Bachelor Thesis in the Degree Program
Computer Science and Media

**Real-Time Set Editing in a Virtual Production
Environment with an Innovative Interface**

submitted by Stefan Seibert

at the Stuttgart Media University
on 26.02.2015

First Examiner: Prof. Dr. Simon Wiest
Second Examiner: Dipl. Media Sys. Wiss. Simon Spielmann

HOCHSCHULE DER MEDIEN

Seibert, Stefan:

Real-Time Set Editing in a Virtual Production
Environment with an Innovative Interface

Bachelor Thesis

Computer Science and Media

Stuttgart Media University, 2015

Eidesstattliche Erklärung | Statutory Declaration

„Hiermit versichere ich, Stefan Seibert, an Eides Statt, dass ich die vorliegende Bachelorarbeit mit dem Titel: „Real-Time Set Editing in a Virtual Production Environment with an Innovative Interface" selbstständig und ohne fremde Hilfe verfasst und keine anderen als die angegebenen Hilfsmittel benutzt habe. Die Stellen der Arbeit, die dem Wortlaut oder dem Sinn nach anderen Werken entnommen wurden, sind in jedem Fall unter Angabe der Quelle kenntlich gemacht. Die Arbeit ist noch nicht veröffentlicht oder in anderer Form als Prüfungsleistung vorgelegt worden. Ich habe die Bedeutung der eidesstattlichen Versicherung und die prüfungsrechtlichen Folgen (§26 Abs. 2 Bachelor-SPO (6 Semester), § 23 Abs. 2 Bachelor-SPO (7 Semester) bzw. § 19 Abs. 2 Master-SPO der HdM) sowie die strafrechtlichen Folgen (gem. § 156 StGB) einer unrichtigen oder unvollständigen eidesstattlichen Versicherung zur Kenntnis genommen."

_____

Unterschrift | Signature

Outline

Acknowledgments

1. Introduction

This work wants to have a look on the technical aspects that have to be considered when virtual production is extended by real-time editing capabilities. For this purpose a prototype implementation of real-time set editing is implemented and evaluated in a virtual production scenario. Based on the experiences from developing the prototype and evaluating it, this paper gives an overview of topics and problems that should be considered when real-time editing is desired.

The whole work is divided in several logically connected chapters. This first chapter provides a short abstract of the thesis in German and English and also delivers a short insight about the scope of this work. Subsequently Chapter 2 gives an introduction what visual effects are, what problems the common working scenarios in visual effects nowadays have, what virtual production is and how virtual production can minimize such problems. Afterwards Chapter 3 presents the most valuable technologies and systems, in the author's point of view, for virtual production and editing in such an environment, so the reader is aware of the common technologies. Chapter 4 is following with an overview of related work, to give an overview of the research taking place in this area at the moment. Becoming more concrete about the done work in this research, one will find an insight description about the developed 3D user interface for real-time set editing and the problems emerged during its development in Chapter 5. Subsequently the evaluation of such an approach is described in Chapter 6. To end it all Chapter 7 delivers a comprehensive conclusion about the project.

Figure 1, pictures from the shooting for evaluation purposes in Ludwigsburg, Germany.

1.1. German Abstract

Ziel dieser Bachelorthesis ist die Beschreibung eines prototypischen 3D Editierverfahrens, das intuitives Editieren von virtuellen Elementen in Echtzeit innerhalb einer virtuellen Produktionsumgebung ermöglichen soll. Die Evaluation dieses Ansatzes geschieht qualitativ. Eine Benutzergruppe, bestehend aus Industrievertretern testet das neue Verfahren und füllt anschließend einen Fragebogen aus.

Der Anteil virtueller, mithilfe von 3D Computergrafik erstellter, Elemente wächst in allen Bereichen der Entertainment Industrie seit Jahren stetig. Trotzdem ist die Bearbeitung von virtuellen Objekten nach wie vor ein komplexer Vorgang, der besonders geschulte Mitarbeiter an speziellen Arbeitsplätzen benötigt. Dies kostet Zeit und Geld. Mit dem Aufkommen neuer Eingabegeräte und verbesserten Tracking Technologien stellt sich die Frage ob es nicht möglich ist diesen Bearbeitungsprozess zu verbessern. Mithilfe des neuen Editierverfahrens soll eine intuitive Oberfläche geschaffen werden die es jedermann ermöglicht direkt noch am Filmset Änderungen an virtuellen Elementen vorzunehmen und gemeinsam an einer Filmszene zu arbeiten ohne dass hierfür besonderes Expertenwissen nötig wäre.

1.2. English Abstract

This bachelor thesis wants to describe a prototypical implementation of a 3D user interface for intuitive real-time set editing in virtual production. Furthermore this approach is evaluated qualitatively through a user group, testing the device and fill in a questionnaire.

The dimension of virtual elements created with computer graphics technology in all areas of entertainment industry is steadily growing since the past years. Nevertheless can the editing process of virtual elements still require a costly process in terms of time and money. With the appearance of new input devices and improved tracking technologies it is interesting to evaluate if a real-time editing process could improve this situation. Being currently bound to experts on special workstations, this could lead to a more intuitive and real-time workflow, enabling everybody on a film set to influence the digital editing process and work collaboratively on the scene consisting of virtual and real elements.

1.3. Scope of this Thesis

This work is published at Stuttgart Media University in the winter term 2014/2015 as bachelor thesis from Stefan Seibert. It is written at the Research & Development Department of the

"Animationsinstitut", which is part of the "Filmakademie Baden-Württemberg", in Germany. Parts of this thesis are written in line with an EU co-funded project Dreamspace (cf. Dreamspace Project). Dreamspace is a three year running project with the goal to research, develop and demonstrate tools to allow creative professionals to collaborate and combine live performances, video and computer-generated imagery in real-time. Dreamspace is part of the Seventh Framework Programme of the European Union, a research and innovation package (cf. Dreamspace Project). Practical work necessary for this paper including the creation of the evaluation questionnaire and the evaluation process itself in a film studio at the "Filmakademie Baden-Württemberg" have been performed in cooperation with Kai Götz, writing his master thesis at the same time about the topic "Virtual Production: Possibilities and Limitations of Virtual Production Environments, Optimization through Implementation of Innovative Interfaces". It is also published in winter term 2014/2015 at the Stuttgart Media University and relates more to the economical and production related aspects of real-time editing, while this work focuses on the technical aspects. The preparation, development, practical and written evaluation of the set editing prototype took place in a 6 month schedule. The complete software development was done by the author, relying on several SDKs. Following this introduction into the work, the next chapter will give a brief overview of how work in visual effects is done normally and how these workflows have been changed through the last years by introducing several technologies.

2. From Visual Effects to Virtual Production

In the following chapter one can find a short overview what visual effects are, how a traditional visual effects production can be described, how changes and editing are fulfilled within this pipeline, what a virtual environment is and how all these technologies developed to a workflow called virtual production. Following there is a short overview which advantages this different way of producing could deliver to an individual before afterwards Chapter 3 will introduce the technologies used in virtual production environments more in detail.

2.1. What are Visual Effects

The Visual Effects Society is an association of visual effects professionals throughout the whole industry. According to this society, the terms "visual effects" or "special effects" are often used for visual effects and special effects at the same time. But these are distinct areas in filmmaking. A visual effect can be specified as any imagery that is created, altered or enhanced because it cannot be achieved with live action shooting and until today this image alteration is nearly always done by digital editing, whether if the image is painted completely digitally or only altered digitally. Special effects in contrast relate to effects that can be done while filming a live action scene for example practical explosions, rain or fire (cf. VES Handbook, pp. 1-2).

This work focuses purely on the former definition of visual effects.

Famous Visual Effects Supervisor Scott Squires describes the importance of visual effects for the creative industry as tremendous. Visual effects are used in nearly all films today, not only science fiction and fantasy. Most of the visual effects are not visible to the audience. The different tasks range from fixing problems on set like wire removal but also to add, change or remove actors, backgrounds or objects to create entirely virtual shots. Visual effects grant filmmakers and other creative professionals a great amount of freedom to create any kind of world they can imagine only. (cf. The Value of Visual Effects).

Despite the fact that visual effects are often discussed in a film producing environment, most of the concepts can easily be adopted for other industries like all kind of moving media, games, television shows, webcasts or commercials (cf. VES Handbook, p. 1). This also applies for virtual production. Since the work and evaluation this paper is written about is also done in a film environment, verbalizations in this paper are focusing on a film production environment, even

though virtual production can be used in other industries also, especially in various fields of media. Since it is known now what visual effects are, we want to have a look how visual effects are traditionally produced.

2.2. The Traditional Visual Effects Production

The traditional production is divided into the three steps preproduction, production and post-production. Each of these steps is very important for producing visual effects (cf. VES Handbook, p. 5). Preproduction relates to all work done in preparation for a shooting like identifying the visual style, concept artworks, building assets, blocking scenes, drawing storyboards, prototyping of new features and so on (cf. Production Pipeline Fundamentals, pp. 29-30). Production (in terms of visual effects) is the process of on set shooting of the real footage and collecting the pictures necessary for producing visual effects afterwards. Postproduction in traditional visual effects includes most of the work for instance editing, matte painting, or modelling, rigging and animating digital characters, adding simulations like fluids or explosions, and so on (cf. Production Pipeline Fundamentals, pp. 35-36). In the next chapter a outlook is provided what limitations this way of producing visual effects can have.

Figure 2, a typical VFX production where actors interact with a placeholder.

2.3. Limited Editing in the Traditional Visual Effects Production

Before inventions like more capable computers and digital image scanners came up in the late 1980s visual effects were done analogically with technologies like stop motion, matte paintings or miniatures. All the effects were done directly while shooting or were inserted into the film image afterwards by analog film processing. Thereby digital visual effects became possible and thus the creation of complete artificial and digital characters like aliens or monsters or

even whole unnatural worlds. But all these digital complements where done off set after live action shooting in postproduction. When digital effects started to hold a portion in the films to a greater extent, it became a major topic to also be able to visualize them in the scene before shooting (cf. The New Art of Virtual Moviemaking, p. 3). This way previsualization evolved from the classic way of only drawing a storyboard on paper as a way to describe what you want to show in a specific shot. In fact with new possibilities like 3D animation tools one could create simplified versions of all objects in the scene and create a digital animated version of your prospective sequence. This already enabled filmmakers to explore their creative ideas in a better way (cf. VES Handbook, pp. 45-48).

In spite of the options previsualization offers, there is still a gap in the traditional visual effects production. First of all the director is not able to interact directly with the creative process. The film production itself remains isolated from the digital content (cf. The New Art of Virtual Moviemaking, p. 3). In movies with a large amount of digital assets, directors can only see half of the movie during production. Where digital environments should be, are only green screens; instead of digital characters with a specific look you can only find actors in motion-capture suits acting with a tennis ball as stand-in for digital extensions for example. This can lead to a lot of expensive mistakes (cf. Production Pipeline Fundamentals, p. 304). The next chapter gives an insight view how the idea of previsualization evolved to virtual production.

2.4. From Previsualization to Virtual Production

But digital previsualization does not serve as a rule today. Furthermore with the aid of real-time graphics, primarily developed for computer games, it became more popular to expand the previsualization also to the film set, like an on-set visualization of the digital contents. A first basic usage of this idea could already be seen in the making of Steven Spielberg's A.I. in 2001 (cf. VES Handbook, p. 196) But virtual production is more than the pure visualization of digital characters on a monitor at the shooting to name one example. It includes many more ideas and concepts. One possible goal is a fully shared asset pipeline, so that digital assets are not created for every step anew, furthermore they are created once in the very early stage of production and then used and altered in the level of details as needed for the current step at a time. In addition with motion capturing (which is explained more in detail in Chapter 3.1.5) it is also imaginable to record the performance of an actor on set and to use it directly in real-time for controlling a digital character for instance. Another idea is bringing production to a new level

through a complete virtual production space, where all departments can work collaboratively and interactively on a project, sharing resources, ideas and decisions and carry them through the whole production process (cf. VES Handbook, pp. 443-444). One could imagine a system which delivers a complete virtual environment that enables doing everything digitally and collaboratively, from the beginning of preproduction to the finished film, like designing costumes, doing the lighting, animations, camera work, framing, blocking and so on. So questions about creative work can be answered much earlier in production (cf. Production Pipeline Fundamentals, p. 304).

The Visual Effects Society defines the term virtual production as follows: „virtual production is a collaborative and interactive digital filmmaking process which begins with virtual design and digital asset development and continues in an interactive, nonlinear process throughout the production" (cf. VES Handbook, p. 444). The most common and important technologies required for a virtual production environment are explained in more detail in Chapter 3. Mentionable projects that used these or part of these technologies were already James Cameron's "Avatar (2009)", Steven Spielberg's "The Adventures of Tin Tin (2011)" and Shawn Levy's "Real Steel (2011)" (cf. VES Handbook, pp. 73-74). But what kind of benefits does this different way of production deliver? One answer to that question can be found in the next chapter.

2.5. Advantages of Virtual Production

Virtual production or virtual cinematography entails a lot of benefits. It enables the harmonical combination of virtual characters, real actors, virtual worlds and existing environments all in real-time. It closes the gap between pre- and postproduction so that creative decisions can be made collaboratively and interactively during the whole proceeding. Assets don't need to be created anew for every step of production, but once in the beginning and then carried through all stages of the filmmaking process. Thus it is possible to see everything together in place not only weeks after production by reviewing the work of a visual effects company, but rather directly while shooting (cf. VES Handbook, pp. 73-74, 443-444). Directors and cinematographers are able to realize a shot completely by themselves or collaboratively, staging it by moving around the camera and not having to do this by computer animators or previsualization artists (cf. The New Art of Virtual Moviemaking, p. 4).

More production related advantages are fewer physical constraints as it is not necessary to wait for the blue hour for instance to get a specific lighting situation. Weather conditions really do

not matter and it is cheaper having the virtual North Pole in the studio then flying a whole film crew and equipment to such an isolated spot. (cf. Methods, Guidelines and Scenarios, p. 8). Virtual production is making it possible to walk around and see everything in place directly on set, watching the digital environment from different angles and perspectives. It offers a deeper impression of a digital character when one is able to orbit him or her. As directors and the other creative decision-makers on set are used to being able to walk around and to frame a scene from different perspectives, seeing the camera image on a monitor immediate, training camera movements and everything before they finally decide which one is the best looking framing and movement, they also want to do that in virtual environments and with computer generated image parts. But what exactly are virtual environments? The next chapter intends to answer this question.

2.6. Virtual Environments, Augmented Reality and Virtual Production

Since there are many definitions, ideas and concepts behind virtual environments and augmented reality these terms should defined clearly for the scope of this work. A virtual environment is often defined as a three dimensional synthetic world, that is seen from a first-person point of view (cf. 3D User Interfaces, p. 7). This means a virtual environment is purely built with virtual elements only. Incorporating collaborative work leads to several clients working in this virtual environment at the same time, therefore every client can have its own first-person point of view into the virtual environment.

Augmented reality rather is known as enhancing the real-world with virtual elements or information (cf. 3D User Interfaces, p. 7). Augmented reality has the advantage that him or her is still able to orient him- or herself in real world constraints, so it is not that likely to feel dizzy. Virtual environments in contrast normally make blind to the real world so that additional safety aspects have to be considered.

In virtual production both setups are feasible. Films and media productions made purely from digital elements can rely on virtual environments, for productions using also live action footage from a real camera, an augmented reality would be preferable. Knowing what a virtual production is and what benefits it entails Chapter 3 presents the most common technologies in the field of virtual production.

3. Virtual Production Systems

This section intends to first introduce and furthermore explain the most important technologies which are used in virtual production systems. This chapter only considers technologies with a high impact to virtual production. In addition there are a lot of technologies used already in traditional visual effects productions that are also common in virtual production scenarios like keying, rotoscoping and so forth, but since this is not the focus of this work and has been part of extensive research and development throughout decades, one can already find a description of such technologies in the VES Handbook (cf. VES Handbook) for example. After these technologies, input and output devices are introduced and subsequently their suitability for the specific case of real-time editing in a 3D environment like virtual production is reviewed. This is followed by three possible ideas about real-time editing on set. At the end of this chapter one will find an overview of tools and systems that are already in use for creating virtual production environments, which limitations they have or which interesting aspects they offer for virtual production.

3.1. Important Technologies for Virtual Production Systems

3.1.1. Tracking

The first technology to mention is tracking. It is used to describe tasks and technologies in various industries. There are also several tasks in the environment of visual effects productions which are referred to as "tracking". In general, tracking can performed in at least two directions. Either it refers to extract the movement from virtual elements out of a picture sequence to make use of it or to track the position and rotation of a real object in space and match virtual elements with this motion. This can either be the camera, an actor or any other object. Tracking is very important for virtual production. To perfectly align virtual and real space, the camera movement and the movement of any actor who represents a virtual character or real objects who should be replaced through virtual objects must be recorded precise. The following tracking technologies are the most important ones developed and frequently made use of.

Magnetic Tracking
Magnetic tracking systems use the altered magnetic field distribution depending on ones posi-

tion for instance. A stationary part emits an electromagnetic field which is received by little devices mounted on the tracked object. It is then possible to determine the position and rotation of the receivers with respect to the stationary part by their induced current. Advantages of this technology are high update rates, low latency and good precision without the need of a line of sight. In contrast to this disadvantages are the sensitivity to magnetic interference, metallic objects and ferromagnetic material, as well as the rapidly decreasing signal quality with increasing distance (cf. Navigation and Interaction, p. 22). Figure 3 shows a magnetic tracking device.

Figure 3, the Razer Hydra a current game controller, which uses magnetic tracking.

Acoustic Tracking

These systems use ultrasonic waves as a signal; the measured time they need to travel to multiple receivers can be used to compute the position and rotation of the tracked objects in space. Only acoustic interferences exist, respective to the used wavelength. But their precision is reasonable, their update rate is low and they require a line of sight (cf. Navigation and Interaction, p. 22).

Inertial Tracking

Inertial tracking works on the basis of accelerometers and gyroscopes by measuring the relative sensor movements. These sensors can be employed at nearly any location. Because they work by measuring the relative distance from a known position they are more susceptible to errors. (cf. Navigation and Interaction, p. 23). Figure 4 shows a magnetic tracking device.

Figure 4, Inertia Cube 4 from Intersense, a inertial tracking device.

Optical Tracking

Tracking is also practicable with optical technologies. Knowing the position of several cameras allows to triangulate the position and rotation of an object by knowing at least the position of three markers on the object. Marker positions are calculated by their positions on the respective camera images. The visibility of the necessary positions in the pictures are often reinforced by making them reflective to a specific infrared wavelength which is sent out by the cameras and received by CCD sensors filtering out any other kind of light. Optical tracking systems can provide accurate positions and rotations. Their disadvantage lies in the required line of sight for most of the cameras and in being limited to the volume generated by the cameras, since the cameras need to have overlapping fields of view (cf. 3D user interfaces, pp. 101-103). Figure 6 shows a optical tracking system.

Depth Based Tracking

Recent research also covered the usage of technologies like time of flight to calculate a depth map of the scene. (cf. Overview of Tracking Technologies). A depth map in conjunction with the coherent camera image can be used for estimating the camera's position and reconstructing the three dimensional scenery of the image (cf. 3D Pose Estimation and Mapping).

Runtime Tracking

Another technology also makes use of triangulation for estimating an objects position and rotation. The calculation is as well possible between multiple senders of electric magnetic waves like satellites, wireless local area networks or broadcasting network base stations. (cf. Location systems for ubiquitous computing). This is the same technology like the one commonly used in GPS (Global Positioning System) for navigating cars around the world (cf. Navigation and Interaction, p. 23).

Summary

All of these tracking technologies have their assets and drawbacks. Some manufacturers try to build their devices as hybrid solutions based on several technologies today to compensate the drawbacks (cf. 3D user interfaces, pp. 103-105). Important features for a virtual production environment is the ability to track in real-time, so the virtual camera can directly aligned with the movement of the real one, or the movement of an actor can be applied to a digital character

instantly. Also the restrictions for allowed movements should be as low as possible. Only this way the creativity of a cinematographer can be absolute.

3.1.2. Real-Time Graphics

Since virtual production intends to combine the real world and a virtual one, computer graphics are also an essential part of this technology. In traditional visual effects productions computer generated images are generated in an offline process which is an established production pipeline. To name the most important steps it starts with the modelling of 3D objects, texturing and shading them, putting them into a complete 3D scenery and after setting up a correct virtual lighting the 3D scenery is rendered back to a flat 2D image. This rendering process is done for every frame of a movie separately and normally there are 24 or 25 of these frames in a second for a typical film production. Renee Dunlop describes all of these production steps very well (cf. Production Pipeline Fundamentals, pp. 45-66). All of these steps take a lot of time from trained persons and many of them are still necessary in virtual production. But the arrangement changes, since for a virtual production the computer graphic elements cannot be created at the same time, this has to be done before the production starts. Also the high amount of iterations until an asset is finished can be lowered, because it is created once in the beginning and used throughout the whole production pipeline and it is not seen for the first time in post-production, which saves money and time. However, if you want to show virtual characters and objects in real-time you cannot wait for hours until everything is processed. How is this feasible? Computer graphics developed in the last years in two distinct ways. One is the above described process, driven by the film industry and aiming towards being as photo realistic as it can be achieved, basing on technologies trying to simulate the real world. Computer generated photo realism is procurable today, but at the cost of long render times. The other driven development was lead by the game industry. Computer games (except from some examples like round based games, or old text based games) are nearly always in the need of real-time graphics. So the possibilities of game graphics quality was growing with the power of computer hardware, always trying to simulate and fake real situations to keep the computation time as low as possible. Contemporary game engines already produce a nearly photo realistic look at acceptable frame rates if you have a glance at the quality of actual software like the Unreal Engine from Epic Games or the Crytek CryEngine. So it is obvious that these real-time capable graphic technologies are adopted to be used in the film industry as well. And there are already trends

leading to fitting game engines, which were originally created as a development platform for games, better to the needs of a film production environment, for example the Crytek Cinebox that is based on top of the CryEngine.

Figure 5, Screenshot from the game Star Citizen, showcasing computer graphic qualities in 2015.

3.1.3. Synchronization

Not only digital assets have to be rendered and displayed in real-time as mentioned in the prior chapter, every kind of data has to be processed as quickly as possible. Digital film cameras record at a certain frame rate, usually 24 or 25 frames per second. Lower frame rates are technically possible but one would see jittering images instead of seamless motion. Higher frame rates are technically also doable and are used in some situations like high-speed recordings or HFR movies (cf. VES Handbook, p. 526), but technically more ambitious and thus more expensive. Additionally motion capture cameras deliver their data in higher frame rates to deliver seamless motion. Display devices in contrast need a fixed frame rate for displaying images. A 3D rendering engine, that processes the motion capture or pre-animated data for providing the virtual elements also renders at a certain frame rate. So for a resulting image of the virtual and real scenery combined that is fluid and not delayed, the virtual production system must be able to synchronize all the different input and output devices and process the necessary computations in a fast way.

3.1.4. Data Management

As mentioned above there are several inputs and outputs in a virtual production environment. Additionally it is also important having a working data management, being able to save everything right in place and in a persistent way, in a database for example. Important data, currently mostly in different file formats, are motion captured data from cameras and actors, video and audio files from recordings, text documents with information, scripts for processing commands, 3D models of digital assets, textures for the 3D models, shader descriptions to know how the 3D objects should interact with the light in the scene, whole scene descriptions and so forth. Versioning of all of this data, meaning to be able to load older versions of the scene is also a must have, so non-destructive editing is possible and unintended changes can be reverted immediately. Metadata like timecodes also have to be included as a way to align all the assets correctly in time. Moreover titles, durations, creators, parent and child relations, lens data, focal length, audio channels, bit depth or color space of the video images and other information about the recorded data have to be saved. Also a communication tool should be included allowing all members to communicate about the data, assuming a role and being able to approve or deny changes (cf. Methods, Guidelines and Scenarios, p. 68).
By the current state of knowledge these many types of data are saved in various file formats. Thinking of a complete virtual production system it would also help to standardize all these formats for a better exchangeability when one part of the system has to be changed, like a new resolution of the final picture or a new camera type for example.

3.1.5. Scene Distribution

Besides being able to save the data it is also an issue how to keep changes and updates always synchronized in the whole system. Questions that should also be answered are: How can changes be applied in real-time throughout the whole system? How is the same information displayable for a big high definition screen and a little mobile phone display at the same time? How is information distributed through the different system parts? Which axis of the object is desired to be the upper side? How small or big is the scenery in relation to the real world and the different sizes of the digitals assets?
One approach to solve most of this questions could be to link all available information of the virtual and real world in a scene description file which is based on a markup language like xml. These description file can then easily be read by the different systems and fitted to the particu-

lar needs. The first ideas of virtual world descriptions like the Open Inventor (cf. Open Inventor) or OpenGL Performer by Silicon Graphics (cf. OpenGL Performer) were based on the idea to have a better level of abstraction between low level rendering engines like OpenGL (cf. OpenGL), which purely render 3D graphics to flat 2D images, so that they can be displayed on output devices and the scene itself. Today these descriptions are called scenegraphs and can be found slightly modified in nearly every 3D application (cf. Scenegraphs). Today these contain all the objects, properties and relations to each other, different materials, necessary sizes, positions and so on. The scenegraph concept moreover evolved to a complete virtual environment modeling language also including inputs and outputs, audio, animation and so on. This lang-uage is called VRML and was developed by the World Wide Web Consortium (W3C) (cf. 3D User Interfaces, p. 24). Its successor X3D became an ISO standard xml-based file format for representing 3D graphics (cf. X3D). Current technological developments like the distributed scene graph (cf. DSG) by Intel add the possibility to use decentralized and heterogeneous hardware which is often the case in virtual production environments. Looking at what is being used in the industry at the moment, actual file formats like FBX, Collada (cf. CG Formats) or the Pixar Scene Storage Format (cf. Pixar USD) are already very powerful but they do not solve the problem of connecting multiple systems and modifying a central scene through multiple clients (cf. Methods, Guidelines and Scenarios, p. 73).

3.1.6. Motion Capture

Body Motion Capture

Being able to capture the performance of an actor is a key feature for real-time editing in virtual production. Otherwise the motion of a digital character would have to be animated by hand which could be done in preparation, but changing the animation in real-time is not practical, as it disables the altering of the scenery live on set. There are several technologies available for capturing motion; considering the scope of this work it is only possible to mention the most common ones briefly. For a good overview of the pros and cons of the different performance capturing and facial capturing technologies a great overview offers (cf. VES Handbook, pp. 385-445). Passive retroreflective optical systems work on optical tracking technology like it is described in Chapter 3.1.1., relying on reflective markers worn on several positions of an actors body. Active optical systems use markers emitting light themselves (cf. VES Handbook, pp. 390-392).

Inertial capture systems use the technology with the same name as described in Chapter 3.1.1. They are very popular in the industry, because no line of sight to several cameras is needed and they can be used in large environments. However they do not provide positions, only rotations (cf. VES Handbook, p. 392). Bend sensors work on the basis of measured angles: a joint like a finger or a body part is bent. As for inertial capturing devices, these technologies miss position values (cf. VES Handbook, p. 397). After capturing the motion captured data are normally revised. Gaps are filled, jitters are cleaned and overall the data is solved for having a clean animation afterwards. (cf. VES Handbook, pp. 418-423). Never the less real-time motion capturing already delivers sufficient results for virtual production and previsualization purposes.

Figure 6, A motion capture studio, showing actors in motion capture suits with passive markers for optical tracking.

Facial Motion Capture

Facial motion capturing describes the task of recording the movement of an actors' face and transfers these facial expressions onto a virtual character. Facial motion capturing is an area of very active research and also its very own topic. This chapter intends to describe the specifics of facial motion capture although this technology was not used or evaluated in our prototype implementation.

As of today, there are several technologies available. First to mention is also just putting markers on the face and recording them likewise in an optical body motion capture system; however the cameras need to be closer to the face or having a higher resolution to capture the correct positions of the normally smaller markers. Another problem is that they tend to fall off and need to be readjusted (cf. VES Handbook, p. 434). Another approach often watched is using single or multiple cameras mounted to a fixed location relative to the face. The motion is then detected through feature detection of high resolution images when working with one camera or

by reconstructing 3D information about the face when having multiple cameras available. One disadvantage is for example that the actor is not able to kiss someone, as the cameras are in the way for several motions (cf. VES Handbook, pp. 435-436). Further information can be found in the VES Handbook (cf. VES Handbook, pp. 424-438).
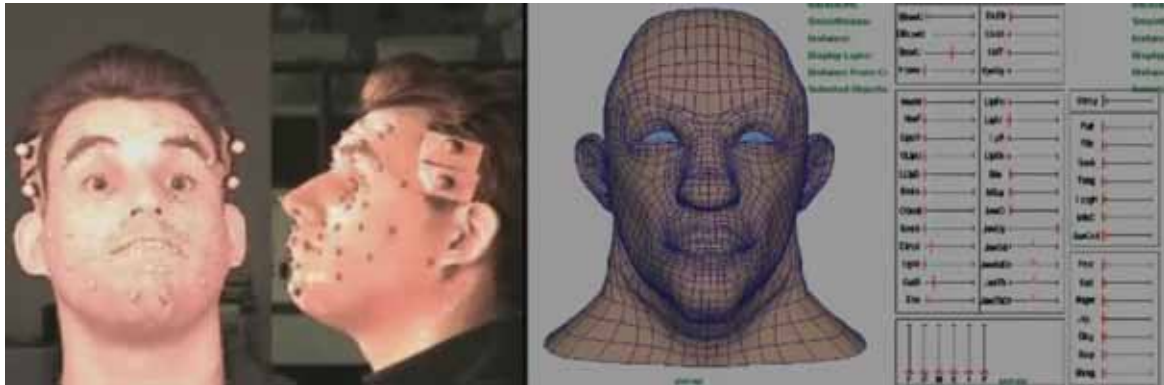
Figure 7, Facial Motion Capture, Motion data from the actor left are transferred to the digital character right.

3.2. Real-Time Editing in Virtual Production

As already mentioned in Chapter 2, first virtual production approaches only focused on getting an insight into how virtual and real content fit together during production, or on the passive act of scouting the scenery augmented with virtual elements, known as virtual scouting. From today's perspective virtual production is more than that. Following the definition in Chapter 2.4 virtual production also includes interaction (cf. VES Handbook, p. 444). The next three chapters describe three possible ways of real-time editing in virtual production. Being able to edit virtual elements directly on set has the potential to bring interaction in virtual production.

3.2.1. Real-Time Set Editing

The idea of set editing is to enhance virtual production through the possibility of editing virtual objects directly on set. Normally the virtual elements of a scene are created in specialized software applications before or after a film is shot. Though, very often it turns out during shooting that changes have to be made, and virtual elements need to be repositioned to fit to the real elements. Adding a real-time set editing option to virtual production enables one to manipulate the position, rotation and scale of an object directly on set. But how is it possible to achieve such thing?

An intuitive user interface is necessary, since the creative people on set like the cinematographer or the director are often not very well educated in 3D applications. The user interface

should these non-expert users offer to make changes without an extensive learning process in advance. The system should be capable of streaming the image of the whole scene to several clients of different technical specifications and from different points of view, so as to be able to make annotations, mark changes that need to be put into execution from other team members or to do changes directly and delivering real-time feedback (cf. Interface Description, p. 23). Set Editing is the real-time technology that is implemented prototypical in this work. The prototype is further described in Chapter 5 and evaluated in Chapter 6.

3.2.2. Real-Time Light Editing

Based on the set editing capabilities described in the chapter before, light editing introduces additionally a way to edit the lighting situation on set and match virtual and real lighting conditions. When editing virtual light values through a user interface the system could automatically change the parameters of real lights, through DMX (cf. DMX512) control commands for example. In the other direction when real lights are changed, the altered parameters could be digitized and transmitted to the system, which fits the virtual light parameters to the real ones. So lighting conditions of real and virtual elements could be kept in relation for the whole production. This can save money and time, since it is currently still necessary to match the lighting of virtual elements to the conditions which could be found on set in an iterating post-production process (cf. Interface Description, p. 24).

3.2.3. Real-Time Animation Editing

Creating good animations for virtual objects is very creative work and needs a lot of experience and training. But animations are also a very important part of virtual elements in a film. Hence it would be a great benefit to the existing workflows to be able to edit animations while exploring their look in a three dimensional environment on set. It would give a deeper immersion into the scene if the animation operator is able to scout the scene, watching it from different perspectives. Simultaneously he or she could be able to alter the animations until they feed his or her creative ideas: Furthermore it would be able to share ideas in real-time with a director, explore them together with other creative people and discuss about the progress of the work. To render this possible there is also the need of a certain user interface, allowing to edit animation data in an intuitive way and not being fixed to a special purpose software that needs a lot of training and the usage of complex commandos (cf. Interface Description pp. 24-25).

3.3. Input and Output in a Virtual Production System

Knowing the most important technologies necessary for virtual production systems, as described in the former chapters, this section provides information about typical classification criteria for input and output devices. With the aid of these classifications it compares different input and output devices. Since there is an unmanageable amount of devices present, the focus is on devices with a higher significance generally and in virtual production environments especially. A more complete list of input and output devices can be found in Faisst (cf. Navigation and Interaction, Appendix E and F) or in 3D User Interfaces (cf. 3D User Interfaces, pp. 29-132). Summing things up this part discusses which kind of devices are the best fitting solution for the primary tasks in a three dimensional environment and how they can be used for developing an innovative 3D user interface for editing. Moreover it also sheds some light on what a 3D user interface is and how it can be distinguished from classical user interfaces.

3.3.1. Classification of Input and Output Devices

Input Devices

Input devices have a lot of different characteristics. The most important is the degree of freedom, short *DOF*, normally classified depending on how many DOF are provided simultaneously. A degree of freedom can be described as an independent way that an object moves in space. Prominent examples are the classical computer mouse which has 2 DOF integrated control, but several tracker devices allow simultaneous control of 3D position (3 DOF) and 3D rotation (3 DOF), so all together 6 DOF integrated control are achievable with this systems (cf. 3D User Interfaces, p. 144).

Furthermore it is important whether the device works *isotonic*, *isometric* or *elastic*. Isotonic devices, called either free moving devices or displacement devices, consist of zero or constant resistance. A classic computer mouse again is a famous example for an isotonic device. Isometric devices in contrast are also often named pressure devices or force devices because they measure how much force acts on them but they do not move. In between these two classifications one can find elastic devices which have a varying resistance. Shumin Zhai distinguishes between spring-loaded devices, when their resistance increases with the force put into account, and viscous devices that have a greater resistance with a higher velocity and inertial devices in case the resistance increases with acceleration (cf. Human Performance in 6 DOF, p. 13).

Another very important feature of input devices is the kind of *mapping* that is used together with the device type or in other words with what kind of function the data is transferred from the device to the application making use of them, therefore this mapping is also referred to as "transfer function" (cf. Human Performance in 6 DOF, p. 17). For mapping, the two best known ways are isomorphic control or *position control* on the one side and isometric control or *rate control* on the other side. Isomorphic control directly uses the position alteration from the device, which is useful for changing an object's position for instance. Isometric control measures the power applied by the user, being more useful for changing speed for example (cf. 3D User Interfaces, p. 145).

Output Devices

Output devices provide feedback and information to the user. They are sometimes also called display devices, referring to visual displays. As information could theoretically be provided for every human sense, most display devices focus on visual, auditory or haptic senses (cf. 3D User Interfaces, p. 29).

For visual displays the most important technical characteristics are *field of regard (FOR)* and *field of view (FOR)*, *spatial resolution*, screen geometry, light transfer mechanism, refresh rate and the kind of supported visual depth cues. To give further information, field of regards means the amount of physical space around a user that is usable for displaying visual images. Field of view describes the maximum number of visual angles in degrees that can be seen instantaneously. Spatial resolution is the amount of pixels that can be displayed in a given screen size. The screen geometry relates to the shape of the device. Light transfer mechanisms describe how the light from the visual image is transferred onto the screen. The refresh rate tells something about how often the device is able to update the visual image (cf. 3D User Interfaces, pp. 31-34).

Auditory and haptic displays exist but are not in the scope of this work. As they are also interesting to be considered for a complete immersive experience one can find a good overview of these devices in 3D User Interfaces (cf. 3D User Interfaces, pp. 59-86).

3.3.2. Important Input Devices

Keyboard

The keyboard is the most classical device for data input. It relies on several buttons or keys which deliver commands to the computer. The most common "QWERTY" keyboards today, named by their typical key arrangement, have about 104 keys but with mode change keys there are far more commands possible. There are a lot of special purpose keyboards available, extra layouts for different languages, space saving layouts, even keyboards that can be rolled in or are waterproof.

Keyboards are divided in the two categories fixed function and variable function. Fixed function keyboards have a fixed mapping for every key, whereas variable function keyboards allow to change the key mapping (cf. Navigation and Interaction, p. 18). The keyboard can be used for navigation or manipulation if the keys are mapped to specific commands, like forward/backward or up/down for instance, but a lot of key changes have to be done and the mapping is not very intuitive. Pressing a key for moving forward or turning around is well known to people playing computer games, but not for average persons.

2D Mouse

Also very common is the 2D mouse. Small cases are moved over a flat surface by the user and the motion of movement is transmitted to the computer either wirelessly or with a cable. The two common principles of how mice are constructed are mechanical or optical mice. Mechanical mice contain a small ball which translates the chassis movement to little wheels which encode the movement by potentiometers or optical encoders. Optical mice scan the surface in a certain speed with a laser and are hereby able to decode the movement out of the changing underground (cf. Navigation and Interaction, p. 19). A 2D Mouse provides 2 DOF simultaneously and is normally used with isomorphic control. 3D Editing Applications that lack a 3D user interface base their editing possibilities on an editing interface where the mouse is used in conjunction with several option keys which permit editing of the third dimension or allow to switch between rotation and translation, but not at once. These option keys can be used from a keyboard or from the mouse itself. Most mice come with two or more keys today. Mice are used with position control, but the mapping of the movement does not have to be linear.

Trackballs

Trackballs are similar to the 2D mouse in many categories. The most important difference is that the user does not move the device on a flat surface, but furthermore a trackball consists of a movable ball sitting in a static frame, so that the user rotates the ball but the device itself does not move. So there is no need for a flat surface of certain size like for the mouse, since the device does not need to move. Modern Trackballs such as the SpaceMouse Pro (cf. SpaceMouse Pro) or the SpaceBall 5000 (cf. Spaceball 5000) by 3Dconnexion also deliver 6 DOF freedom input, but those are still built as desktop input devices (cf. Navigation and Interaction, p. 21).

3 DOF / 6 DOF Tracker

Based on several of the tracking technologies described in Chapter 3.1.1 very important input devices for virtual production are also trackers providing position or orientation in space for 3 DOF or both together for 6 DOF input. They can used to track parts of a human body or any other kind of object that is for interest (cf. Navigation and Interaction, pp. 22-23). One example can be seen in Figure 4. Mostly trackers are used in combination with other devices. Using trackers together with a head-mounted display allows the computer to know where the user is looking at, or combining them with data gloves lets the computer know where the operators hands are currently.

3D Mouse

3D mice combine 3 DOF or 6 DOF tracker devices with several additional input options like buttons for instance or knobs for additional DOFs. The range is wide, from handheld devices to user-worn devices that are put on a finger of the operator. Mentionable devices that fall in this category are the recently developed Razer Hydra (Figure 3), the Cubic Mouse (Figure 8) or the Wanda input device (cf. 3D User Interfaces, pp. 110-114).

Figure 8, The Cubic Mouse.

Joysticks

Joysticks are also well-known devices. They are mostly built as sticks which are fixed on a chassis at the bottom and only their upper tip can be moved up to a certain degree. Typically they are used as desktop devices.

Faisst classifies three different kind of joystick types (cf. Navigation and Interaction, pp. 20-21): Firstly, digital joysticks deliver only values for the eight directions one can turn the stick towards, that is forward, backward, left, right and the corners in between which tell the system if the stick is pushed into that direction. So the joystick is not capable to detect how far the stick is pushed into a certain distance, he delivers only the value one to the computer if he is pushed in a specific distance and zero if this is not the case. A typical interpretation of these commands would be to move an object in a program or a computer game at a certain speed constantly into the activated direction. Secondly, displacement joysticks gain a higher resistance with the more they are tilted in any direction. These joysticks can be used for position or rate control either. Lastly force joysticks have a fixed, rigid stick which only measures the amount of pressure an operator applies to the device and the direction the force is applied towards. These devices are better suitable for rate control because of their little to no displacement (cf. Navigation and Interaction, pp. 20-21).

Joysticks were first introduced with arcade machines and became popular with computer games like car racing games or flight simulators throughout the years. Their concept offers good possibilities for a 4 DOF mapped rate control. Car driving simulators can use the forward and backward movements for increasing speed or making use of the break and left or right for respective car turns. Flight simulators normally map the forward and backward movement of the joystick to pitch the aircraft and the left or right movement to roll it. With the exception of special 3D joysticks, which include a third axis by twisting the stick in clockwise or counterclockwise direction (cf. Navigation and Interaction, p. 21), conventional joysticks lack this supplementary axis which is a requirement for editing tasks in a three dimensional environment.

Data Gloves

Another interesting input device are data gloves, which are wearable. They consist of a glove that measures the bending of the operator's fingers. Thus it is able to detect specific gestures. It is possible to enhance these devices with a 6-DOF tracking device and/or a tactile feedback system for the finger tips for instance. (cf. Navigation and Interaction, p. 24). Since the human

hand in theory has 27 DOF (cf. Handrix, p. 3) there are great possibilities for editing in 3D using data gloves through the recognition of gestures the operator forms with his bare hands. This is especially true when one is wearing a HMD which restricts your sight, so you do not have to hold a device. However a disadvantage is the high price of these systems. Another problem is that the glove has to fit to the human hand which is not always the case due to fixed sizes (cf. 3D User Interface, p. 108).

Figure 9, The Peregrine Glove, a touch based data glove.

Direct Gesture Control

Another interesting input form for gesture control besides data gloves is the gesture recognition either by surfaces or by cameras. Surface-based recognition is often used on touchscreens when multiple finger gestures are mapped to specific commands. Camera-based systems work with computer vision technology to recognize the human hand's gestures (cf. 3D User Interface, p. 272). They had a growing market presence throughout the last years, since computer vision algorithms and camera technologies got better.

A very interesting device in this area is the Leap Motion (cf. Leap Motion), published by the company of the same name. It takes only little space (13mm width and 76mm depth) and consists of three infrared LEDs and two cameras who gather the infrared light reflections. The infrared light spectrum is outside of the visible light spectrum. From the light reflections the Leap Motion identifies the position and rotation of the operator's hands (cf. Leap Motion System). Using this device entails several benefits; on the one hand, the Leap Motion controller enables gesture recognition without the need of any further equipment. Above all, the amount of DOF is not coupled to the technical specifications of a glove or any other input device; rather, it is crucial which gestures the operator can execute and the Leap Motion controller itself can detect. This requires that the desired range of gestures have been implemented beforehand

with the appropriate software API. Acting on the assumption that the hand has 27 DOF (cf. Handrix, p. 3) this provides a broad range of interaction.

The Microsoft Kinect as another current example based on the same technology recognizes full body motion and makes use of this in computer games and other software applications.

Figure 10, The Leap Motion, a gesture recognition device.

Tablets

Constituting a mixture of an input and an output device, tables are widely used for various purposes today. The computer screen acts as the output device; then there are two ways to directly interaction with the screen - this could either be done by a pen, in pen-based tablets (cf. 3D User Interfaces, p. 92), or by the operator's fingers in touch based devices.

Some tools include a fully functional computer system, others can be seen as the combination of a 2D mouse and a monitor. The input is very closely related to classical 2D computer mice. The interaction is always based on plane 2D coordinates, defined by the finger's or pen's position on the screen. Latest developments like Project Tango (cf. Project Tango) from Google try to add 6 DOF tracking capabilities to tablet devices.

3.3.3. Important Output Devices

Monitors

The most common visual displays are monitors, available in different sizes and technical specifications. In connection with different types of glasses it is even possible to view stereoscopic pictures. Monitors have their limitations in their degree of immersion and their small FOR, as the user is not able to move the head a lot without losing sight of the screen (cf. 3D User Interfaces, p. 42). Smaller, portable monitors are available as known from tablets or smartphones, but the user has to hold the device, which prevents the hands from doing some other task such as editing.

Surround-Screen Displays

These devices consist of three or more big surfaces for projecting visual images. Normally these are placed so as to surround the operator. The usage of rear-projection technology, where the image is displayed onto the surfaces from behind, avoids shadows of users or spectators on the surfaces. They come with high spatial resolutions besides a big FOR and FOV. Unfortunately, disadvantages are the high price of such systems and the required space (cf. 3D User Interfaces, pp. 43-35).

Workbenches

Workbenches are projection-based displays developed for usage in conjunction with tables. They can be used in combination with stereoscopic glasses and user-tracking devices for displaying and editing data in 3D. Since the operator also relies on the workbenches' screen position to see the objects and these devices normally are normally not movable, workbenches are only feasible for editing tasks on a fixed position and not for moving around (cf. 3D User Interfaces, p. 47).

Head-Mounted Displays

As their name already reveals, the main unique feature of head-mounted displays (HMD) is the possibility to wear them on the operator's head, so the display position is always relative to the user's FOV. They consist of either one visual display when watching monoscopically, or two separate visual displays, one for every human eye when watching stereoscopically. Often lenses help the operator to focus the displays at such close distance as an HMD delivers them (cf. 3D User Interfaces, p. 49).

A big asset of HMDs is that they are capable of complete visual immersion. This means the operator is able to move his or her head freely and have a 360° look around in the virtual environment. This is facilitated by tracking devices usually integrated into head-mounted displays. Another advantage is the portability of these devices.

However, the visual displays built into HMDs are normally of lower quality in terms of spatial resolution so they are less heavy and easier to wear on the user's head. (cf. User Interfaces, pp. 49-52). Also the blocked vision of a user is a problem, which is sometimes avoided by so-called see-through head-mounted displays. Rolland and Fuchs describe two types of see-through displays: Optical see-through devices enable the user to see the world through half-

transparent mirrors directly; video see-through devices record the real world with cameras and electronically combine the virtual elements with the video representation of the real world (cf. See-Through Displays). HMDs have been a topic of research for a long time but until a few years ago they were high-cost devices, often only affordable for research in military or special training applications. These devices also had problems with the delivered FOV, only providing 30 – 60 degrees horizontally (cf. User Interfaces, p. 50).

Recent interesting developments in this market are the Samsung Gear VR (cf. Gear VR) and the Google Cardboard (cf. Cardboard), two devices only providing lenses and a case. They work only with a smartphone injected into an available slot, making use of the smartphone's display and its inertial tracking for measuring the user's head rotations. The Samsung Gear VR was developed in cooperation with a company called Oculus VR, which was bought by Facebook in 2014. Oculus VR developed a full HMD itself, called the Oculus Rift (OVR).

The OVR is currently (early 2015) still in development, and Oculus VR releases new development kits from time to time, showing their actual development status. The technical specifications of the Oculus Rift lead to an impetus for the virtual environment market and pushed other companies to focus also on the development of affordable HMD devices for the mass market, like the computer games industry. The Oculus Rift DK2 (cf. Oculus Rift DK2) consists of a very good nominal FOV of 100 degrees, 960x1080 pixels resolution per eye as well as position and rotation tracking. The DK2 is available for 350 US Dollars, which is much less than other systems before. In comparison, the Wide5 HMD System by Fakespace Labs was sold to the US government for 32500 US Dollars in 2007 (cf. Wide5). The Oculus Rift DK2 delivers no see-through capabilities. The low price of these newer systems and the high degree of immersion makes them very interesting for usage in virtual environments.

Figure 11, The Oculus Rift DK2, a head-mounted display.

Virtual Retinal Displays

Another approach to deliver visual images to the user's eye is using virtual retinal displays. This technology works by displaying images directly onto the eye's retina. The so-called virtual retinal displays (VRD) or light-scanning displays create coherent beams of light from photon sources. These light beams are used to draw a rasterized image onto the retina. VRDs offer a high potential, approaching the FOV of human vision by presenting bright, high-resolution and stereoscopic images (cf. 3D User Interfaces, p. 54). But research in the field of VRDs still reveals some issues that have to be eliminated, like unforeseen eye rotations and problems with correct stereoscopic images. More information about the actual research topics and what problems need to be overcome for a brighter usage of VRDs can be found in 3D User Interfaces (cf. 3D User Interfaces, pp. 54-55). For sure it is interesting where these developments can lead towards in the future and what kind of interfaces and devices are possible. However, it is interesting to read that a company called Magic Leap has recently raised a half billion dollars from Google to work on retina based technology (cf. Magic Leap).

3.3.4. Devices and Interfaces for Editing Tasks in a 3D Environment

After it was given a look on several input and output devices, it is discussed in this section what devices fit best for editing in a 3D environment. The most common devices for interacting with classic user interfaces of a computer today are the keyboard and a mouse. Especially for the two dimensional principles like drag and drop, interface widgets like buttons, menus and windows these devices are widely accepted.

However, working in a three dimensional environment is different and devices like the ones mentioned above are often inappropriate. Placing an object anywhere in 3D space with a 2D mouse is insufficient (cf. 3D User Interfaces, pp. 3-5). Zhai has shown that for 3D manipulation tasks multiple DOF devices which integrate control of all input dimensions are the best choice (cf. 3D User Interfaces, p. 144). Since editing in a virtual environment consists of three dimensions for translation and three dimensions for rotation, 6 DOF already provide sufficient results. A device which offers more DOF simultaneously could be able to offer more tasks at the same time. Research has shown that it works well for users teing able to navigate and manipulate in 3D space parallel with an elastic 12-DOF device (cf. Evaluation of 12-DOF). 3D user interfaces all in all match better with the characteristics of 3D environments and can provide a sense of "presence", also called immersion, which enables users to make better use

of their natural skills (cf. 3D User Interfaces, p. 4). It is also important to mention that 3D user interfaces which provide more DOF to control simultaneously are important for users to specify points, directions, gestures and other actions in a 3D space (cf. 3D User Interfaces, p. 17) People don't need to learn how to walk or how to grab something, they normally learn that already in the course of becoming mature. Additionally wearable computing devices like head-mounted displays try to bridge the gap between collecting information for some task and performing the task itself, for example instead of turning a view to know how an object looks from another side, just walk there yourself and have a look (cf. Be-greifbare Interaktionen, p. 236). There are lots of tasks one can imagine in 3D, but except from some cases that should be target by special research, almost every action can be broken down into small tasks. The so-called basic manipulation tasks are: *selection,* meaning the task of identifying an object out of a set of objects, *positioning* as translating an object in space and *rotation* as the task of an orientation modification (cf. 3D User Interfaces, pp. 141-142). Also important is *traveling* as the task of getting from the current location to a new location. *Wayfinding* is the task of finding a path through a environment, but wayfinding relates mostly to artificial intelligence systems like robots or other computer systems. Since a real-time editing system for humans should be achieved, wayfinding can be further ignored. *System control* is the task to trigger an action or an event and lastly *symbolic input* as the task to hand values over to a system like numbers, text or other information.

As seen in Chapter 3.2 a real-time editing interface should be lightweight and transportable, so the operator is not fixed to a desk and can explore the scene while editing from any position. Additionally, the desired 3D user interface should not block the users sight, allowing him or her to walk around, it should also track the users position and orientation in space so it is able to display the virtual elements correctly. Furthermore it should allow an intuitive multi DOF input and real-time output. So the next chapter takes a look what currently existing systems can offer,. Afterwards several graphic engines are evaluated to select the best choice for developing a real-time editing interface. Subsequently current research projects related to these ideas are presented in Chapter 4 and the prototype system for real-time set editing is introduced in Chapter 5. This prototypical system already implements most of the here described basic manipulation tasks.

3.4. Existing Virtual Production Devices

There are already some virtual production systems on the market. Most of these systems are combinations of different devices, software and hardware components. To get an impression of the current state in the industry this chapter gives a very short view on up-to-date systems.

OptiTrack Virtual Camera System

The virtual camera system from OptiTrack (cf. OptiTrack VCS), the company which also sells motion capture systems provides a trackable rig, that can be worn like a shoulder camera. Used in combination with a motion capture system, it provides position and rotation tracking of the camera, being able to scout the virtual scene through the mounted monitor. The motion data from an actor and/or the camera can be streamed in a 3D application and the cameras perspective from the 3D application is then displayed on the monitor. Also live scouting of motion capture data transferred onto virtual characters is possible. But the system lacks of any real-time editing features.

Figure 12, The Virtual Camera System from OptiTrack at work.

Vicon Virtual Camera

The Vicon virtual camera (cf. Vicon VC) is a very similar system to the OptiTrack VCS. It is also a shoulder rig for hand camera movements with a monitor and is normally used in correlation with the tracking data from Vicon's motion capture systems.

nCam Camera System

The nCam Camera System consists of a workstation computer and a tracking bar. The tracking bar can be mounted onto every industry typical film camera with front rods and provides optical and inertial tracking technology. So the system is able to estimate its position and orientation in space without the need of an external motion capture system. It also takes into account the cameras focus, iris and zoom values by reading them from a Preston wireless lens and camera control system. The camera image and the tracking data are delivered to the workstation and can be processed together there. Providing a suitable virtual scene from a 3D application can be composited in the server and an augmented picture is then streamed back to the camera, so the camera operator is able to see the augmented picture in its viewfinder. The nCam system works with a delay of 2.5 frames (cf. nCam System). It also provides basic color correction of the cg elements in the server, basic rendering techniques and a real-time keying function.

Figure 13, The nCam camera tracking system (bottom) mounted onto a movie camera.

Simulcam

Pioneering the field of virtual production, famous director James Cameron used a system called Simulcam for his feature film Avatar. The system was invented for that purpose and already combines cg and live-action in real-time (cf. Simulcam).

LightWave and VCam

The VCam from Intersense together with LightWave 10 from NewTek together already combine a 3D software and a virtual camera system. The VCam is capable of real-time camera

tracking for virtual scouting and producing virtual content (cf. VCam).

Virtual Backlot

Virtual Backlot (VB) is a proprietary technique from Stargate Studios. The system is capable of bringing location footage from recorded scenes all over the world and material from a green screen studio together in real-time and has with VB Live also a tool for real-time previsualization and compositing (cf. Methods, Guidelines and Scenarios, p. 44).

3.5. Evaluation of Real-Time Graphic Engines for Virtual Production

This chapter illustrates the decision making process to pick out a suitable software development basis to realize the prototypical real-time editing system, further described in Chapter 5. After a first pre selection based on experiences, the available documentation and the software's feature lists (table 1), it was decided to further investigate XML3D, Blender, Unity, Motion-Builder and the Shark3D engine. For that a small task was constructed which shall be solved within the several frameworks and measured progress and effort it takes for the implementation. Next to this, also the quality of documentation was included, together with the existence of a community and the presence of project related tools. The task contains the import of an animated object out of a DCC application into the engine and the manipulation of the object by a simple GUI element. The manipulation itself was a simple rotation along one axis in space.

Pre Evaluation of Several Frameworks

	Blender	XML3D	Frapper	Unity	MotionBuilder	Shark 3D	Unreal Engine	CryEngine
Viewport Interaction	+	~	~	+	+	+	~	+
Platforms	~	+	-	+	-	+	+	+
Existing Tools	+	-	-	+	+	~	~	~
Video Input	~	~	+	+	+	+	~	+
Modularity	-	+	+	~	-	+	~	~
Frame Sync	~	-	~	~	+	~	~	~
Animation	+	~	+	+	+	~	+	+
Existing Experience	~	-	+	+	~	-	-	~
Open Source	+	+	+	-	-	~*	~*	~*

+ available

~ partly available, needs further development

- not available or need of new development

* source can be inspected but not edited

Table 1, graphic engine feature comparism.

Viewport Interaction

Possible ways for navigation, selection and manipulation in a virtual scene.

Platforms

Supported platforms e.g. Windows, Mac OS, Mobile (Android, iOS).

Existing Tools

Already available tools for interaction with a virtual scene and for animation editing.

Video Input

If a SDI and/or USB video input is possible or not.

Modularity

Possibility to: write plugins for the framework, include external libraries, run the framework and editors itself as a plugin in other applications.

Frame Sync

Synchronize SDI Video input with rendered image and image output.

Animation

Supported animation formats and techniques, e.g. skeletal animation, skinning, vertex animation, complex rigs, simulation, point caches

Existing Experience

Does anyone involved in the project has experience with the framework in other projects.

Open Source

Is the source code of the framework fully accessible and editable.

Unity Engine

3D Object Import

Unity supports a wide range of data formats. Because of practical reasons it was decided to use the quasi standards Maya ASCII which is the Autodesk Maya internal scene description and the latest version (2014.1) of the Autodesk Filmbox format. Next to textured static geometry these formats also support animations as well as more complex hierarchies, rigs, controllers and basic shaders.

Object Manipulation

The user has access to every object within a running game context and via registered network events also to objects in other game contexts. So it is trivial to write a script to find the related object and change a property like the actual rotation.

Creating GUI elements

Unity provides two base classes of GUI elements which are fully accessible through the scripting engine. One to create interfaces for the Editor during "edit time" and one class for run time environment interfaces.

Assessment

Required time: minimal (2h)

Quality of documentation: very good

Popularity: High

Programming language: C#, Javascript

The Unity API provides a good structured, wide range of functionality and a well-made documentation which is rich of examples and tutorials. This results in a low level entry and very short prototyping times. Related to the huge community a wide range of implementations over the whole functional spectrum are free available.

However the engines source is closed and the coding restrictions are clearly defined. This means that development is only possible to some point. Changing or accessing the engines core functionality is not possible. This could be problematic regarding to changes or extensions on core functionalities e.g. the rendering pipeline. Furthermore the use of external C/C++ librar-

ies requires the implementation of an additional interface and a wrapper for all library specific functionality.
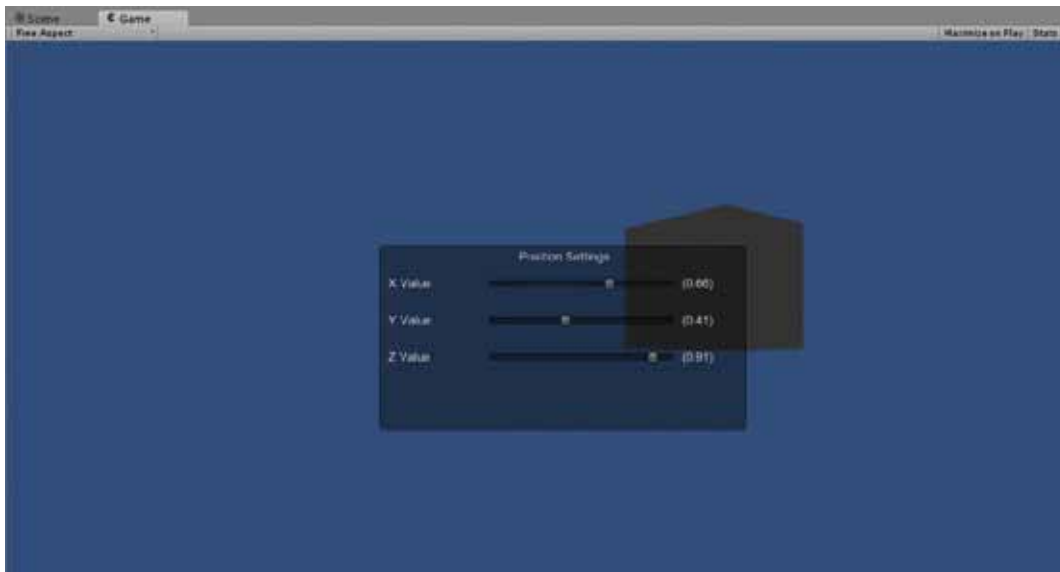

Figure 14, The evaluation setup in Unity.

XML3D

3D Object Import

XML3D uses a customary file format, which is able to export static geometry. Exporters are currently only available for Blender and Maxon Cinema 4D. Complex rigs and shaders are not supported by the existing exporters even though the Framework would support it. Vertex and skeleton animations are supported by using the XFLOW extensions and the JSON data format. But also here the data exchange pipeline is hard to use and needs further development of ready to use tools for content developers.

Object Manipulation

For manipulating the transformation properties of a scene object, the objects scene node in the XML description has to be changed. The changes can be done by manipulating the XML description via the DOM structure by a simple JavaScript code snippet.

Creating GUI elements

In XML3D all web based GUI elements are available. A test was performed with classic HTML elements like <input>-fields as well as JavaScript based JQuery GUI elements. As long as WebGL is supported, the access to web based elements also allows a seamless and easy integration into every web context.

Assessment

Required time: minimal (2h)

Quality of documentation: moderate (less tutorials, missing parts)

Popularity: low

Programming language: XML, JavaScript

The easy use of web based programming and the possibility to run on every platform that supports WebGL are the biggest advantages of XML3D. The documentation could be more comprehensive and related to the small community also the amount of available examples and tutorials are barely available. Other problems are the lag of functionality in animation, editing, data exchange and the limitations dictated by the WebGL standard. The software is open source and all these points are addressable, nevertheless it would require further development and time. The usage of a web browser as central piece of software in a virtual production environment also appears rather cumbersome.
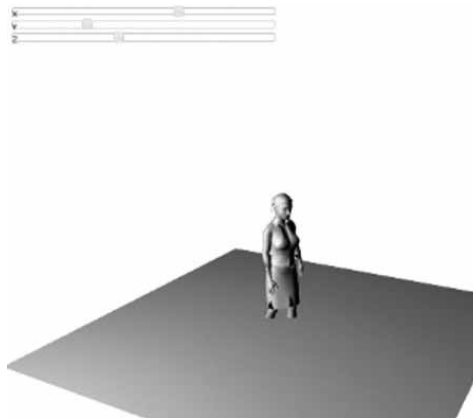

Figure 15, XML3D: Web side using JQuery GUI elements to manipulate 3d object.

Blender

3D Object Import

Next to the functionality to create assets including animation, lighting and shading directly, Blender is able to load in FBX files. Not all features of the FBX file format are supported (e.g. constrains), but enough to build up an Object import pipeline from common DCC Tools, including animation, basic texturing, shading and simple rigging.

Object Manipulation

The manipulation of objects in a scene is possible via the Python scripting API, which is also well documented.

Creating GUI elements

With the tested version of Blender (2.7.1) one was not able to solve the task within the given time just by using Blender internal functionality. Over the last versions of the Blender API, a variety of options for creating GUI elements popped up and disappeared. As a result the Documentation at this point is ambiguous and inconsequent. At the current point, the actual API documentation seems to be fragmentary incomplete.

Assessment

Required time: high (4h, still incomplete)

Quality of documentation: low, incomplete

Popularity: high

Programming language: C, Python

Blender is a full grown DCC package providing a wide toolset for creative asset generation and a powerful scripting interface. The poor documentation is the biggest drawback. Parts of it refer to functionality, which is no longer available or has changed over the last versions. Contingent on the C API and the lag of a plugin interface, the implementation of external libraries would require rebuilding the whole blender framework every time some functionality should be added or modified.

Autodesk MotionBuilder

3D Object Import

A huge set of asset formats, including animation and complex rigs is supported. The exchange from a DCC package into the software was absolutely unproblematic.

Object Manipulation

MotionBuilder provides very well documented API in both domains, for scripting with Python during the runtime and for creating precompiled C++ plugins. For the test case the required functionality manipulating the test object by using the Python API, as well as the C++ plugin version could easily be implemented.

Creating GUI elements

A specialized part of the SDK provides all necessary functionality to develop GUI elements. Guided by the documentation it was possible to create the test GUI elements in a time efficient end easy way.

Assessment

Required time: low (1.5h)

Quality of documentation: very good

Popularity: high (but specific context: Motion Capturing)

Programming language: C++, Python

MotionBuilder offers a good structured and well-made documentation of all available interfaces and extension possibilities. The test example could be realized fast and easy. In addition it offers a huge set of already existing, professional functionality for animation and animation editing. On the downside, MotionBuilder is closed software with clear restrictions. Especially the build in real-time rendering is not in keeping with the times. Another drawback is the high price.
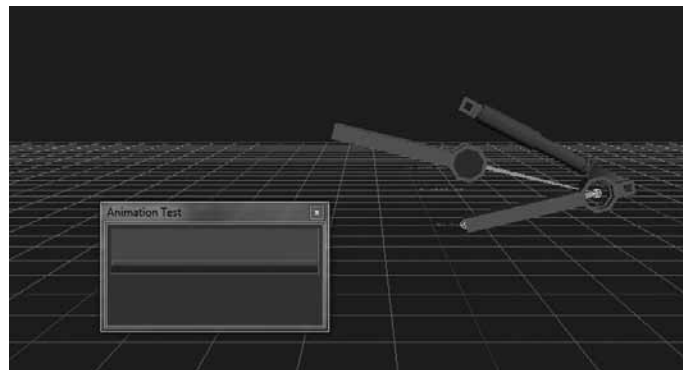

Figure 16, Test Plug-In developed to evaluate the functionality in Autodesk MotionBuilder

Shark3D

Assessment

Required time: not tested

Quality of documentation: good, direct contact to developer

Popularity: low, (specific context : TV and live events, games)

Programming language: C++, Python

Shark3D was not evaluated within the test cases. However the engine, the tools and workflows were presented during a live demo at Filmakademie. The engine provides a powerful way for real-time rendering and also the possibility to exchange the existing renderer with complete other systems (e.g. ray-tracers). The source code is open, but for the core, code editing is not allowed. However if it should be necessary to edit it, the company would rate this as a bug in their software concept and try to fix the problem. The core and plugins are written in C++ the user interface is based on WxWidgets using Python Scripting. The user interface is completely

independent and just communicates over an open protocol with the core engine. So an complete Headless use or an remote control is easy to realize. The Engine and all the concepts are really complex and it needs some time to get used to it. A good amount of basic functionality for animation and animation editing already exist. But nevertheless the community using and extending the engine is small and it will take a lot of effort and time to develop eventually missing functionality.

Conclusion

Due to the insufficient documentation, missing API functionality and the lag of a proper plugin interface Blender drops out of consideration. XML3D seems to be in an early stage of development. Next to the limitations of Web3D based systems, it needs further testing iterations and better tool and data import functionality. The documentation of XLM3D could also be more detailed. Even though there are good connections to the authors of the software, decision was made against the use of XML3D for the implementation of the next prototype. The effort to add the missing functionality and to push the software to be an efficient platform for the next prototype is still too high and in the available time far beyond any possibility. From this evaluations point of view, MotionBuilder and Unity are close to each other. Both systems have closed sources, but well-made documentations and a huge set of already existing functionality. Shark3D is somewhere in-between. At least for understanding underlying core functionality, the sources are open and a close connation to the developers exists. But still, Shark3D is a complex and comprehensive software and the existing community is small. It would take some time to understand the mechanisms and philosophy behind the architecture. To catch up with the available functionality of Unity or MotionBuilder also Shark3D would need some further development. Unity offers the access to a huge community providing tutorials examples and a rich pool of other functionality. Most of the basic elements, which for the other frameworks need to be developed in addition, are already available. The good data import pipeline including the possibility to export more complex rigs and animation and the proper documentation are final arguments to choose Unity as first prototype development platform. For ongoing, future developments also the possibility to easily port developments to other platforms like mobile devices are interesting features and reasons for the use of Unity. Looking on the projects limitations described in Chapter 5.1 it is clear to see that Unity is the best choice for the desired set editing evaluation.

4. Related work

This chapter is designated to introduce work that is a good read in relation to the topics of this thesis. Not every work presented here corresponds completely to the research goals of this paper, but in the authors view has at least one interesting aspect to consider.

Virtual Cinematography: Beyond Big Studio Production

Balakrishnan and Diefenbach show in their work an idea of virtual cinematography bypassing expensive hardware. The system consists of a tablet device, connected with a Playstation Move controller, that can be tracked and two Playstation controllers for navigation. Using an intuitive user interface the system called "MobileVCS" is capable of free-space movement and controller-based navigation. Being also able to record camera motion and export those data into common 3D software packages is another property. It is also able to perform rudimentary scene editing. Goal of the research is to provide a system at low costs, enabling everybody to visualize and scout their virtual scenes. Bringing together a lot of good ideas the system seems to lack a fully integrated real-time editing and the possibility to work with live motion capture data (cf. Virtual Cinematography).

A Collaborative Real-Time Previsualization Tool for Video Games and Film

Northam, Istead and Kaplan are presenting a previsualization tool, incorporating a lot of interesting features already. A client-server based software is capable of connecting several tools and software packages by providing a plugin architecture. The solution is able to share a virtual scene collaborative by connecting multiple clients. It also is capable of using live motion capture data for virtual characters. The system does not involve real-time editing on set (cf. Real-Time Previsualization).

Dark Matter - A Tale of Virtual Production

This project by Sahin, Spielmann and Backhaus from the year 2014 included already a nearly complete virtual production pipeline. It was capable of live on set previsualization of real and digital environments combined. The tracking relied completely on optical tracking systems and the problems involved like jittering camera movement records. The project was working with editing virtual objects on set, but relied on a classic desktop based setup, by moving the virtual

objects with mouse and keyboard in a 3D application (cf. Dark Matter).

A Wearable Input Device for 3D Interaction

Since the interface of our real-time evaluation is based on gesture recognition it is also very interesting to see that research is done on other devices for reliable gesture control. Ahmad and Musilek are working on the fusion of techniques requiring the user to wear instruments to measure its hands movements and machine vision based techniques by segmenting the hands and determine hand position using a kinematic model. In this work a wrist-worn camera is presented, allowing to determine each fingers position by using skin color segmentation. Furthermore it is possible to generate a 3D model of the hand using the fingers tracking data (cf. UbiHand).

Using Motion Capture for Interactive Motion Editing

A possible solution for using the idea of animation editing from Chapter 3.2.3 could be found in an approach described by Oshita and Muranaka in their paper. They propose an interactive motion editing system. The system can be used to capture motion and edit the captured motion using the same device, going back and forth in the editing process (cf. Interactive Motion Editing).

Superiority of the Mouse over 3D Input Devices in a 3D Placement Task

In this paper, Bérard et al. are describing their research work on the superiority of a classical 2D mouse device against 3D input devices, which is an interesting opposite view to studies, showing that 3D input devices are preferred for 3D environments. They report two studies showing that a standard 2D mouse is outperforming three other 3 DOF input devices in a 3D placement task (cf. Superiority of the Mouse).

Navigation and interaction in virtual worlds

Faisstnauer describes a mapping device in his master thesis, trying to overcome the problem of the variety of different 3d input devices existing today, by creating a mapping device, simplifying the input of 3D editing data in a standardized way (cf. Navigation and Interaction).

Conclusion

All of the presented projects have at least one interesting idea, but none of them presents a intuitive way of real-time editing in virtual production on set. However, interesting ideas and fundamental knowledge for virtual production setups can be found in this papers. This work wants to evaluate what kind of benefits a real-time editing approach could deliver. For this case a prototype interface is built, allowing real-time set editing with gesture control and a simple menu system, allowing every user without any further knowledge to edit objects on set.

5. Prototypic Implementation of Real-Time Set Editing

This chapter describes the prototype system implementation for evaluating real-time set editing in virtual production. Chapter 3.2 described some ideas for real-time editing in a virtual production. Chapter 3.3 showed that 3D user interfaces are best suitable for such tasks and Chapter 3.5 lead to the Unity Engine as a development platform for implementing real-time editing with innovative interfaces. This chapter describes the thoughts in preparation of the development, what considerations have been taken, what limitations are set and what requirements do come up with such an approach. Finally it summarizes the taken approach what leads to Chapter 6 where real-time set editing is qualitatively evaluated through a questionnaire.

5.1. Limitations of this Research and the Prototype System

To give the reader a better idea of the context this research is settled in, some thoughts about the limitations have to be taken. The goal is to evaluate real-time set editing qualitatively. Since there is no out of the box solution that delivers all the features, the set editing software and hardware solution have to be tailored completely. In 16 weeks the best fitting hardware has to be chosen, the software platform as basis needs to be selected, what SDKs can be used to build the prototype on top of it and what features have to be there in the end and what is the best method to tailor all the software and hardware components together in a proper way. After answering all these questions the software itself needs to be developed by a single person and the hardware setup has to be build up and tested nearly without any research budget. Related to this limitations the project is desired to evaluate the benefits of real-time set editing qualitatively only by giving a user group an idea of how set editing could work in virtual production and how this could improve their daily work by enabling them to test the interface and perform basic manipulation tasks. Subsequently they were asked to answer a questionnaire about the interface. Qualitative evaluation as method was also preferred since the amount of work for performing a sustainable quantitative evaluation of the new interfaces was too high. This could be a interesting point to proceed this research, by further improving the new interface with a larger feature set and evaluate this improved version quantitatively.

5.2. Hardware and Software Components of the Real-Time Set Editing System

To develop an interface enabling an operator to perform intuitive real-time set editing in a virtual production, several hardware and software components had to be combined. This chapter describes the software and hardware that is used, manipulated or developed to form the prototype system.

Hardware

As described in Chapter 3.3.4 the hardware should consist of a lightweight output device that enables the user to explore a real scenery, augmented with virtual objects. The hardware should also include an input device that enables object edits in an intuitive way, such as a direct gesture control. The available head-mounted display Oculus Rift DK2 (described in Chapter 3.3.3 and further named "OVR") was chosen as output and as input, the project relies on the Leap Motion Controller (described in Chapter 3.3.2 and further named "LMC") for recognizing the operators hand gestures. The OVR is capable of tracking its own rotation by inertial tracking built into the device, but relies on optical infrared tracking for position tracking. Normally, an external camera for this optical tracking is necessary that is shipped with the OVR and built as a desktop device with a limited field of view. So it only works when the operator sits or stands in front of it, what limits his action radius tremendous. So the already existing optical motion capture system was not only used for capturing an actors movements, but also to track the position and rotation of the OVR in space. So the OVR is in this test scenario only used to display the augmented reality in a portable way, relying on external tracking. It has a 1920x1080 pixel OLED display, delivering 960x1080 pixels per eye and is capable of a nominal FOV of 100° (cf. Oculus Rift DK2). The image is brought to the OVR through an HDMI signal from a workstation computer, already rendered by the Unity Engine in a special way to provide the two distinct views for the left and right eye lenses. Additionally a Microsoft LifeCam Studio, a simple webcam, was mounted on top of the OVR. The webcam was connected to the workstation computer through USB. By bringing the webcam video live feed into the Unity Engine it was possible to enhance the OVR with monoscopic video see-through capability. The LMC was tested at an early stage for tracking the operators hand by attaching it to the front of the OVR, but since it is only able to see the hands from their back in this position the hand tracking is not reliable. Therefore the decision was made to position the LMC about 25cm away of an operators waist, so that he can operate with it as it would be located on a desk in front of

him. For this approach a hardware rig was built, consisting of a L shaped metal holder fixed by a belt. The hand tracking data is delivered through USB to a workstation computer. A Black-magic Design Decklink HD Extreme 3D video card was installed into the workstation, provid-ing SDI video input and output. The workstation is a commercial 64-bit workstation, equipped with a Intel Xeon processor architecture, running at 3.1Ghz with 32GB memory installed. An NVIDIA Geforce GTX 580 was used as a graphics card.

Figure 17, The prototypical real-time set editing interface.

Software

As evaluated in Chapter 3.5 the basis of the software development is the Unity Engine. Most development is done in the scripting language C#. The manufacturer of the OVR as well as of the LMC are providing SDKs with sample code in Unity for how to connect these devices to the engine. This software combines multiple design patterns (cf. Design Patterns) for soft-ware development and object-oriented programming. The core of the system is a Finite State Machine (FSM), implemented as a singleton class. The states of the FSM are representing interaction states of the user. When the software is started, it is running in an idle state, wait-ing for the user to select an object. In this state selection by ray-casting is active. This selection technique is further described in the next chapter. After an object has been selected the selec-tion state is active, it then waits for the user to choose a manipulation task from the visible menu. After that, several states for the different manipulation tasks are available. The FSM always distinguishes if a state is running, a state is entered, or a state is exited and perform-

ing appropriate tasks. While it is moving between two states, the FSM is in transition mode. Additionally, the software is divided into several, independent feature sets, each is important for the whole system. The first important feature set is the ability to connect the engine to a OptiTrack motion capture system and receive the tracked motion capture data through a network interface, writing this data into a suitable self-defined class structure and create graphical representations in the engine for every rigid body or actor the system is currently tracking. This makes it possible to drive digital characters like a robot for instance, in real-time by motion capture data recorded from an artist. The second important feature set is the ability to connect to a nCam camera tracking system over Ethernet and write the necessary information like camera position, rotation or current field of view into a fitting self-defined class structure. This can be used for matching the virtual camera of the Unity Engine and a real camera on set for virtual scene scouting. The third feature is to be able to integrate video of the real world through a plugin that is delivered from third party (cf. AvProLiveCam), either coming from the webcam or a professional digital camera via SDI in combination with a video card. The FOV of the real camera footage and the FOV of the virtual world rendering are correctly matched in the engine. While the OVR is capable of displaying an image with a FOV of 100° and the FOV of the webcam is only 66° horizontal, the representing image of the real world is respectively smaller. The fourth important feature is a 3D user interface that is developed on basis of the freeform menus (cf. Freeform Menu). Based on this menu system, multiple menus are provided to enable a user to select between manipulation options. This 3D user interface is further described in Chapter 5.3. The software examples from the LMC already come with a 3D representation of the users tracked hands, so the hands can be seen as virtual copies inside the Unity engine. Furthermore, a 3D representation of the volume where the LMC is able to detect the hands was integrated to guide new users if they are placing their hands correctly at the moment. Combining all these features provides a way to combine the real and virtual world in real-time, selecting virtual objects, and subsequently manipulate them.

5.3. The 3D User Interface of the Real-Time Set Editing System

The 3D user interface for the real-time set editing was developed based on the considerations in Chapter 3.3.4. This interface consists of three modes: one for selecting objects, a second for the basic manipulation and a third where the manipulation can be done, by selecting several options. The actions made possibly by these basic manipulation tasks are further described

subsequently.

Traveling and Navigation

The evaluation prototype system provides a way for traveling by simply walking around in the physical world (cf. 3D User Interfaces, p. 192). Through the motion capture system used to track the OVR, the translation and rotation of the users motion is directly mapped onto the virtual camera, representing his view into the augmented reality. This is very natural for the user and because of the video see-through capabilities of the system the user is able to see where he is walking towards. This way of traveling has four restrictions at the moment. One is the cable length of all connections between the OVR, the webcam, the LMC and the work-station computer. As a result, in the evaluation setup the walking distance was restricted to a maximum of a circle with a radius of ca. 5 meters. The second restriction is the capture volume of the motion capture system. In the evaluation setup this was a square with ca. 5 meters side length. The third restriction is the monoscopic view of the real world, since only one webcam was used. This could be improved by using two webcams correctly aligned for real stereoscopic sight. Last restriction is the limited field of view of the webcam in relation to the natural field of view of the users eyes.

Selection

For the selection task a simple ray-casting method was chosen. The user points with the pointing finger of his right hand into the virtual space, defining a virtual ray, calculated by the pointing direction of the users virtual hand and the 3D position of the virtual hand. (cf. 3D User Interfaces, pp. 151-152). It was decided to keep the ray visible to guide the user where he is pointing towards at the moment. The ray is represented by a red line, going from the point-ing fingers tip into the pointing direction. The first object that is hit by the ray gets selected, which is visible that the objects are marked through a glowing rim. The glowing gets brighter with the time the user keeps the ray on the object and after an adjustable time (for the proto-type software around 1,5 seconds) the object gets finally selected, which is visible through the rim being completely white. Now the so called editing menu appears and several manipulation options are available. Deselecting an object is done by choosing the appropriate option in the editing menu.

Figure 18, A user is selecting a virtual object.

System Control

Real-time set editing enables for a bright variety of system control commands. The most important were implemented in two menus used. Both menus are 2D menus, placed in the middle of the screen and parented to the cameras movement, so that they always stay at the screens center. The menus consist of a center position with a name, written as text and several menu options, appearing in circular shape around the center position. The menu options are selected by moving the left hand outwards from the menu center in a straight line. There is a clear differentiation between the left hand, for selecting menu options and doing the system control, and the right hand, which is only for selection and manipulation tasks.

Edit Menu

The edit menu is the first menu, visible after the user selected an object. It offers options for translation, rotation, scaling, annotation, animation and exiting. Annotation and animation are there for the purpose of being implemented later maybe, but currently offer no functionality. With exit the user can deselect the object again and can select another one afterwards. Translation, rotation and scaling brings the user to the respective editing action and opens the axis menu.
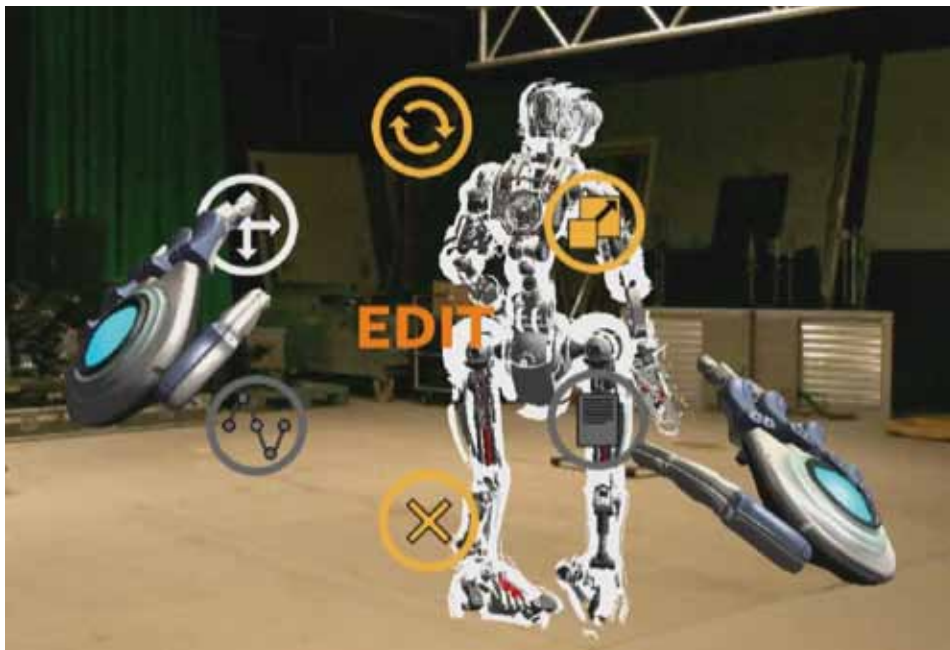
Figure 19, The edit menu from an operators view, while being activated.

Axis Menu

The axis menu can be used parallel to the manipulation process. Being able to see the axis menu also indicates that the user is now able to manipulate the object. The selected object also has 3D widgets, so the user can see how the three main axes X, Y and Z, which represent the three dimensions in space are positioned in space. In our setup the world orientation for the axis is: positive X is right, positive Y is up and positive Z is forward. The axis menu brings options for the X, Y and Z axis. These options work as a toggle, either activating or deactivating the respective axis for editing. The scale option activates or deactivates a multiplication factor to the editing, so bigger distances are possible. The global/local switch facilitates the user to select if he wants to edit the object by its own local axis or the global coordinate axis of the virtual world. One option also enables the user to undo the manipulation and going back to the edit menu. Lastly there is an option that triggers no explicit action and is furthermore only present for being able to close the menu if the user accessed it by accident.

Manipulation

Since the LMC already delivered a good working virtual representation of the users hands, the manipulation was oriented on the simple virtual hand technique (cf. 3D User Interfaces, pp. 159-160). The manipulation can be started by closing the right hand. The left/right movement

of the user is are directly mapped onto the X axis of the object. The X axis represents left/right movement in the Unity engine. The forward/back movement is likewise mapped onto the Z axis of the object. And the up/down movement of the users right hand is also mapped directly onto the Y axis. When moving the object slowly, the mapping is only position control, so stopping the hands movement stops the object. It is also possible to give the object a push with a faster, stronger hand movement, what results in a rate control mapping and the object is moved continuously at that speed in that direction until it stops because the user opens his right hand again.

Symbolic Input

Some of the limitations of this project were that the prototype relied on the above described editing possibilities only and there was no option for symbolic input. As this could be interesting, for doing object annotations or changing a value to a specific number for instance, one can find more information about techniques for symbolic input in (cf. 3D User Interfaces, pp. 294-305).

5.4. Test Scenario for Evaluating the Real-Time Set Editing System

Since the set editing system should be evaluated in a virtual production environment it is necessary to provide an evaluation situation that corresponds to virtual production. For achieving such a goal several existing hardware and software components worked together with the system to provide a virtual production scenario. In preparation of the evaluation a 3D model of a robot was built, the so called "Camdroid". This robot was designated to work as an actor in this virtual production.
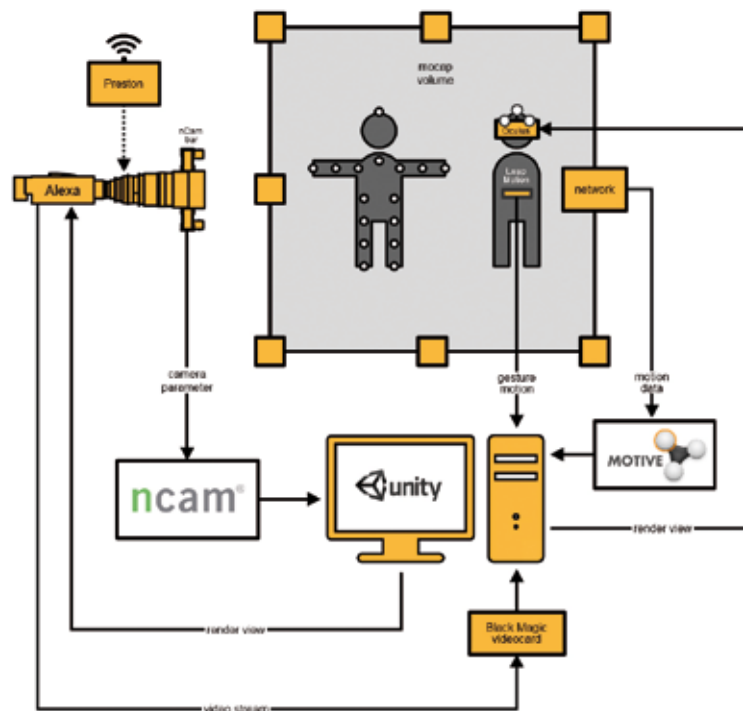
Figure 20, The complete hardware setup of the set editing prototype and the evaluation setup.

Hardware

For the production itself a motion capture system was set up. A OptiTrack system (cf. OptiTrack) was used, with optical tracking and passive markers and consisting of 24 cameras. This system was connected through Ethernet with a workstation computer. The cameras were distributed on different heights on a rig. Motion captured data was collected from an actor in a motion capture suit and/or the OVR. For a digital main camera, an Arri Alexa (cf. Alexa) with an Arri Alura zoom lens (cf. Alura) was used in combination with a nCam camera tracking system for providing the cameras position and orientation in space in real-time. A Preston remote control was used to digitize the values of focus, iris and focal length. These values were streamed in real-time to the Unity engine through a special workstation computer, also called the nCam server. The main camera streamed their image to this server and to the Unity engine as well. The combined image from the engine was delivered to a HDMI to SDI converter, and then to an SDI recorder that recorded the image on a portable SSD hard drive. The recorder also sent the image to a control monitor, so it was possible to either watch the live scene from the workstation or watching the replay of recorded material from the SDI recorder. The second output of the HDMI to SDI converter was brought back to the Arri main camera, so the director of photography was able to watch the combined scenery directly in the viewfinder of

the camera and scout the scenery with his camera like he would in a normal film production. Figure 20 shows the complete evaluation setup as a diagram.

Figure 21, The cinematographer sees a virtual character instead of the actor while scouting the scene.

Software

The nCam server was running a special proprietary software from nCam, which helps to calibrate the system and to align the virtual positioning data of the server along with the rest of the setup. It was also intended that the server records the camera movement during production, allowing the aligment of the camera movement and the motion capture data again in post-production in a 3D software. Due to a bug in the nCam software licensing, the recorded camera data was empty and could not be used to produce the shots as scheduled afterwards. The workstation computer from the motion capture system was running OptiTrack own software called Motive, designated to record the motion capture data and stream them parallel to the Unity engine. On the Unity engine workstation, the real camera movement was combined with the virtual camera movement, the real world video feed was added to the scene, the digital robot was controlled by a motion capture artist and everything was combined and rendered to one composite image.
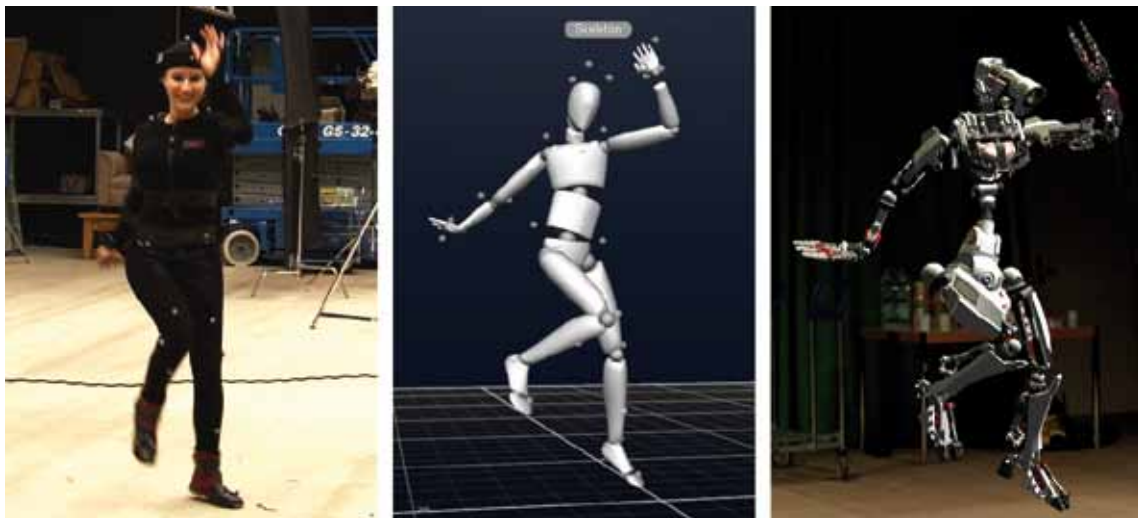
Figure 22, The process to transfer motion capture data from the actor onto a virtual character.

5.5. Additional Requirements From Real-Time Editing

The prototype system described in the previous chapters is mainly intended to evaluate real-time set editing as a possible tool in virtual production itself. But if it is intended to integrate such editing possibilities further into an existing pipeline or into a functioning, well established system several points have to considered. Most of these points are out of the scope of this thesis, the most important that emerged through the evaluation process are described here.

Environment Restrictions

A good reason for the great success of visual effects are the endless possibilities. One is able to create any world he can imagine. An editing interface shouldn't restrict this creative freedom too much. The action radius of the operator was restricted to cable length as described in Chapter 5.3. Also one always has to check that all the cables aren't blocked or squeezed somewhere. A wireless solution would improve the experience. Another improvement would be a solution for the editing environment itself being scalable. If one thinks of a science-fiction movie, in one scene a simple box should be edited for instance, in the next scene a whole planet. Thinking of a market-ready solution wireless systems are preferable.

Data Processing and Data Handling

This topics were introduced already in Chapter 3.1.4 but virtual production only moves steps of the creative process to another point in time. Further thoughts have to be given about how to

integrate real-time editing into an existing workflow. A possible solution could be to establish a file format that is able to track the evolution of a film scene from its creation to the finished production. This would enable a production also to edit things in real-time on set. For example the movement of an object could be established in a digital 3D storyboard, being enhanced in a form of previsualization digitally, then used as an orientation for the camera operator in virtual production and after being accepted from the leading department directly available for post-production purposes afterwards without being touched any further. The envisioned file format has the capability to handle all this kind of stages in production.

Multiple User Input

Since virtual production should enable the collaborative work on a set, it could be possible that several people are editing the same object. A virtual production system with real-time editing features should also be able to grant different people different roles. So it is clearly to see that a specific comment to an object is made by the DOP for instance. It should also be discussed how the system handles an object if is edited at the same time by two persons. What if a person wants to move an object and another one tries to remove the virtual object from the set at that moment, is the action from the person with the higher role only executed? Are both persons informed from the system that their access is concurrent? This has to be stated clear for production.

Data Persistence

In relation to point one and two and Chapter 3.1.4 , it is far more important to keep the data persistent and reliable with real-time editing in a collaborative workspace. Since data are not only displayed, furthermore altered all the time it has to be clear at any time of the process who did which kind of editing also with the option to undo such change. What if the camera operator wants to see the camera movement from 2 days ago, without loading a whole scene with all the objects and everything from the system? All the information has to be put under version control and additionally in some kind of database system which has some naming conventions, so everything is on its place and can be found very fast. One does not simply want to search complex folder structures if he or she decides to import a new virtual object into the scene or load a texture from anywhere on a huge file system when an artist dated it up. These things should be as much automatically as possible.

Network Traffic Structure

By delivering a collaborative workplace for several people, all working on the same data, it is important to first think about suitable networking capabilities before. What information have to be transferred in what kind of way to the individual users and when did they need to be updated so that the existing network capabilities. It needs also to be considered how updates are handled. Does someone who changes anything in the scene inform anyone in the network through a multicast for example? Or is it better to change nothing directly and only send change requests to a server that then informs everybody registered about the change. The second options seems more reliable in terms of synchronization since every participant is getting the information at nearly same time and a delay between the users could only happen because of different processing time inside of the users device.

5.6. Still Existing Problems

Chapter 5.5 showed several topics that should be discussed if real-time editing is to be used fluidly. Chapter 6 shows the way of how evaluation of the real-time set editing interface was performed. Between these two chapters some problems are listed that also came up during the evaluation process but there was no clear solution for them in sight, although one may think of a possible one.

Tactile Feedback

There are several developments attempting to imitate tactile feedback when interacting with virtual objects, but when an actor is trying to interact with a virtual character that is of non-human size for example giving a suitable tactile feedback is still challenging. For instance if an actor has to run into this virtual character, the interaction would be very different if the virtual character would be taller and/or heavier as the person imitating him. For interacting with virtual rigid bodies, like everyday things maybe a pair of data gloves with tactile feedback could improve the experience, but there are open problems in this area as well.

Delay

Synchronization was already described as one of the main topics of virtual production in general in Chapter 3.1.3., this is an increasing problem with real-time editing. Because, as one can think of an solution for synchronization with ideas like an sync signal, provided by an hard-

ware device in combination with system components all of being able to receive and sync their calculations to this signal, the delay of these systems is far more challenging as expected and has to be concerned. When the camera operator is looking through an HMD into the scenery for instance and walking around, the real image and the camera respectively the operators position have to be delivered to a computer, meanwhile the motion data of the actors have to be executed onto the virtual character either they are performed in real-time or loaded from an hard drive, everything has to be rendered in a correct way while recognizing an possible change the operator is performing, and then send back to the displaying device. Fast movements of the operator are increasing the problem. Maybe some kind of movement prediction algorithms to estimate the next steps of the operator could be established to support the reduction of delay times but this has not been evaluated.

Reliability

Since the real and the virtual world are in the need of being combined for real-time editing, the way of how the real world is measured needs to be reliable. Tracking of actors, the camera and everything else in the scene needs to be dependable at any time. The nCam camera system was tested with dense fog and strong glare light. Since it combines optical and inertial tracking it was still working well, but if it is missing feature points to track the cameras position, the inertial tracking can only deliver rotation values and no position changes. The optical tracking system for recording the movement of the real-time editing system and also an actors movement becomes problematic with strong reflective materials. Working with fluids or a lot of metallic materials is not recommended. Also the restricting size of the motion capture volume is not always satisfying. Bigger motion capture volumes are feasible by increasing the amount of used cameras, but scenes with a lot of movement in one direction, or a lot of probes hiding the line of sight could benefit from another solution.

Real-Time Pitfalls

Often the many iterations in traditional visual effects productions are used as an argument for high costs and a low effectivity, but editing in real-time can also have its own pitfalls. A director may want to think over an decision if it is an important scene for the entire product and want to try out several options. With traditional editing he could just hand out the scene to three artists and give them different instructions, leading to three variations. This is also pos-

sible with real-time editing by recording the three variations and playing them back. It is hard to say which method is the better one, this has to be proved by using these new methods in real productions. One should also think of the errors that are often seen first when watching the material multiple times, or by another operator. This means many iterations are not always something bad.

Input Recognition

The LMC is officially capable of detecting several gestures already implemented from the manufacturer, like pointing onto a virtual plane, wiping or posing a circle with the hand. Except the gesture of closing the hand, none of these worked in a reliable way that it was feasible to use them for the real-time editing device. One benefit of gesture recognition is the high amount of DOF a human hand can provide, as described in Chapter 3.3.2., but when the recognition device cannot detect that much DOF, it is not possible to make use of this freedom. It remains to be seen if the LMC improves so more gestures can be detected or if there are inexpensive input devices that can provide better recognition, like data gloves.

Production Costs

Costs for shooting big movies are tremendous. So are the costs if a film production is scheduled to be shot; everything is set up, the crew is on position but everything is on hold because something doesn't work properly or the director wants to try out something different. This has to be taken into account when thinking about real-time editing. A lot of technology comes into play here, so it has to be reliable to keep a high-cost production running. One also has to think about how to integrate real-time editing into an existing film production workflow on set or how to modify this workflow correctly so that it is not necessary for the whole staff to wait until the virtual operator has placed everything on its correct spot. Hardware and software devices for real-time editing themselves also incorporate a lot of additional costs to the production budget.

6. Evaluation

In addition to the topics already described in Chapters 5.5 and 5.6 a sustainable evaluation method has to be used for evaluating the developed real-time editing interface. This was done by giving an audience of people working in the visual effects industry the ability to test the developed interface and asking them to fill in a questionnaire afterwards (cf. 3D User Interfaces, p. 356). The questionnaires results are presented in Chapter 6.1. Chapter 6.2 summarizes all the advantages from the survey including the developers own perception and Chapter 6.3 does this in the same way with all the disadvantages. Afterwards Chapter 7 gives a conclusion about the project.

6.1. A Survey of an Expert User Group

The questionnaire was constructed by the author and Kai Götz together. It was structured to answer questions about the impact of virtual production in general and how the tested real time editing interface was perceived. So it can be used to get a feedback to the new interface, analyzed in this work and other topics discussed in the thesis of Kai Götz (more in Chapter 1.3). The following evaluation is based on questions more related to the interface.

6.2. Results of the Survey

20 participants used the prototypical set editing interface and filled in the questionnaire. 55% of them where under 30 years and 45% over 30 years old. With the exception of one person from the education sector, everybody worked directly in film or in another media related field like interactive media, photography or design. 60% saw themselves as young professionals , while 15% had 5 years or more experience in their industry and 20% had 10 years or more experience. Only 5% reported having no work experience. 60% of the participants worked directly in visual effects.
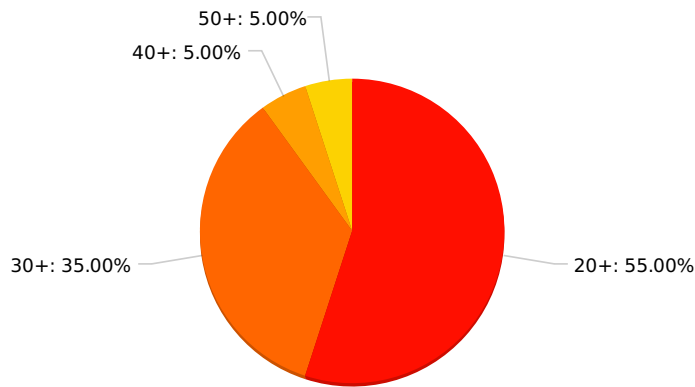
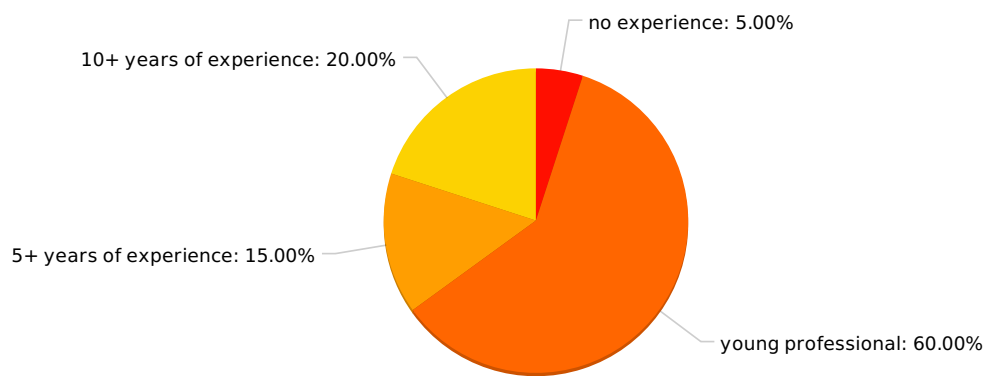**Figure 23,** The age distribution among the test users in years.

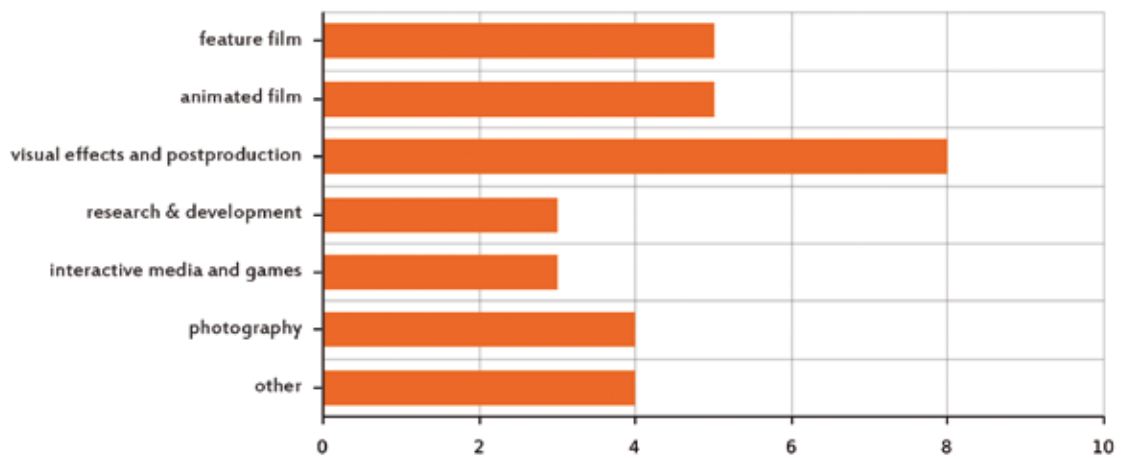**Figure 24,** The level of experience.

Figure 25, The different working areas of the test users.

VFX Workflow Improvement

The first question not regarding the test persons themselves was

"From your point of view, which parts of the current VFX workflow need improvement urgently?".

Multiple participants expressed the wish of a better integration between the different production stages to reduce overhead work. This relates much to the already stated ideas of data processing described in Chapter 5.5.

Others said they wanted improved motion tracking or keying, improved handling of 3D data in combination with 2D live video footage for a better interaction between virtual objects and live video images, compositing that can be done on set, the reuse of assets and ideas, easy ways for integrating metadata information, virtual production at all for reducing time loss because of the direct integration and interactive feedback.

The answers to this question leads to the following conclusions: the current way of working with 3D elements is not satisfying; the industry needs tools that give more immediate feedback of the work; a better integrated pipeline from beginning to the end of production.

Idea of Virtual Production

Another question in the survey stated *"Would you like to share your idea of virtual production?"*.

It is interesting to note that multiple participants mentioned that they see editing on set as part of the virtual production concept. Interesting statements are "saving money through virtual editing possibilities on real set", "...fully produce a scene (inclusive postproduction) live on set.", "real-time production of virtual content", "...experiences involved interactivity...", "walk through a set, being able to see everything at its place and change it in an intuitive and immersive way." This statements support the idea of being able to edit virtual elements in real-time on set and interact with them as intuitively as it would be done with real objects. So it can be said that real-time editing is desired. In contrast, when asked if the users would use virtual production itself, it was also stated that one would not use the real-time editing tools since they are not portable enough, so it is too extensive bringing them to different locations. This is indeed a problem since the tracking is currently provided through an optical motion capture system and it is for most productions too complex and expensive to move the whole system anywhere, in the woods or a desert for example. A production ready system could be improved by changing the tracking system to another more portable solution for instance.
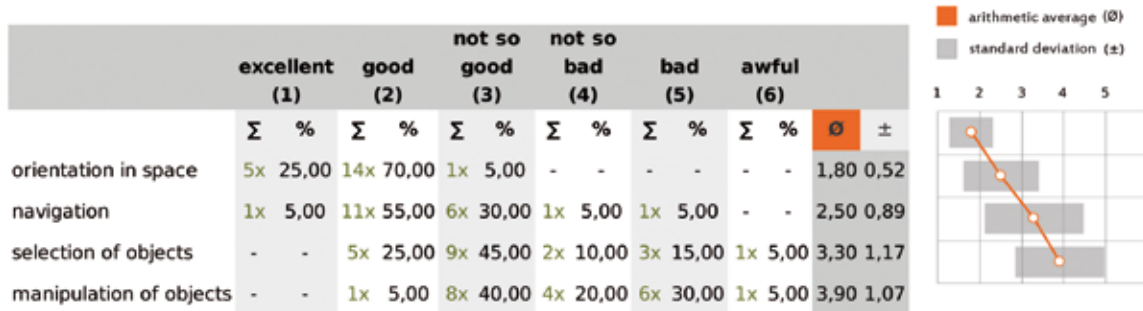
Quality and Intuitivity of Interaction Techniques

	excellent (1)		good (2)		not so good (3)		not so bad (4)		bad (5)		awful (6)		Ø	±
	Σ	%	Σ	%	Σ	%	Σ	%	Σ	%	Σ	%		
orientation in space	5x	25,00	14x	70,00	1x	5,00	-	-	-	-	-	-	1,80	0,52
navigation	1x	5,00	11x	55,00	6x	30,00	1x	5,00	1x	5,00	-	-	2,50	0,89
selection of objects	-	-	5x	25,00	9x	45,00	2x	10,00	3x	15,00	1x	5,00	3,30	1,17
manipulation of objects	-	-	1x	5,00	8x	40,00	4x	20,00	6x	30,00	1x	5,00	3,90	1,07

Figure 26, Results for the quality of the interaction techniques.

Another important question was *"How well did the following interaction technologies work?"*. Here the users could rate the orientation in space, the navigation, the selection of objects and the manipulation of objects separately by choosing a mark between 1 for excellent and 6 for awful. Orientation was getting an average mark of 1,8 which is very good, navigation got 2,5 while selection got the average mark of 3,3 and manipulation got 3,9. Selection and manipulation had also a big standard deviation of 1,17 and 1,07. This outcome fits with the overall impressions from the evaluation. Direct navigation and orientation by turning your head and walking around works very well, people are used to it, and lower marks can be related amongst other things to the limited FOV from the webcam in regards to the normal FOV of human sight and the lower image quality. Selection and manipulation were done in a way that was internally evaluated as most intuitive. Anyway some participants had problems with the usage. Asking how people would describe the intuitivity of the components, the subjects rated the gesture control with a 2,9 in average and the graphical user interface with an average 1,85. Coming to the conclusion that the interface on the whole was intuitive in its usage, but hard to use due to technical problems, relying to the often wrong hand tracking of the LMC, leading to jumping pointers in selection modus or unintentionally picked interface menu options.
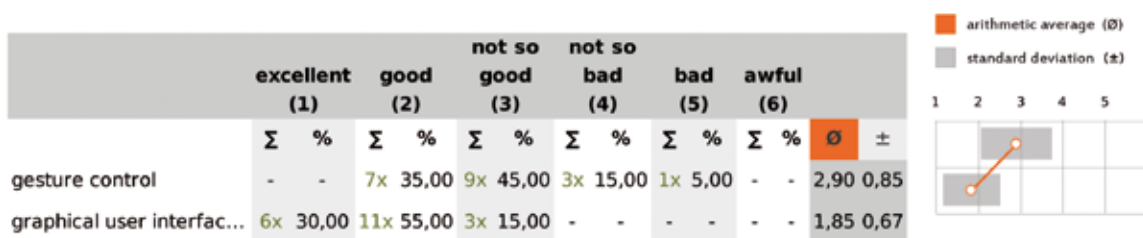
	excellent (1)		good (2)		not so good (3)		not so bad (4)		bad (5)		awful (6)		Ø	±
	Σ	%	Σ	%	Σ	%	Σ	%	Σ	%	Σ	%		
gesture control	-	-	7x	35,00	9x	45,00	3x	15,00	1x	5,00	-	-	2,90	0,85
graphical user interfac...	6x	30,00	11x	55,00	3x	15,00	-	-	-	-	-	-	1,85	0,67

Figure 27, Results for the intuitivity of the interaction techniques.

Improvement of Interaction Techniques

After asking about the quality, the next question was *"How could our interface be improved?"*. As stated before, several participants proposed to improve the gesture recognition for better interaction, for example using data gloves for a more reliable detection. The missing tactile feedback for instance when making a selection in the menus was referred as well. Either people directly mentioned that they are missing such feature or they stated that it is hard to do a selection when no feedback is given. Also brought up multiple times was a way of introduction into the systems. This was overseen in the developing process and is a very good point. Besides all the thoughts about intuitivity a tutorial always helps a lot, explaining the first steps of the device and how it should be used. After explaining the first steps on a computer for multiple participants during the evaluation, it could be seen that they had less problems with the usage at all. Also mentioned was "mapping for manipulations should be overwritten". This is very interesting, since it could be seen during evaluation that nearly every test person trying to rotate an object, rotated its own hand, thinking of the idea, the rotation of the object is mapped directly to the rotation of the hand. In reality rotation was also mapped by the movement of the users hand. This was due to the fact that rotation detection worked not very good during development process with the LMC and it was not clear to see which rotation axis from the users hand should be mapped onto which rotation axis of the object, since they must not be oriented in the same way. Holding your hand in front of your own body, makes it hard to rotate the hand to a greater extent in any other direction as by rotating your forearm, because the hand (having no ball joints) naturally restricts how it can be rotated. Also editing through a tracked hand-held controller was suggested what relates to tactile feedback again in the authors point of view, since holding a controller to perform the task relates a lot to have a tactile feedback where one is moving something towards.

Possible Additional Features

The last question of the survey was *"Which additional features should a market-ready solution include?"* Nearly all answers to this questions relate to interesting features for future versions. It was mentioned that a hardware solution combining the head-mounted display and the gesture control together more fluently in a device would be preferred. Also suggested was voice control, which is a very interesting point and should be considered, because of the research improvements in natural language processing through the past years. Reasonable additions that have

been recommended are more options like being able to attach objects directly to the ground, visibility options, further status information or buttons for special system control commands combined with the gesture control, collaborative working by several people, recording or saving of changes with a complete undo/redo history or technical improvements in resolution or lower delay as already discussed in Chapter 5.6. Most of the input is understandable and can be related to the reduced feature set of a prototypical implementation. Also light editing was mentioned as good addition, which is already described as an idea in Chapter 3.2.2.

6.3. Advantages of the New Approaches

To judge the given feedback through the questionnaire and the own perception of the implementation after all, the idea of editing in real-time on set is a good extension to common technologies. People don't want to learn very complex 3D software applications to perform basic tasks. However, these applications have their qualification for special tasks, like creating digital characters or whole worlds. Nevertheless some functionality should be available intuitively to everyone in real-time. The measured problems with the editing interfaces should be removable by using improved hardware. Also the often mentioned better integration of data into one pipeline from beginning to the end of production would improve the workflow in virtual production and save time and money. The artistic benefits are seen by multiple people. By building a virtual production system that integrates the complete set of suggested features and bringing improved hardware for reduced lag and wireless, lightweight handling could become widely accepted.

6.4. Problems of the New Approaches

In contrast the approaches do have their limitations and barriers. New ways of editing and displaying like gesture control or HMD are not accepted by every user. Different departments or artists have different demands on what they want to have or need in such a system, so it would be tricky to satisfy all of them. The individual hardware pieces that could provide most of the needed technical features are available at high costs. It remains to be seen whether a complete system is possible in the future with affordable prices for a lot of production companies or if such technologies will be to the advantage of large-scale productions only. Also to mention the problems in Chapter 5.6, that have to be solved.

7. Comprehensive Conclusion

In this project a glance was thrown on the problems in current time-based media productions, especially in the field of visual effects. It was identified that the usage of an high amount of virtual elements leads to the necessity of an intuitive way to edit these virtual elements directly on set. For that purpose necessary virtual production technologies as well as current input and output devices where analyzed.

After selecting the current hardware and software that seem to be the best choice for implementing prototypic real-time set editing, such interface was developed. Based on the Oculus Rift DK2, a Microsoft Webcam and the Leap Motion Controller for gesture recognition a set editing prototype was developed in the Unity Engine.

To evaluate the new interface a complete virtual production scenario was established. This virtual production scenario consisted of a robot as virtual character, which could be driven live from a motion capture artist. Making use of this scenario a evaluation was performed, bringing together the real-time set editing prototype in combination with current virtual production hardware. A nCam camera tracking system was used to track a principal film camera, using the rendering of the set editing prototype to enable the camera operator scouting the virtual environment combined with the real world through the viewfinder of his camera. Live motion capture data were used to drive the virtual character and to record the position and rotation of the set editing prototype. The whole setup was also subsequently evaluated by an invited user group, who tested the new set editing interface and filled in a survey afterwards.

The evaluation of the work and the questionnaire lead to several conclusions. People working in the industry want to have the current workflow in producing media content, especially film or time-based media, changed, referring to the questionnaire answers in Chapter 6.1. Real-time editing is also perceived as a great plus for virtual production. Also light editing in addition to match real and virtual light sources or animation editing to fit animation curves directly to ones needed on a set could extend the possibilities in a great way, but could not be further investigated in the prototype.

Otherwise, real-time editing changes currently existing workflows, so a lot of topics have to be taken into account and questions to be answered when this approach should be used in a professional pipeline. Also, the available, used hardware still has its drawbacks. Through the limited FOV of the webcam in combination with the high FOV of the OVR, people had a

limited sight into the real world. Perhaps another device would fit better here. Recently released Meta Glasses (cf. Meta) would be an idea, since they provide optical see-through capabilities meaning that virtual objects can simply augmented. Also for gesture control, either the LMC can be improved further in a prospective version or another gesture input device like a data glove combined with 6 DOF tracking for instance is presented, that combines tactile feedback and better tracking capabilities at low costs. A lot of new devices like the Microsoft HoloLens (cf. HoloLens) and others from the field of virtual environments and augmented reality are developed as well at the moment, so it will be interesting to see what possibilities will arise in the near future. Besides the technical options, it remains in question whether the different approach of real-time editing could become widely accepted. This would also be linked with further software developments, for more robust, intuitive software products with a lot of additional feature sets combined in on solution, easy to understand and to manage. However, the Dreamspace research project this work is written in line with is still ongoing for the next one and a half years and it is possible that further prototypes are implemented, using new hardware devices and improving the user interface. So it keeps exciting where the current innovations are leading towards.

Appendix

A. Glossary

[2D]

two dimensional.

[3D]

three dimensional.

[Accelerometer]

An electromechanical device that measures acceleration forces.

Retrieved from: http://www.dimensionengineering.com/info/accelerometers.

[Algorithm]

A defined sequence of orders a computer can execute.

[API]

Application Programming Interface. Description for the software interface normally provided from a hardware or software manufacturer to allow other software developers writing external functionalities for their products.

[Blue Hour]

A short time during twilight where the sky has a deep blue hue with a cold color temperature. Very popular with photographers and video artists.

Retrieved from: http://petapixel.com/2014/06/11/understanding-golden-hour-blue-hour-twilights/

[CCD]

Charge-Coupled Device. A light-sensitive integrated circuit that stores and displays the data for an image in such a way that each pixel in the image is converted into an electrical charge.

Retrieved from: http://searchstorage.techtarget.com/definition/charge-coupled-device

[Computer Vision]

Field in the area of computer science to acquire, process, analye and understand images.

Retrieved from: http://en.wikipedia.org/wiki/Computer_vision

[Data Solving]

The smoothing and removing of errors from data in the context of visual effects.

[Depth Map]

A image representing instead of colors with its pixels the distance of that pixel in space.

[FSM]

Finite State Machine, a device that stores the status of something at a given time and operates

on input to change the status or cause an action.

Retrieved from: http://whatis.techtarget.com/definition/finite-state-machine

[Game Engine]

Software framework for developing computer games.

Retrieved from: http://encyclopedia.thefreedictionary.com/game+engine

[Gyroscope]

A device consisting of a spinning mass, mounted on a gimbal so that its axis can turn freely

in one ore more direections and maintain thereby its orientation. Therefore this device can be

used to measure orientation.

Retrieved from: http://www.thefreedictionary.com/gyroscope

[HFR]

High Frame Rates, definition for film or video material, recorded with frame rates,

higher then the long time industry standards of 24 or 25 frames per seconds.

Retrieved from: http://en.wikipedia.org/wiki/High_frame_rate

[Immersion]

The perception how someone feels physically present in a non-physical world.

Retrieved from: http://en.wikipedia.org/wiki/Immersion_(virtual_reality)

[ISO]

International Organization for Standardization,

a organisation working worldwide on developing and publishing standards.

[LED]

Light emitting diode, a semiconductor emitting light by the movement of electrons.

Retrieved from: http://electronics.howstuffworks.com/led.htm

[Lighting]

The process of setting virtual lights in digital 3D environments.

[Monoscopic]

Using only one eye and therefore do not have a perception of depth.

Retrieved from: https://www.wordnik.com/words/monoscopic

[OpenGL]

Widely-used graphics standard API for rendering 2D and 3D images.

Retrieved from: https://developer.nvidia.com/opengl

[Potentiometer]

A resistor with a sliding or rotating contact that delivers adjustable voltages,

depending on the position of the contact.

Retrieved from: http://en.wikipedia.org/wiki/Potentiometer

[Retroreflective]

Material that reflects the light to its source.

Retrieved from: http://www.roadvista.com/retroreflection/

[Shading]

The process of simulating material characteristics in computer graphics.

[Singleton]

A design pattern where only one instance of a class is allowed to exist at any time.

[Stereoscopic]

Enhancing the illusion of depth in an image.

Retrieved from: http://en.wikipedia.org/wiki/Stereoscopy

[Texturing]

Process of manipulating the visual appearance of computer graphics

by mapping images or painting on their surface.

[Time of Flight]

Technology used to measure the time it takes for an object to travel a certain distance.

Used in conjuncting wih light, the distance of an object can be estimated by the time

it takes for the light to travel from the sender to the object and back.

Retrieved from: http://en.wikipedia.org/wiki/Time_of_flight

B. Bibliography

[VES Handbook]

J. Okun, S. Zwerman, The VES Handbook of Visual Effects, Second Edition, Focal Press, New York 2015

[Design Patterns]

E. Gamma, Design Patterns: elements of reusable object-oriented software, Adison-Wesley, 1998

[3D User Interfaces]

D. Bowman, E. Kruijff, J. LaViola Jr., I. Poupyrev, 3D User Interfaces Theory and Practice, Addison-Wesley, Boston 2005

[Production Pipeline Fundamentals]

R. Dunlop, Production Pipeline Fundamentals for Film and Games, First Edition, Focal Press, New York 2014

[Be-greifbare Interaktionen]

B. Robben, H. Schelhowe, Be-greifbare Interaktionen, transcript Verlag, Bielefeld, 2012

[The New Art of Virtual Moviemaking]

M. Patel, Autodesk Inc., The New Art of Virtual Moviemaking, White Paper 2009

[Location systems for ubiquitous computing]

J. Hightower and G. Borriello, Location systems for ubiquitous computing, Computer Journal, Volume 34, Issue 8, pages 57–66, 2001

[Navigation and Interaction]

C. Faisstnauer, Navigation and Interaction in Virtual Environments, Institute of Computer Graphics, Vienna University of Technology, 1997

[Handrix]

G. El Koura and K. Singh, Handrix: Animating the Human Hand, Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on computer animation, pages 110-119, 2003

[Virtual Cinematography]

G. Balakrishnan, P. Diefenbach, Virtual Cinematography: Beyond Big Studio Production, ACM SIGGRAPH 2013 Studio Talks, 2013

[Real-Time Previsualization]

L. Northam, J. Istead, C. S. Kaplan, A Collaborative Real Time Previsualization Tool for Video Games and Film, ACM SIGGRAPH 2012 Posters, 2012

[Dark Matter]

Y. Sahin, S. Spielmann, M. Backhaus, Dark Matter – A Tale of Virtual Production, ACM SIG-GRAPH 2014 Studio Talks, 2014

[UbiHand]

F. Ahmad, P. Musilek, UbiHand: A Wearable Input Device for 3D Interaction, ACM SIGGRAPH 2006 Research Posters, 2006

[Interactive Motion Editing]

M. Oshita, H. Muranaka, Using motion capture for interactive motion editing, Proceeding of the ACM VRCAI 2014 conference, pages 65-69, 2014

[Superiority of the Mouse]

F. Bérard et al., Did "Minority Report" Get it wrong? Superiority of the Mouse over 3D Input Devices in a 3D Placement Task, INTERACT 2009, Part II, LNCS 5727, pages 400-414, 2009

[See-Through Displays]

J. Rolland, H. Fuchs, Optical Versus Video See-Through Head-Mounted Displays in Medical Visualization, Presence Vol. 9, No. 3, June 2000, pages 287-309, Massachusetts Institute of Technology, 2000

[Methods, Guidelines and Scenarios]

T. Knop, Virtual Production – Methods, Guidelines and Scenarios, Dreamspace Research Project, Document D2.1.1, retrieved from www.dreamspaceproject.eu/Documents on 15.02.16, Stargate Studios Germany 2014

[Evaluation of 12-DOF]

A. Huckauf et al., Evaluation of 12-DOF Input Devices for Navigation and Manipulation in Virtual Environments, Faculty of Media, Bauhaus-University Weimar, Germany, 2005

[Interface Description]

V. Helzle, S. Spielmann, Draft Specification and Interface Description for Virtual Production Editing Tools, Dreamspace Research Project, Document D5.1.1, version from 14.03.22, retrieved from www.dreamspaceproject.eu/Documents on 15.02.16, Filmakademie Baden-Württemberg, 2014

[3D Pose Estimation and Mapping]

S. May, D. Droeschel, D. Holz, C. Wiesen, S. Fuchs, 3D Pose Estimation and Mapping with Time-of-Flight Cameras, Fraunhofer Institute for Intelligent Analysis and Information Systems, German Aerospace Center – Institute of Robotics and Mechatronics, Sankt Augustin and Wessling, 2008, publically retrieved through www.researchgate.net

[OpenGL Performer]

G. Eckel, K. Jones, T. Domeier, OpenGL Performer Getting Started Guide Version 3.2, Silicon Graphics Inc., 2004, publically retrieved from http://techpubs.sgi.com/library/manuals/3000/007-3560-005/pdf/007-3560-005.pdf

[The Value of Visual Effects]

S. Squires, "http://effectscorner.blogspot.de/2012/07/value-of-visual-effects.html", retrieved on 14.12.02

[Overview of Tracking Technologies]

Boger Yuval, "http://www.roadtovr.com/overview-of-positional-tracking-technologies-virtual-reality", retrieved on 14.12.16

[Scenegraphs]

A. Bar-Zeev, "http://www.realityprime.com/blog/2007/06/scenegraphs-past-present-and-future/", retrieved on 15.01.27

[Project Tango]

J. Lee, "https://www.google.com/atap/projecttango/", retrieved on 15.01.28

[Leap Motion]

Leap Motion Inc., "https://www.leapmotion.com/product", retrieved on 15.01.28

[Leap Motion System]

Leap Motion Inc., "http://blog.leapmotion.com/hardware-to-software-how-does-the-leap-motion-controller-work/" retrieved on 15.01.29

[SpaceMouse Pro]

3Dconnexcion Inc., "http://www.3dconnexion.de/products/spacemousepro.html", retrieved on 15.01.29

[SpaceBall 5000]

evermotion.org, "http://www.evermotion.org/tutorials/show/7916/3dconnexion-s-spaceball-5000-review", retrieved on 15.02.24

[Cardboard]

Google Inc., "https://www.google.com/get/cardboard/", retrieved on 15.01.29

[Gear VR]

Samsung Inc., "http://www.samsung.com/global/microsite/gearvr/gearvr_features.html", retrieved on 15.01.29

[OptiTrack]

OptiTrack Inc., "http://www.optitrack.com/", retrieved on 15.02.23

[Oculus Rift DK2]

Oculus VR Inc., "https://www.oculus.com/dk2/", retrieved on 15.01.29

[Alexa]

ARRI Inc., "http://www.arri.com/camera/alexa/", retrieved on 15.02.23

[Alura]

ARRI Inc., "http://www.arri.de/camera/cine_lenses/zoom_lenses/alura_studio_zooms/", retrieved on 15.02.23

[Wide5]

United States of America Government, "https://www.fbo.gov/index?s=opportunity &mode=form&tab=core&id=6160dc7a41ec5458d1e914ffb508d5ab", retrieved on 15.01.29

[OptiTrack VCS]

OptiTrack Inc., "https://www.naturalpoint.com/optitrack/products/insight-vcs/specs.html", retrieved on 15.01.30

[Open Inventor]

Silicon Graphics Inc., "http://oss.sgi.com/projects/inventor/", retrieved on 15.02.23

[OpenGL]

Khronos Group, "https://www.opengl.org/", retrieved on 15.02.23

[X3D]

Web3D Consortium, "http://www.web3d.org/x3d/what-x3d/", retrieved on 15.02.23

[DSG]

OpenSimulator.org, "http://opensimulator.org/wiki/Distributed_Scene_Graph", retrieved on 15.02.23

[DMX512]

USITT, United States Institute for Theatre Technology, "http://www.usitt.org/content. asp?contentid=373", retrieved on 15.02.23

[CG Formats]

Digital-Tutors, "http://blog.digitaltutors.com/cg-file-formats-you-need-to-know-understanding-obj-fbx-alembic-and-more/", retrieved on 15.02.23

[Pixar USD]

Pixar, "http://graphics.pixar.com/usd/overview.html", retrieved on 15.02.23

[Vicon VC]

Vicon Inc., "http://www.vicon.com/System/VirtualCamera", retrieved on 15.01.30

[nCam System]

Ncam Inc., "http://www.ncam-tech.com/info/advantages", retrieved on 15.01.30

[Magic Leap]

MIT Technology Review "http://www.technologyreview.com/featuredstory/534971/magic-leap/", retrieved on 15.02.23

[Simulcam]

A. Billington, "http://www.firstshowing.net/2008/a-brief-look-at-the-technological-advancements-in-james-camerons-avatar/", retrieved on 15.01.30

[VCam]

J. Dachman, "http://sportsvideo.org/main/blog/2010/07/newtek-integrtes-lightwave-10-with-intersense-vcam-to-create-complete-3d-solution/", retrieved on 15.01.30

[AVProLiveCam]

Renderheads Inc., "http://www.renderheads.com/portfolio/UnityAVProLiveCamera/", retrieved on 15.02.03

[Freeform Menu]

Leap Motion Inc., "http://u3d.as/content/leap-motion/leap-motion-freeform-menus/6TU", retrieved on 15.02.03

[Meta]

Meta Inc., "https://www.spaceglasses.com/products", retrieved on 15.02.11

[HoloLens]

Microsoft Inc., "http://www.microsoft.com/microsoft-hololens/en-us/get-ready", retrieved on 15.02.11

[Dreamspace Project]

Dreamspace Project, "http://www.dreamspaceproject.eu/", retrieved on 15.02.16

C. List of Figures

[Figure 1]

Götz K.: Handmade collage of pictures from the shooting for evaluation

purposes, with permission by the author, Ludwigsburg, 2015.

[Figure 2]

Image showing a typical vfx production where actor interact with a placeholder, retrieved from:

http://www.fxguide.com/featured/effects-you-never-knew-were-there/ on 2015.02.18

[Figure 3]

Image showing the Razer Hydra, a input controller using magnetic tracking, retrieved from:

http://www.maximumpc.com/article/reviews/razer_hydra_review on 2015.02.18

[Figure 4]

Image showing the Inertia Cube 4, a device for inertial tracking, retrieved from:

http://research.vuse.vanderbilt.edu/rasl/facilities/ on 2015.02.18

[Figure 5]

Image showing a screenshot from the game Star Citizen, using the CryEngine, retrieved from:

http://www.cryengine.com/showcases/star-citizen on 2015.02.18

[Figure 6]

Image showing a motion capture studio with two actors, retrieved from:

http://blogs.chapman.edu/dodge/2014/01/17/interterm-robot-attack-student-transformers-in-the-folino/ on 2015.02.19

[Figure 7]

Image showing motion capture performance, Filmakademie Baden-Württemberg, 2009

[Figure 8]

Image showing the cubic mouse, retrieved from: http://www.chip.de/bildergalerie/Die-abge-fahrensten-Eingabegeraete-Galerie_29338681.html?show=1 on 2015.02.18

[Figure 9]

Image showing the peregrine glove, a touch based glove, retrieved from:

http://www.mellottsvrpage.com/index.php/peregrine-glove/ on 2015.02.19

[Figure 10]

Image showing Leap Motion, a gesture recognition device, retrieved from:

http://www.robotshop.com/en/leap-motion-3d-motion-controller.html on 2015.02.19

[Figure 23]

image showing the age distribution from the survey, work is automatically exported

from the online questionnaire, created at http://www.umfrageonline.de/.

 [Figure 24]

image showing the experience distribution from the survey, work is automatically exported

from the online questionnaire, created at http://www.umfrageonline.de/.

[Figure 25]

image showing the area of work distribution from the survey, work is automatically exported

from the online questionnaire, created at http://www.umfrageonline.de/.

[Figure 26]

image showing the quality of interaction, work is automatically exported

from the online questionnaire, created at http://www.umfrageonline.de/.

[Figure 27]

image showing the intuitivity of interaction, work is automatically exported

from the online questionnaire, created at http://www.umfrageonline.de/.