESTION 2)
If there was anything cumbersome, what was it and why?

UserA2: I think for me the big thing was the movement, coz I did this sculpting thing on an iPad when you start with basic shapes and you add arms and things (limbs) and you can move that. I always found that that felt very natural to do that ... and I get confused with things very easily. I think that was probably my main thing. If I could move easily through this world, I would do crazy things with it. I would be able to do such cool things coz its so wonderful. Maybe it's just coz I'm an idiot, and it should be made more idiot proof in terms of the movement of the stuff.

Interviewer: No absolutely, what you say makes sense, because you have to design software in a way that everyone should be able to can use it. If it's software like this which is expects you to have little to no knowledge or experience of let's just say previz . That was the whole point of this.

UserA1: The only thing I found cumbersome was moving around in the 3D space the way I wanted to. But in the same way it felt like what would make it a bit easier would be to have less freedom, as much as that might contradict itself.

UserA2: I agree with you.

UserA1: It almost felt like because I can move literally on all axes, it made it weird. Whereas if I could lock it and then know that now I can move forwards and backwards, I think that would have made it easier. Or like we discussed, having that snap function: once you're on a perfectly horizontal level, it snaps.

UserA2: But I agree, it's almost like you have too much freedom. Too much choice in terms of movement. I think if it was a bit slower, a little bit less ... like I had to do more because otherwise it runs away from you. You want to have complete control over it.

QUESTION 3)
How suitable do you think multitouch is for previz and as a means of collaboration?

Low fidelity previz.

UserA1: I think it is very suitable if you have, especially if you discuss what it is that you want to do and you have three people in the room and everyone knows what their job is to get this previz done as quickly and efficiently as possible, then all three of us can work at the same time and get to do what ever it is we want to do.

UserA2: Can you for example manipulate 3 different objects at once or two objects with the camera at once?

Interviewer: Yes you can.

UserA2: But I have to say it's weird that this hasn't been done before, and I think that is always the mark of a really great idea. You ask yourself: why the hell has this not been done before? I've been needing this, you know.

Interviewer reassures that something like this may already exist, but nothing quite like this was found during the research process (i.e. literature review). Explains the overlap of research areas in this project.

UserA2: I think it is about time for something like this, because it's awesome.

UserA1: I also think it's great. And I also think there is potential of creating it in a way that... You know it's nice, because now all three of us are here and we can work off the tablet together, which is really cool. But I think as you progress and go further, if you have certain things in place, like if all three of us had a really stable and fast internet connection, it would also be nice to make it virtual, i.e. networked collaboration. I could sit on my computer and we could all work off the same project file and do exactly what we did as if we were in the same room. I think the potential for that is incredible. ... Because that exists, but not in real time. So you can work off the same project file, but it's not like you're in a virtual room together.

Interviewer: Like a game for instance?

UserA1: Exactly!

Interviewer: Double check, do you think it makes sense for previz of this kind?

UserA2: Absolutely,it makes sense. It makes sense for me, coz I wanna do that. I wanna move the camera directly as opposed to sitting with the mouse trying to work through it.

UserA1: Now more than ever, because we're getting more used to doing touch screen related stuff anyway.

UserA2: If you had it on an iPad on set or whatever, imagine how great that would be? If there was an actual jet and you had to explain to the guy how to fly go, this is kind of what I want you to do.

QUESTION 4)
What held you back the most when using the system?

UserA2: It's just the movement thing for me. Too many options. (Moving through the environment). Too me it felt like it should just, again, be a lot more limiting/limited .. which is also a weird thing to say, but I only need a little bit and then I will work for it. Then I feel like I'm engaging with

it. Whereas it's almost like it runs away from you a little bit, coz it moves too fast. That's the only thing.

UserA3: This is now excluding the bugs and things?

UserA1: Yeah, because that is also what I wanted to say. Because, without the glitches now when it came to the objects, and especially the jet for instance, umhh.. I don't think I'd have the same opinion about it that I have now. Because if it wasn't for that I feel like I would have worked towards it and it would have worked out and I would have had a different opinion about it. But now that I didn't really get to see it's full potential, my opinion might change. Because at the moment that's what held me back. The glitches held me back, but not the interface.

Interviewer: (functionality has a drastic impact on the overall user experience)

QUESTION 6 & 7)

What do you think should be improved/removed, and what other fundamental features would you desire in this system?

UserA1: I think we mentioned, let's just go through what we mentioned while we were working with it:

Group listing features to add/improve:

- (frustum) there was an indication of what the camera is capturing on a two dimensional space
- horizontal 180 degree snap thing and a vertical 90 degree snap thing
- undo for objects and undo for perspective
- an option to make the keyframes a curve maybe even interacting with the actual line, using a handle on the curve... but this is for much more specific animation.
- the keyframes on the timeline as well, being able to move them around, because if you actually start doing proper animation (high fidelity?) you should be able to fine-tune it a lot more
- a list of all objects in the scene, so that you can select them there to go to them in the VE and even hide them

UserA1: And in incorporating your way to navigating the environment, for example like you would do in maps app on a phone. If you want to get from point A to point B, you could do it via quick gestures without having to use the pinch zoom or the slow movement. So I can have options. If I want to fine tune my position or my perspective, then I should have that option, but I should also have the option to quickly go from point A to point B.

Interviewer: Mentions the double tap to go to a location

QUESTION 8)

Were there any issues with simultaneous user interaction, and if so, please elaborate?

UserA2: We didn't actually test that that much.

UserA1: When we did to it it actually worked, when you (UserA3) were gonna work on the camera angle, you could do so. It wasn't buggy in any way there, because you could move that around and you could do it, I could see that something changed. So I think we didn't see it at it's full potential, because if we would have all gotten to the point where we collaborated, from what we saw... from the little that we saw it seemed to work.

Interviewer: Mentions flaw: Unity GUI won't let more than one button be pressed at the same time (expecting the first point to be the mouse)

UserA1: What might actually be an interesting way of doing this is because working on a tablet this size, I felt almost like it's overwhelming how much space there was. Whereas if you could actually do it in a split screen function it might actually work.

Interviewer: So everyone get's their own viewport?

UserA1: Exactly! You get your own viewport, you can change and do whatever you want. I mean it's obviously going to be hard on the processing power, so I don't know the limitations there. If 4 people work at the same time but you have your own section on the screen, and you have the entire map on that quarter.

Interviewer: We also considered user specific viewports.

UserA1: It's going to be very computing intensive I guess, I don't know. Where everyone has a duplicate of all the functions and UI components available for themselves. Especially when all viewports also have to synchronise after every change to the scene.

UserA3: What would the workaround be for Unity to allow multiple buttons to be pressed simultaneously?

Interviewer: One option would be to cross check every touch point with every button, to see whether it falls within the button's interaction region. This could further be optimised with something like a quad-tree to only cross check touch points against GUI elements within their region.

QUESTION 9
Was the user interface helpful or harmful for aiding with collaboration?

UserA1: I would say helpful.
UserA2: I second that
UserA3: I third that.

QUESTION 10)
Were there any issues as to which user was in control of a particular component of the scene?

UserA2: I think maybe we didn't explore that.
UserA1: Yeah

QUESTION 11)
Did the interface help you to work together in a team compared to past experiences?

UserA1: I think we didn't explore it enough, but just the fact that the way the software is laid out it immediately at least promotes teamwork. Just the fact that we all have our own panels and that we're working on one screen already made us have to work together, otherwise it's just three monkeys hitting buttons (giggling) you know, it wouldn't make sense.

QUESTION 12)
Did the system improve your collaboration/team work in any way?

UserA2: I think so yes, because honestly for me normally this isn't something I would want to do in a team, but I think that just maybe a personal thing it probably is something I probably want to do by myself. Sitting with UserA3 and UserA1, doing that I felt like I needed their input. I wanted to know what they think, because that's the feel for this engine. For this thing to happen, we need to do it together. And I think that will probably end up making the product better.

UserA1: That's actually such a great point, because if you work together on the previz as a team, every step of the way each of you will have a better understanding of what it is that the scene requires.

UserA2: And it takes time to go through those stages of approval...

UserA1: And then having to explain if you do it by yourself and you explain it to us, we'll definitely have an understanding of what it is that you visualise as the director, but if we sit with you and do it step by step together and we collaborate on creating this previz scene, then there's not gonna be a question about, e.g. where the camera is. Coz we've done it together.

UserA2: And you save so much time.

Interviewer: Load shedding is going to kick in in 3 minutes. 3 questions left. Don't worry, the load shedding won't affect us.

22:00 QUESTION 13)

How do you feel about this attempt at creating a more involved collaborative experience for constructing scenes?

UserA2: I think, yeah, I think it's good. You know, um, as a director I normally, like I said, would have to do a lot of that by myself, but there's such a large component of it that's also collaborative. It's so hard for me to always draw the line there. With the film that I'm doing now I'm not sure whether I have to do the shot list again or that previz and storyboarding by myself or do it with the POP, whereas with this I think it's best of both, because that person can choose to either be a part of it or not, and if that person is a part of it you get everything done and it's a check, it's a done. You're done with this, it doesn't have to then go through the process of sending it to the DOP sending it to this person to have a look at it. We know exactly what we're doing and that's also something where I would maybe like to add as well, I don't know if that's maybe ridiculous, but notes for example. If you could at any point make a little note. So if I give this to UserA3, who is for example the DOP and I want to add all the production design and I want to say I would like for all these walls to be red or let's use a 60mm lens for this shot, you know, I think that's also something that might be cool, because it's something I would normally have to ask someone else to do.

UserA1: It's great if you could do it in notes for now to bridge the gap, but I think ultimately what you should be able to do is actually incorporate those notes into the software. If you want those notes to be red then you should be able to do it, and then have the notes function as an added extra. It would be nice to have a little marker if you click on that marker there can be notes just to maybe describe something in more detail or to bridge the limitations that the software might put on you. Because there are always going to be limitations, it doesn't matter how refined the software is ... there is always something that could be done better.

24:00 QUESTION 14)

How effective do you think this type of system would be for previsualisation in your profession, whether with your company or generally in the industry?

UserA2: I would say that's a resounding HELL YES!

UserA1: Yeah!

UserA2: Absolutely! I would use this a lot. HELL YES! Honestly man, this is something that I would use extensively, you know. I'm still on paper, so I would still have to draw my storyboards and stuff, and it's so difficult to try and explain that. I think something like this, if I had my iPad on set I either way will have something, like my shot list and my references and all that kind of stuff. If I could have this and could then quickly construct a scene and talk through a scene in the way that it couldn't possibly be clearer unless you shoot the scene or the shot. I just think it's the best thing. I think it's incredible.

User Testing Monday 13th July 2015

QUESTION 1)
How intuitive and consistent did you find the system's UI and interaction?

UserB1: Very consistent.
UserB3: The view and the camera all reacted the same... so it's very consistent. We found that rotation was unintuitive and the view. Everything had a fairly similar UI with mostly the same set of controls ... or functionality, in the context of the object.
UserB1: Yeah, there really wasn't that much to the interface, so it was quite easy to pick it up. Of course you need to learn how to use any interface in the beginning.

QUESTION 2)
If there was anything cumbersome, what was it and why?

UserB3: What else did we say that was unintuitive?
UserB1: Rotation!
UserB2: The movement of the view was a little bit unintuitive.
UserB3: The view ... we found that it didn't have that two finger rotation. You know, to rotate around the z axis like Google Maps or Google Earth. We felt that is what made it more difficult.
UserB2: Yeah that was cumbersome.
UserB1: The rotation made it cumbersome and the fact that only one person could control the view at a time.

QUESTION 3)
How suitable do you think multitouch is for previz and as a means of collaboration?

UserB2: I think yeah, good for previz maybe?!
UserB1: So, for collaboration ... it's good to get people working together, and it's fun ... people talk and discuss ideas. The only issue is that, because it's so dependent on the view, only one person is really in control at a time... and because you can only select one object at a time, it's still only one person in control. So it's not very collaborative in working asynchronously, but it's good for communication.

QUESTION 4)
What held you back the most when using the system?

UserB1: The view was a bit of an issue. Always readjusting that held everyone else back from interacting with the system at all.

QUESTION 6 & 7)
What do you think should be improved/removed, and what other fundamental features would you desire in this system?

UserB1: The copy thing to make a clone. To duplicate the user widget. I think it would be better to do it in a different screen or something. Like, if there was a toolbar or something that you could create widgets from. Otherwise I thought the system was very minimalistic.
UserB3: It was very simplistic. A list of all the objects would have helped a lot. That is, a list of all the objects in the scene. And the timeline, if those objects were always on the timeline.

UserB1: Yeah, kind of like how Unity does it, with the hierarchy list. Another feature that would have been cool is if the camera could be locked to arbitrary objects. That way you wouldn't have to always worry about moving the camera. And if the timeline was always there.

You mentioned that, when someone is in control of the view, that they have their colour in the border. Maybe that should be more visible, because we didn't realise that immediately.

UserB3: I think the entire screen could be sort of greyed out, or dimmed or something, so that its very obvious that it's the view mode.

UserB1: The rotation could also be improved. So you have more like a game object thing... a transform gizmo/manipulator, so you can perform these transformations more easily. You can do this around a particular axis or arbitrarily. Or even have the camera locked to any object at will. The way it works, the camera is locked to the landscape. In that way you could just move the object of interest and the camera would follow it, so you don't have to worry that much about setting up the camera path. By animating the object the camera would automatically get a follow animation that you could just change or adjust later.

UserB2: What would also be interesting and potentially useful is if you could make your view the actual camera.

UserB3: I also thought maybe, custom view points. So like a view of the scene side on, so that each user would orient the camera to suit whatever side of the screen they're on.

QUESTION 8)
Were there any issues with simultaneous user interaction, and if so, please elaborate?

UserB3: It was really just the bottleneck of adjust the view before being able to continue with the animation. Simply the fact that only one user could control the view point at a time, made it difficult to parallelise tasks.

UserB1: So what would be cool is if there are 3 different screens. I don't know if that would work, but they're all integrated and they all have their own view.

UserB2: And everyone can select their own objects.

UserB1: Yeah, everyone does their own thing and then when you launch it (press play), it kind of brings it together.

QUESTION 9
Was the user interface helpful or harmful for aiding with collaboration?

UserB1: For working together it was useful, because we you all work on the same device. However, for simultaneous user interaction it was harmful, because of that bottleneck due to the single user view point interaction.

QUESTION 10)
Were there any issues as to which user was in control of a particular component of the scene?

UserB3: Not the components. We were only really working on the jet and the camera.

UserB1: So rotation was a bit of an issue.

Interviewer: This question is about who was in control of what. So rotation is clearly an issue.

UserB1: And the view. What we also mentioned was that when you move and then rotated the camera it almost felt like a dimension was missing. So like a wheel rotation so like a wheel rotation would be nice, so that you can do more than one directional movement of the camera at a time.

QUESTION 11)

Did the interface help you to work together in a team compared to past experiences?

UserB1: We don't really have past experiences of this.

QUESTION 12)
Did the system improve your collaboration/team work in any way?

UserB1: Communication definitely.
UserB3: We tend to take like a role-based approach, which I think was quite good.
UserB1: The one thing that was actually quite cool is, even though we took role-based approaches, we all get very involved in our roles. So that forced you to know exactly what was going on in all the aspects.
Interviewer: Everyone was on the same page?
UserB1: Yeah... Except for the view.

22:00 QUESTION 13)
How do you feel about this attempt at creating a more involved collaborative experience for constructing scenes?

UserB3: I thought it would be a very good thing for previsualisation, which I then realised is the whole point of this project.
UserB1: But to build like a big scene..
UserB3: If there is gonna be detail, I think it might be a bit finicky ... a bit painful.
Interviewer: Especially with touch based interaction.
Group: Yeah
UserB3: Yeah like the accuracy of getting it down to the very detail of where you want to put and item or something.
UserB2: Yeah, won't work either.

24:00 QUESTION 14)
How effective do you think this type of system would be for previsualisation in your profession, whether with your company or generally in the industry?

N/A

QUESTION 15)
UserB3: I found it quite interesting actually, because I created a similar system in the past. It was for internet browsing. For like opening multiple tabs or like whatever and opening multiple buttons. And I used all the similar patterns that you did. The circular UI elements, that always use the same real estate, no matter what way you rotate them.
UserB1: So it's very natural to come up with something like that. And both of you individually came up with the same thing, for completely different programs. So it's a very natural thing of interacting with touch tables. Boom Baby.
Interviewer: Awesome!