VR viewer

Tobias Eilertsen

Master's thesis in Software Engineering at

Department of Computer science, Electrical engineering and Mathematical sciences,
Western Norway University of Applied Sciences

Department of Informatics,
University of Bergen

May 2022

Western Norway
University of
Applied Sciences

Abstract

During orthopedic surgery planning, the surgeons use CT images and 3D printing to analyze the fracture and help prepare for the surgery. Visualising three-dimensional data as 2D images limits how the data is comprehended. 3D printing has drawbacks related to the printing process, such as printing time and the need for support structures.

With Virtual Reality technology, the data can be visualized in 3D using a real-life scale to perceive depth and distances better. The improved visualisation can improve the preparation phase by increasing the anatomical understanding or removing the need for 3d printing. A VR application for viewing CT scans was created with assistance from surgeons at Haukland University Hospital.

The VR application is tested in order to evaluate how it affects the surgery planning. Medical professionals have tried the VR applications and participated in semi-structured interviews.

Contents

List of Figures

Chapter 1

Introduction

This thesis is written as part of the University of Bergen and Western University of Applied Sciences master's program in software engineering. The project is a collaboration between Helse Vest IKT, HVL department of computer science and orthopedic surgeons at Haukeland University Hospital.

1.1 Context and Approach

A Computed Tomography (CT) scan is often performed when a hospital receives an injured patient considered for orthopedic surgery. The CT scan is displayed as 2D images or as a 3D model. The model allows the surgeons to better plan the surgery by understanding the anatomy of the fracture. If a surgeon has a good understanding of the anatomy and better plans the surgery, the surgery has less risk of complications and could give a better result.

1.2 Problem Description

Visualizing the model in 2D limits the understanding of the fracture because of the lack of scale and depth. The problem applies to both 2D images and 3D models rendered in 3D. A considered solution is visualizing the model in Augmented Reality (AR) or Virtual Reality (VR) to give medical personnel a good understanding of the problem area. VR has multiple potential benefits, and the surgery planning process can benefit from this. As the users are already looking at 3D models on 2D screens, visualizing in VR could improve the surgeon's fracture understanding and by this ensure patient safety. The entire planning process could also be more effective by removing or reducing the need for 3D printed models.

1.2.1 Research question

The problem description leads us to the following research question and sub-questions.

How can VR technology improve orthopedic surgery planning?

- Can VR give a better preparation for surgery?
- Can VR replace 3D printing while using fewer resources or less time?

1.3 Methodology

The project includes a minimum viable product VR application of a standard viewer that is user-friendly enough to test with non-technical persons and with functionality that covers the use cases of a printed model. The application is tested on medical personnel to evaluate any benefits on anatomical understanding and how it affects surgery planning and effectiveness. The application will be available for further development and study.

Firstly, a application VR viewer will be created with the help of VR frameworks and guidance from both orthopedic surgeons and developers with experience in medical technology.

In order to answer the research question, we have evaluate the usability of the final application. This thesis has used qualitative methods by interviewing related personnel to evaluate the planning improvements, including anatomical understanding, cooperation, and the effectiveness of the planning process. The functionality will also be compared to the existing solutions, possibly by making a direct comparison by using a 3D model, printed model, and VR viewer on the same case.

1.4 Related works

Several other variants of VR viewers already exist. Typical functionality for existing solutions includes viewing DICOM files as 3D models and a basic VR interface for inspecting the model and viewing DICOM slice images. Some applications are part of more extensive enterprise solutions including other medical imaging tools, management systems, and more. No solutions found are open-source or free. The existing solutions differ in visualizing the model as a solid mesh or as a volume. ImmersiveView VR [24] and MedicalImagingVR [29] renders the scan as a transparent volume. Most solutions also include a standard 2D plane rendering for viewing slices. Some solutions like medical holodeck [23] $MedicalimagingXR$ offer a detailed visualization including anatomical layers like muscle and bone. However, the data is created manually with a custom dataset and can not be used for viewing a specific patient. Some solutions are also created specifically for other areas than orthopedics, like Sentiar CommandEP [10] for heart surgery. Early AR solutions exist intended for use during surgery, including surgical theater [50] intended for neurosurgery [3]. Other similar solutions are *The body VR: anatomy Viewer* [48], DICOM VR [12], Ceevra [6] and Dicom Director Intravision XR [13].

The application developed in this project differs itself by vizualising the model in separate pieces and allowing users to adjust the model. In addition, no other open-source solutions are found specifically for viewing DICOM data in VR, and few free VR viewers are found.

A 2019 study in visualising Patient data with VR [49] implemented a VR viewer

for DICOM data and tried to measure anatomical understanding compared to 2D images. The study did not investigate the efficiency of the planning phase, and it did not consider 3D printing.

1.5 Contribution

1.6 Outline

Chapter 2

Background

This chapter will present some of the knowledge that our research is built upon.

2.1 Orthopedic surgery

Orthopedic surgery is surgery involving the musculoskeletal system. Cases range from trauma surgery, where high impact forces cause injuries to infections and tumors [43].

Orthopedic surgeons do both elective (planned) and emergency surgery. Surgery includes prosthesis surgery, tumors and infections. In elective surgery, the surgeons will have days or weeks to plan out the surgery. A team of usually two surgeons will plan the surgery together. The surgeons will diagnose the patient from the following features: history, clinical examination, medical imaging, and any special investigations. History includes the patient's complaint and any previous injuries. Clinical investigation means examining the sources of the symptoms and the body as a whole. Medical imaging, including X-ray, ultrasound, CT, and Magnetic Resonance Imaging (MRI), gives the surgeons a detailed insight into bones or soft tissue structures [43]. Medical imaging is used to locate the fracture or the number of fractures and inform the surgeon on the anatomy, such as fracture line and fracture type. The data can also show bone condition if any joints are involved and swelling of soft tissue. All this information helps confirm the diagnosis, study the fracture and plan the treatment [15]. The different imaging types have different uses and advantages. X-ray is always the first choice as it simpler and cheaper [15]. A CT scan is used to get better information in complicated fractures. MRI can additionally detect soft tissue and ligament injuries.

When deemed necessary by the surgeon, a 3D printed model of the fracture is created. The 3D print requires a previous three-dimensional scan such as a CT. The model can improve the surgeon's understanding of the fracture, and allow the surgeon to adjust and practice with implants. The medical imaging and 3D printing is done by radiologists at the hospital.

This project was based on elective surgery planning in a complicated fracture.

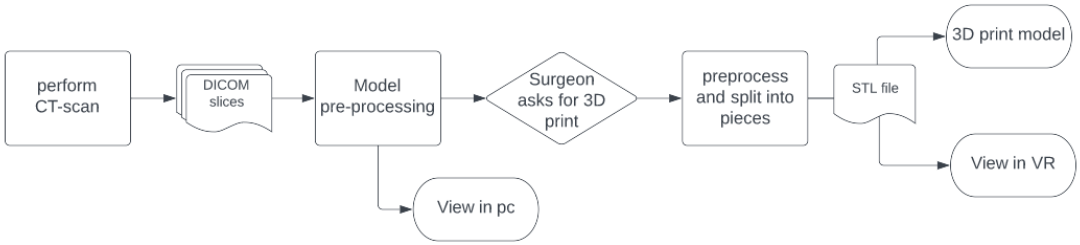


Figure 2.1: Workflow of how medical imaging is used

2.1.1 Orthopedic methods

Fracture treatment in orthopedics can be divided into several methods outlined in Textbook of Orthopedics [15]. Nonoperative methods include straps, slings or casts. Nonoperative methods have no infection risk or surgical risk. When surgery is decided, the fracture needs to be fixed internally by implants. Internal Fixation means to internally set and stabilize bones [15]. Implants include screws, nails, and plates made from steel, titanium, or plastics. Operative methods are used for anatomic reduction. Treatment of fractures by external fixation is also possible. External fixation means a metallic frame is connected to the bone with pins.

2.1.2 Brief history of medical imaging

The first used imaging tool was the X-Ray, discovered in 1895 [22] [42]. As the energy in the radiation is absorbed at a different rate by tissue and bone mass, it is possible to create an image of the bone. The image is displayed as a projection from the X-ray angle. During the first half of the 20th century, additional techniques with several X-Rays allowed to isolate a slice of bone without over- and underlying tissue. Often X-ray from several angles is required to view the fracture without being obscured by other bone masses [15].

A big leap in medical imaging was the CT scan (computed tomography) [5]. CT scans (Computed Tomography) were invented during the 1970s. During a CT scan, an X-ray tube is rotated around the tissue, scanning from all angles while detecting the absorption/reflection of the tissue. CT scans overcome the two-dimensional X-ray limits and create detailed image data that can be visualized in any plane without superimposing the image with tissue above and below the selected layer [22]. The detail of a CT scan depends on the hardware used, as well as the trade-off where higher resolution gives the patient higher radiation [5]. The higher resolution is in complex cases preferred over X-ray to evaluate intra-articular fractures (crossing a joint) or locate smaller fractures [15]. CT scans can eliminate the need for repeated imaging in the case of a trauma patient [43]. MRI (Magnetic Resonance Imaging) was also developed during the 70s. It uses a strong magnetic field and radio signal frequencies to scan. MRI has comparable accuracy to CT but can also view soft tissue. MRI also avoids radiation [43].

The imaging used for testing in this report is from CT scans done by Radiological department of Haukeland University Hospital.

2.1.3 Visualisation of medical imaging

The output of both CT and MRI scans is a three-dimensional scan. The output is typically represented as slices, where a slice is a 2D picture repeating along an angle. The pixel at coordinate *(x, y)* at slice number z represent the absorption at the point *(x, y, z)* [8]. Each pixel in a slice represents a voxel in a volume, and all the slices combined make up volumetric data or a three-dimensional point cloud [8]. The slice thickness of a slice can be below 1 mm, giving a high-resolution scan where a single point is less than one cubic millimeter [22]. Figure 2.2 shows the workflow of converting from slices to voxels and further to a mesh model.

When rendering a 2D image, the volume must be projected to two-dimensional space. The values in a single slice can be used to view the intersection at a specific slice number. A line of values along all the slices can be used to view the intersection at another plane. All slices can be combined to show either the average or the maximum intensity [19]. The values are converted to greyscale by mapping ranges of values to greyscale pixels. This mapping function determines the brightness and contrast of the final image.

A specific algorithm is needed to render the scan as a 3D surface mesh. Rendering the model includes preprocessing the volume and classification to determine the type of tissue based on voxel value. A simple approach is thresholding, a binary classification where a polygon is created on any volume point that matches the threshold value. A different threshold can be selected to visualize tissue with different densities, typically bone [19].

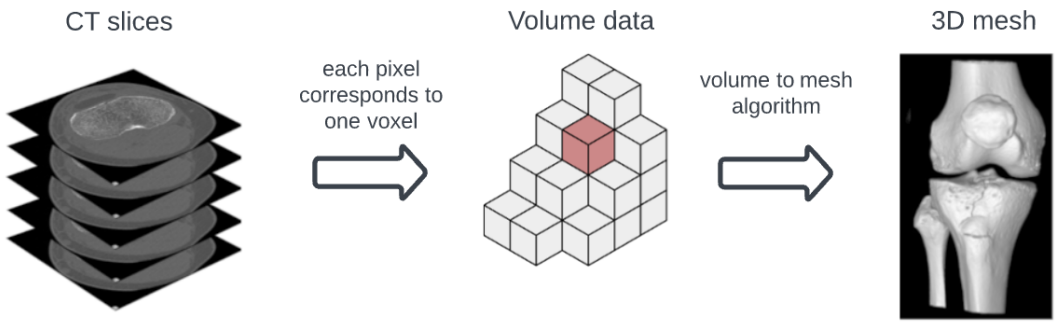


Figure 2.2: DICOM data conversion to a polygon mesh

Data formats

The output of most medical imaging is standardized with the Digital Imaging and Communications in Medicine (DICOM) format [11]. The DICOM format includes metadata and pixel data. CT scans are saved as several files, where each file represents one slice in the scan.

If the data is represented as a 3D mesh, several formats can be used. The Wavefront Object file format (OBJ) [**obj**] is a standard format for objects defined by polygons and surfaces. 3D mesh formats such as OBJ differs from DICOM by not storing volume data. Any density data is deleted when converting from DICOM to OBJ. Another mesh format is the Stereo lithography file (STL) [**stl**]. The STL standard is created specifically for 3D printing. STL includes polygon info similar to OBJ files.

3D printing

An alternative to digital representation is to print the model with a 3D printer, illustrated in figure 2.3. A 3D model can help increase accuracy when performing a procedure [41]. 3D printing has many advantages, such as the surgeon physically holding the model, measuring the model, trying out equipment, and practicing with the model. The most significant disadvantage to 3D printing is that the printing process can take more than 24 hours, depending on model size, materials, and printer. The waiting time is, in some cases, too long. Another drawback is not having any digital tools such as transparency control, displaying cross-sections, or being able to alter the model after it is printed. A physical plastic model also needs support structures that can lead to an inaccurate representation of the fracture or get in the way of viewing the model. The support structure issue is especially obvious in a fracture with many small bone fragments. A similar problem is not being able to print any fractures on the inside of volumes.

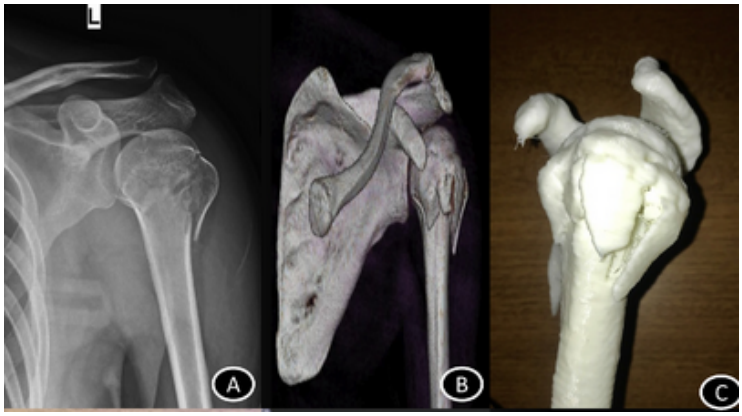


Figure 2.3: Humeral (upper arm) fracture
A fracture from initial scan to finished print. (A) preoperative X-ray; (B) 3D model; (C) 3D printed model. Image from study on virtual preoperative planning [34]

The cost of a 3D printer and material depends on printer type and size requirements but is typically low. Extrusion-based plastic printers are commercially available at as low as 200 USD/ 1800 NOK, with very cheap materials. The cost of 3D printing in industry or medicine comes from personnel, training, and time usage [41].

2.2 Extended Reality

Extended Reality (XR) "is devices capable of overlaying digital information onto the physical world or incorporating aspects of the physical world into virtual scenes" [2]. XR is typically split into Virtual Reality (VR) and Augmented Reality (AR). Virtual Reality is the use of VR technology to sense the user's state and actions and augment sensory feedback to immerse the user in a 3D virtual environment [32].

The virtual reality system works by tricking the senses by displaying computer-generated stimuli that replace stimuli from the real world. The user typically perceives the virtual environment through a Head-mounted Display (HMD), sound, and haptic feedback (vibration). The virtual environment is the computer-generated objects that the user interacts with. The virtual environment will often mimic properties in the real world, such as shape, color, or functionality.

With more specialized hardware, additional stimuli such as temperature, smell, and more are possible [17]. To make the virtual environment seem real, it must respond to the user's actions. Current commercial Virtual Reality headsets track the user's head and hands and allow for button inputs [35]. Modern HMDs use 6 Degrees of Freedom (DOF), which means the user is tracked in three-dimensional position and rotation [28]. The tracking is used by the VR application to simulate walking, picking up objects, and more.

2.2.1 Augmented and Mixed Reality

Augmented Reality and Mixed Reality (MR) is an addition to VR where the real world is viewed together with the digital environment [21]. Several definitions exists, according to Intel [**vrandmr**] AR is a digital overlay onto the real world while MR means interacting with both digital and real elements. For simplicity this thesis will use the term Augmented Reality to describe devices showing both a virtual and a real environment.

AR comes in different variants; some mobile apps use the camera to create an augmented environment shown in 2D. More expensive AR HMDs work similarly to VR HMDs, except for the transparent viewing glass. Mixed Reality devices primarily exists as expensive HMDs, such as the Hololens [**hololens**]. AR HMDs are used in medical, industrial, and military devices to show critical information while users operate devices or perform a job in real life. Examples of this are a surgeon viewing medical information during a surgery or a pilot viewing a Heads Up Display while flying [32] [31]. AR has some uses not relevant to VR because it does not completely disconnect the user from the real world, but has some disadvantages such as reduced field of view compared to VR, poor visibility in bright light [21] and drastically higher cost [23].

2.2.2 Professional usage of XR

While becoming more popular in mainstream entertainment, XR has been used in professional environments for many years. An advantage of XR compared to mouse/keyboard devices is touch-free interfaces with hand tracking, making it better for a sterile environment [2].

VR has been developed by the US Airforce since the 1980s for pilot interfaces [31]. VR and AR are often used in the medical field for training or education because the actual situation would be unpractical or dangerous [20]. VR is used in cardiology and neurology for monitoring and is emerging in other medical fields such as rehabilitation and training [25]. VR is used in cardiology and neurology for monitoring and is emerging in other medical fields such as rehabilitation and training [25].

2.2.3 Advantages and disadvantages

A disadvantage with wearing an HMD is fatigue, both physical fatigue caused by the weight or eye fatigue and motion sickness [30]. Image imperfections cause eye fatigue in the HMD [27], and motion sickness is caused by sensory conflict. According to a study on motion sickness factors, 59 % of the subjects experienced motion sickness after 14 minutes on average. The remaining subjects did not experience any illness [27]. However, Motion sickness can be mitigated in several ways described later.

A use-case of XR is simulating a subject physically out of reach, such as a planet in space or the inside of a patient's knee. For physically impaired users, this advantage is even more relevant. XR can also simulate situations that would otherwise be dangerous, for example, an untrained surgeon performing surgery alone. Another advantage is that VR is more immersive than other mediums. The immersion makes the user feel more stress or fear and makes it feasible to prepare personnel for stressful situations, such as the police. The added immersion also makes teaching or training more motivating for the student [20] while improving learning outcomes [**cynthia**].

2.2.4 Designing for VR

VR is relatively new in mainstream media, and as such, there are few agreed-upon standards for designing the User Experience (UX). Common VR tips from game designers [**vrdesign**, **vrdesignadobe**] are immersing the user in a enviroment with real life interactions instead of just text. Other tips include keeping the user comfortable by keeping the user in control of movement, avoiding bright colors and placing objects at a appropriate height.

Some practical measures to counter motion sickness and fatigue are reducing Field of View (FOV), varying latency between user input and display update, flickering, moving content in the virtual environment, and using several stimuli (audio, haptic feedback) [7]. Some of these are hardware-dependent and not relevant for this project, so moving content is the most relevant measure in this project.

2.2.5 Game Engine

A game engine is a framework specifically used for developing games [**gameengine**]. Several third party game engines are available, the most popular being Unity and Unreal Engine. Using a game engine speeds up the development as it includes systems needed, like rendering, animations, and physics.

The Unity game engine [44] was used for developing the application. Unity is widely used for game development [14] and is well documented online with a lot of community-supported plugins. The author also had previous experience with Unity development. Unity has good support for different VR and XR platforms [46] including the OpenXR standard widely used by commercial VR headsets [36]. OpenXR allows creating VR games with Unity handling most VR logic.

Part of the Unity VR support is the XR interaction Toolkit (XRTK) [51]. It is a high-level framework for using XR interactions with unity events. The toolkit also includes components for selecting/grabbing, haptic feedback, and button interaction.

To develop for several XR platforms, the different controllers have a mutual interface for setting keybindings, called an XR controller. Almost all commercial VR controllers support an index finger trigger, a joystick/trackpad with 2D directional input, a grip button, and at least one extra button [44].

Entity Component System

Development in unity is based around the Entity Component System [**entitycomponent**]. An entity is every object populating the game. The base class for all entities is the GameObject. Every model, player and menu is a GameObject.

Components attaches to one or more Entities to define the behaviour. Entities can have several components. Components can provide game logic, movement, user input and much more. Components are scripted in C# and the .NET framework.

Chapter 3

Design and Implementation

The author developed the application in 6 months. The author had no medical experience and was supported by staff at Helse vest with guidance and expertise in orthopedics, medical imaging and Unity, and VR development.

3.1 Demonstration

gameplay

3.2 Design methods

3.2.1 Research Method

The research question was answered by evaluating the VR application regarding the research questions. A mixed testing method was used to evaluate the application.

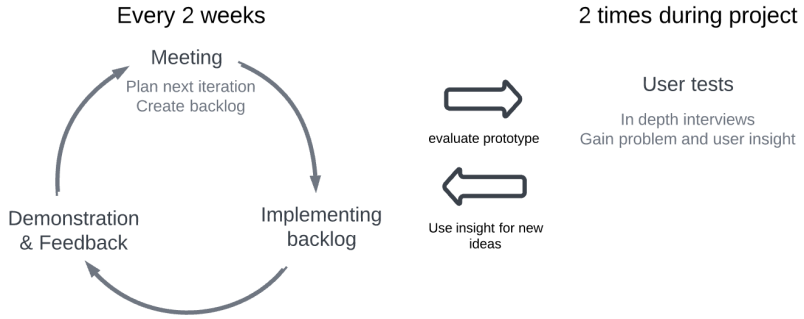
The most significant part of the study was developing the application that fills the need for viewing CT data. This was accomplished by uncovering user needs with interviews and testing the application on users. When the application met minimum specifications, an evaluation was performed.

The evaluation at the end of the project aims to answer both research questions. The evaluation was done in the following way: The project participants use the application on their own in a relevant scenario. The users respond to a System Usability Scale form and a semi-structured interview. The questions are specific to the research questions.

The responses are gathered and used to answer the questions.

3.2.2 Iterative Development

The application is developed using iterative development and elements from Design Thinking.



(a)

Figure 3.1: Development method
weekly iterations illustrated on the left, and user testing on the right.

Iterative development was used to cooperate effectively with the team and increase the development speed. The team had regular meetings, and this fits well with the iterative cycles (figure 3.1). The cycles facilitated frequent feedback from the domain experts. The feedback could then be implemented and tested at the next meeting.

TODO: bilde av early prototype?

The team had regular biweekly meetings with a demonstration and next iteration planning. The meeting was a valuable way of sharing interdisciplinary information and getting feedback from domain experts. Scrum elements such as Sprint retrospective and daily scrum were not relevant as only one person worked full time on the project. After the demonstration, the backlog was updated with tasks prioritized by value. The value is determined by the time needed to implement compared to how important the functionality is to the product owner. The top tasks that fit within two weeks are added to the backlog, and the remaining are stored in the product backlog and considered for the next cycle. During the next two weeks, as many tasks as possible were implemented and demonstrated in the next meeting.

The frequent meeting demonstrations worked really well with VR development, as the different features are hard to understand without trying the application. Fast prototypes are essential to cooperate efficiently when the product owner or test users have a limited understanding of the technology.

Design thinking elements was used because both the medical domain and user needs were challenging to understand. Specifically the empathise, define and ideate stages was used. Frequent prototyping would be too time consuming too be relevant. Design Thinking helps to understand an unfamiliar domain and what the needs of the users are. The application demos served as smaller user testing sessions. The medical personnel in the team would try out the application and any new features and give feedback and ideas during testing. User testing would often lead to new ideas. Optimally the test subjects should have been personnel outside the project without any bias, but busy schedules and COVID restrictions limited this.

A more in-depth user testing was done twice in the project. This included testing the application with orthopedic surgeons to gain more insight into user experience and user needs. Optimally this would be done frequently at the end of iterations, but the test users were unavailable. The tests are described in detail in a later chapter.

3.3 Project overview

3.4 Design

The final product is a VR viewer for viewing STL files and DICOM data. A VR solution was selected for faster development and better hardware availability than AR, while the AR advantages were not relevant for the planning phase. The solution uses the STL files and DICOM data already produced at the hospital and requires no extra data other than what is used for 3D printing.

The primary goal for the UX design is ease of use for non-technical users, as current VR solutions at Haukland University Hospital are cumbersome programs. A secondary goal is to make users understand the fracture well and facilitate planning the surgery.

3.4.1 Hardware

The VR devices used for this development and testing was the Oculus Quest 1 and Quest 2 [35]. The hardware was selected out of availability reasons and the practicality of having a standalone device. The Quest 1 and 2 are very popular consumer standalone VR headsets. When in standalone mode, the headset uses integrated graphics and is battery powered. There are other headsets with better displays and motion tracking, but the Quest devices are sufficient.

Testing in standalone mode at different locations was very convenient, but it had a drawback as the project needed to be tailored to run on a low performance android device. This led to problems with both high resolution data and different bugs related to the android build.

3.4.2 VR interface

To make the VR interface easy to use, the focus was primarily on familiarity with current tools and a simple design. I used MicroDICOM [12] as a reference for a familiar tool. Familiarity with current tools means taking interactions from programs such as CT viewers and adopting the same interactions to VR to make it recognizable and consistent to the user. An example of this is navigating a menu in VR the same way one would navigate a menu with a pc mouse. All unnecessary interactions and information is removed to allow new users with little VR experience to start using the program quickly. The goal is to make the experience less overwhelming and lower the skill required to use the viewer effectively.

Architecture

The architecture of the Unity application is based around a singleton 'GameController' module. The GameController manages important states and contains references to the other modules, shown in figure 3.2. The modules have different responsibilities like multiplayer or menu interaction and communicates through the GameController. This solution is easy to extend and prevents a lot of cross-references between components. A module typically consists of a GameObject with the same name, and one or more components attached to the GameObject. In some cases the GameObject has several GameObjects as children. As an example the Model Container has a child for every part of the model. This is indicated by one to many relationships in the figure 3.2.

The components related to the model have several instances and are drawn as one-to-many relations in the figure. This allows to swap between models while keeping info on positions and implants.

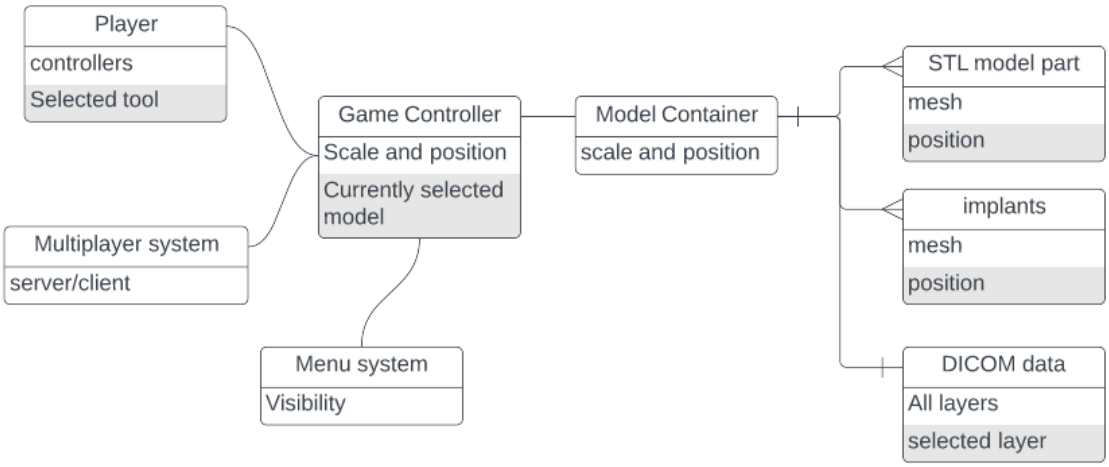


Figure 3.2: Unity project modules

Learning process

A video game style tutorial was considered, where the user would go through tasks with hints or descriptions of the features in the application. However, this was not implemented as early testing revealed users were learning the controls quickly.

Every button has a description displayed as text above the button to remind the user of the keybindings as in figure 3.3. An image can optionally be displayed, showing the user the keybindings. The hints were implemented as users repeatedly forgot button controls in testing.

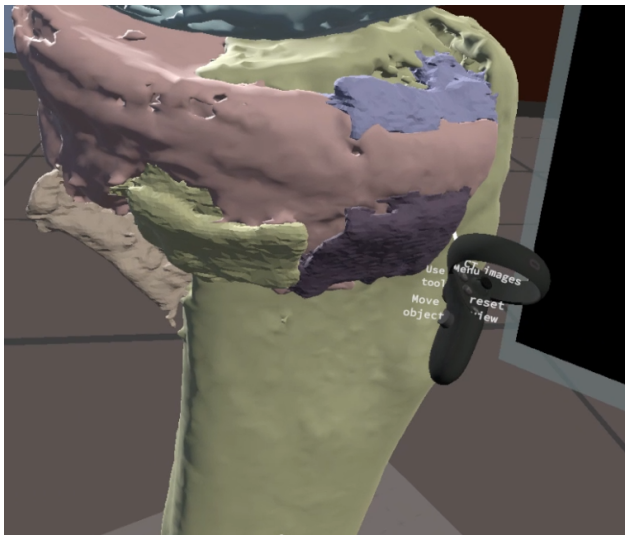
Grab interaction

The most used part of the user interface is moving fracture parts to inspect or rearrange them. Rearranging could be important for fixation planning. The

(a)

Figure 3.3: Controller hints

grab should mimic how users move objects in real life to make this easy to use. When the user places the hand close to a model part, the color is tinted to highlight the model. When the grab button is pressed, the highlighted model part is picked up, and a short audio clip is played.



(a)

Figure 3.4: Grabbing an object
Notice how the model part close to the controller is focused with a darker color.

One issue is whether the user wants to move either the entire model or move a part of the model when grabbing. Including both could introduce mode issues [16], so the grab can only have one use. It was decided that the grab interaction always picks up a model part. A joystick input moves the entire model, which is less precise than grabbing.

The grab functionality is implemented using Unity's built-in *Interactable* system for XR [51] development. This allows to set up grabbing, throwing, and more. By default, the grab will move the center of the interactable to the user's hand. The built-in grab action is cumbersome when precisely adjusting larger objects, as moving the interactable a small distance requires repositioning it from wherever the center is. To solve this, the interactable script is modified to keep the current position when grabbed and then follow the relative motion of the hand.

To allow for grabbing small objects or objects that are partly overlapping, the closest object is always selected. The default Unity XR behavior is selecting a new object in every frame, and this would lead to the user grabbing several objects simultaneously. To solve this, any interactable is highlighted when the users hand is within the collision box, as in figure 3.4. It stays highlighted until the hand is outside of the collision box. The grab action is only allowed to select the hovered object, until the grab is released.

Using Occulus hand tracking [40] could be beneficial, but is currently not recommended for high degrees of precision. Future versions of hand tracking or Hololens hand tracking could improve the interact experience in the future.

Separation mode

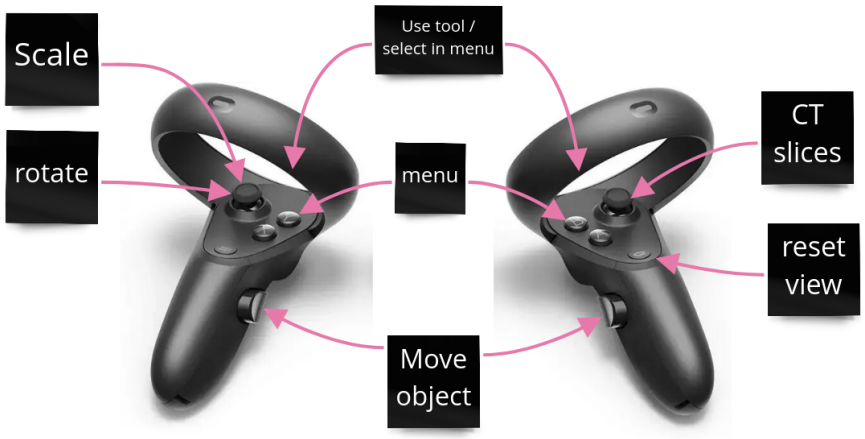
The grabbing functionality introduces a problem: If the user moves a model part, the CT images will no longer be similar to the model displayed. Also, if the user is measuring the distance between two model parts, it gives an invalid result if the parts are moved. The solution to this is a separation mode. Moving any model part enters separation mode and reverting to the original position exits separation mode. In separation mode, the CT images are hidden, and measurements are hidden. A large button appears to display the current mode, prompting the user to click the button to reset the model and once again view CT images.

Keybindings

Controller keybindings shown in figure 3.5 was set to make the controllers intuitive and easy to use. This configuration seemed to work well in testing.

The most common operations are bound to trigger and grip button, which is easily reachable on most controllers. The controller trigger mimics a mouse click on a pc, and so it is used for clicking buttons and using selected tools. The grip button mimics a real-world grabbing gesture and is used for grabbing a model part. The primary button is used for opening the menu. For ease of use, the two controllers are mostly mirrored, except for the joystick. The right joystick is used for scrolling through CT slices, similar to how the scroll wheel is used in CT image viewers. The left joystick rotates and scales the entire model while keeping all the model parts correctly positioned.

The trigger button is also used for menu selection for consistency with other VR apps. This introduces conflicting keybindings with the *use tool* button. This is solved by only performing one of the actions based on the context. If a menu is



(a)

Figure 3.5: Keybindings on Quest controllers
This illustration is also used within the application.

open or a button is hovered, a menu click is performed. Otherwise, *use tool* is performed. This workaround simplifies the controls by using less buttons.

Menu system

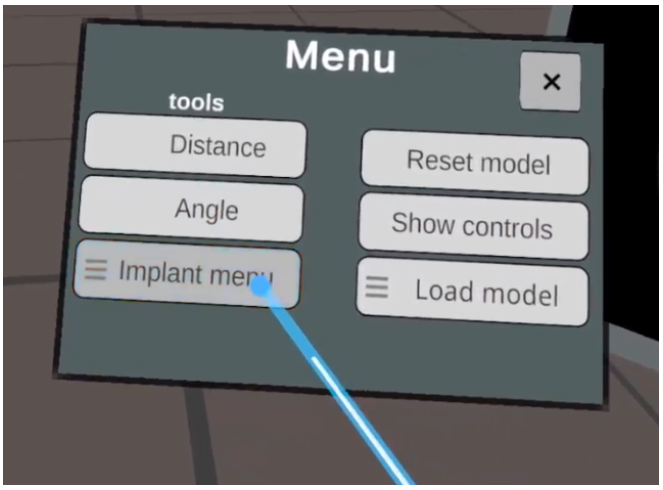
A menu system is implemented to allow the user to navigate less common operations. The different operations are shown in 3.6. When the menu button is pressed, the menu screen appears in front of the user. It is shown in world space, which means the text appears as a physical screen. The model is temporarily hidden to ensure the menu is always in the user's field of view. Placing content at the center of the screen avoids a typical VR UX problem where some information is not visible to the user.

The menu includes access to less frequent actions that do not require their own keybinding. The initial menu includes fast access to simple tools (e.g., measurement) and buttons to sub-menus. A hamburger menu icon suggests the button leads to a sub-menu.

The menu layout consists of one or more vertical lists of buttons. Several columns of lists are used to separate buttons into categories. To keep the menu simple and consistent, only buttons are used for interacting with the menu. The 'cross' button from pc programs is used for exiting the menu. The sub-menus are implemented when multiple options are present. It is used for selecting an implant type or loading a new dataset.

Measurements

The most used functionality available in desktop CT viewers should also be available in the VR viewer. The needed tools are, according to early testing, measurement of distance and measurement of angles. Taking measurements



(a)

Figure 3.6: Menu screen
The main menu interface. The user is pointing at one of the buttons.

allows surgeons to estimate sizes and plan what tools to use during a surgery. Figure 3.7 shows a distance measurement.

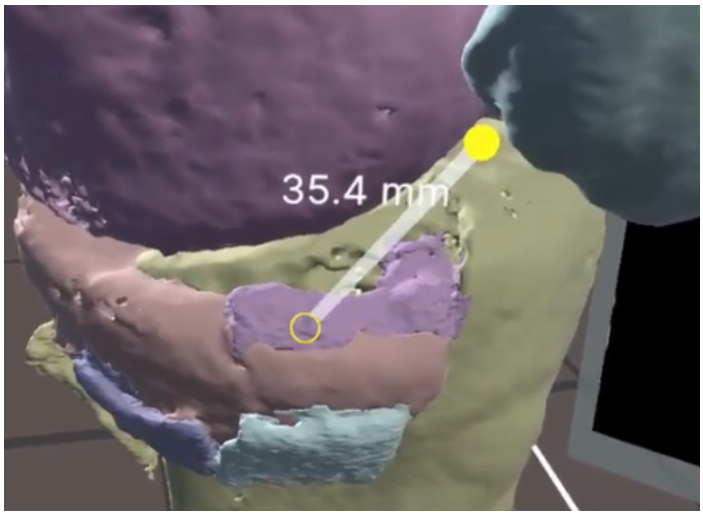
The measure tools works by selecting a start and an end point. The angle measurement additionally requires a middle point. After placing the first point, a preview of the measurement result is shown as a line to give feedback to the user of the current state. Using the outline shader mentioned previously makes it possible to see the placed points even when obscured by the model, and the line between the points is always visible. This, combined with depth in VR makes it easy to understand where the points are placed and what the length and angle is.

Audio

Audio is mainly used for interaction feedback. If the user successfully picks up a model part, a sound is played to indicate the user is holding something. Audio is also used for feedback on the measurement tools. Audio in VR increases presence and investment and reduces distractions [26].

Motion sickness

The most relevant motion sickness factor for this project was the use of motion and static content. The only moving content is the fracture model, which only moves because of player input. The environment is designed like an office room and is always static. All menus and other objects are also static, with a few exceptions. During multiplayer, the model may move caused by another player's input, which might cause some motion sickness if the model is viewed by the user. This could be solved by hiding the model when it moves, but it could interfere with the multiplayer experience if the model suddenly disappears.



(a)

Figure 3.7: Distance measurement
The yellow hollow circle is inside the model with outline rendered on top. The filled circle is in front of the model. Distance in mm also drawn on top.

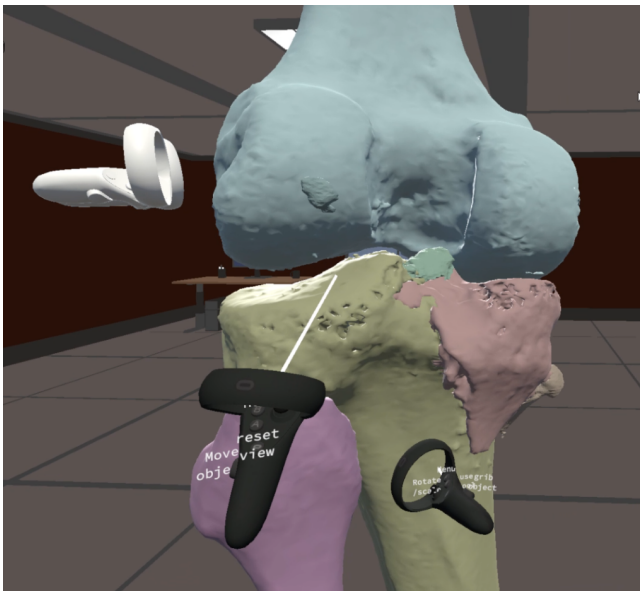
3.4.3 Multiplayer design

The network discovery and session connection parts required for pairing two Quest devices over local networks or internet was not completed due to some network issues and lack of time. Multiplayer was not used in testing, but questions were still asked regarding its usefulness during interviews.

The goal of the multiplayer feature was to allow two or more surgeons to cooperate effectively. Having one surgeon using a VR headset and another using a screen is likely interfering because the surgeons are seeing different models. The multiplayer component allows the surgeons to use the same tools while seeing the same model, and should increase communication and cooperation. Multiplayer should not make the application more difficult to use. All interactions should be similar to the usual experience, and cooperation should not require any extra steps, except connecting to the multiplayer server. Any important information should be available to all players, and irrelevant information (another player opening a menu) should not disturb users.

The multiplayer interface is a menu shown at startup. The user can select either to start a new session in which other users can join, or join an existing session. The sessions are hard-coded to always run on the same network port and have no authentication, as this would take too much development time and give no value. The server is also hosted by the user starting the session instead of a dedicated server, for easier development.

All model parts are synchronized between players, so any models and movements is visible for all users. The position is continuously shared whenever a user is grabbing a model. As shown in figure 3.8 other users' controllers are also visible in order to communicate and point at locations.



(a)

Figure 3.8: Multiplayer
Image showing a session with another user with white controllers.

Mirror

The networking was built with Mirror, a downloadable Unity asset. Mirror is a high-level networking API built on deprecated Unity networking [33]. Mirror has good support for creating a client/server pattern and classic video game related operations such as spawning in players at the start of the game and synchronizing objects across all clients.

3.5 DICOM data visualisation

I received several anonymized CT scan files from Helse Vest to use as input data. The CT scans are sent as DICOM packages, where the scan is stored as several files, each representing a slice. The model is also split up into separate bone fragments by radiologists at Haukeland University Hospital. The files include 3D surface models as STL files, where each file contains one bone.

3.5.1 Unity with custom file types

To be able to read the DICOM files, the build produced by Unity needs access to the files. Unity has several built-in solutions to this. The most common is the Resources functionality [38] that builds the application and includes all files in a 'Resources' folder. Android projects are built in the standard APK format and also require special file paths and using the built-in web request to read files. This is not well documented, especially for the Quest, so this method was scrapped.

The easiest way of referencing files is usually setting a reference in the unity editor, and the file will be included in the build. This works great with common files like images, but Unity does not allow custom file types without 'hacking' the Unity inspector and creating a custom file importer [39], that has lacking documentation. The solution to this is to rename all DICOM files to name.bytes. This makes Unity handle the files as 'TextAsset' text files [47]. The files are then opened as text files and create a byte stream. The byte stream is then used for rendering with the DICOM library.

3.5.2 Rendering CT image

The user can view CT slices in 2D while simultaneously viewing the model as illustrated in figure 3.9. This is done because of two reasons: to bridge the gap going from traditional 2D images to VR and retain the density data lost when converting to a mesh model. An additional benefit is the use case of practicing reading CT images while viewing the corresponding model.

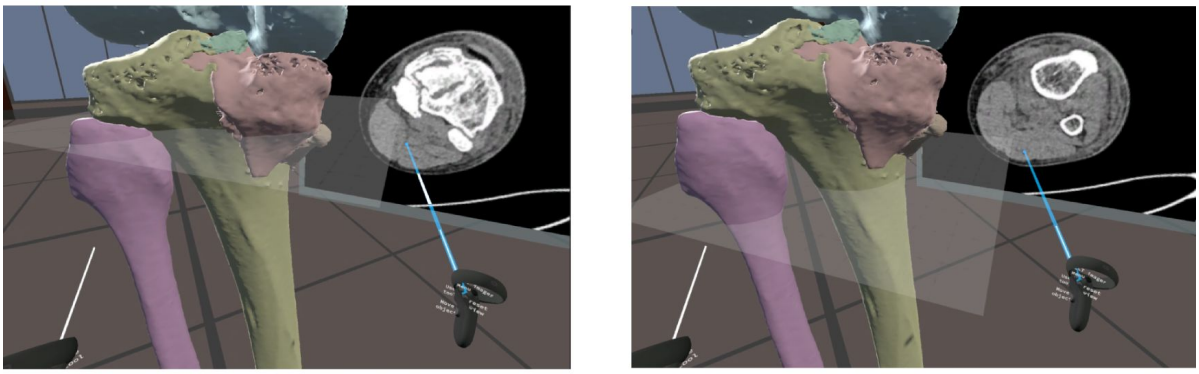


Figure 3.9: CT image
The user is moving the CT slice plane (transparent grey) up and down to inspect the Tibia and Fibula (Shinbone/Calf Bone). The CT image shows the CT scan of the two bones. [34]

To render CT images from the DICOM data, the .NET library *Fellow Oak DICOM* [18] is used. It is an open-source library for parsing DICOM data and image rendering. The data from the DICOM files are used to create a *Slice* object for each of the files. The slice object implements reading the density value at position (x, y) at that slice. A Unity Texture is created for each slice to render the image in Unity.

Every pixel on the texture is set to the greyscale color correlating to the density value. To find the color in a position, calculate $(density - minDensity)/(maxDensity - minDensity)$. If result is 1, it has maximum density and is set to white color.

The slice number is calculated to convert between the selected height and the rendered CT image. This is done by getting the model's height and reading the y position of the selection plane. The height percentage of the plane is rounded down to the closest slice number, and that slice is shown.

3.5.3 Rendering bone fragments

In STL files all vertices are stored coordinates relative to the center, so by inserting all STL files at the same world space coordinate, the bone fragments will be placed correctly relative to each other. The STL files are exported to .obj files using blender and rendered with Unity.

Accurate scaling of the model is important, both to give a realistic view for the user and for tools like measurement and implants. The STL files used for input does not contain physical size data, but uses a convention where one unit corresponds to one mm. In Unity a single unit is rendered as one meter in VR. By scaling the Unity object so a model unit is 1/1000 of a Unity unit the sizes are perceived as true to life. When scaling the model, all measurements and implants are scaled simultaneously. Being able to view life-sized models could make the planning closer to the surgery and make tools more familiar to the user.

Separating bone fragments

Adding some form of manual splitting of meshes was considered. This would allow the user to split the model pieces into separate pieces further. This functionality would also make it possible to use models that are not split into pieces beforehand.

An example open-source framework called Ezy-slice [4] was tested. This was later abandoned as the model was manually segmented at Haukeland University Hospital.

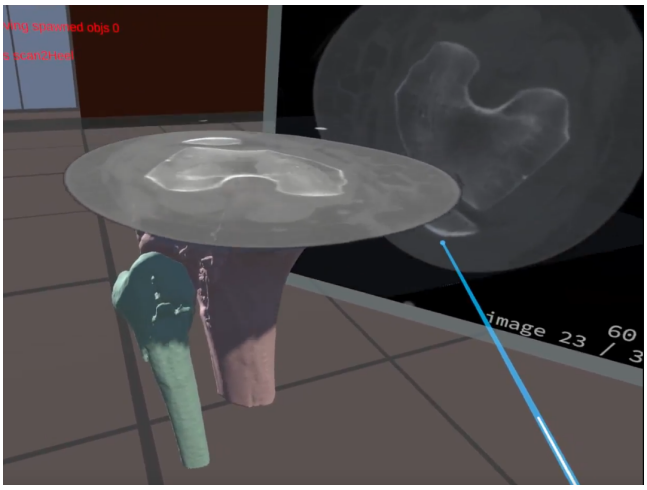
3.5.4 Alternative slice visualisation

While rendering the selected slice alongside the model was useful, there was still a some disconnect between the model and the CT image. To improve upon this, the CT image was rendered at the exact position where it would intersect the model (figure 3.10). Now the user sees both the model and the CT data simultaneously.

This required removing the part of the model above/under the intersection plane, so it did not obscure the image. For this, a Unity Assetstore shader was used to view the cross-section of a mesh [1].

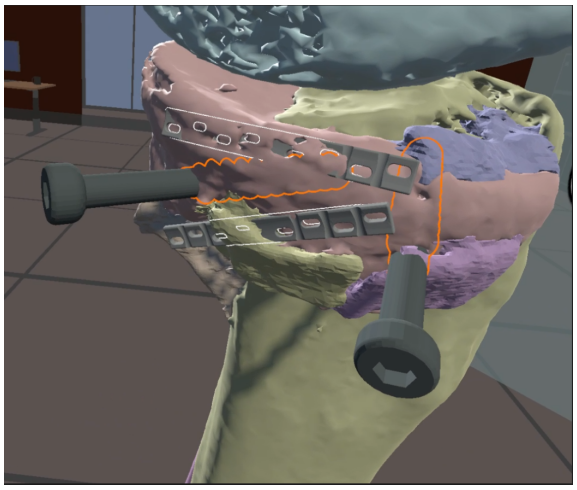
This effect can be disturbing as the entire model is not visible and the image can obscure objects behind. To allow for standard rendering, intersection rendering is made optional. The option defaults to showing the entire model because it is easier to understand. The toggle button is shown in the CT image options.

When overlaying the slice texture onto the intersection plane, the image will completely obscure the model when looking down at the image. This removes the point of overlaying the CT image, so the picture is created as partially transparent. When the texture is created, the grey value is used as input for calculating the transparency of the pixel. This results in most of the outside of the image being completely transparent.



(a)

Figure 3.10: CT intersection
The selected CT slice is shown both in 2D and as a overlay on top of model. Notice that the upper part of the model is not rendered.



(a)

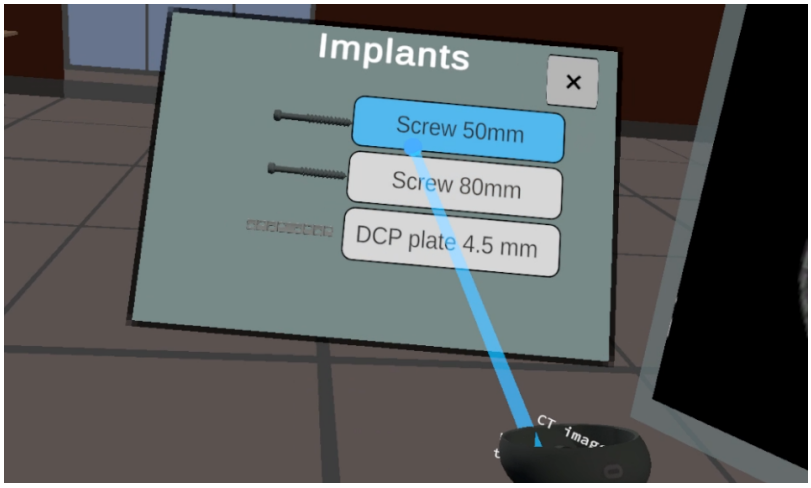
Figure 3.11: Implants
Picture showing 4 implants placed on and inside the fracture. Outline makes it easy to understand the exact position.

3.5.5 Implants

The menu has a sub-menu for choosing what implant to add to the scene (figure 3.12). It consists of a list with a description and a preview of the implant. Pressing a button will spawn the selected implant in front of the user, and can then be moved in the same way a model part can be moved.

To improve the surgeon's ability to line up screws and find what parts of the fracture is penetrated by the screw/object an outline is shown as in figure 3.11.

This allows the user to see where an object is compared to any other parts, even if it is entirely obscured by the model. This is achieved by a shader using a depth buffer to only draw the outline when obscured by other meshes [45].



(a)

Figure 3.12: Implants menu
The menu used for selecting implants, with a side by side preview.

3.6 Performance

Performance guidelines for Occulus Quest hardware are outlined by the Occulus developers for the Quest 1 in the documentation [35].

The recommended triangle count is between 350k and 500k triangles [37]. The imported meshes can have a high vertex count and are reduced manually using remeshing in MeshLab [9] to bring the total to under 500k. The reduced quality is barely noticable when viewed in the resolution of the Quest HMD. The mesh simplification could be automated, but this limit would be irrelevant if using a headset with a modern GPU for dedicated rendering.

TODO: performance data?

Chapter 4

Analysis and assessment

Two user tests were performed during the project to evaluate the application and try to answer the research questions. In this chapter both tests are described.

4.1 Evaluations

4.1.1 Participants

Two surgeons -few participants -previous experience

4.1.2 Test nr. 1

date: 16 Jan. 2022

This was the first test done with professional end users. Previous to this test only the surgeon on the team had proposed what functionality was needed in the product. The goal of the test was to figure out if any of the proposed use cases and features was relevant for other users. The intended use cases was also described. At this point the intended use cases was simply increasing the understanding compared to traditional slice visualisation. The test was performed by a short introduction on the application and a introduction on how to use the VR controllers. Afterwards each surgeon would have 5 to 10 minutes to try out all features at their own pace. The HMD was mirrored to a screen to observe the user. Minimal guidance was given during the testing, except for any breaking bugs or clarifying questions.

Both surgeons agreed that using VR to understand the fractures was an insignificant improvement. The reason stated was that both surgeons are very experienced with reading 2D CT images,

Occupation	Experience	proficiency with VR	comments
hline Orthopedic surgeon	10+ years	None	The density of bone is important to know parts are solid enough for drilling. Also, some parts below the density threshold are missing.
Orthopedic surgeon	10+ years	None	Smaller parts and bone fragments should be moveable to "reponere" bruddet.

Chapter 5

Discussion

Chapter 6

Conclusion

6.1 Compared to previous tools

6.2 Compared to 3D printing

-price -time -equal to or better performance

Chapter 7

Further Work

Appendix A

Source code

The source code for the plug-in is available at this URL: `https://github.com/....`

The source code for the underlying ...: `https://github.com/....`

Bibliography

[1] Abdullah Aldandarawy. *Unity Cross Section Shader Using Shader Graph.* en. Dec. 2019. URL: https://codeburst.io/unity-cross-section-shader-using-shader-graph-31c3fed0fa4f (visited on Jan. 27, 2022).

[2] Christopher Andrews et al. "Extended Reality in Medical Practice." eng. In: *Current Treatment Options in Cardiovascular Medicine* 21.4 (Mar. 2019), p. 18. ISSN: 1092-8464. DOI: 10.1007/s11936-019-0722-7.

[3] Diana Anthony et al. "Patient-specific virtual reality technology for complex neurosurgical cases: illustrative cases." EN. In: *Journal of Neurosurgery: Case Lessons* 1.23 (June 2021). Publisher: American Association of Neurological Surgeons Section: Journal of Neurosurgery: Case Lessons. DOI: 10.3171/CASE21114. URL: https://thejns.org/caselessons/view/journals/j-neurosurg-case-lessons/1/23/article-CASE21114.xml (visited on Mar. 16, 2022).

[4] David Arayan. *DavidArayan/ezy-slice.* original-date: 2015-11-17T08:38:41Z. Feb. 2022. URL: https://github.com/DavidArayan/ezy-slice (visited on Feb. 14, 2022).

[5] William Bradley. "History of Medical Imaging." In: *Proceedings of the American Philosophical Society* 152 (Sept. 2008), pp. 349–61.

[6] Ceevra, Inc. *Using Virtual Reality (VR) Models for Preoperative Planning.* Clinical trial registration NCT03334344. submitted: November 1, 2017. clinicaltrials.gov, June 2019. URL: https://clinicaltrials.gov/ct2/show/NCT03334344 (visited on Sept. 26, 2021).

[7] Eunhee Chang, Hyun Taek Kim, and Byounghyun Yoo. "Virtual Reality Sickness: A Review of Causes and Measurements." In: *International Journal of Human–Computer Interaction* 36.17 (Oct. 2020). Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/10447318.2020.1778351, pp. 1658–1682. ISSN: 1044-7318. DOI: 10.1080/10447318.2020.1778351. URL: https://doi.org/10.1080/10447318.2020.1778351 (visited on Nov. 15, 2021).

[8] Chougule, Mulay, and N. Ahuja. "Conversions of CT Scan Images into 3 D Point Cloud Data for the Development of 3 D Solid Model using B-Rep Scheme." en. In: *undefined* (2013). URL: https://www.semanticscholar.org/paper/Conversions-of-CT-Scan-Images-into-3-D-Point-Cloud-Chougule-Mulay/d4875163b4162ba44843a326490f20d6c7fa33e7 (visited on Dec. 1, 2021).

[9] Paolo Cignoni et al. "MeshLab: an Open-Source Mesh Processing Tool." en. In: *Eurographics Italian Chapter Conference* (2008). Artwork Size: 8 pages ISBN: 9783905673685 Publisher: The Eurographics Association, 8

pages. DOI: 10.2312/LOCALCHAPTEREVENTS/ITALCHAP/ITALIANCHAPCONF2008/129-136. URL: http://diglib.eg.org/handle/10.2312/LocalChapterEvents.ItalChap.ItalianChapConf2008.129-136 (visited on Apr. 10, 2022).

[10] *CommandEP – SentiAR*. en-US. URL: https://sentiar.com/commandep/ (visited on Mar. 16, 2022).

[11] *DICOM*. URL: https://www.dicomstandard.org/ (visited on Jan. 21, 2022).

[12] *DICOM VR – Visualizing and Manipulating Medical Imaging in a New Dimension*. en-US. URL: http://www.dicomvr.com/ (visited on Mar. 16, 2022).

[13] dicomdirector.com. *Surgeons — Solutions For DICOM Done Better, Faster + 3D on every device*. en-US. URL: https://www.dicomdirector.com/for-surgeons/ (visited on Sept. 1, 2021).

[14] Lars Doucet, Anthony Pecorella September 02, and 2021. *Game engines on Steam: The definitive breakdown*. en. Section: business. Sept. 2021. URL: https://www.gamedeveloper.com/business/game-engines-on-steam-the-definitive-breakdown (visited on Feb. 6, 2022).

[15] John Ebnezar and Rakesh John. *Textbook of Orthopedics*. en. Google-Books-ID: 0efhjwEACAAJ. JP Medical Ltd, Dec. 2016. ISBN: 978-93-86056-68-9.

[16] World Leaders in Research-Based User Experience. *Modes in User Interfaces: When They Help and When They Hurt Users*. en. URL: https://www.nngroup.com/articles/modes/ (visited on Feb. 7, 2022).

[17] *FEELREAL Multisensory VR Mask*. en. URL: https://feelreal.com/ (visited on Nov. 3, 2021).

[18] *Fellow Oak DICOM*. original-date: 2015-05-09T13:35:00Z. Feb. 2022. URL: https://github.com/fo-dicom/fo-dicom (visited on Feb. 11, 2022).

[19] Elliot K. Fishman et al. "Volume Rendering versus Maximum Intensity Projection in CT Angiography: What Works Best, When, and Why." In: *RadioGraphics* 26.3 (May 2006). Publisher: Radiological Society of North America, pp. 905–922. ISSN: 0271-5333. DOI: 10.1148/rg.263055186. URL: https://pubs.rsna.org/doi/10.1148/rg.263055186 (visited on Dec. 6, 2021).

[20] Laura Freina and Michela Ott. "Immersive Virtual Reality in Education: State of the Art and Perspectives." English. In: *The International Scientific Conference eLearning and Software for Education*. Vol. 1. Num Pages: 9. Bucharest, Romania: "Carol I" National Defence University, 2015, pp. 133–141. URL: https://www.proquest.com/docview/1681252932/abstract/D4B2F610BED4D5BPQ/1 (visited on Nov. 11, 2021).

[21] Matthew Hackett and Michael Proctor. "Three-Dimensional Display Technologies for Anatomical Education: A Literature Review." en. In: *Journal of Science Education and Technology* 25.4 (Aug. 2016), pp. 641–654. ISSN: 1573-1839. DOI: 10.1007/s10956-016-9619-3. URL: https://doi.org/10.1007/s10956-016-9619-3 (visited on Nov. 3, 2021).

[22] David L. Hamblen and A Hamish R.W. Simpson. *outline of orthopaedics*. 14. edition. 2010.

[23] Medical Holodeck. *Medicalholodeck. The Virtual Reality Platform for Medical Teamwork*. en. URL: https://www.medicalholodeck.com/en/ (visited on Sept. 1, 2021).

[24] *ImmersiveView™ VR*. en-US. URL: https://www.immersivetouch.com/immersiveview-vr (visited on Mar. 16, 2022).

[25] Mohd Javaid and Abid Haleem. "Virtual reality applications toward medical field." en. In: *Clinical Epidemiology and Global Health* 8.2 (June 2020), pp. 600–605. ISSN: 2213-3984. DOI: 10.1016/j.cegh.2019.12.010. URL: https://www.sciencedirect.com/science/article/pii/S2213398419304294 (visited on Apr. 7, 2022).

[26] Angelika C. Kern and Wolfgang Ellermeier. "Audio in VR: Effects of a Soundscape and Movement-Triggered Step Sounds on Presence." In: *Frontiers in Robotics and AI* 7 (2020). ISSN: 2296-9144. URL: https://www.frontiersin.org/article/10.3389/frobt.2020.00020 (visited on Apr. 10, 2022).

[27] Frank L. Kooi and Alexander Toet. "Visual comfort of binocular and 3D displays." en. In: *Displays* 25.2 (Aug. 2004), pp. 99–108. ISSN: 0141-9382. DOI: 10.1016/j.displa.2004.07.004. URL: https://www.sciencedirect.com/science/article/pii/S0141938204000502 (visited on Nov. 3, 2021).

[28] Ben Lang. *An Introduction to Positional Tracking and Degrees of Freedom (DOF)*. en-US. Section: Body Tracking. Feb. 2013. URL: https://www.roadtovr.com/introduction-positional-tracking-degrees-freedom-dof/ (visited on Nov. 11, 2021).

[29] *MedicalImagingVR on Steam*. en. URL: https://store.steampowered.com/app/1471980/MedicalImagingVR/ (visited on Mar. 16, 2022).

[30] Omar Merhi et al. "Motion Sickness, Console Video Games, and Head-Mounted Displays." en. In: *Human Factors: The Journal of the Human Factors and Ergonomics Society* 49.5 (Oct. 2007), pp. 920–934. ISSN: 0018-7208, 1547-8181. DOI: 10.1518/001872007X230262. URL: http://journals.sagepub.com/doi/10.1518/001872007X230262 (visited on Nov. 11, 2021).

[31] Leslie Mertz. "Virtual Reality Pioneer Tom Furness on the Past, Present, and Future of VR in Health Care." en. In: *IEEE Pulse* 10.3 (May 2019), pp. 9–11. ISSN: 2154-2287, 2154-2317. DOI: 10.1109/MPULS.2019.2911808. URL: https://ieeexplore.ieee.org/document/8720292/ (visited on Apr. 4, 2022).

[32] Matjaž Mihelj, Domen Novak, and Samo Beguš. *Virtual Reality Technology and Applications*. en. Vol. 68. Intelligent Systems, Control and Automation: Science and Engineering. Dordrecht: Springer Netherlands, 2014. ISBN: 978-94-007-6909-0 978-94-007-6910-6. DOI: 10.1007/978-94-007-6910-6. URL: http://link.springer.com/10.1007/978-94-007-6910-6 (visited on Nov. 3, 2021).

[33] *Mirror Networking – Open Source Networking for Unity*. en-US. URL: https://mirror-networking.com/ (visited on Jan. 31, 2022).

[34] Abhishek Mishra et al. "Virtual preoperative planning and 3D printing are valuable for the management of complex orthopaedic trauma." eng. In: *Chinese Journal of Traumatology = Zhonghua Chuang Shang Za Zhi* 22.6 (Dec. 2019), pp. 350–355. ISSN: 1008-1275. DOI: 10.1016/j.cjtee.2019.07.006.

[35] *Oculus Quest 2: Our Most Advanced New All-in-One VR Headset — Oculus*. en. URL: https://www.oculus.com/quest-2/ (visited on Nov. 3, 2021).

[36] *OpenXR - High-performance access to AR and VR —collectively known as XR— platforms and devices*. en. Section: API. Dec. 2016. URL: https://www.khronos.org/openxr/ (visited on Feb. 6, 2022).

[37] *Performance and Optimization — Oculus Developers*. URL: https://developer.oculus.com/documentation/unity/unity-perf/ (visited on Feb. 14, 2022).

[38] Unity resources.load. *Unity - Scripting API: Resources.Load*. en. URL: https://docs.unity3d.com/ScriptReference/Resources.Load.html (visited on Feb. 11, 2022).

[39] Unity scriptedimporters. *Unity - Manual: Scripted Importers*. en. URL: https://docs.unity3d.com/Manual/ScriptedImporters.html (visited on Feb. 11, 2022).

[40] *Set Up Hand Tracking — Oculus Developers*. URL: https://developer.oculus.com/documentation/unity/unity-handtracking/ (visited on Apr. 10, 2022).

[41] N. Shahrubudin, T. C. Lee, and R. Ramlan. "An Overview on 3D Printing Technology: Technological, Materials, and Applications." en. In: *Procedia Manufacturing*. The 2nd International Conference on Sustainable Materials Processing and Manufacturing, SMPM 2019, 8-10 March 2019, Sun City, South Africa 35 (Jan. 2019), pp. 1286–1296. ISSN: 2351-9789. DOI: 10.1016/j.promfg.2019.06.089. URL: https://www.sciencedirect.com/science/article/pii/S2351978919308169 (visited on Apr. 5, 2022).

[42] Paul Suetens. *Fundamentals of Medical Imaging*. en. Google-Books-ID: U11EDgAAQBAJ. Cambridge University Press, May 2017. ISBN: 978-1-107-15978-5.

[43] Marc F. Swiontkowski and Steven D. Stovitz, eds. *Manual of orthopaedics*. 7th ed. Philadelphia: Wolters Kluwer/Lippincott Williams & Wilkins Health, 2013. ISBN: 978-1-4511-1592-5.

[44] Unity Technologies. *Unity - Manual: Unity XR Input*. en. URL: https://docs.unity3d.com/Manual/xr_input.html (visited on Feb. 2, 2022).

[45] Unity Technologies. *Unity - Manual: Using Depth Textures*. en. URL: https://docs.unity3d.com/2019.3/Documentation/Manual/SL-DepthTextures.html (visited on Feb. 4, 2022).

[46] Unity Technologies. *Unity - Manual: XR*. en. URL: https://docs.unity3d.com/Manual/XR.html (visited on Feb. 6, 2022).

[47] Unity textassets. *Unity - Manual: Text assets*. en. URL: https://docs.unity3d.com/Manual/class-TextAsset.html (visited on Feb. 11, 2022).

[48] *The Body VR: Anatomy Viewer on Steam*. en. URL: https://store.steampowered.com/app/579620/The_Body_VR_Anatomy_Viewer/ (visited on Mar. 16, 2022).

[49] Maurizio Vertemati et al. "A Virtual Reality Environment to Visualize Three-Dimensional Patient-Specific Models by a Mobile Head-Mounted Display." en. In: *Surgical Innovation* 26.3 (June 2019). Publisher: SAGE Publications Inc, pp. 359–370. ISSN: 1553-3506. DOI: 10.1177/1553350618822860. URL: https://doi.org/10.1177/1553350618822860 (visited on June 1, 2021).

[50] *Virtual Reality for Surgery — Precison XR*. en-US. URL: https://surgicaltheater.com/ (visited on Mar. 16, 2022).

[51] *XR Interaction Toolkit — XR Interaction Toolkit — 2.0.0-pre.7*. URL: https://docs.unity3d.com/Packages/com.unity.xr.interaction.toolkit@2.0/manual/index.html (visited on Feb. 7, 2022).