3D lab & Trauma Surgery, Radboudumc

Thesis Technical Medicine

Surgical navigation technology for placement of patient-specific osseointegration implants and locking screws in patients with short femoral stumps

E.P.J. Smits
s1361309

Medical Supervisor:
Dr. J.P.M. Frölke, *Trauma Surgeon*

Technical Supervisor 3D lab:
Dr. Ir. L.M. Verhamme

Technical Supervisor UT:
Prof. Dr. Ir. C.H. Slump

Process Supervisor UT:
Drs. R.J. Haarman

Radboudumc

UNIVERSITEIT TWENTE.

3rd December, 2018

Abstract

Patients with lower limb amputation often experience discomfort from their prosthetic leg. For some patients, a suitable alternative to a socket prosthesis is an osseointegrated prosthesis. This consists of a bone-anchored implant that is surgically placed in the medullary canal of the remaining femur. Distally, the prosthetic leg can be attached to the implant with a transcutaneous adapter. In the Radboudumc in The Netherlands, a patient-specific implant is designed and 3D-printed for patients with short femoral stumps. To ensure implant stability, the implant is supported with a locking screw that is placed through the top of the implant and into the head and neck of the femur.

Placing the patient-specific implant is challenging because the depth and rotation of the implant inside the femur determines the position of the locking screw in the femoral head. Conventionally, with intraoperative 2D X-ray imaging, the position of the implant in the bone is visualized. The surgeon needs to determine the correct drilling position on the cortex of the femur to drill through the gap in the implant and place the locking screw. Precisely aligning the C-arm to determine the down-the-beam direction necessary for drilling is time-consuming and often leads to considerable radiation doses for the patient and surgical team.

Surgical navigation technology is an alternative to fluoroscopy. With the use of optical tracking, real-time feedback about the rotation and depth of the implant in the bone can be given. The main goal of this master's thesis is to design and develop a prototype of optical navigation software for precise placement of femoral implants and locking screws. The software is built in Unity, a game engine that allows programming in the C# language and builds a visualization of the surgical scene in 3D. The visualization shows the planned position of the implant and the locking screw in the bone. Using an optical tracking system (PST base) the real-time location of the implant, the bone and instruments are acquired. The software shows the position of the tracked objects in relation to the ideal position. Different visualizations are developed to allow the surgeon to combine and change perspectives. A feedback mechanism is developed that changes the colour of virtual objects from red to green if the planned position is reached.

A pre-clinical test is carried out to determine which visualization guides the user most accurately and efficiently. Different visualizations were tested by 18 users. Objective measurement of the distance between the implant and the surgical planning is obtained, as well as subjective user preferences acquired with a questionnaire. No significant differences between visualizations were found.

In a second test, designed as a proof of concept for guided excavation of the femur, the intraoperatively calculated distance measures are compared to the postoperatively measured distances in a CBCT scan. The intraoperative distance was 6 mm larger than the postoperatively derived measure. This could be an indication of the inaccuracy that is introduced in the navigation software by usage of the optical tracker and by registration of the implant model.

Recommendation for future tests and developments in the software are made to enable testing in a more clinical setting. Surgical navigation technology is able to provide a real-time 3D visualization and give feedback on reaching the planned position without the use of radiation. More research is needed to quantify how much the radiation dose and if surgical time could be decreased.

Acknowledgements

Firstly, I would like to express my gratitude to my supervisors Luc Verhamme, Jan Paul Frölke, Kees Slump and Rian Haarman for their help, knowledge, comments and support during this graduation project.

Without the additional help of the staff of the 3D Lab in the Radboudumc, it would not have been possible to design my software and experimental set-up. A special thank you to Dylan, Jene and Han for helping me understand Unity.

I want to thank my fellow students and co-workers at the 3D Lab for brainstorming, for fun moments during coffee breaks and for participating in my user tests.

My thanks go out to the Orthopaedic Research Lab, especially Richard van Swam, for providing me with a drill I could use for testing guided drilling.

On top of that, I would like to thank my parents, sister, brother and friends for supporting me during this graduation project and during my time as a student. Last but not least, Thijs, thank you for your loving support and for standing by me in everything I do.

Table of Contents

List Of Abbreviations

2D two-dimensional. 4, 7, 9, 11, 14, 24, 47

3D three-dimensional. 4, 6, 9, 11, 14, 15, 18, 23, 24, 27, 31, 47

ANOVA ANalysis Of VAriance. 32, 34

AP anterior-posterior. 27

BAP bone-anchored prosthesis. 1, 48

C# C Sharp. 11, 14

CBCT Cone Beam CT. 39–44, 46, 49

CCS Camera Coordinate System. 10, 16, 17, 31

CT Computer Tomography. 4, 12, 23

DBP euclidean Distance Between Points. 16

DCA Dual Cone Adapter. 13, 24

FC Front Camera. 27, 28, 31, 34, 36

HU Hounsfield Units. 41, 44

IC Implant Camera. 27, 28, 31, 34, 36

ILP Integral Leg Prothesis. 1, 2

k-wire Kirschner wire. 4

MITeC Medical Innovation and Technology expert Center. 10, 11, 13, 24, 49

OIP osseointegrated prosthesis. 1, 47

ONS Optical Navigation Server. 14–16, 23

OpenIGTlink Open Network Interface for Image-Guided Therapy. 10

OPL Osseointegrated Prosthetic Limb. 1, 2, 4

OPRA Osseointegrated Prostheses for the Rehabilitation of Amputees. 1, 2

OR Operating Room. 4, 6, 9–11, 13, 23, 24

Radboudumc Radboud University Medical Centre. 2, 6, 11, 13, 23, 39, 47

RMS Root Mean Square. 16, 24, 25, 31, 41, 48

SC Side Camera. 27, 28, 31, 34, 36

SLS Selective Laser Sintering. 12, 25

SNT Surgical Navigation Technology. iii, 6, 7

TAD tip-to-apex distance. 5, 47

TRE Target Registration Error. 16, 24, 48

UCS Unity Coordinate System. 16, 18, 19, 22, 31

UI User Interface. 7

CHAPTER 1

Introduction

1.1 Lower-extremity amputation

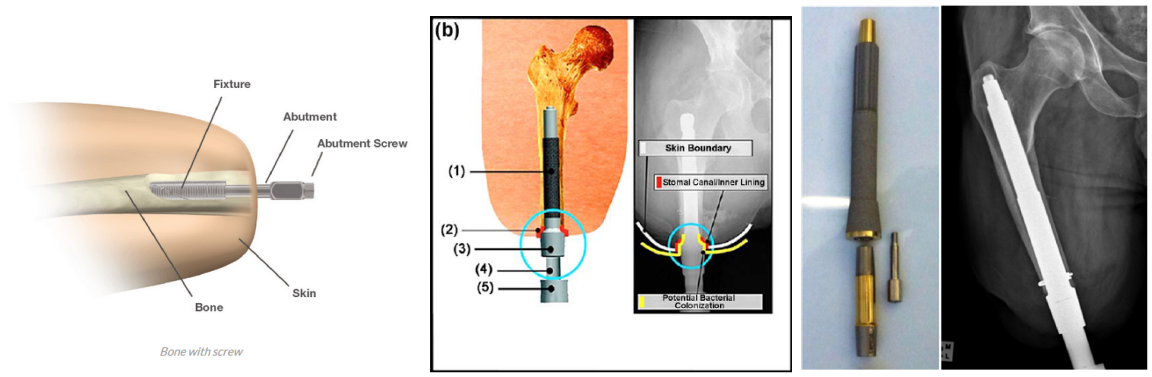
Lower extremity amputation has a severe impact on the quality of life and mobility of patients. In the Netherlands, lower limb amputation (transfemoral or transtibial) occurs in 18 – 20 out of 100 000 people. [1] Common causes of amputation are peripheral arterial disease (90 – 94%), trauma (3%) and tumour resection (3%). [1]

Usually, prosthetic limbs are attached to the residual limb by use of a socket. The socket must closely fit the soft tissue of the limb to enable the transfer of forces from the prosthesis to the residual limb. Many patients experience discomfort because of a poor socket fit caused by atrophy or swelling of soft tissue. [2] Socket-related problems include heat and sweating problems, unreliable suspension of the prosthesis, discomfort while sitting, (chronic) skin irritation and pressure sores, lower back pain and residual limb pain. [2–10] As an effect, patients with poorly fitted socket prostheses experience a reduced mobility and decreased quality of life.

1.1.1 Bone-anchored prostheses

An alternative option for socket prostheses is a bone-anchored implant to which an external prosthesis is transcutaneously attached. Through a stoma, a permanent opening in the soft tissues and skin at the distal end of the stump, the prosthesis is attached to the implant. Thus, the prosthetic leg directly transfers the load onto the skeletal system. [1, 3] The bone-anchored prosthesis (BAP), also called osseointegrated prosthesis (OIP), has become a part of clinical practice in several countries (mainly The Netherlands, Germany, Sweden and Australia) since the introduction in 1990. [1] There are three different types of BAP currently commercially available (Figure 1.1). [2, 3] These are the OPRA (Osseointegrated Prostheses for the Rehabilitation of Amputees, Integrum AB, Sweden) [11], ILP (Integral Leg Prothesis, Orthodynamics, Germany) [12–14] and the OPL (Osseointegrated Prosthetic Limb, Permedica s.p.a., Italy) [15].

Several investigators report how the design of the implants has changed over the years, hereby decreasing the incidence of implant infection, implant failure and stoma problems. [3, 11, 13, 14] The most important difference between the OPRA and the other two implants is the fixation approach. The OPRA (Figure 1.1a) is fixated in the medullary canal with a threaded implant, contrary to the press-fit design of the ILP and OPL. [3] The advantage of the press-fit



(A) OPRA (B) ILP (C) OPL

FIGURE 1.1: Three different commercially available bone-anchored prosthesis implants. The OPRA implant is fixated in the bone by a threaded fixture (A), while the ILP (B) and OPL (C) implants are press-fitted into the bone. [3, 13, 15]

design is that the time between implantation of the ILP and OPL implants and full weight-bearing is approximately 3, and 4 – 5 months, respectively. While the recommended period between implantation and weight-bearing is 12 months for the threaded OPRA implant. [3]

1.1.2 Surgical procedure

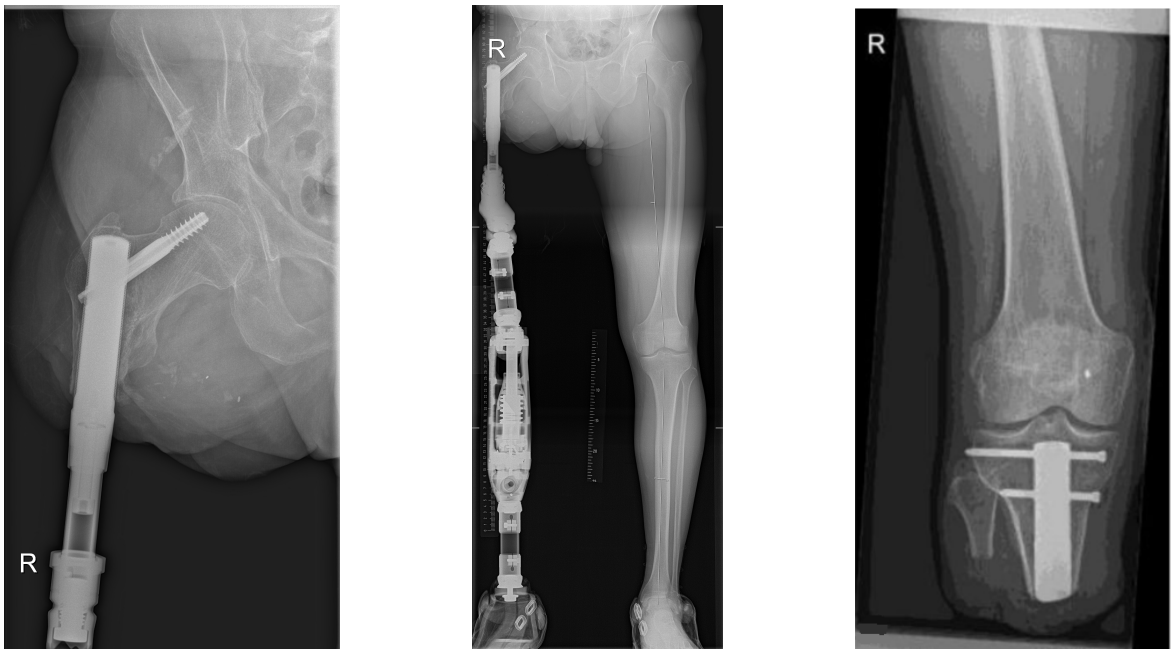
In general, a bone-anchored prosthesis is placed in a two-stage surgical procedure. During the first surgery, the implant is placed in the medullary canal. The soft tissues and skin are closed to allow wound healing and osseointegration of the implant under sterile conditions, i.e. without an opening through the skin. [16] After approximately 6 – 8 weeks, the second surgery takes place during which the stoma is created by cutting a small circular skin incision. The transcuteneous adapter (Figure 1.1b, number 3) is fixated in the head of the femoral stem. [16] After the second surgery, patients may carefully start rehabilitation with guidance of physiotherapists after attaching the artificial limb to the implant with a coupling device.

Complications of BAP surgery reported in literature are skin infection, implant infection, implant loosening, periprosthetic fractures and removal of the implant. [6] In the review by Van Eck et al., the implant infection rate ranged from 2 to 41%, implant loosening occurred in 2 – 6% of patients and implant explantation was needed in 3 – 20% of the cases. [6] However, these numbers are based on studies that evaluated different implant designs. Also, studies which date back more than ten years are included, while implant designs have improved over time to reduce the described complications.

1.1.3 Clinical practice in the Radboud University Medical Centre

In the Radboud University Medical Centre (Radboudumc) (Nijmegen, The Netherlands), several designs are used in clinical practice for both transfemoral and transtibial amputees. Patients with a transfemoral amputation receive the standard Osseointegrated Prosthetic Limb (OPL) implant. The length of the residual femur determines whether a standard implant, with a length of 160 mm (Figure 1.1c), can be used. [16]. The primary stability of the OPL implant is provided by the slightly curved design, in agreement with the curved anatomy of the femur. Secondary stability, i.e. osseointegration of the implant, is facilitated by a plasma-sprayed rough titanium coating. [3]

For transtibial amputees, a custom-made patient-specific implant is designed to fit the droplet-shaped anatomy of the tibia. One or two fixation screws are positioned perpendicular to the implant (Figure 1.2b) for primary stabilisation.



(A) (B) (C)

FIGURE 1.2: Custom-made femoral (A) and tibial implants (C). The prosthesis is attached with a transcutaneous adapter (B). The implants are secured with locking screws. The femoral implant is specifically designed for patients with short femoral remnants. The tibial implant is designed to fit the unique droplet-shaped anatomy of the tibia. Images taken from Frölke et al. [16]

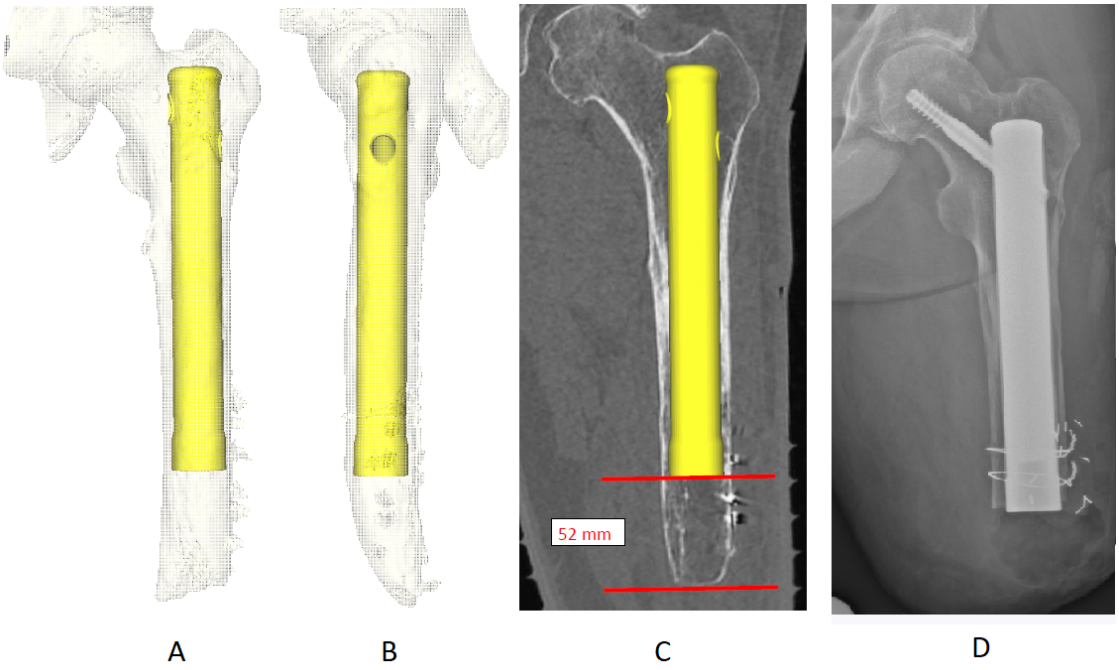


FIGURE 1.3: Planning of patient-specific femoral implant, with a frontal view of the femur in A and a lateral view in B. In image C the implant's position is shown in a x-ray image of the patient. A cylindrical hole, through which a locking screw is inserted during surgery, is designed such that the locking screw is placed in the centre of the femoral neck and head. Also shown in image C is the required shortening how much of the femur before the implant is inserted. The position of the implant and locking screw after the first surgery are shown in D.

Patients with short residual femurs (shorter than 160 mm) receive a patient-specifically designed implant, see Figures 1.2 and 1.3. These implants are designed according to the patients anatomy based on a lower-extremity Computer Tomography (CT) scan. The design differs from the OPL, because the implants include a cylindrical hole through which a locking screw can be placed. The implants are made of 3D printed titanium, including a 1 mm macro-porous 3D mesh coating for osseointegration. The mesh increases the surface area for osseointegration. [16] The implants' primary stability is ensured with the locking screw which prevents rotation of the implant inside the femur. The locking screw is also intended to prevent femoral neck fractures and stress risers (places where stress lines from the applied forces concentrate in the implant and femur). As seen in Figure 1.2a), the locking screw is positioned in the head and neck of the femur, similar to lag screws used to fixate femoral neck fractures. [17]

1.2 Accurately placing the implant and locking screw

During the first stage surgery, fluoroscopy is used as an intraoperative imaging technique to determine the correct way to position the implant. For custom-made implants in patients with a short residual femur, the rotation of the implant in the femur dictates the direction in which the locking screw will be placed in the femoral neck and head. The patient-specific implant is designed such that the locking screw will be placed in the center of the femoral neck and will not protrude through the femoral head (Figure 1.3).

The orientation of the implant after placement in the femur is essential for a good position of the locking screw. Additionally, the correct proximal/distal position of the implant in the medullary canal is important for the final position of the screw.

In some cases, the femur is shortened to the length of the implant before insertion, as shown in Figure 1.3c. This is carefully measured, as too much shortening of the femur could lead to a wrong implant position in the proximal femur. Subsequently, the femur is reamed in steps, with reamers increasing in diameter. To determine the correct rotation, a hand-held aiming device is used to determine where the locking screw will be positioned with different implant rotations. A mark is drawn on the femoral cortex that will be aligned with a marked line on the implant, thus indicating how the implant needs to be positioned in the femur. The implant is then inserted by hammering to achieve a press-fit stabilisation. The only means of validating the depth of insertion and rotation of the implant during insertion is fluoroscopy. A mobile C-arm in the Operating Room (OR) is used to make two-dimensional (2D) images of the surgical area. To examine the actual position and rotation of the implant in the femur during insertion, different viewing directions are necessary. This requires constant repositioning of the C-arm in the operating area, because the C-arm is rotated between anterior-posterior and lateral-medial viewing directions.

The fixation of the implant with the locking screw is also done with radiographic guidance. Initially, a Kirschner wire (k-wire), a guide wire for the drill, is placed through the cylindrical locking hole and predrilling takes place before the locking screw is inserted.

The major limitation of fluoroscopy is that it delivers two-dimensional projection images. If the holes in the implant are do not show up as circular, an adjustment to the C-arm position or beam direction is necessary (Figure 1.4b). This way, making a good projection image for guidance during the placement of the locking screw can be time-consuming. Furthermore, the total amount of fluoroscopy needed during the surgery could can up to a significant radiation dose for the patient and the surgical team. [18–26]

1.3 Alternatives to fluoroscopy

To find the correct entry position and direction for placing a locking screw, in this case to fixate the patient-specific implant, is not a new issue in the field of orthopaedic trauma surgery. For

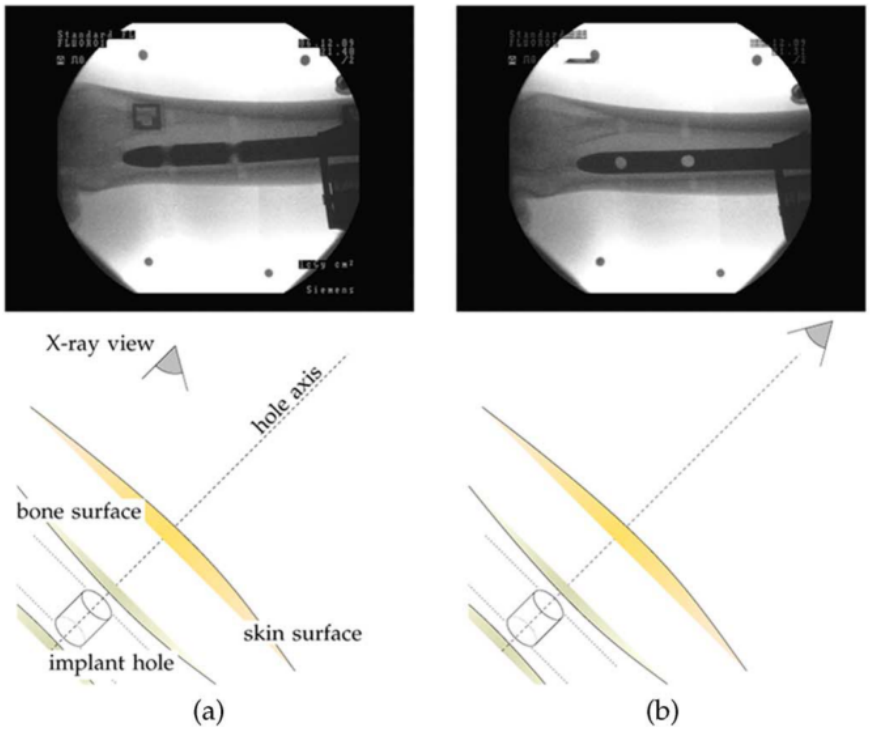


FIGURE 1.4: Alignment of the fluoroscope to the locking holes of an intramedullary nail. In (a) the x-ray view is not aligned to the hole axis. In (b) the fluoroscope has been aligned to the hole axis, which displays the holes as perfect circles. Down-the-beam positioning of the drill can now be commenced. Images taken from Diotte et al. [21]

instance, this is encountered in the locking of intramedullary nails, which are used to stabilize femoral or tibial fractures without opening the fracture site. An example of an intramedullary nail can be seen in Figure 1.4. Similarly, the locking of gamma nails for the stabilization of proximal femoral fractures is done with a lag screw inserted in the head of the femur (see Figure 1.5).

The ideal position of the lag screw discussed in literature is dependent on the tip-to-apex distance (TAD) in the femoral head and the position of the screw on lateral and anterior-posterior radiographic images. Risk factors for screw cut-out are a TAD $>$ 25 mm and a position off the center of the femur when viewed in different directions. [17, 27]

Conventionally, fluoroscopy is used to determine the position of the locking holes. The mobile C-arm must be accurately aligned with the locking holes, which should appear as perfectly circular holes. However, there are other techniques available for guidance during the placement of locking screws.

1.3.1 Different guidance methods in literature

Many different approaches to simplify distal locking of intramedullary nails are proposed in literature. [18–26] Different targeting devices are described to replace fluoroscopic free-hand targeting. A few examples of these are hand-held guides, fluoroscope-mounted targeting devices, nail-mounted guides and self-locking nailing systems. [18, 26] Each of these devices comes with their own problems and inaccuracies. [18] More recently, research has focused on surgical navigation technology, video-assisted fluoroscopy and augmented reality. [20–24, 26, 28] Commercially available electromagnetic navigation systems exist that aid the surgeon in positioning the drill correctly. [20, 26] However, these require the use of special intramedullary nails to allow guidance in an electromagnetic field. A study by Lilly et al. reported the use of another

type of computer-assisted navigation system. [17] This software is used to determine the tip-to-apex distance of the lag screw in the neck and head of the femur, shown as a projection on the standard fluoroscopic images (Figure 1.5). Leung et al. have described a similar application of fluoronavigation for different trauma procedures, including treatment of femoral neck, acetabulum and sacroiliac fractures. [19]

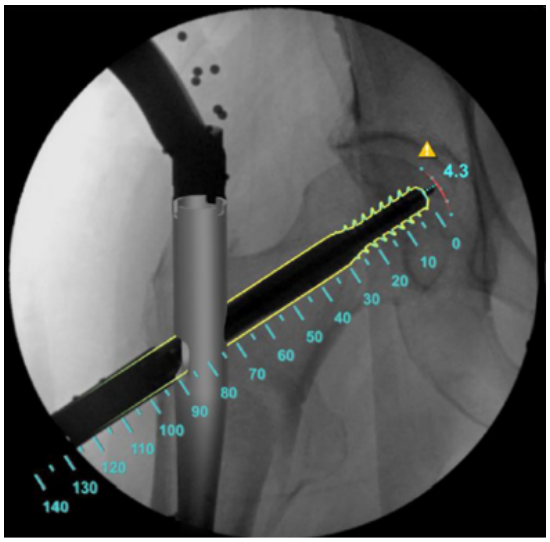


FIGURE 1.5: The navigation software projects information on the live fluoroscopic images. The nail and screw are visualized and a tip-to-apex distance (TAD) can be calculated. Image taken from Lilly et al. [17].

1.3.2 Surgical Navigation Technology (SNT)

Surgical navigation systems offer real-time image guidance during surgery by visualizing virtual models of the instruments in relation to the patient's anatomy, in e.g. preoperative planning images. [29,30] By means of optical or electromagnetic tracking, the position of instruments can be registered to preoperative or intraoperative image data and shown in different image planes simultaneously. [19] SNT can thus aid the surgeon in positioning instruments or in following a predefined trajectory, without radiation exposure to the patients and surgical team. [31]

In the field of surgical navigation, there are two methods of tracking: optical tracking with infra-red cameras and electromagnetic tracking with an induced magnetic field in the operating area. Electromagnetic tracking can be impaired by ferromagnetic (iron, cobalt) or conductive (titanium, stainless steel) metal objects that distort the magnetic field and is not a suitable technique to be used during the insertion of implants and screws.

Optical tracking is facilitated by attaching reflective markers/spheres to the patient and instruments. A stereotactic camera can determine the position of these object in three-dimensional (3D) space by receiving infra-red light reflected by the markers.

In the Radboud University Medical Centre (Radboudumc), an example of an available SNT system is the Brainlab Curve navigation system (Brainlab AG, Munich, Germany). The accompanying software is designed for guidance of different procedures, e.g. guidance during neurosurgery.

1.4 Specific software application

Using an existing navigational system in the OR is a possible alternative for the time-consuming use of fluoroscopy. The hardware of the system, i.e. the stereotactic camera and the instruments that are tracked, could be used during the placement of the implant and locking screw. However,

there is no specific software application designed for guidance during this procedure. There are different visualizations possible of which an example is shown in Figure 1.5. An application could offer projected information on fluoroscopic images or show a three-dimensional model of the instruments and the patient's anatomy. In the design of specific SNT software, the particular limitations which are caused by using fluoroscopy, e.g. the 2D visualization of cylindrical holes, may be overcome. Ideally, the use of surgical navigation decreases the time needed for locking screw placement and dismisses the need of using radiation for imaging. And last but not least, it should help the surgeon in carrying out the preoperative planning accurately and intuitively.

1.5 Research question and aims

In this study, the possible use of surgical navigation technology to overcome the described limitations will be investigated.The main research question that is answered in this thesis is:
How can the insertion of patient-specific osseointegration implants in short residual femurs and the fixation with locking screws be improved and simplified with the application of surgical navigation technology? Can this lead to a decreased radiation exposure to the patient and surgical team, a reduction of surgery time and more intuitive surgery?

Three sub-questions were formulated:

1. What are the requirements for the SNT software and hardware to provide guidance for determining the correct amputation level in case of femur shortening, for the correct rotation and depth of the implant in the medullary canal and for placing the locking screw?
2. How should the surgical setting be visualized in the navigational software to optimally assist the surgeon with placing the implant and locking screw?
3. How well does the postoperative position of the implant match the preoperatively planned position with guidance of the surgical navigation software?

1.5.1 Design and create a software application

The first aim is to design and develop a surgical navigation application for image-guided insertion of the implant and locking screw. An ex vivo experimental set-up is necessary to test the navigational software during development. Certain hardware is needed, including a stereotactic camera system, a patient phantom and instruments used during surgery. A list of requirements, the design process and the outcomes of sub-question 1 are reported in Chapter 2.

1.5.2 Test the use of different visualisations

In addition to the algorithm facilitating tracking during surgery, a virtual visualization of the surgical area and a User Interface (UI) need to be designed to guides the surgeon. Different visualization approaches were developed and user tests were performed to determine which visualization offers the best guidance. The visualizations and user tests are described in Chapter 3.

1.5.3 Test the accuracy of the developed software

Finally, to answer sub-question 3, an experiment is designed to determine the precision with which the implant can be positioned. Furthermore, the experiment is used to validate the measures the software calculates, e.g. the distance between the planned position and actual position of the implant. The results are presented in Chapter 4.

CHAPTER 2

Software Design

2.1 List of requirements

At the start of developing a specific software solution for placing the implant and locking screw in the OR by optical tracking, a list of requirements was composed. It is divided in two sections. First, the "material requirements": the necessary tools that were needed to develop the software. The second section named "software requirements" lists the conditions that the developed navigation application needs to meet. The list of requirements contains a few larger necessities (I - XI) that are complemented with sub-requirements ((a) etc.).

Material requirements:

I A stereotactic infra-red camera.

- (a) The camera is able to track objects and measure their position and orientation in three dimensions based on infra-red tracking.
- (b) The camera is able to recognize different objects simultaneously.
- (c) The camera can forward the measured information to the software application in real time.
- (d) The camera tracking is accurate enough, i.e. differences in position and orientation that are clinically relevant can be measured.
- (e) The camera can be used in the OR.

II A program used to create a visualization of the surgical scene.

- (a) The program in which the visualization is built should be able to create 2D and 3D visualizations.
- (b) The program is capable of real-time updating the visualization according to the sent information from the camera.

III Experimental set-up.

- (a) The use of objects, such as a femur model and surgical instruments, that represent the objects in the surgical setting.

(b) Reflective markers that allow tracking of these objects.

(c) Virtual models of the used objects to be used in the visualization of the surgical scene.

Software requirements:

IV A method to send information from the camera to the software.

V Registration between the Camera Coordinate System (CCS) and the visualization coordinate system.

VI Different visualizations of the surgical area can be designed.

(a) The visualizations can be switched on the screen or possibly all shown together.

VII The possibility to load a surgical plan in the developed software.

VIII A function that calculates the difference – in distance and in orientation – between the tracked objects' and the planned objects' position.

IX The software should provide feedback on how close the user is to reaching the planned position.

X The software should provide guidance during reaming of the femur and during pre-drilling before insertion of the locking screw.

XI The software can be run from a computer in the OR.

2.2 Material requirements

Before development of the software started, different choices needed to be made, determining which materials and software would be used during the internship. Keeping in mind that all the demands on the list of requirements needed to be met, different options were considered. The following section describes which options were present, what the considerations were and why an option was chosen.

2.2.1 A stereotactic infra-red camera

Two different camera systems were considered: the PST Base (PS-Tech, Amsterdam) and the Brainlab Curve camera system (Brainlab AG, Munich, Germany) in the Medical Innovation and Technology expert Center (MITeC) OR in the Radboudumc hospital (Figure 2.1). Both of the cameras have an accuracy that meets the clinical demands. [32, 33] The advantage of using the Brainlab camera system over the PST camera is that it has clinically been approved for use in the OR. The PST Base has other advantages, particularly that a connection between any computer and the camera is free and consists of plugging a cable into the computer and running the accompanying free software (PST Client). The Brainlab camera system is closed, i.e. it can only be used with the Brainlab software on the Brainlab system. A connection between the Brainlab camera and a personal computer is possible with the Open Network Interface for Image-Guided Therapy (OpenIGTlink) communication protocol, for which Brainlab charges a price.

Other considerations gave the PST Base the advantage over Brainlab. Firstly, the possibility to track any object defined in the PST Client, in contrast to tracking Brainlab's own instruments exclusively. Secondly, the PST Base system can be used in a non-clinical environment and experimental set-up, which is easier in the developmental stage of testing a software application.

FIGURE 2.1: The Brainlab Curve camera (left) and the PST Base camera (right).

The PST Base camera is the chosen hardware for this project. During the development of the software and experimental set-up, it was kept in mind that at a later stage the software could be connected to the Brainlab system in the OR as well. In a further stage of testing, it may be desirable to switch to the Brainlab system in the MITeC OR.

2.2.2 A program used to create a visualization of the surgical scene

After choosing the camera system, the next step was to determine how and with which programs the navigation software would be developed. Evidently, this program should run well with the PST camera. The considered alternatives were:

- MATLAB® (The MathWorks, Inc., Natick, Massachusetts, USA).
- Unity (Unity Technologies ApS, Copenhagen, Denmark).
- The C++ programming language with an appropriate editor in which to write the source code.

Requirements II(a) and (b) should be met, but IV to X are also influenced by the choice of program. Matlab is capable of creating a 3D visualization. However, based on previous experience, it was doubted whether Matlab would be able to build the visualization and update it smoothly in 3D. Also, there was no previous experience combining the PST Base and Matlab. In previous projects in which the PST Base camera was used in the Radboudumc, Unity was used to build a visualization. Unity is a game engine and thus primarily used to build 2D or 3D games with which a player can interact real-time. A programmer can use either of the programming languages C Sharp (C#) and Java to write scripts that influence the objects in the game. It has some built-in functions, e.g. to transform a rotation from the quaternion notation to Euler angles. Additionally, packages can be downloaded from the 'Asset store' for more specific functionalities. Unity is capable of meeting all of the requirements involved and was used in combination with the PST camera before. All of this is also possible in C++, but the advantage of Unity over C++ is that Unity offers the programmer tools to build a visualization. Considering all of the above, the program that was chosen to develop the navigation software and visualizations of the objects is Unity (version 2017.3.1f1).

2.2.3 Experimental set-up

With the PST Base camera, any object with a minimum of four reflective markers or reflective stickers can be trained in the PST Client software and tracked with the camera. This presented

the opportunity to use any object to represent the surgical situation. Training in the PST Client software requires the user to hold the object with markers in front of the camera and move it slightly around. From this recorded movement, the specific configuration of the markers is saved in camera coordinates and the object is given a name and an id-code.

The actual objects in the set-up need to be represented in the virtual visualization of the navigation application. Ideally, The movement of the object that is tracked by the camera is displayed on the screen with a virtual model of the object. This is an important requirement and leads to 3D-printing as a proposed production method of the physical objects. Mainly because a virtual three-dimensional design of the object is necessary before 3D-printing of the model is possible. This design can be used in Unity to represent the physical objects in the experimental set-up.

Designing the objects was done in Solidworks (Dassault Systèmes SOLIDWORKS Corp., Waltham, MA, USA), 3Ds Max (Autodesk Inc, San Rafael, CA, USA) and Meshmixer (Autodesk Inc, San Rafael, CA, USA). The accuracy with which a virtual model is produced by a 3D-printer depends on the printer-model. The printing company (Oceanz, Ede, The Netherlands) that produced the models in this study uses Selective Laser Sintering (SLS), a powder-based fusion technique. A laser beam is used to sinter, or solidify, a polymer powder to build up an object layer by layer with a tolerance of 0.3 mm. [34]

The objects (Figure 2.2) that were designed and 3D-printed for the experimental set-up include:

- A femur model.
- An implant (and locking screw).
- Reference stars (with unique configurations) to attach a minimum of four reflective markers to the objects.
- A pointer instrument to indicate different location on the models for registration purposes.
- An instrument that resembles the drill used in surgery.

Design of the models

The femur and implant model were created from an existing design for a patient-specific implant. The bone was segmented from the Computer Tomography (CT) scan and modified with 3Ds Max to create a smooth surface. The femur model was shortened with 3 cm, according to the surgical plan. Also, the implant dimensions were used to excavate the femur model. The created cavity simulates the reamed medullary canal and fits the implant with a diameter of 21 mm (including the 1mm macro-porous mesh on the surface of the implant). The implant design is by L. Verhamme of the 3D Lab in the Radboudumc.

The femur model needs to be tracked by the camera. The easiest solution for the initial experimental set-up was to attach four reflective stickers to the model, as seen in Figure 2.2, number 5.

For registration purposes, described in detail in section 2.3.2, seven pits were designed on the surface of the femur model. These pits are non-anatomical landmarks created to facilitate point-based registration. Point-based registration is used to align two datasets of the same landmarks in different coordinate systems. In case of the landmarks on the femur model, one dataset consists of the virtual positions in Unity and the other contains the landmark positions on the physical femur model, which are indicated by a pointer tool (Figure 2.2, number 3).

The pointer is thus a tool used to indicate the landmarks on the femur model. For the experimental set-up used during development of the navigation software, any pointer that can be tracked would have sufficed. However, with future research in mind, the design of the

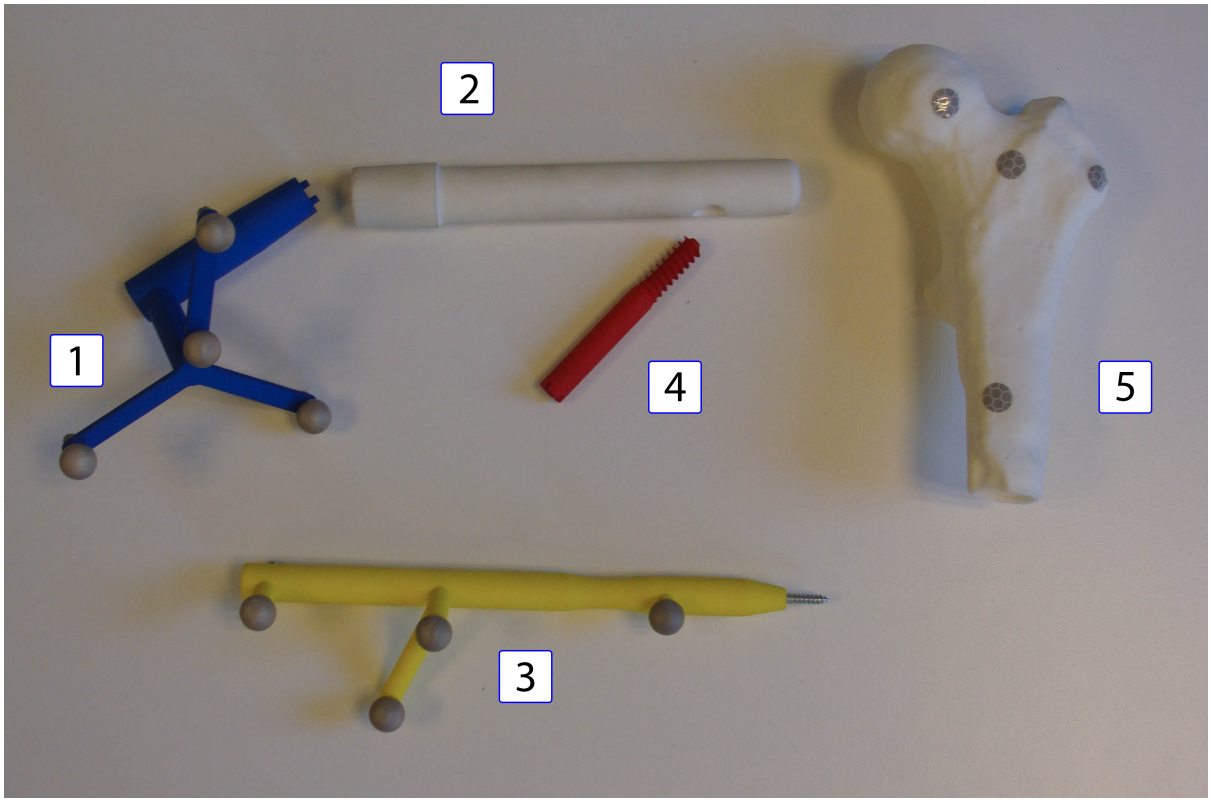


FIGURE 2.2: 3D-printed models used during the development of the software. Indicated by (1) is the Dual Cone Adapter (DCA) that can be attached to the implant (2) to facilitate tracking. A model of the locking screw inserted through the cylindrical hole in the implant is denoted with (4). The pointer (3) is used to register the femur model (5) by indicating non-anatomical landmarks on the surface.

pointer is based on the design of the Brainlab pointer. In a more clinical test setting, such as the MITeC OR in the Radboudumc, the designed pointer model would not need to be adapted to enable tracking with the Brainlab system. It has the same marker configuration as the Brainlab pointer, only with an additional fourth marker that the PST camera requires. This fourth marker can be removed if the Brainlab camera is used. To determine the exact positions of the reflective markers with respect to each other, a scan of the Brainlab instrument was made and the reflective markers were segmented to find their position. With these positions, a pointer model was designed in Solidworks and 3D-printed. At the tip of the pointer a threaded opening is made, in which a tip can be screwed. For this study, a screw is inserted in the opening, which functions as the tip of the pointer tool (visible in Figure 2.2).

Similarly to the pointer, the design of the Dual Cone Adapter (DCA) that is attached to the implant has the configuration of a Brainlab reference star. The additional fourth marker is put on one of the bars of the star and can be removed for future use with the Brainlab camera. The DCA is attached to the connector of the implant with the two pins that fit the holes in the inside of the implant. This is not a unique attachment, considering the fact that the reference star can point upwards or downwards if the DCA is rotated 180° with respect to the implant.

2.3 Software requirements

In the previous section, the used materials and design of the experimental set-up are described. This section elaborates on the design of different functionalities in the navigation software that is built in Unity. An overview of Unity terminology mentioned in the following sections, highlighted in *italic*, is provided in Appendix A.

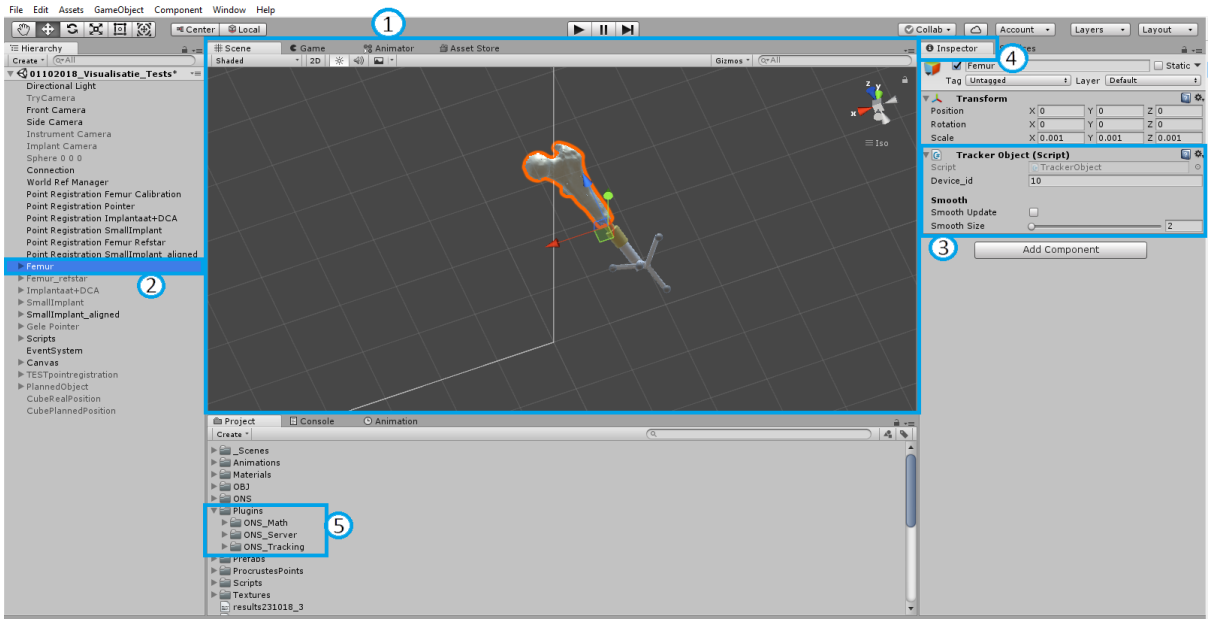


FIGURE 2.3: The interface of Unity is displayed. Denoted with (1) is the *scene*, a three-dimensional space in which the project, or game, is built. Different *GameObjects*, such as *Camera* objects and the virtual model of the femur (2), are added to the scene. The position and orientation of an object can be changed in the *Inspector* window (4), as well as the different scripts and components linked to the object (3). In the *Project window*, all assets that a project consists of are displayed. These could be described as the resources (different components, such as scripts, colours, textures) that can be added to a GameObject. Highlighted with (5) is the package of scripts that is needed to connect to the Optical Navigation Server (ONS). The different scripts, e.g. "Tracker Object" (3) are linked to different GameObjects, which is also illustrated in Figure 2.5 and 2.9.

Unity is a game engine that can build 2D or 3D games. A *project*, or game, is built in a *scene* (indicated in Figure 2.3 by number 1). The scene is a three-dimensional space with its own coordinate system, in which *GameObjects* (Figure 2.3, number 2) can be created and moved around. This gives Unity an object-oriented programming approach. This means that each GameObject can be given certain properties and that the game is based on the functionalities, or *components* (Figure 2.3, number 3), that are attached to the different GameObjects. Some properties, like the appearance, can be edited by attaching standard components (e.g. a *Mesh Renderer*) to the GameObject.

In the *Inspector* window in Unity (Figure 2.3, number 4), the component values can be adjusted. Besides the standard components, self-developed scripts can be attached. The scripts are written in the C# language and add a certain behaviour or function to the GameObject that the script is attached to. The following sections explain how the different functionalities described in the list of requirementes are designed and built in Unity.

2.3.1 Sending information from the camera to Unity

A server called Optical Navigation Server (ONS) was built by J. Duits of the 3D Lab (the version used in this project is 0.1.1). It receives information about the position (${}^{CCS}\mathbf{p}_i = (x_i, y_i, z_i)$, with i the object id-code) and rotation (${}^{CCS}\mathbf{q}_i = (w_i, x_i, y_i, z_i)$) of objects that are tracked by the PST camera and sends it to Unity. As part of the ONS, the project in Unity must contain several script packages (see Figure 2.3, number 5). This *plug-in* contains a "Connection" script, for example. This Connection script procures access to the server with a given server address and port. Additionally to adding the package of scripts to the project folder, some of these scripts must also be attached to GameObjects.

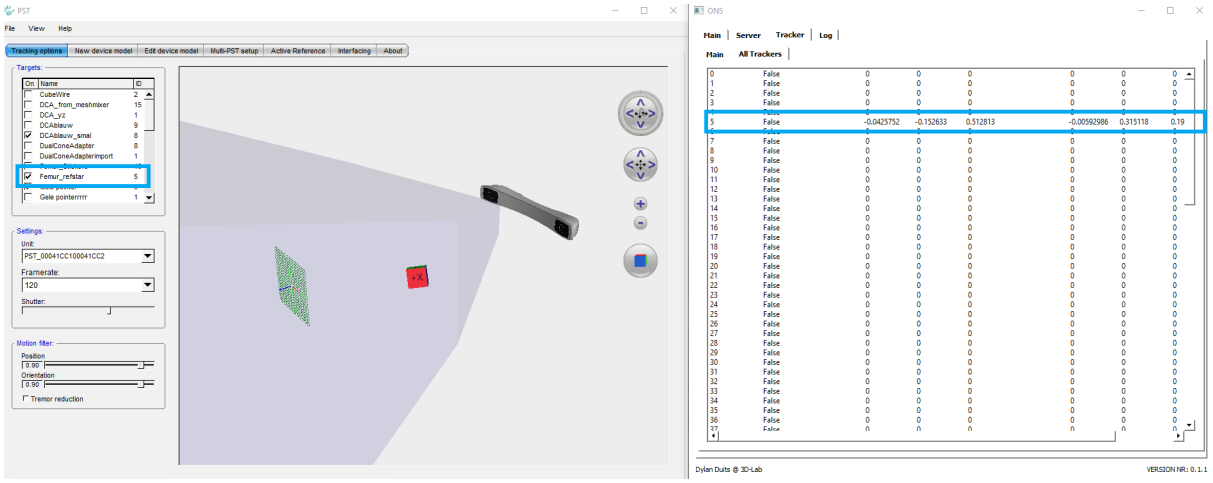


FIGURE 2.4: Two screen-shots are displayed, with the PST Client software on the left and the interface of the ONS on the right. On the left, the tracked object is shows as a cube in the field of view of the camera. Only objects which are checked in the left window are tracked. Highlighted with blue is the femur object, with id-code 5. This code corresponds to the code with which the server sends information about position and orientation, which is highlighted in the right screen-shot.

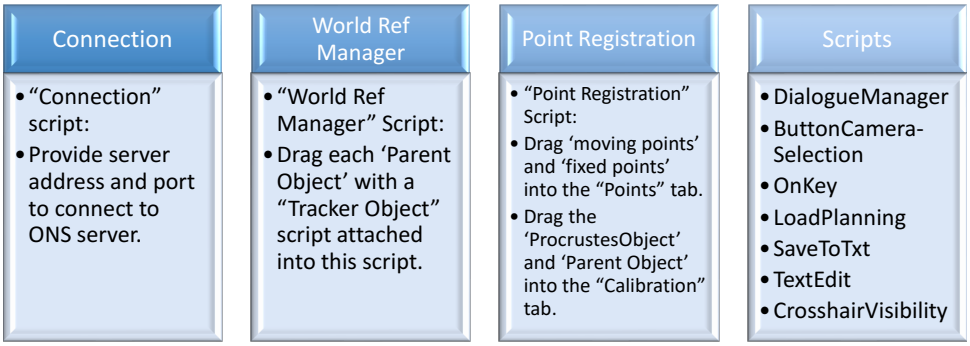


FIGURE 2.5: Different GameObjects that are created in Unity to enable a specific function. The GameObjects do not feature a visible object in the scene and are therefore called 'empty' GameObjects. However, due to the attached scripts, the objects provide certain functionalities. The Connection object gains access to the ONS server. In the World Ref Manager, the different GameObjects that are tracked by the camera are defined. The Point Registration object carries out the registration between the landmark positions in two coordinate systems (the moving points in Unity coordinates and the fixed points in camera coordinates). To add other functionalities that are not specifically attached to a visible model or other GameObject, the empty Scripts object is created.

An attached script is shown as a component in the Inspector window of the object, visible in Figure 2.3, number 3. If a script changes the properties of other GameObjects, it may need information about which GameObjects are referenced in the script. To specify which GameObjects are influenced by a script, they can be dragged from the project onto the script. Examples of different GameObjects used in this project and their attached scripts are presented in Figure 2.5.

In the PST Client software, each object is given an id-code. If the object is selected in the interface, it can be tracked by the camera and the position in the 3D world is shown in a visualization of the field of view of the camera. The server makes use of these id-codes. For an image of the PST and ONS software, see Figure 2.4. The positions and orientations of a "Tracker" with id "5" are sent to the Unity project. To move a GameObject in Unity according

to the sent information, this GameObject should have a "Tracker Object " (see Figure 2.3, number 3) script attached to it, one of the elements of the ONS plug-in in the project. In this component, the id-code should match the id-code of the object in the PST Client. Also, the GameObject with attached Tracker Object script should be added onto the "World Ref Manager" component (Figure 2.5).

2.3.2 Registration method

The objects that are tracked by the PST camera are given a position ($\mathbf{p}_i$) and orientation ($\mathbf{q}_i$) in the Camera Coordinate System (CCS). However, the virtual objects in the Unity visualization are positioned and oriented according to the Unity Coordinate System (UCS) (Figure 2.6). When an object is tracked, the measured camera coordinates are sent to Unity. The objects in Unity therefore need to move as if they are in the CCS. To transform the Unity coordinates to the CCS, a registration algorithm is necessary. A registration can be performed with two datasets of the same objects or positions in both coordinate systems. E.g. registration of a reference star attached to the implant is done by calculating a transformation based on the markers' configuration in the PST software (${}^{CCS}\mathbf{A}_{m\times n}$, with m the number of markers (4) and $n = (x, y, z)$) and the position of the markers in Unity (${}^{UCS}\mathbf{B}_{m\times n}$ with $m = 4$ and $n = 3$). With the Procrustes algorithm, see equation (2.1), a registration is performed that gives the approximated positions ${}^{CCS}\hat{\mathbf{A}}_{m\times n}$ of the reference star in camera coordinates that can be used in the Unity scene.

$$ {}^{CCS}\hat{\mathbf{A}}_{m\times n} = S \cdot {}^{CCS}\hat{\mathbf{R}}_{UCS} \cdot {}^{UCS}\mathbf{B}_{m\times n} + {}^{CCS}\hat{\mathbf{t}}_{UCS} \tag{2.1} $$

Procrustes estimates a scale, S, a rotation matrix, ${}^{CCS}\hat{\mathbf{R}}_{UCS}$, and a translation vector ${}^{CCS}\hat{\mathbf{t}}_{UCS}$. The accuracy of the registration can be expressed in the Target Registration Error (TRE) (equation (2.2)) and the Root Mean Square (RMS) (equation (2.3)).

$$ \mathbf{TRE}_m = {}^{CCS}\mathbf{A}_m - {}^{CCS}\hat{\mathbf{A}}_m, \tag{2.2} $$

with m the marker number. The RMS is a sum of the squared values of the individual TRE. It expresses how well the registration succeeded.

$$ RMS = \sqrt{\frac{1}{m}\sum_{m=1}^{m} \|\mathbf{TRE}\|_m^2} \tag{2.3} $$

with $\|\mathbf{TRE}\|_m^2$ the squared magnitude of the displacement vector.

For all of the objects that are tracked with the PST camera, a registration is performed in Unity. This is done with a "Point Registration" script for the pointer, femur model, implant reference star and drill (Figure 2.7). For the pointer and implant reference star, a registration is calculated with the positions of the four markers as Unity GameObjects and the configuration of the four markers in PST coordinates. The femur model, as shown in Figure 2.2, is tracked with stickers. Registration is therefore done with use of the pointer tip.

The femur model was designed with seven pits in the surface. The coordinates of these pits in Unity are known. These are called the moving points, or ${}^{UCS}\mathbf{B}_{7\times 3}$. The pointer, previously registered to camera coordinates, is positioned in each of the pits to indicate the fixed points, the positions of the pits in camera coordinates (${}^{CCS}\mathbf{A}_{7\times 3}$). An image is shown in Figure 2.7. The scene shows the difference between the pits indicated by the pointer and the pits' true position in Unity with light blue cubes. For each of the seven pits, a TRE is calculated. This can be seen in the Inspector window of the Point Registration script, as the euclidean Distance Between Points (DBP). The RMS is automatically calculated as well (the unit is meter).

The results of the pointer and implant registrations are constant, the manual registration of the femur model varies each time the registration is performed. The values of the TRE and RMS errors are presented in Table 2.1.

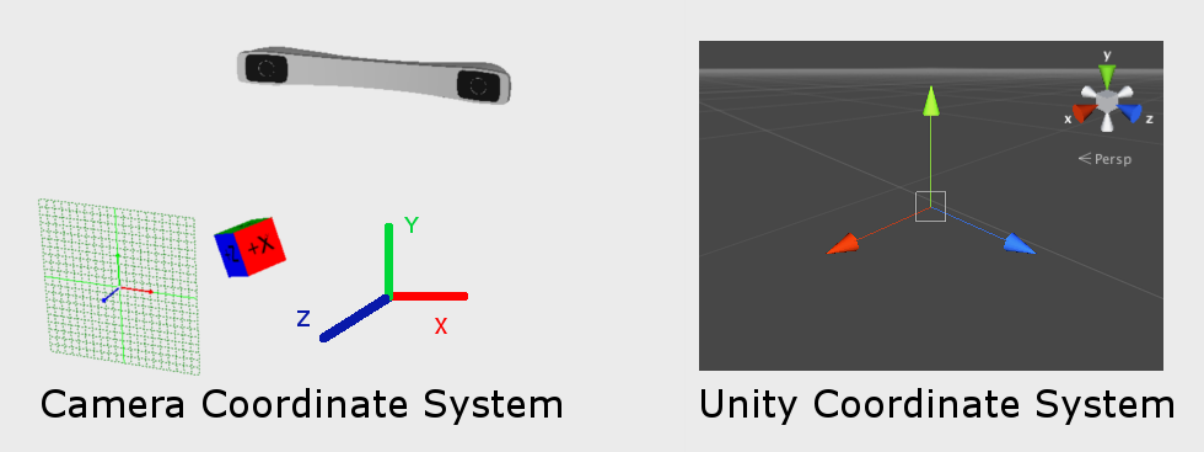


FIGURE 2.6: The differences between the camera coordinate system and the Unity coordinate system. Unity has a left-handed coordinate system, with the y-axis indicating up and the z-axis indicating forward. The camera coordinate system is right-handed. This caused some disarray in the first stages of development. By inverting the z-coordinates from the camera system, the camera coordinates were consistent with the Unity coordinate system.

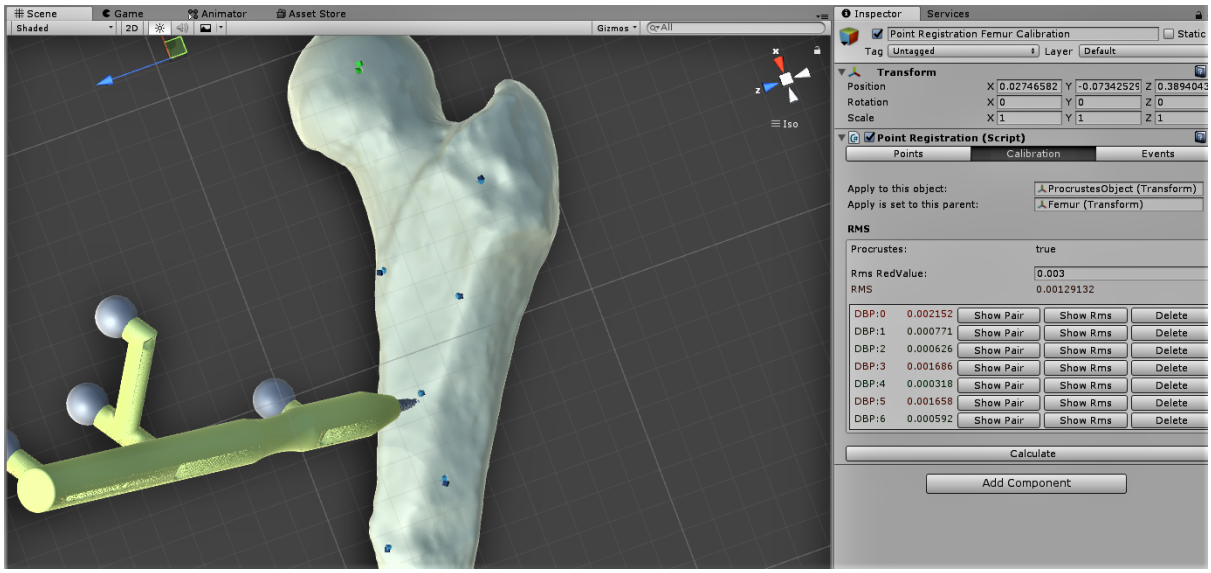


FIGURE 2.7: Registration of the femur from Unity to the CCS is done by pointing at the 7 pits on the physical object with the pointer (visible with dark blue cubes) and calculating the transformation between the two datasets – the indicated points and the positions of the pits in the Unity GameObject (light blue cubes)– with the Procrustes algorithm.

TABLE 2.1: The calculated target registration errors (in mm) and the root mean square error of registration for each model. The femur parameters vary for each registration, because of manually indicated points.

Model	**TRE (mm)**	**RMS (mm)**	**Model**	**TRE (mm)**	**RMS (mm)**	**Model**	**TRE (mm)**	**RMS (mm)**
Implant	1.034	0.876	Pointer	1.269	0.773	Femur	2.152	1.291
	0.186			0.693			0.771	
	0.909			0.336			0.626	
	1.068			0.429			1.686	
							0.318	
							1.658	
							0.592	

2.3.3 Different camera views

Different visualizations of the surgical setting were designed. These are designed in Unity in the following manner. A *Camera* object in Unity is a specific GameObject that determines from which viewing point the scene is rendered. As explained before, the scene is a 3D space in which the game is developed. However, the scene is not what the user sees in the actual game. It depends on where the camera GameObjects are positioned in that scene.

For instance, a camera has a specified field of view and a certain position towards objects in the UCS. Different cameras viewing different sides of the objects can be switched on and off, or shown simultaneously. With a script, the behaviour of the camera can be altered additionally. For example, it can be programmed to follow an object while it moves in the scene.

For this project, five different camera functionalities were developed. These are the "Front Camera", a "Side Camera", an "Implant Camera", TryCamera and a "Crosshair Camera". An overview is provided in Figure 2.8a. The design of the TryCamera for the visualization tests is further described in Chapter 3 and the Crosshair Camera is developed for the validation test of which the details can be found in Chapter 4.

The Front Camera gives the user a 3D anterior-posterior view of the surgical scene. The camera position and orientation are fixed on the virtual femur model. When the physical femur model is moved in space, the virtual femur model moves accordingly. However, the camera object is programmed to follow the movement and orientation of the femur and therefore moves along with the virtual model. This means that the user always views the virtual scene in the same way. The Front Camera view is intended as a reference view, or overview of the surgical scene. This functionality is added by the "SmoothFollow" script.

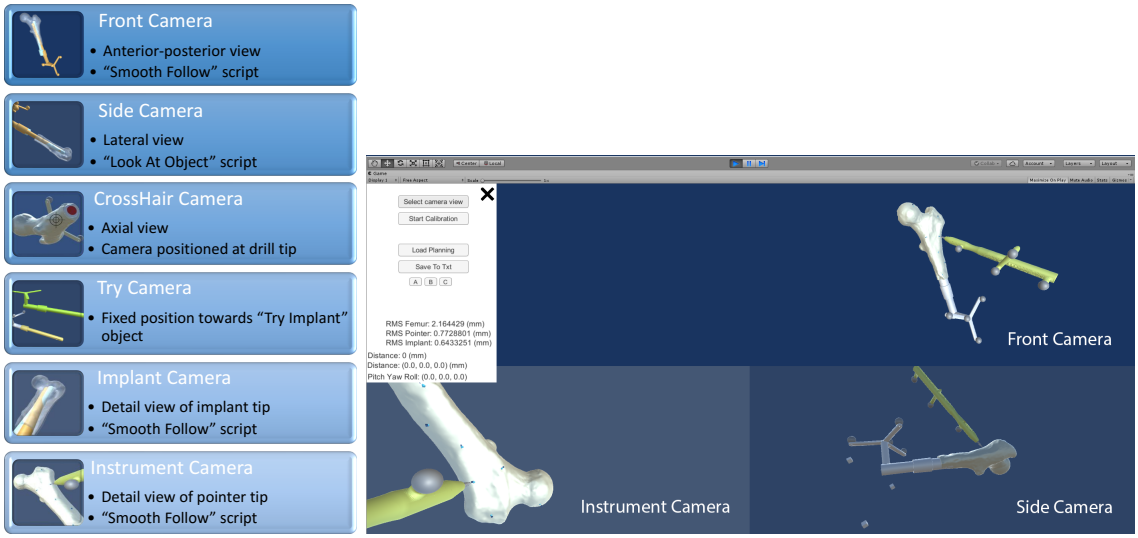
The Side Camera is designed similarly to the Front Camera. It is meant to provide a lateral view of the virtual surgical scene. A script "LookAtObject " is attached to the Side Camera GameObject which makes sure it always positioned sideways of the femur model. If the position of the virtual femur model is updated, the position of the camera is updated as well. To illustrate the difference with the behaviour of the Front Camera: the femur model can be rotated in the physical world and the Side Camera would remain at a fixed position towards the femur, but would show the rotation. Whereas, the Front Camera also rotates with a femur rotation and keeps the same view of the femur. The Side Camera functionality is added with the "LookAtObject " script.

The Implant and Instrument Camera GameObjects follow the implant or instrument objects when they move in the scene. The same script applied to the Front Camera is applied here. The target is not the femur model, but the tip of the implant. This means that the Implant Camera is always positioned and rotated in the same way with respect to the implant tip. This relative position is specified as a distance value (the distance between the implant's and camera's position in the z-axis) and a height value (the difference between the implant's and camera's position in Unity y-coordinates). The Implant Camera is intended as a detailed view of the surgical scene, used to determine the position of the implant in the bone or look at the implant tip while inserting the locking screw. The script can also be attached to the pointer, to follow the pointer tip during registration of the femur model (see Figure 2.8b).

While running the project, a menu with different buttons and text is displayed to give the user the ability to tweak the virtual surgical scene during different stages of surgery. The button "Select camera view"allows the user to see all active cameras and switch to a full-screen view of one camera by clicking on the part of the screen where that particular camera view is shown. This is illustrated in Figure 2.8b.

2.3.4 Loading a surgical planning

Requirement VII in section 2.1 states that the developed software must be able to load a surgical planning. The planning consists of the design of the patient-specific femoral implant and its



(A) All cameras (B) Switch views

FIGURE 2.8: Different camera views are illustrated. In (A) all the different camera viewpoints are presented. The Try Camera was specifically used in the visualization tests (chapter 3) and the design of the Crosshair camera for the validation tests is reported in chapter 4. In (B), the functionality that allows the user to switch from viewpoint is displayed. By clicking on the button 'Select camera view' (shown in the menu on the left), the three active cameras in the scene are shown. By clicking on the camera of choice, the full-screen rendering from that camera perspective is activated.

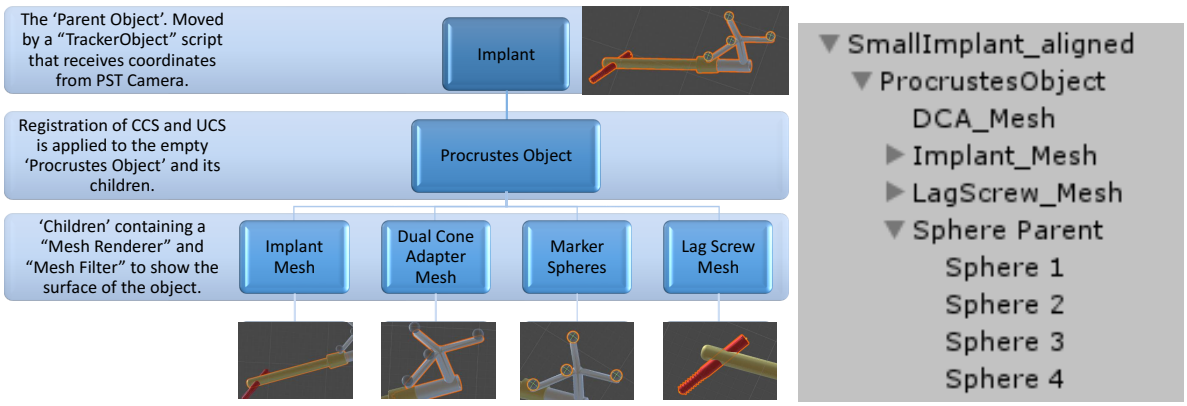
ideal position in the femur. This is linked to the ideal position of the locking screw in the femoral head. Additionally, the surgical planning may contain information about the necessity of shortening the femur before implantation.

To provide guidance during surgery, the planned position of the implant in the femur must be displayed along with the tracked implant and the user should be able to interpret the difference between the actual position of the implant and the ideal, planned position. In Unity, the virtual implant objects have the same dimensions, which enables a comparison between the two GameObjects to determine how close the user reaches the planned position. The GameObjects are assembled in a specific manner, explained below and demonstrated in Figure 2.9.

Parent-child hierarchy in Unity

The *parent* object, Implant in Figure 2.9a, is an empty GameObject given a position and orientation in Unity according to the tracked physical implant model. All the underlying GameObjects, the *children* move along with the parent. To explain the parent-child relation, think of the children all being attached to the parent at a particular distance. When the parent moves, the children all move along with it but remain at their fixed position relative to the parent. To illustrate this in the case of the implant object, the objects that make up the virtual implant model, i.e. the implant mesh, the dual cone adapter mesh, the marker spheres and lag screw mesh are all set at a particular position relative to the parent to put the implant model together. It is the parent however, that is moved in the UCS in correlation to the movement of the tracked implant model.

The hierarchy explained above (visible in Figure 2.9b) is also applied to assemble the planned implant model. The same mesh objects are used. However, the planned implant position is fixed in relation to the femur model, i.e. the planned implant is a child of the femur model. Also, the planned implant is not tracked by the camera, therefore it does not have a 'ProcrustesObject'. Nonetheless, the planned implant object is a child of the ProcrustesObject of the Femur object. This is necessary, because the femur model is registered to camera coordinates and positioned



(A) Schedule of the the Implant model (B) GameObject hierarchy

FIGURE 2.9: Each virtual model in Unity is composed of different GameObjects. The different GameObjects put the virtual model together in Unity. (A): The parent object's position, Implant, is controlled by the position of the physical object that is tracked by the PST camera. The ProcrustesObject is given a certain position and orientation according to the transformation calculated by the Procrustes algorithm. This ensures that all its children move properly along with the parent object. In (B), the hierarchy of GameObjects in Unity is displayed.

according to camera coordinates. With the child-construction, it is ensured that the planned implant position in the femur remains the same, no matter the position of the femur model.

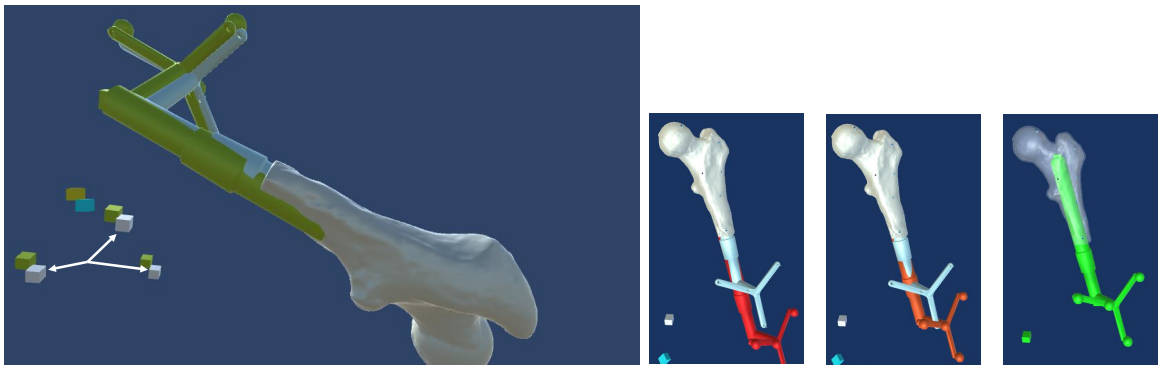
The menu that allows the user to interact with the software while running the project (seen in Figure 2.8b in the white window), has a button "Load Planning" to start navigation towards the surgical planning. The planned implant object is thus only activated and visible in the femur model when the user is ready to start navigation of the implant.

2.3.5 Calculation of the difference in position and orientation

In the previous paragraphs, the assembly of the virtual model of the tracked and planned implant are explained. The user is able to move the tracked implant and navigate towards the planned implant's position rendered in the virtual femur model. To provide the user with information about how close the implant is to reaching the planned position, different feedback mechanisms are developed and described in this section.

First of all, the distance between the two 'Implant Mesh' objects is calculated and displayed real-time in the menu. This is calculated with three GameObjects, standard cubical objects, that are instantiated when the button "Load Planning" is pressed. The cubes are shown in Figure 2.10a. The script "Load Planning" generates three cubes positioned in a range around the tracked implant mesh and the planned implant mesh object. With these three cubes, the distance between the two objects can be visualized and calculated.

Visualizing the difference in position and orientation is done by changing the colour of the tracked implant based on how close it is to the planned position. The distance between the three generated cubes is compared to three threshold values: "Far Margin", "Close Margin" and "Very Close Margin". If the distance between the implants is larger than the Far Margin (default value is 125 mm), the implant is coloured red. Between the Far Margin value and the Close Margin value (set to 10 mm), the colour is linearly interpolated between red and orange. If the distance is between the Close Margin and Very Close Margin (4 mm), the colour is somewhere between orange and green. Finally, for distance values below the Very Close Margin, the implant is coloured green. The different colours and distances are illustrated in Figure 2.10b.



(A) Three cubes (B) Colour interpolation

FIGURE 2.10: The screenshot in (A) shows how the tracked implant (in green) is close to the planned implant position (light blue). The white cubes indicated with the arrows are the GameObjects created around the planned implant position in Unity. The green cubes are generated in the same manner around the tracked implant. The cubes are used to determine the distance between the two objects in three directions. in (B), the visual feedback mechanism is demonstrated. In the left image, the distance is larger than the Far Margin, which turns the implant red. If the distance decreases, the colour is interpolated between red and orange (middle image). On the right, the distance has become smaller than the Very Close Margin, which turns the implant bright green.

In addition to visual feedback, two different distances are displayed in the menu, the euclidean distance (i.e the shortest straight-line distance between two objects) and the distance between the objects in each axis. Additionally, the difference in orientation is calculated as a "pitch", "yaw" and "roll" angle (unit degrees).

Distance calculation

If the positions of cube 1, 2 and 3 around the tracked implant in Unity coordinates are given by $\mathbf{T}_i = (T_x, T_y, T_z)$ with $i =$ cube 1, 2 or 3 and the position of the cubes around the planned implant are given by $\mathbf{P}_i = (P_x, P_y, P_z)$ with $i = 1$, 2, or 3, then the euclidean distance between the implant objects is defined as follows:

$$d(\mathbf{T},\mathbf{P}) = \frac{1}{3}\sum_{i=1}^{3}\left(\sqrt{(T_x - P_x)^2 + (T_y - P_y)^2 + (T_z - P_z)^2}\right)_i (m) \tag{2.4}$$

The distances between the planned and tracked implant in three axes provides information about the direction in which the user should move the implant towards the planned position. This is calculated as the absolute difference between the positions of the cubes in the x, y and z axis in Unity:

$$D_x = \frac{1}{3}\sum_{i=1}^{3} ||(T_x - P_x)||_i (m) \tag{2.5}$$

$$D_y = \frac{1}{3}\sum_{i=1}^{3} ||(T_y - P_y)||_i (m) \tag{2.6}$$

$$D_z = \frac{1}{3}\sum_{i=1}^{3} ||(T_z - P_z)||_i (m) \tag{2.7}$$

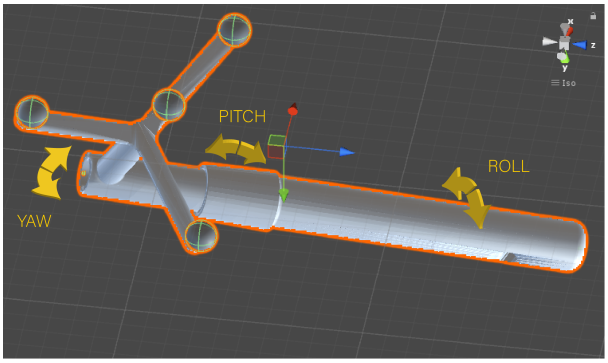


FIGURE 2.11: The definition of the pitch, yaw and roll angles that describes the orientation of the implant in three axes.

Pitch, yaw and roll angles

The planned implant object has an orientation in the UCS described with a quaternion $\mathbf{q}_p = (p_x, p_y, p_z, p_w)$. The tracked implant quaternion is defined similarly ($\mathbf{q}_t = (t_x, t_y, t_z, t_w)$). To calculate the difference in orientation between the two objects, the following formula is applied:

$$\mathbf{r} = \mathbf{q}_p \cdot \mathbf{q}_t^{-1} \tag{2.8}$$

This is based on the principles of quaternion multiplication. Multiplying a quaternion with the inverse of a quaternion is equal to subtracting a quaternion. The quaternion $\mathbf{r} = (r_x, r_y, r_z, r_w)$ represents the difference between the orientations of the planned and tracked implants objects. This is converted to three angles for interpretation purposes (Figure 2.11). The formulas for the pitch, yaw and roll angles are given in equations 2.9, 2.10 and 2.11. To converse the angles from radians to degrees, they are additionally multiplied with the constant $\frac{180°}{\pi}$.

$$pitch = \arctan 2\left(\frac{2 \cdot r_x \cdot r_w - 2 \cdot r_y \cdot r_z}{1 - 2 \cdot r_x^2 - 2 \cdot r_z^2}\right)\frac{180°}{\pi}(°) \tag{2.9}$$

$$yaw = \arcsin\left(2 \cdot r_x \cdot r_y + 2 \cdot r_z \cdot r_w\right)\frac{180°}{\pi}(°) \tag{2.10}$$

$$roll = \arctan 2\left(\frac{2 \cdot r_y \cdot r_w - 2 \cdot r_x \cdot r_z}{1 - 2 \cdot r_y^2 - 2 \cdot r_z^2}\right)\frac{180°}{\pi}(°) \tag{2.11}$$

The pitch yaw and roll angles can be represented as a vector $\mathbf{v} = (pitch, yaw, roll)$. This vector has a certain magnitude and direction in the coordinate system of Unity (Figure 2.6). The Unity coordinate system however, is not necessarily in accordance with the direction of the implant mesh. I.e. the longitudinal axis of the implant mesh is not aligned to the direction of the the x-, y- or z-axis of the UCS. This introduces a problem in interpreting the pitch, yaw and roll angles, because they represent directions that are not related to the implant model in Unity. To solve this interpretation problem, the longitudinal axis of the implant is defined as a vector pointing from the back of the implant to the tip of the implant. By projection of the vector $\mathbf{v}$ on this axis ($\mathbf{a}$) (equation 2.12), the pitch, yaw and roll angles are redefined in a local coordinate system that is aligned to the longitudinal axis of the implant.

$$\mathbf{b} = \left(\mathbf{v} \cdot \frac{\mathbf{a}}{|\mathbf{a}|}\right)\hat{\mathbf{a}} \tag{2.12}$$

where $\cdot$ denotes a dot-product, $|\mathbf{a}|$ is the length of vector $\mathbf{a}$ and $\hat{\mathbf{a}}$ is the unit-vector in the direction of $\mathbf{a}$.

The vector **b** becomes closer to zero if the vector **v**, containing the pitch, yaw and roll angles, is more and more aligned to the axis **a**. Angles close to zero or zero thus provide the user with the knowledge that the tracked implant has an equal orientation as the planned implant. Together with the distance measures, this gives insight into how close the planned position of the implant and thus the locking screw will be reached.

2.4 Discussion & Conclusion

An initial objective of this dissertation was to identify the requirements that the developed software needs to satisfy. Based on material and software-related requirements, choices were made in the design of the navigation software.

2.4.1 Discussion of the used materials

The choice between the PST Base camera and the Brainlab Curve camera was based on several practical issues, such as the availability and ease of use of the PST system and the capability of tracking any object. There is only one requirement that the PST camera does not meet: the possibility of using the camera in a clinical setting in the OR. Further testing in a clinical setting is therefore limited with the PST camera. To alleviate this limitation, most of the tracked objects that were designed for the experimental set-up are also compatible with the Brainlab camera and software.

The PST Base camera was chosen for tracking the objects and the ONS server application was used to communicate between the tracker and Unity. A difficulty that arose was the fact that the Unity coordinate system and camera coordinate system were not equally defined, i.e. Unity has a left-handed coordinate system and the camera a right-handed one. This caused a discrepancy between movement in the camera coordinate system and movement of a virtual object in Unity. Registration of the two coordinate systems initially failed. This was eventually solved by inverting the z-coordinates of the markers in camera coordinates before using these marker positions for registration with the Procrustes algorithm. It is very important to apply this rule consistently, to prevent registration errors. In future use, perhaps with the Brainlab Curve camera, careful analysis of the difference between coordinate systems is advised.

Based on previous experience in the Radboudumc and an available connection between Unity and the PST camera, Unity was the chosen program in which the software was built. In reviewing the literature, little was found of the application of Unity for medical applications. One paper used segmentations of patient scans in Unity for medical training purposes. [35] Another publication reports similar medical training uses. [36] Another research area in which Unity is used is augmented reality in image-guided surgery. [37] Alternative developing platforms for surgical navigation mentioned in literature are 3D Slicer and other open-source software, such as ITK. [30, 38] Based on literature alone, Unity may not seem a likely development platform for medical software. A limitation of Unity that was encountered during this intership is that it does not support the DICOM file format. Visualizing slices from a CT scan is therefore not supported in Unity, something which is possible in 3D Slicer. However, the ease and speed with which three-dimensional anatomical models are rendered in great detail and the many options for user-interaction with the models are remarkable capacities of Unity. It encounters no problems in rendering multiple 3D models real-time and with the use of specific scripts any functionality can be added to the project.

2.4.2 Interpretation of the software functionalities

In the current prototype of the navigation software, the user receives both visual and numeric feedback on the relative position and orientation of the implant to the surgical planning. The

effectiveness of these feedback mechanisms needs to be tested in future experiments to investigate the way the user interprets this information and how well the software is able to guide the user to the ideal position.

The developed visualizations in the prototype, the different viewpoints of Camera objects in Unity, are all three-dimensional. However, the surgeon is used to visualizing the patient's anatomy in 2D slices or projection images. Three-dimensional visualization has the advantage over two-dimensional images that depth is interpretable. Furthermore, by rotating around the virtual 3D objects, much more information can conveyed about e.g. the surgical progress than from a single 2D image. To examine the suggested advantages of 3D images over 2D images, the software should be tested by different users to determine the ease of use and other user experiences such as the time it takes users to complete a task with the guiding software.

The developed software is able to provide the surgeon with real-time information about the distance and orientation differences between the implant and the ideal implant position in the surgical planning. In a surgical setting, this may save time that is now used for fluoroscopy. After the initialization steps, such as the infra-red camera positioning in the OR and registration of the coordinate systems is done, the surgeon can focus solely on the insertion of the implant and locking screw, without pausing to allow fluoroscopic check-ups during insertion. Future research is needed to examine the amount of hypothesized time-saving. The same applies to the hypothesized decrease of radiation dose to the patient and surgical team.

2.4.3 Introduction of inaccuracies in the software

Inaccuracies in the software arise from (manual) registration processes and manually aligning the reference markers in the Unity scene. Additionally, the PST camera can produce an uncertainty during tracking and the 3D-printed objects that are used may differ slightly from the virtual models because of printing errors.

Registration: First of all, errors in the registration process are addressed. Registration is performed based on reference marker positions in both coordinate systems. The marker positions defined by the PST Client software can be exported to a '.txt-file'. The other dataset consists of the positions of the spherical GameObjects of the 3D model in the Unity coordinate system. These spheres are added manually by aligning the GameObjects to the surface of the reference star model in Unity. Errors may be produced by this. When running the project, registration of the two matrices containing these marker positions in Unity is instantly performed. The virtual models of the implant and pointer are therefore positioned in Unity according to the marker configuration in the camera coordinate system from the start. The TRE and RMS of these registrations are constant and presented in Table 2.1. Registering each object with an individual transformation calculated by Procrustes adds up to the total inaccuracy of the navigation software. By adding the values of the pointer, implant and femur registration RMS errors, the total error introduced by registration is 3 mm.

For the femur, manual registration was done by indicating non-anatomical landmarks on the surface with the previously registered pointer object. This resulted in a different registration accuracy each time registration is performed. Also, the method of registering is not applicable in a clinical setting, because the surface of the femur is not apparent during surgery. To avoid the manual registration and to move towards a more clinically realistic registration of the femur, a reference star may be attached to the greater trochanter. Therefore, a different femur model was developed, which is discussed in Chapters 3 and 4.

The configuration of the markers of the DCA attached to the implant and the pointer are duplicates of Brainlab instruments. All of the 3D printed models were tested on the MITeC OR and were recognized by the Brainlab software as inherent Brainlab instruments. For future testing on the OR, this signifies that the models developed for the experimental set-up may be

used instead of sterile Brainlab instruments. The pointer, for example, can be employed for point-based registration with the Brainlab software.

A final discussion point on the registration of the models is that registration is only performed on that part of the object that features reference markers. For the pointer, this does not pose a major problem, given the fact that at the tip and the back of the pointer a marker is present. It it hypothesized that for the implant model however, the position of the reference star does introduce an unknown error at the tip of the implant. The Procrustes algorithm produces an optimal registration for reference star at the back of the implant, but this may not be the best registration of the tip of the implant. Further research is advised to quantify how much the tip of the implant may deviate due to registration on the back of the implant. For example, by repeatedly placing the tip of the implant in a cylindrical opening of an object and rotating the implant to examine if the tip is positioned correctly with different rotations of the reference star in relation to the camera.

Optical tracking: Secondly, the PST camera may introduce an error during optical tracking. This can be subdivided into the inaccuracy of the camera itself, errors produced by the use of damaged reference markers and the position of the tracked objects in the field of view. The manufacturer supplies two values for the accuracy of the system: a RMS error for the position of objects lower than 0.5 mm and for the orientation lower than 1°. [33] This inaccuracy of the camera is experienced during tracking of a stationary object positioned on the table. The tracked object's position in Unity varies with an estimated value of 0.5 to 1 mm.

Tracking is made less reliable with the use of damaged markers, e.g. by scratches on the surface of the markers introduced over time. It is advised to exchange the markers by new ones after a long period of use or when damages are detected.

Where the objects are positioned in the field of view of the camera may also affect the accuracy of tracking. It is advised to position the objects in the centre of the field of view as much as possible.

3D-printing: The third possible source of inaccuracies in the software stated in the introduction of this section is the inaccuracy of 3D-printing. One can imagine that if the pointer object differs from the virtual model by the given inaccuracy of SLS printing (0.3 mm), this influences the registration and the use of the objects.

2.4.4 Further research recommendations

While analysing the list of requirements, it is found that one requirement specifically has not been met yet. That is, guidance during reaming of the medullary canal of the femur and guidance during locking screw insertion has not been achieved yet. These are functionalities that should be provided in future versions of the prototype before clinical testing is possible.

The software-related requirements that are met, e.g. different visualizations and guidance toward the planning with visual and numeric methods should be tested by users to determine the effectiveness and examine subjects of improvement.

To answer the research question and sub-questions formulated in this thesis, different tests are designed. The first, a user test examining the different visualization options and the capability to guide the user towards the ideal position is presented in Chapter 3. Secondly, a validation experiment is performed to analyse the validity of the distance and orientation measures that are provided during guidance.

2.4.5 Conclusion

The current design of the prototype was tested extensively during development. The different functionalities were added roughly in the order they were reported in this chapter. Co-operation

of the various processes, i.e. tracking, registration, guidance and calculation of the position of different objects in the project has been achieved. The introduced inaccuracy of different processes is analysed. Nevertheless, with the reported design of the software, several questions remain unanswered. Further work is required to examine how the user interacts with the software, what the most precise guidance method is and ultimately the prototype needs to be validated with additional testing.

CHAPTER 3

Visualization Tests

3.1 Introduction

In the previous chapter, the design of the surgical navigation application is reported. The developed software is capable of guiding the user towards a planned situation of the implant in the femur. Also, a method of providing feedback to the user is created by changing the implant colour based on how close the implant is positioned near the planned position. A software feature that calculates the distance between and difference in orientation of two Implant objects in the scene has also been built. Different visualizations were created, to display the virtual surgical scene from different perspectives. The three camera objects that render different views of the virtual models during guidance are:

1. The Front Camera (FC). A static 3D anterior-posterior (AP) overview of the surgical scene.

2. The Side Camera (SC). A lateral 3D view of the surgical scene.

3. The Implant Camera (IC). A detailed 3D view of the tip of the implant, moving along with it.

To answer subquestion 2 formulated in the introduction of this thesis (*How should the surgical scene be visualized in the navigational software to optimally assist the surgeon with placing the implant and locking screw?*), a user test is designed. In this experiment, different users test the visualizations and data is saved to assess which visualization gives the best guidance. It is useful to collect data on how fast users reach their goal with each visualization and how close they have come to the planned implant position. This may indicate which view guides the surgeon in the most efficient way. Also, do users prefer one view over the other if two different camera views are shown side by side?

The main goal of this experiment is to find out how well users position the implant according to the different visualizations on the screen and how long this takes in each visualization. A secondary goal is to discover if one of the visualizations guides the user significantly faster or to a significantly more accurate position. A third aim is to find out which visualization the users prefer and why.

How the experiment was designed and conducted is explained in the following section. This is followed by the results of the experiment and an interpretation of these results. Lastly, a conclusion and recommendations are presented.

3.2 Method

3.2.1 Subjects

The user tests were performed by 18 users with different backgrounds. The users can be divided into three categories: medical background (2 subjects), technical background (5 subjects) and Technical Medicine background (11 subjects). The selected users are co-workers, supervisors and medical staff from the department of surgery.

3.2.2 Design

Adaptations to the existing design of the software application and the experimental set-up described in Chapter 2 were made to create a version of the software developed to test the different visualizations and collect the necessary data.

Adaptations to the software To test which visualization is dominantly used over the other visualizations during guidance, three different combinations of a split-screen were created. In each split-screen, one of the three cameras is shown on the left and one on the right. In combination A, the FC is positioned left and the SC is positioned right. In combination B, the FC is shown with the IC on the right. Lastly, combination C consists of a split-screen of the SC and IC. Each user tests all three combinations, but in a randomized order. Users 1 to 6 test the three combinations in the following order: ABC, ACB, BAC, BCA, CAB and CBA. This sequence is repeated for users 7 to 12 and 13 tot 18. Randomization is done to prevent bias, which may be caused by a developing learning curve after inserting the implant once or twice.

Another adaptation to the software is the addition of an object that represents the drill. This instrument is shown in Figure 3.2. It is used in the final step of the experiment, when the implant has been positioned in the femur and the drill would be used to pre-drill a path for the locking screw.

The user test consists of five steps:

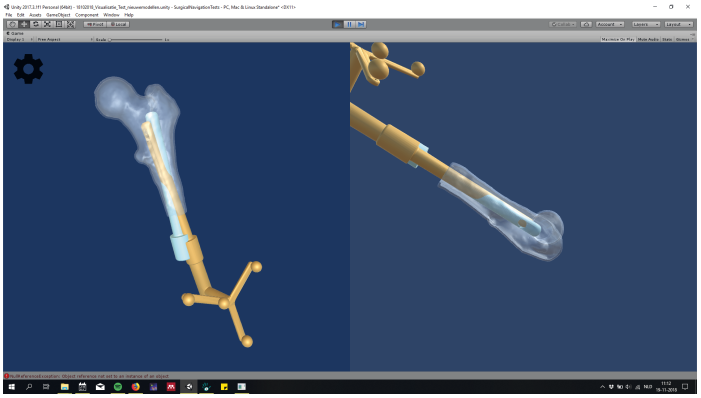
1. The user is given a warm-up exercise to get acquainted with the navigational software. The physical implant object must be moved towards a planned position in space shown on the screen.

2. The femur and a planned implant position are shown in the first combination (A,B or C) of a randomized order and the user positions the implant as close to the planned position as possible.

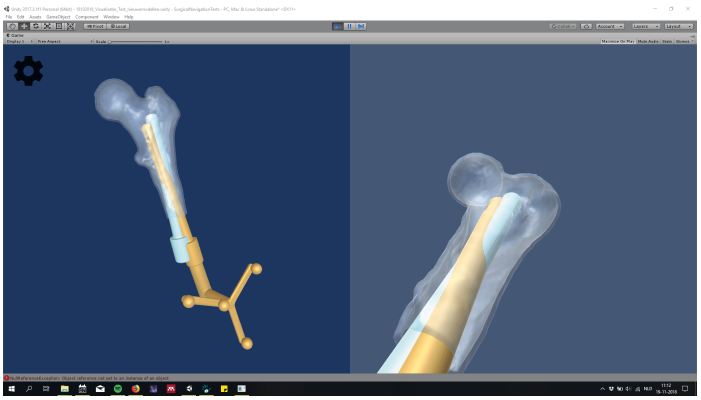
3. This is repeated for the second combination of visualizations.

4. This is repeated for the third combination of visualizations.

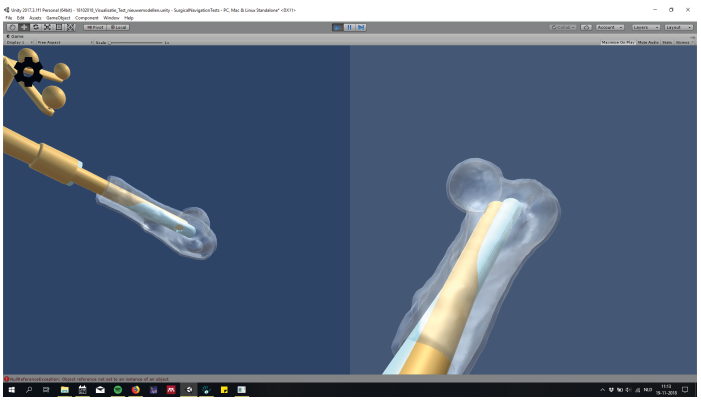
5. The user moves the drill instrument towards the opening in the implant where the locking screw would be inserted. Each user performs this step in the last combination of cameras they tested.



Combination A

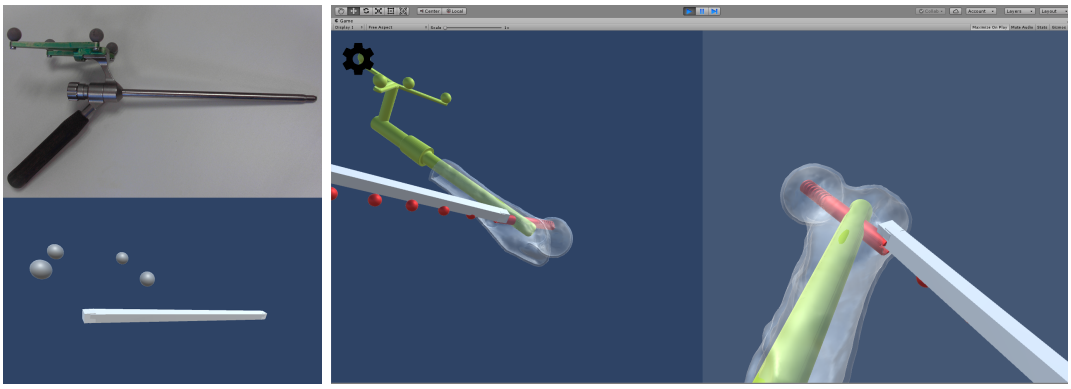


Combination B



Combination C

FIGURE 3.1: The three combinations that are compared in this test. In Combination A, the Front Camera is shown on the left and the Side Camera on the right. B combines the Front Camera and Implant Camera. Finally, in C the Side Camera and Implant Camera are combined.



(A) (B)

FIGURE 3.2: (A): The physical object representing a drill (above) and the virtual object (below). (B): The drill is used to find the gap in the implant through which the screw is inserted. Guidance is provided with the red spheres indicating the right orientation of the drill.

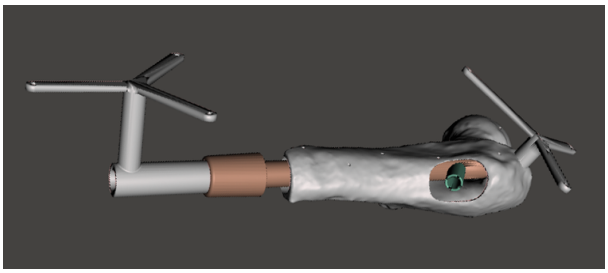
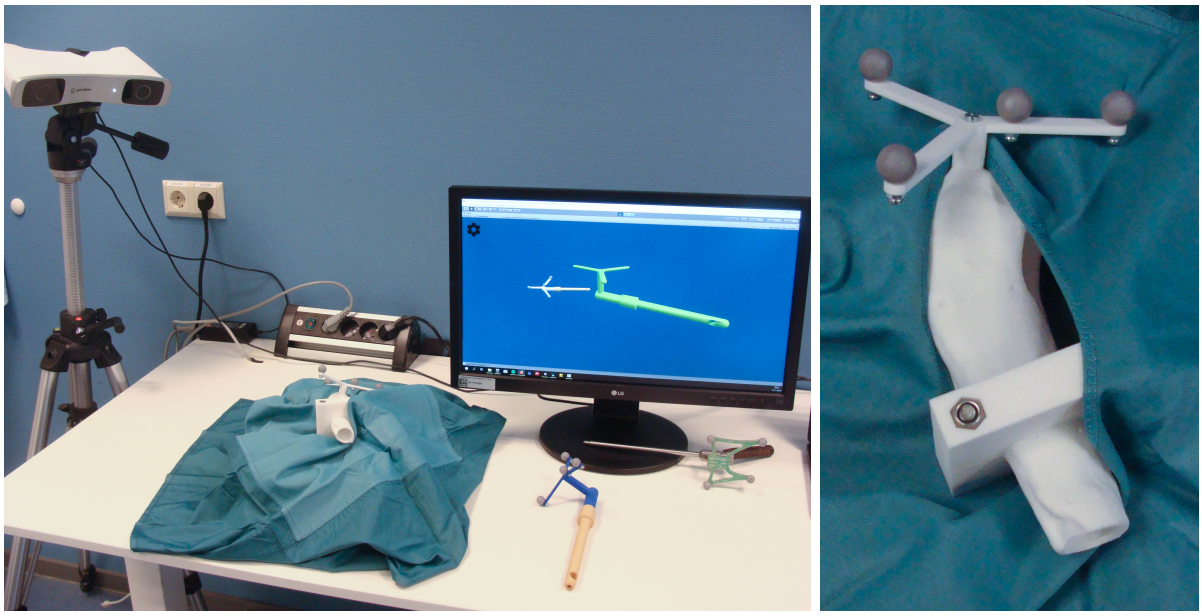


FIGURE 3.3: The design of the models produced for the visualisation experiment. The implant diameter is smaller (15 mm), to allow movement of the implant inside the hollow cavity (diameter of 21 mm). A non-anatomical gap on the lateral side of the femur model is designed to provide the opportunity of navigating a drill for locking screw pre-drilling. The femur contains a fixed reference star on the greater trochanter, which eliminates the manual registration step.

Adaptations to the experimental set-up The existing femur model and implant model used during development of the software were not suitable for this user test. This has several reasons. First of all, the implant model is designed to fit the hollow cavity of the femur model exactly. This means that the implant can only be inserted with a perfect fit. Secondly, the femur model needs to be manually calibrated each time the software is ran. By using the pointer, the seven pits on the surface of the femur model are indicated and a transformation is calculated with the Procrustes algorithm. This would consequently introduce differences between users because the registration accuracy would vary for each user.

To overcome these limitations, a new implant model and femur model were designed. The implant diameter was decreased from 21 to 15 mm. This allows the implant to move freely in the hollow cavity of the femur model. To omit manual registration of the femur model, a reference star was designed to be attached to the greater trochanter of the femur model. Also, a gap in the lateral side of the femur was created, through which the instrument representing the drill can reach the cylindrical hole in the implant after it has been positioned inside the femur. The new test objects are illustrated in Figure 3.3.

The experimental set-up is shown in Figure 3.4. It shows the camera position relative to the tracked objects. The femur model is covered by surgical drapes, to prevent the user from seeing through the gap on the lateral side of the femur while inserting the implant.



(A) (B)

FIGURE 3.4: The set-up for the user tests is shown in (A). A detail of the reference star attached to the femur model and coverage with drapes is shown in (B).

3.2.3 Methods of measurement

In the warm-up experiment, users can become acquainted with navigating the implant in free space. The users are asked to announce when the implant position is the closest to the visualized position they are trying to reach on the screen. By pressing a button, relevant information of that moment is saved. E.g. the time it took users and the distance between the tracked implant's position and the planned position on the screen. This is done with the script "SaveToTxt ", which saves certain parameters in Unity to a '.txt-file'. These parameters include the positions of the tracked implant and planned implant, as well as the orientations in the 3D Unity coordinate system (UCS). Additionally, the distance between the two positions is automatically calculated and saved. Two different distance measures are applied. The first is the euclidean distance expressed in mm. The second measure is an absolute difference in each axis. E.g. a distance between the two positions in the x-direction of the UCS.

A difference in orientation is calculated as the angle between the objects in three directions: the pitch, yaw and roll measured relative to the longitudinal axis of the implant. The time at the start of running the software and at the start of each step during the user test is saved for further analysis. An overview of what is saved at which step during the user test is provided in Table 3.1. Before each user test, the file-name is specified in "SaveToTxt", to ensure each dataset is saved in a separate file.

In a questionnaire (Appendix B), users are asked which of the camera views they needed most during implant positioning and during usage of the drill instrument. The three camera views, FC, SC and IC, are alternately shown on the left and right in the three combinations A, B and C. Moreover, users were asked to vote for the best visualization: i.e. which combination had showed them the most information during guidance.

Registration of the CCS and UCS is performed on the new implant and femur models and the RMS values are provided in the results section.

TABLE 3.1: Different parameters from Unity are saved at each step of the user test. The script "SaveToTxt" is programmed to save different parameters at different events during testing. The time is saved at each event to allow analysis of how long different phases of the test took.

Parameters	Starting the software	Warm-up experiment	Positioning implant in femur (for each of 3 combinations)	Positioning drill at implant hole
Start time	✓	✓	✓	✓
Positions of tracked implant and planned implant		✓	✓	
Which combination of camera views in Unity? (A, B or C)			✓	✓
Distance between tracked implant and planned implant		✓	✓	
Pitch, Yaw, Roll angles between tracked implant and planned implant		✓	✓	
Distance between the tip of the drill and the implant's cylindrical hole				✓

3.2.4 Analysis

The saved parameters from Unity were analysed using Microsoft Excel (2016). Descriptive statistics, the mean euclidean distance (and standard deviation) between the tracked and planned implant are calculated. Also, the mean difference in orientation of all 18 users is calculated for each of the combinations A, B and C. Each of these parameters in the different groups is assumed to have a normal distribution. A one-way ANalysis Of VAriance (ANOVA) test is performed to examine if the differences between groups A, B and C are statistically significant. The mean time necessary to insert the implant in each combination is also presented and statistically tested between the three combinations with an ANOVA test. After finishing implant positioning in all three combinations, the drill instrument is used to navigate towards the implant opening. The distance between the opening on the implant object and the tip of the drill instrument is saved. Again, a statistical difference between achieved distances in the three combinations is examined.

3.3 Results

3.3.1 Registration

The registration parameters of the implant and femur models are tabulated in Table 3.2.

3.3.2 Euclidean distance and angular difference

In Table 3.3, a comparison between the mean measurements (SD) in the different combinations is shown. Overall, euclidean distances < 10 mm were measured. The results from the warm-up experiment are shown, but not taken into account in the calculation of the p-value. In Figures 3.5 and 3.6, box plots are shown that illustrate the variation in the measurements of distance and angular differences in groups A, B and C.

TABLE 3.2: The calculated target registration errors (in mm) and the root mean square error of registration for the new implant and femur models.

Model	TRE (mm)	RMS (mm)	Model	TRE (mm)	RMS (mm)
Implant	0.861	0.643	Femur	0.957	0.872
	0.162			0.551	
	0.671			0.948	
	0.660			0.963	

TABLE 3.3: The differences between the tracked implant and the planned implant object in Unity are denoted as a mean (SD) per combination A, B and C. In the last row, the calculated euclidean distance between the tip of the drill and the the implant opening are added. [a]Group differences were tested with one-way ANOVA tests. [b]The values for the warm-up experiment are added in the last column, but are not included in the calculation of group differences.

Parameter	A	B	C	p-value[a]	Warm-up[b]
Euclidean distance (mm)	6.58 (3.97)	7.16 (6.06)	9.83 (5.81)	0.162	8.51 (0.64)
Pitch (°)	0.48 (0.34)	0.59 (0.43)	0.61 (0.43)	0.955	4.69 (4.73)
Yaw (°)	0.59 (0.49)	0.71 (0.59)	0.69 (0.55)	0.844	0.86 (1.11)
Roll (°)	0.90 (0.71)	1.07 (0.84)	1.08 (0.86)	0.850	3.97 (6.44)
Elapsed time (s)	54.28 (42.01)	35.89 (16.37)	84.28 (91.05)	0.052	121.67 (93.78)
Euclidean distance (mm) of drill tip to implant opening	8.96 (2.05)	10.10 (6.93)	9.10 (6.63)	0.921	

In Figure 3.5, two outliers are visible, one in combination B (distance of 27.52 mm) and one in combination C (distance of 27.48 mm). The mean distance of combination C is higher than the means in A and B. However, the 'whiskers' extending vertically from the boxes show that the measurements in each combination have a similar variability.

.

3.3.3 Finding the locking screw position in the implant

The results of navigating the drill tip to the implant gap through which the locking screw would be inserted are presented in Table 3.3 and Figure 3.7. There are no significant differences in how close the drill tip is positioned at the implant opening between different combinations of visualizations (p-value of 0.921), tested with a one-way ANOVA test.

3.3.4 Elapsed time

The results of the elapsed time in different visualizations are presented in Table 3.3 and Figure 3.8. An outlier is present in group C (time of 431 s, or 7 minutes and 11 s). The results of the one-way ANOVA test indicate a near-significant difference between the variation in groups A, B and C (p-value of 0.052). The variation in group B is markedly lower than in A and C. Apart from two outliers, the measurements were below 50 seconds.

3.3.5 User preferences

In the questionnaire, users were asked whether the left or right view on the split screen was watched more during guidance towards the planned position. The results for are shown in Figure 3.9. The FC was preferred over the SC and IC in combinations A and B. In Figure 3.10, which views were appreciated most during navigation of the drill tip to the implant opening are displayed. In contrast to using the views to position the implant, there was no preference for the Front Camera in combination A.

FIGURE 3.5: A box plot showing the variation of the distance measurements in each combination.

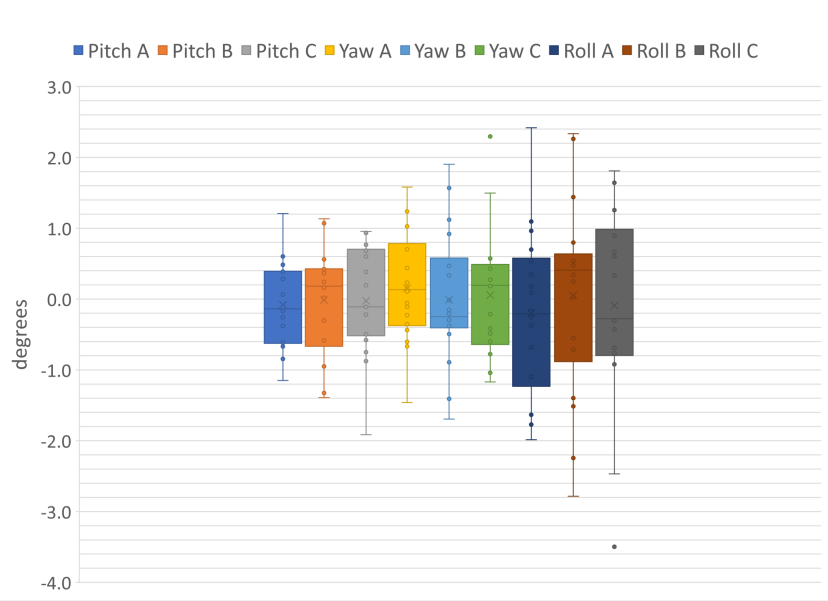


FIGURE 3.6: A box plot showing the variation of the angle differences in each combination. From left to right, first the Pitch variable is plotted in the three combinations, then the Yaw variable and lastly the Roll variable.

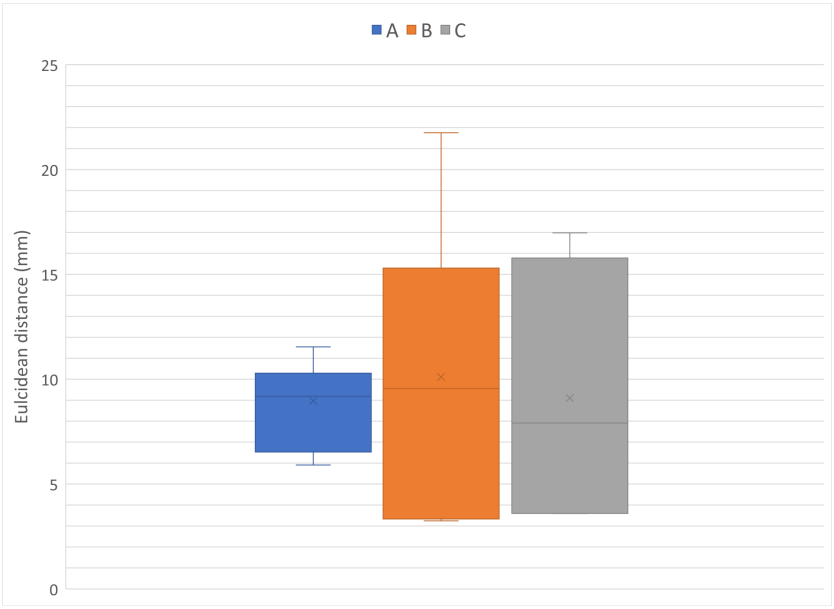


FIGURE 3.7: A box plot of the variability in the measurements of drill tip to implant opening distances is displayed. In combination A, the user variability is lowest.

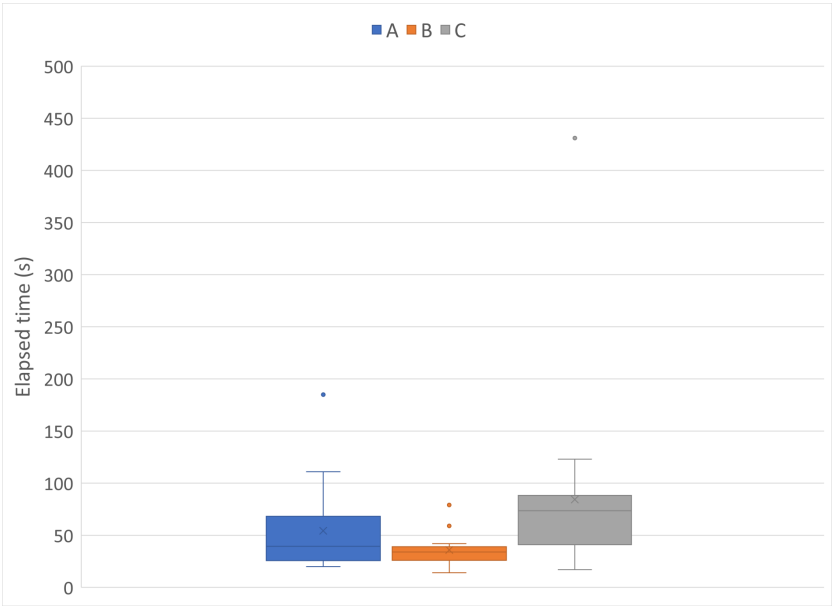


FIGURE 3.8: The results of the time it took users to move towards the planned position in each combination. The blue box plot shows the variation of the time measurements in combination A, the orange plot the variation in B and the grey plot in C.

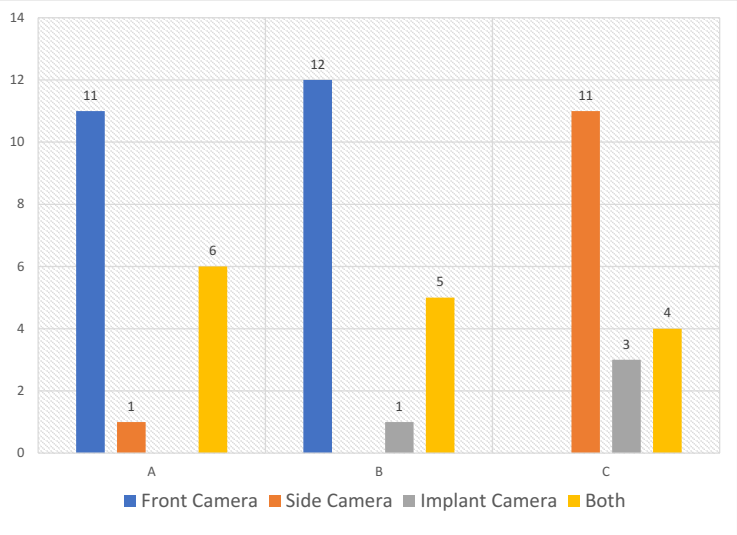


FIGURE 3.9: The dominantly watched camera viewpoints in combinations A, B and C while placing the implant. In combination A, 11 users preferred the Front Camera view versus 1 the Side Camera view. Six users needed both views equally. The Front Camera view is also voted best in comparison to the Implant Camera in combination B. Placement of the implant with guidance of the Side Camera and Implant Camera in combination C resulted in a user preference (11) for the Side Camera.

Additionally, users voted for the best visualization. The combination of the FC and SC (combination A) was voted best with 7 votes. Combination B (FC on the left and IC on the right) received one vote less and 5 users preferred combination C (SC and IC).

3.4 Discussion

There is no significant difference between how close the user was able to position the implant with different visualizations. Each combination of views is therefore capable of guiding the user towards the planned position, but how easily the user gets there may be different. In combination C, users took distinctly longer to find the correct position and orientation. This suggests that the users would like a front camera for overview (not present in combination C). This can also be concluded from the questionnaire.

The calculated pitch and yaw angles varied similarly in all three views. The roll angles showed a larger variation in the three different visualization. This may indicate that users could interpret differences in the pitch and yaw of the implant better and guidance of the right roll angle of the implant was more difficult in the current visualization.

The study design allowed the users to get to know the software with a warm-up experiment. Users found this very helpful. This starting experiment also provided information about the use of only one three-dimensional view. The depth in this single view was difficult to determine. Also, the warm-up experiment gave the users no feedback on how close they were. They had to interpret the differences between the tracked implant and planned implant on the screen themselves. This is what made this warm-up experiment hard, indicating that a feedback mechanism is necessary besides a visualization.

In this experiment, the calculated distance and orientation measures were not shown to the user. Users only had the visualization and colour information, which provides information about the capability of these features to guide the user without providing numbers on how close they were. With the feedback mechanism, which turned the implant bright green at a very close position and orientation, users were guided to a close orientation and position. The feedback mechanism, however, prevented the users from positioning the implant far from the planned

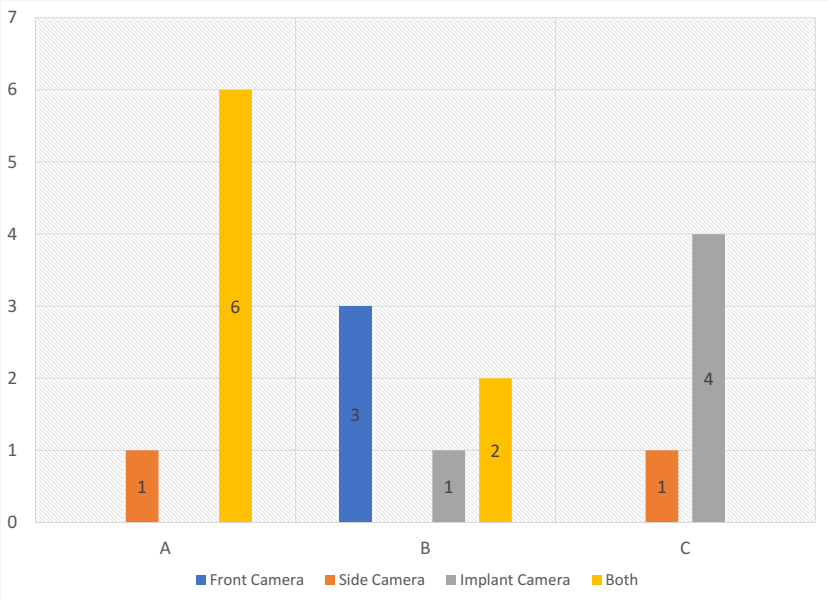


FIGURE 3.10: User preferences documented while positioning the drill tip at the implant opening. Users that performed this step in combination A generally declared to need both the Front Camera and Side Camera. On average, users testing combination B did express a clear preference for either or both views. For combination C, 4 users declared to need both views and one declared to prefer the Side Camera over the Implant Camera.

position. If a certain camera viewpoint did not show all the information that the user needs to interpret depth for example, the user would try to find the right position by looking at the change of colour of the implant. This signifies that, eventually, they would succeed in getting close to the planned position, even if the visualization does not provide sufficient information.

The user tests were conducted by people with a different background, which introduced interesting differences in user preference. People that are used to watching the anatomy of the patient in 2D perpendicular views, for instance doctors and users with a technical medicine background, reported that they would like to view the scene in the three basic anatomical planes. In a future study, it is recommended to examine the difference between the standardized anatomical plane views and more unusual views like the implant camera with user groups that are equal in size.

The insignificant differences that were found between the three combinations can be explained by a small amount of test subjects. Also, because of the colour interpretation, most users were capable of closely reaching the planned position. In future research, user tests could be performed with more test subjects and with equal groups of testers, e.g. equal amounts of subjects with a medical background and with a technical background.

The findings of this study imply that in clinical use, the surgeon needs at least two three-dimensional views to assess distances and angular differences in the virtual surgical scene. The subjective results of the questionnaire indicate that the Front Camera view should be one of the chosen visualizations. Some users gave feedback on the type of view that was missed between the provided options. One view came forward especially, a view similar to an axial slice. This view could also be described as a dartboard view. The viewpoint would be from the tip of the implant or drill. It is speculated that choosing a direction before insertion and orientating the object is improved with such a view. This view may be developed and tested in future research.

3.5 Conclusion

Each visualization was able to lead the user to the planned position, with help of the feedback mechanism. The mean distance to the planned position was < 1 cm. The combination of the lateral view and detailed implant view was found to give slower results, therefore this combination is not recommended. The warm-up experiment showed the need for at least two 3D views. Users preferred an overview from above and a second view to assess distances in multiple axes. It is recommended to always include the overview and it may be interesting to test whether an 'axial'-like view could complement the front and lateral views.

CHAPTER 4

Validation Test

4.1 Introduction

In the previous chapter, an experiment was designed and conducted to determine the efficiency of different visualizations and usage of a feedback mechanism. Without providing numbers about the difference in position and orientation during implant placement, supplying feedback by changing the colour of the implant is a mechanism that can lead the user towards a planned position and orientation.

During these experiments, however, the focus was put on implant positioning inside the hollow cavity of the femur model. This is no representative of the clinical situation, in which the medullary canal of the femur must be excavated before implant insertion. The direction in which and the depth to which the femur is excavated is essential for the final position of the implant. To use surgical navigation during this step, guidance during reaming must be provided by the software. This feature is added to the software and tested in an experimental set-up.

The question that can be answered with this experiment is how well guided drilling and guided implant insertion leads to the correct planned position in the femur. If the software measures a certain euclidean distance between the planned position and the final implant position, how well does this agree with the true implant position in the femur model?

After guided excavation and guided implant insertion, a Cone Beam CT (CBCT) scan of the models is made (at the apartment of dentistry in the Radboudumc). The position of the implant in the femur model can be segmented from the scan and compared to the planned position of the implant. The difference between the segmented position of the implant and the planned position that was navigated towards is measured in the scan. This distance measure is compared with the distance measure in the software to examine the validity of the calculated measures in Unity.

4.2 Method

4.2.1 Design

The experiment is performed once as a proof of concept for navigated drilling. New materials were needed to perform the experiment, such as a material to fill the hollow cavity in the femur

model and a drill that can be tracked by the camera and is used for excavation. Additionally, a visualization of the drill and a registration method were developed. The experiment consists of the following steps:

1. A calibration block that is tracked by the camera is used to calibrate the tip of the drill with respect to the reference star on the drill.
2. In the software, a cylinder with the dimensions of the implant is visualized for guided drilling of the required cavity. Guidance is provided with distance measures calculated in Unity and with colour interpolation of the drill object.
3. After guided excavation has been completed, the planned implant position is displayed in the software and navigation towards the planned position is enabled. The same guidance features are provided.
4. After guided insertion of the implant, the calculated distances and angles between the implant position and the surgical planning are saved.
5. A CBCT scan of the femur and implant model is made.
6. The models in the scan are segmented and the implant position is compared to the planned position.
7. The distance calculated from the scan and the euclidean distance calculated in Unity are compared.

Adaptations to the software Based on the outcomes of the visualization experiment, the Front Camera was recommended, alongside a second view to asses distances in multiple axes. The chosen view was the Side View because it adds a second overview of the scene. Additionally to the existing views, a new type of camera was created in the Unity scene (Figure 4.1). The camera is positioned in front of the tip of a GameObject, in this case the drill tip, and shows a "cross hair" at the direction the GameObject is facing.

The three views that are thus shown during this experiment could be called an anterior-posterior view (Front Camera), a lateral view (Side Camera) and an axial view (Crosshair Camera).

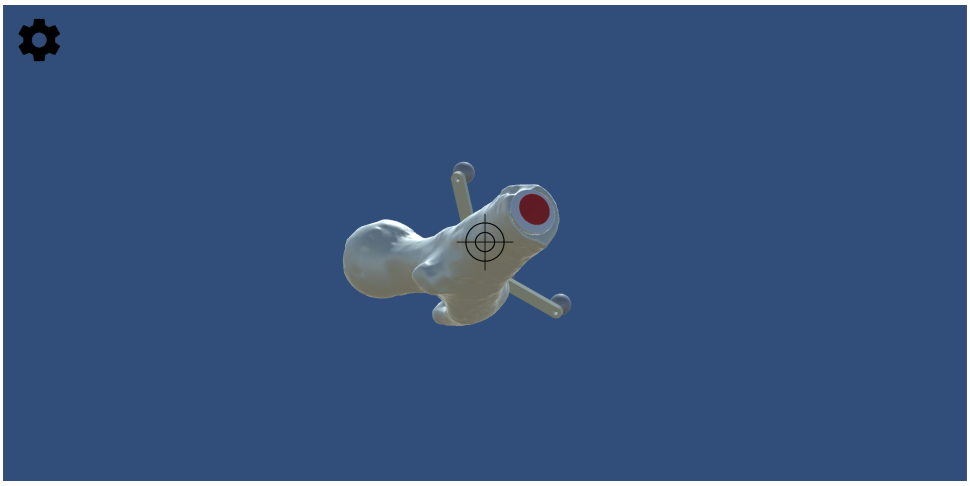
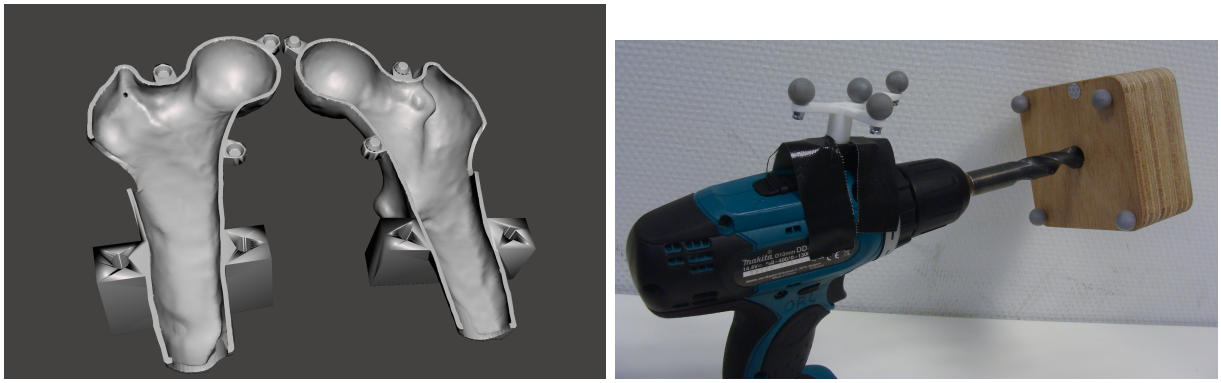


FIGURE 4.1: The crosshair camera is positioned at the tip of the drill used for excavation and shows a crosshair in the center of the screen. Subsequently, if the crosshair is positioned straight at the red target visible in the femur, the drill is directed in the right position for excavation.



(A) (B)

FIGURE 4.2: The set-up for the validation test. In (A) the digital design of the two halves of the femur model that can be filled and closed again are shown. (B) shows how the drill is calibrated using the wooden calibration block, which can be tracked by the optical camera.

Adaptations to the experimental set-up To be able to drill a cavity in the femur, the hollow model is filled with 'putty'(AlgiNot, Kerr Dental). This is a paste used in dentistry to make impressions of the dentition. To enable filling of the model, a model that can be opened up, filled with putty and closed again is utilized. A femur model (Figure 4.2a) is designed that consists of two halves which can be attached to each other at several locking points. This model was also used during the user tests and has a reference star attached to the greater trochanter to facilitate optical tracking. Due to the fixed position of the reference star, the registration parameters of the femur model are constant each time the project is used. The RMS values of the implant and femur registration are the same as during the visualization experiment and can be looked up in Table 3.2.

With a hand-held drill with a diameter of 15 mm (matching the thinner implant designed for the visualization experiment), a cavity is drilled in the filled femur model. To enable navigation, the drill is calibrated in Unity. A calibration block is made for this purpose (Figure 4.2b). The drill is positioned in the gap with a known depth (3 cm) in the middle. The drill tip position and orientation in Unity are set to the depth and orientation of the gap in the digital model of the calibration block. The configuration of the drill reference star and drill tip is thus calibrated with the known position of the gap in the calibration block.

4.2.2 Methods of measurement

The script "SaveToTxt" is able to save the positions and orientations of the planned implant and tracked implant in Unity. The euclidean distance and angle measures are therefore the same as used in the visualization experiment described in the previous chapter.

As an addition, the distance between the position of the implant in the physical femur model and the virtual planned position is determined. The physical positions are based on the CBCT scan made of the models after guided implant insertion.

4.2.3 Analysis

A segmentation of the femur model and implant in the CBCT scan are made with the software program 'Maxillim', which is used at the department of oral and maxillofacial surgery. Segmentation is based on thresholding, utilizing the difference in Hounsfield Units (HU) of the nylon material of which the femur and implant model are composed and the putty material. To enable comparison between the segmented position of the implant in the scan and the planned position used in Unity, the virtual implant and femur models are loaded into Maxillim as well. By registering the virtual femur model (containing the planned position of the implant) with

FIGURE 4.3: Picture of the result of excavating the femur model and insertion of the implant.

the segmentation of the physical femur model, the planned and segmented objects are aligned in Maxillim. Registration is performed with the "Surface-based Matching" tool in Maxillim. With the distance measurement tool in Maxillim, the euclidean distance between the tips of the virtual planned implant and the segmented implant model can be calculated. This distance measure is compared to the calculated measures in Unity to examine the validity of the latter.

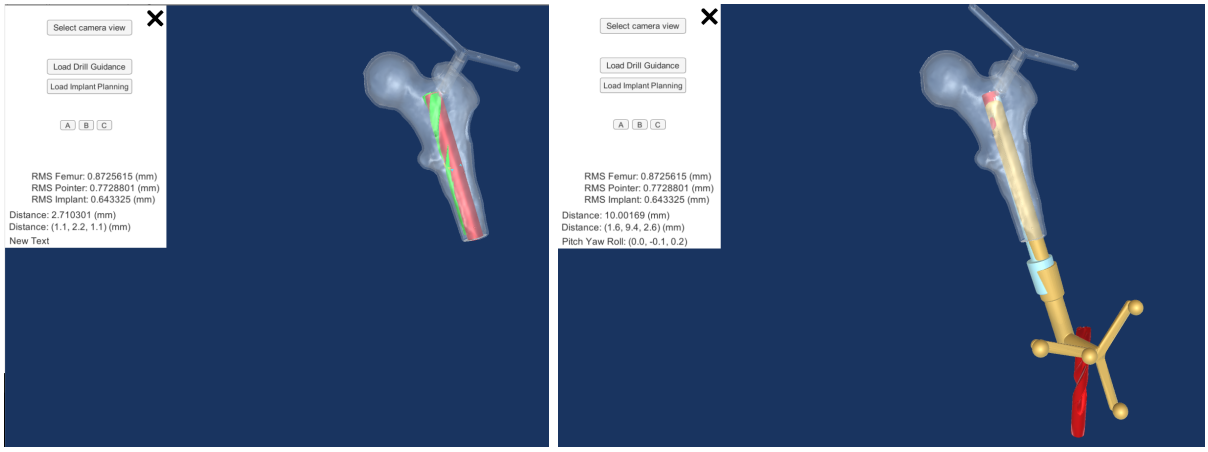
4.3 Results

Navigated drilling was successful. The putty material and femur model were able to undergo drilling and as a result a perfectly cylindrical cavity in which the implant is inserted was excavated. The result is presented in Figure 4.3. The distance measures that were used during guided drilling are shown in the menu in Figure 4.4a. The excavated cavity in the femur fit the implant exactly. After navigated insertion, the implant was tightly fitted in the putty material. An image of navigated implant insertion is given in Figure 4.4b. The distance measures exported from Unity after finishing the implant placement are presented in Table 4.1. The calculated euclidean distance between the planned and tracked implant position is 9.91 mm.

Slices of the CBCT scan are visible in Figure 4.6. The femur model (Figure 4.5a), putty (Figure 4.5b) and implant model were segmented from the scan. Additionally to segmenting the models from the scan, the surgical planning was also loaded in the program. The surface of the femur model and the segmentation of the femur model were matched (Figure 4.8). The planning was compared to the segmented position of the implant. Comparison is done at the tip of the implant, shown in Figure 4.7. The distance measurement tool provided a distance of 3.6 mm, also added to Table 4.1.

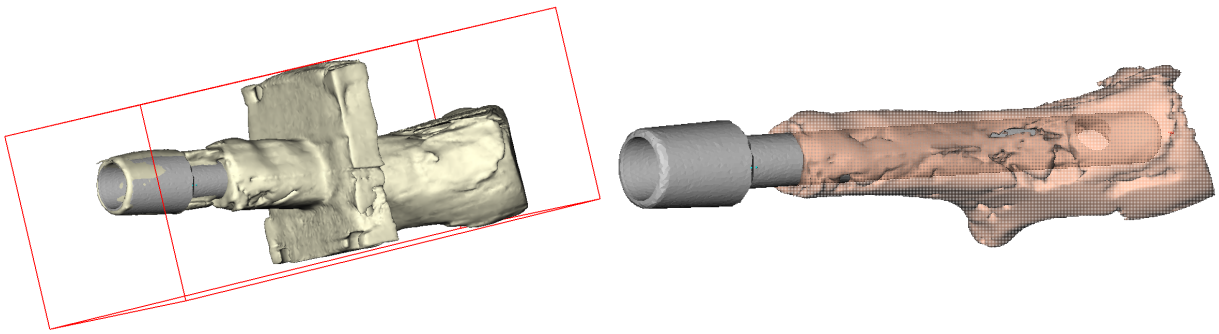
4.4 Discussion

This experiment was used as a proof of concept for navigated drilling and excavation of the femur model. Also, the distance measures calculated in Unity could be validated by scanning the models after guidance.



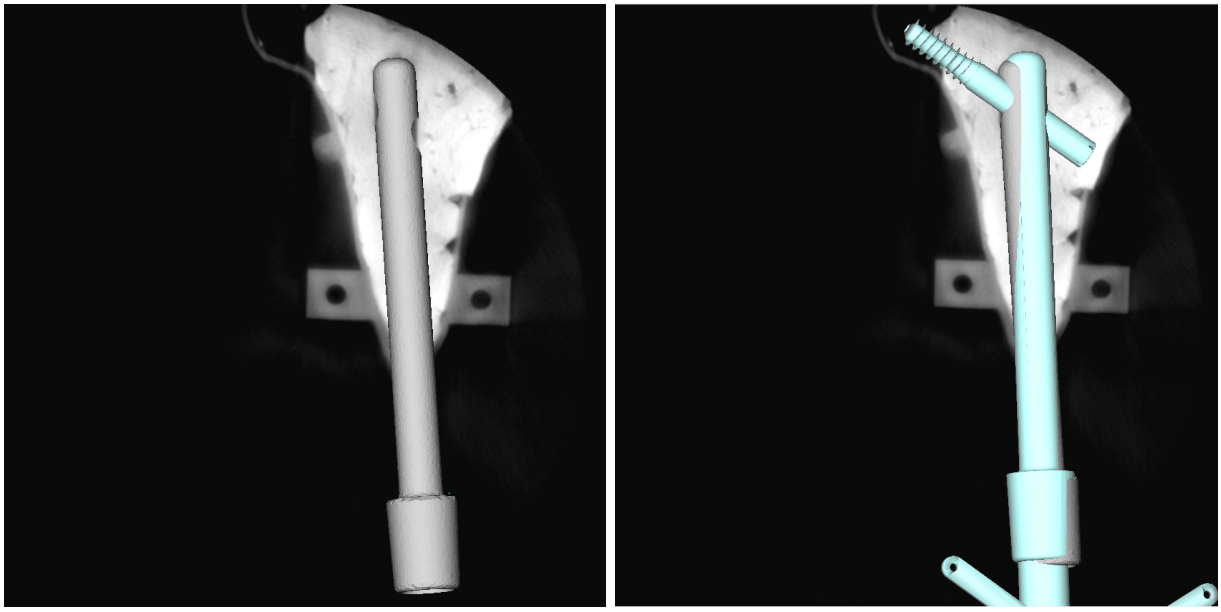
(A) Navigated drilling (B) Navigated implant insertion

FIGURE 4.4: In (A) a screen-shot of the project is shown during guided drilling. The drill, visualized with a 3D drill mesh with the dimensions of the drill that was used in this experiment, is coloured green. It it positioned in a red cylinder with the dimensions and position of the planned implant model. Figure (B) depicts the implant (coloured orange) and planned implant (in blue) during navigated insertion of the implant. The measures displayed in the menu were used for guidance.



(A) Segmented femur model (B) Segmented putty and implant

FIGURE 4.5: The segmentation of the femur model from the CBCT scan is shown in (A). The segmented position of the implant model is depicted in grey. In (B), the putty with which the femur model was filled is segmented and displayed with the segmentation of the implant.



(A) Segmentation of implant (B) Planned position in blue

FIGURE 4.6: Screen-shots of the analysis in Maxillim. Slices of the CBCT scan are diplayed, the putty material has a high HU compared to the 3D-printed nylon objects. The segmented implant model is visible in (A) and additionally the planned implant model and locking screw position in (B).

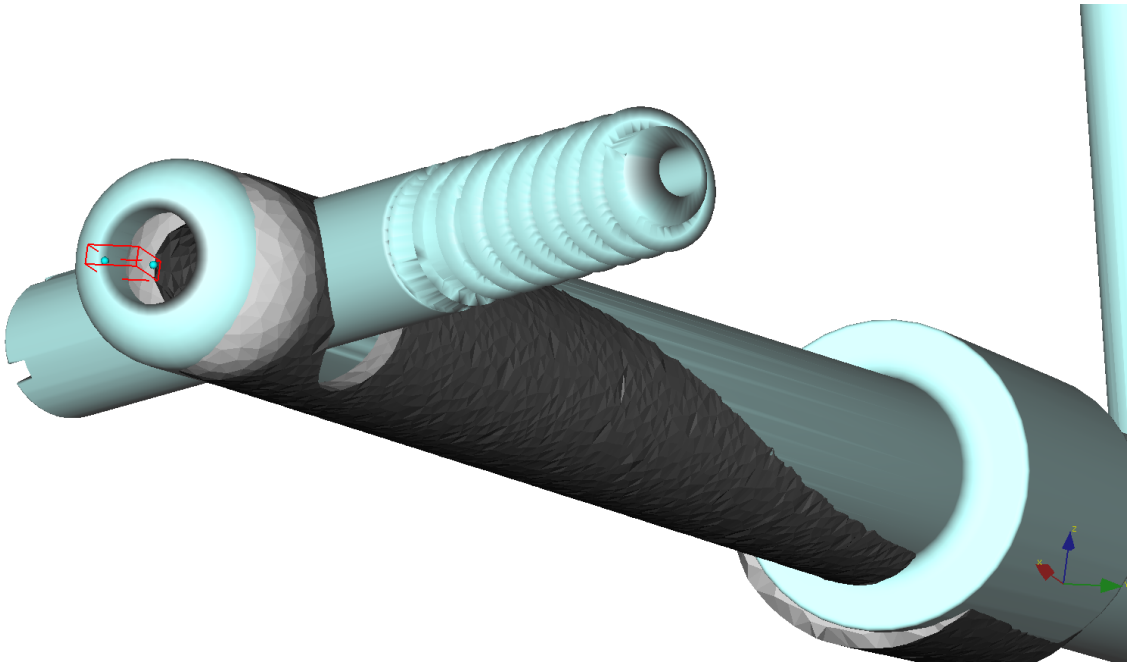


FIGURE 4.7: Screen-shot of the euclidean distance measured between the implant tips in Maxillim. The measured distance is 3.6 mm. This image also shows an unquantified difference in orientation, visible between the implant opening in the segmentation and the locking screw position in the planning.

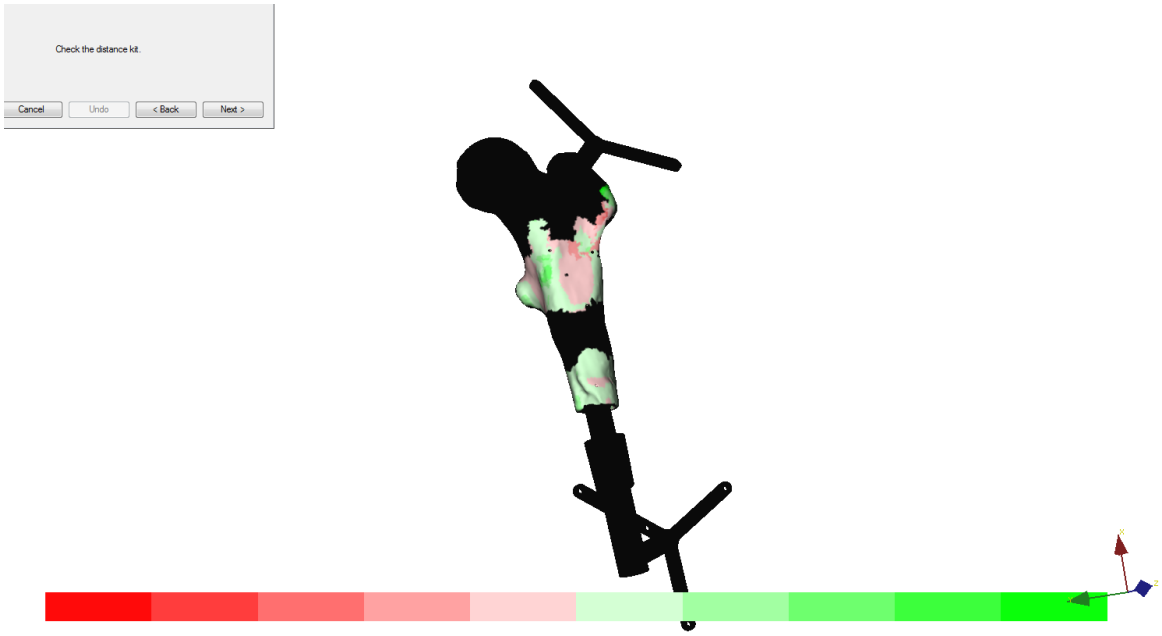


FIGURE 4.8: During Surface-based Matching, a distance kit is provided for assessment of registration accuracy. This is a display of the matched surface in which colours, green and red areas, represent parts of the surface that did not match the other surface well. The legend in the bottom of the screen-shot depicts how much the matching deviated.

Calibration of the drill was straightforward and fast. The calibration block was used to give the drill object in Unity the correct offset relative to the tracked reference star on top of the drill. With this calibration step, registration with the Procrustes algorithm is avoided. In a clinical setting, using a calibration block can also be a method for registering different reamers or drills.

The extra viewpoint (crosshair camera) gave a very detailed view of the drilling direction. A point of improvement could be to add a similar, second view, not at the tip but at the back of the drill. By aligning the tip and the back view with the crosshair, the direction of the drill may be found even easier.

After drilling, navigated implant insertion was begun. It was seen that the direction of insertion was slightly different than the planned direction. The direction in which the implant was inserted could not be changed, the putty material was unyielding. It was not possible to reach the full depth of the implant that the planning showed. Insertion became more and more

TABLE 4.1: Resulting parameters of the experiment. The euclidean distance between the planned implant position and the implant model position calculated by Unity is approximately 1 cm, while the distance measured in the scan is 3.6 mm. An image of the difference between planned and tracked implant positions in Unity is available in Figure 4.4b. The positions in the scan can be seen in Figure 4.6b

Parameters	**Calculated in Unity**	**Derived from segmentation of scan**
Euclidean distance (mm)	9.91	3.6
Absolute distance x-axis (mm)	1.70	
Absolute distance y-axis (mm)	9.35	
Absolute distance z-axis (mm)	2.50	
Pitch angle (°)	-0.02	
Yaw angle (°)	-0.13	
Roll angle (°)	0.14	

difficult with greater depth. Either because the cavity was not as well excavated at the tip of the implant than at the back, or because of added friction from the implant and putty. Eventually, implant insertion was stopped with a euclidean distance between tracked and planned implant of 9.91 mm. The orientation of the implant was adjusted in the final position of the implant according to the roll angle.

The euclidean distance measured between the planned and scanned implant position is 3.6mm. The software provided the user with a distance measure of 9.91 mm. The accumulated error caused by registration and camera inaccuracies may be the source of the additional 6 mm error measured in the software. However, during guided drilling in the femur model, the drill object deviated from the planned cavity with approximately 2.7 mm (Figure 4.4a). It may be possible that the cavity that was drilled is approximately 3 mm off target, which is somewhat close to the deviation found in the scan. It is speculated that the implant object is less accurately registered and therefore, the object is positioned with a larger inaccuracy in the excavated femur. As previously explained on page 25, the tip of the implant is far from the area on which registration is performed (i.e. the reference star at the back). Perhaps if the implant tip was calibrated with the calibration block used for the drill, the tip position in the software could be closer to the actual position of the tip. This is a possible explanation for the difference between the two presented distance measures in Table 4.1.

A small inaccuracy could have been introduced in the analysis in Maxillim. The Surface-based matching tool was used to register the surgical planning to the segmented surface of the femur model. Figure 4.5a shows that the segmented surface was rough. The registration of the two surfaces may therefore introduce an error. Maxillim provides a visualization of the registration accuracy in the form of a 'distance kit' (Figure 4.8). The distance kit shows that some areas of the rough surface were not a 100% match with the smooth surface of the femur model. The implant planning is attached to the femur model matched to the scan, introducing a possible error in the planned implant position in Maxillim.

4.5 Conclusion

This chapter describes the design and execution of an experiment aimed at guided reaming of the femur model. After excavation of the putty material, the implant is inserted with help of the navigation software. The euclidean distance measured by the implant is 9.91 mm. The analysis of a CBCT scan produced a euclidean distance measure of 3.6 mm. Inaccuracies in implant registration and tracking are speculated to be the cause of this discrepancy.

CHAPTER 5

Discussion & Conclusion

This thesis presents the design of and experiments with a novel surgical navigation system intended to guide the surgeon with reaching the surgical planned positions of the patient-specific implant and locking screw. The developed prototype can provide information about the depth and rotation of the implant, which are the most important factors influencing the final position of the locking screw in the head and neck of the femur. Additionally, guided reaming of the femur was tested and the user experiences were taken into account in determining which visualization can optimally guide the surgeon.

The developed software is specifically designed for the placement of patient-specific osseointegration implants and locking screws. Therefore, it differs from more generic surgical navigation systems that are commercially available, such as the Brainlab navigation system available in the Radboudumc. Also, it provides a novel visualization method during osseointegrated prosthesis (OIP) surgeries. Instead of 2D projection images, three-dimensional virtual models are visualised real-time on a screen. Utilizing 3D images enables the interpretation of depth and distances between virtual objects. The distances between the planned positions on the screen and the actual positions of the physical objects is calculated. In the validation experiment described in Chapter 4, the calculated distance measure in the software was compared to the actual measure derived from a scan of the objects. A difference of 6 mm was found between the calculated and scan-derived distance measure. This discrepancy is most likely due to the accumulated errors in the optical navigation software that influence the calculated distance in the software. Registration of the optical camera coordinate system and the navigation software coordinate system increases the inaccuracy of optical tracking. Additionally, the optical tracker itself has an estimated inaccuracy of 0.5 to 1 mm.

To discuss the clinical relevance of these findings, a clinically acceptable margin of the distance measure between the planning and actual implant and locking screw position is needed. There is no literature found that examines the effect of locking screw position on clinical outcome measures in patients with a femoral OIP. However, literature on intramedullary nail fixation and lag-screw placement mentions that a tip-to-apex distance (TAD) < 25 mm provides the most optimal fixation of the intramedullary nail. [17] This study reports that the ideal orientation of the lag-screw in the femoral head is a center-center position. This is a centralized position of the screw in an anterior-posterior radiographic view and in a lateral view. There are no objective measures that signify how much the locking screw position may deviate from the

center. One study did model the effect of different clinically encountered postoperative lag-screw positions to assess whether some positions increase the risk of high strains and yielding of the bone. [39] Their models suggested that the safest positions for the lag-screw are inferior-middle and inferior-posterior positions of the lag-screw in the femoral head. [39] In future studies, it would be interesting to assess if the locking screw position in the femoral head is a predictor for clinical outcome measures reported for patients with a bone-anchored prosthesis (BAP), such as quality of life, functioning levels, and gait quality of patients. [7] In addition, it may be interesting to investigate the difference of the surgical planning and postoperative positions of the implant in a retrospective patient study. The relation between implant position and possible complications after surgery and revalidation may provide information about the ability to predict clinical outcomes based on the implant position.

5.1 Limitations of the developed software

5.1.1 Interpretation of software functionalities

The different software functionalities are previously discussed in Chapter 2.4. The guidance during implant position was tested with the user tests described in Chapter 3. The feedback mechanism based on colour interpolation was able to guide the user to a close position (with a distance on average < 1 cm) in each visualization. Based on the objective measures, there was no statistical difference between how well different visualization guided the user towards the planned position. One visualization resulted in a nearly significantly ($p = 0.052$) longer elapsed time during implant placement and was therefore not advised. With the information gained in Chapter 3, it is recommended to allow combination of multiple viewing points and switching of desired viewing points during the different surgical steps. On the one hand, because of different user preferences and on the other hand because the experiment indicated that a combination of views is necessary for interpretation of distances in different axes.

Some functionalities are not yet fully developed, for example a guidance method for pre-drilling before locking screw insertion. Also, information from the surgical planning about the amputation level at the distal femur stump is not yet incorporated in the software and should be developed in future versions. For guidance of the drill used in locking screw placement, a feedback mechanism based on colour interpolation may suffice. The lack of a feedback mechanism during the user tests (Figure 3.2b) made it difficult for users to interpret whether the correct orientation of the instrument was obtained.

5.1.2 Accuracy of the software

Different inaccuracies are introduced in the design of the software, as elaborated on in Chapter 2.4. These can be divided into inaccuracies of the optical tracking camera, inaccuracies due to registration and inaccuracies due to the design of the developed software.

The evaluation of the registration errors is done by analysing the Target Registration Error (TRE) and Root Mean Square (RMS) values. By leaving out the manual indication of non-anatomical landmarks, a possible error introduced by how well the user indicates the landmarks is avoided. The RMS value obtained by registering on the fixed reference star provided a constant RMS value of 0.872 mm.

After the proof of concept experiment concerning guided drilling was performed, a difference was found in the distance measure that was derived from the scan and the measure calculated in the software. To analyse the origin of this disagreement, different follow-up experiments may be designed. For example, a possible cause may be an inaccurate registration of the implant model. To test the accuracy of the implant model registration, the implant can be positioned in an object like the calibration block used for drill tip calibration. By rotating the implant in the gap in the block, inaccuracies of the tip position and rotation may be visualized and quantified.

To validate the pitch, yaw and roll numbers calculated by the software, additional testing by inserting the implant in an object with a known rotational stance should be done.

The accuracy of the calculated distance and orientation numbers must be improved before clinical testing is possible. To eliminate the inaccuracy introduced by the PST Base camera, a different camera may be employed, such as the PST Base HD camera. This is a model similar to the PST Base camera, with improved specifications in the field of measurement volume, accuracy and frame rate. A study comparing the accuracy of the PST Base Camera to the PST Base HD camera and the Brainlab Curve camera may help to discover which camera is most accurate and should be used in clinical tests.

5.2 Recommendations

The tests performed during this graduation project were pre-clinical tests necessary to test the prototype. To move towards a more clinical test setting, some features that are now developed to work with the experimental set-up must be adapted to a clinical test setting.

For registration of the femur model, the reference star is attached at the designed position at the greater trochanter. In other words, the reference star position is known. However, for clinical testing of the software, the reference star would be attached to the greater trochanter at an unknown position. To register the femur of the patient to the femur model in the software, the position of the reference star must be determined. In the MITeC OR, the intra-operative CBCT scanner (Artis Zeego, Siemens) could be employed. This scanner is already registered to the camera coordinate system of the Brainlab navigation set-up. Optimally, the Brainlab camera may be used as an optical tracker for the developed software and the Artis Zeego may be operated for registration of the patient to the camera coordinate system. To enable this, some software features must be adjusted, such as the communication between the camera and the software and the registration of the femur.

To enable tracking of different instruments used during surgery, each instrument must be added to the visualization in the software and instruments may need to be calibrated before use. A calibration block with gaps of different diameters could be utilized for different instruments.

To quantify a decrease in radiation dose and a change in procedural time by utilizing the surgical navigation software, an experiment should be set up to compare the software to fluoroscopy. With sawbone femur models, the surgeon could insert the implant according to a certain planning twice. Once with the conventional fluoroscopy images and once with the surgical navigation software. Outcome measures could be the elapsed time and the radiation dose, as well as the accuracy with which the planned position was reached.

Another recommended adjustment of the software is to expand navigation to the guidance of patient-specific tibia implant insertion. A difference between the surgical procedures for femoral and tibial implants is the placement of two locking screws in tibial implants. Different modules in the software can be developed to load either a femur planning or a tibia planning.

5.3 Conclusion

To answer the research question and sub-questions defined in the introduction of this thesis, navigation software was designed and developed and tests with the prototype were conducted.

The first aim was to design and create a software application. The developed software is able to provide guidance during excavation of the femur, during implant placement and enables the user to position an instrument in the implant gap through which the locking screw is inserted. The surgical navigation is shown on a screen and the visualization is updated in real time. This replaces time-consuming repositioning of the conventional C-arm used to produce fluoroscopic images from different viewing points. The 3D models in the visualization give the surgeon an image in which depth is interpretable, as opposed to 2D fluoroscopic images.

To simplify implant and locking screw placement, a numerical and visual feedback mechanism is provided during guidance. The distance between the preoperatively planned position and the intraoperative position of the implant is calculated in real-time. A difference in orientation is calculated with the angles around three axes defined in the implant. The visual feedback mechanism is based on the numerical values and gives the implant or drill object in the software a colour representative of how close the user is to the planned position. These software features presented alongside the visualization provides the surgeon with information that fluoroscopic images cannot supply.

The second aim was to investigate how the surgical setting should be visualized to guide the user in the most efficient and convenient way. The virtual surgical setting can be visualized from different viewing points. Additionally, specific features were developed, e.g. to follow and focus on the tip of an instrument during surgery. The results of the user tests were that at least two 3D views were necessary to interpret depth in different axes. Users preferred a combination of views of which one provides an overview, similar to an anterior-posterior (AP) image. A visualization in which the AP overview was not present resulted in an almost significantly longer elapsed time. The saved distance and orientation parameters did not vary significantly between different visualizations.

The third aim was to test the accuracy of the developed software. The postoperative distance of the implant as well as the intraoperatively calculated distance was compared to the preoperatively planned implant position. The postoperative - preoperative distance was 3.6 mm and the intraoperative - preoperative distance was 9.91 mm. This disagreement could be explained by the inaccuracies introduced in the software. The experiment provided a proof of concept for guided excavation of the femur.

To conclude, this thesis presents a novel surgical navigation application providing the surgeon with live visualizations and information about the extent to which the surgical planning is reached. Additional tests and development should be aimed at improving the accuracy of the software and at experiments in a more clinical setting. More research is needed to quantify the amount with which the radiation exposure and surgical time could be decreased.

APPENDIX A

Unity Terminology

Term	Definition from Unity Glossary: [40]
Asset	"Any media or data that can be used in your game or Project. An asset may come from a file created outside of Unity, such as a 3D model, an audio file or an image."
Camera	"A component which creates an image of particular viewpoint in your scene. The output is either drawn to the screen or captured as a texture."
Component	"A functional part of a GameObject. A GameObject can contain any number of components. Unity has many built-in components, and you can create your own by writing scripts that inherit from MonoBehaviour."
GameObject	"The fundamental object in Unity scenes, which can represent characters, props, scenery, cameras, waypoints, and more. A GameObject's functionality is defined by the Components attached to it."
Inspector	"A Unity window that displays information about the currently selected GameObject, Asset or Project Settings, allowing you to inspect and edit the values."
Mesh Filter	"A mesh component that takes a mesh from your assets and passes it to the Mesh Renderer for rendering on the screen."
Mesh Renderer	"A mesh component that takes the geometry from the Mesh Filter and renders it at the position defined by the object's Transform component."
Parent	"An object that contains child objects in a hierarchy. When a GameObject is a Parent of another GameObject, the Child GameObject will move, rotate, and scale exactly as the Parent does. You can think of parenting as being like the relationship between your arms and your body; whenever your body moves, your arms also move along with it."

continues on next page

Plug-in	"A set of code created outside of Unity that creates functionality in Unity. There are two kinds of plug-ins you can use in Unity: Managed plug-ins (managed .NET-assemblies created with tools like Visual Studio) and Native plug-ins (platform-specific native code libraries)."
Project	"In Unity, you use a Project to design and develop a game. A Project stores all of the files that are related to a game, such as the Asset and Scene files."
Project Window	"In this view, you can access and manage the assets that belong to your project."
Scene	"A Scene contains the environments and menus of your game. Think of each unique Scene file as a unique level. In each Scene, you place your environments, obstacles, and decorations, essentially designing and building your game in pieces."
Script	"A piece of code that allows you to create your own Components, trigger game events, modify Component properties over time and respond to user input in any way you like."
World	"The area in your scene in which all objects reside. Often used to specify that coordinates are world-relative, as opposed to object-relative."

APPENDIX B

QUESTIONNAIRE

Tester:	1	Datum:	
Visualisatievolgorde:	ABC	Filename:	Tester1_ABC

1. Positie implantaat bereiken

Zou je het object graag ook van een andere kant willen bekijken, zo ja welke kant?

2. Visualisatie 1

Welke visualisatie keek je voornamelijk naar, links of rechts?

Links	Rechts

3. Visualisatie 2

Links	Rechts

4. Visualisatie 3

Links	Rechts

5. Schroefgat in visualisatie 3

Links	Rechts

Welke visualisatie vond je het best werken?

A	B	C

[1] R. A. Leijendekkers, G. van Hinte, J. P. Frölke, H. van de Meent, M. W. Nijhuis-van der Sanden, and J. B. Staal, "Comparison of bone-anchored prostheses and socket prostheses for patients with a lower extremity amputation: a systematic review," *Disability and Rehabilitation*, vol. 39, no. 11, pp. 1045–1058, may 2017. [Online]. Available: https://www.tandfonline.com/doi/full/10.1080/09638288.2016.1186752

[2] J. S. Hebert, M. Rehani, and R. Stiegelmar, "Osseointegration for Lower-Limb Amputation," *JBJS Reviews*, vol. 5, no. 10, p. e10, oct 2017. [Online]. Available: http://insights.ovid.com/crossref?an=01874474-201710000-00003

[3] A. Thesleff, R. Brånemark, B. Håkansson, and M. Ortiz-Catalan, "Biomechanical Characterisation of Bone-anchored Implant Systems for Amputation Limb Prostheses: A Systematic Review," *Annals of Biomedical Engineering*, pp. 1–15, jan 2018. [Online]. Available: http://link.springer.com/10.1007/s10439-017-1976-4

[4] H. Van De Meent, M. T. Hopman, and J. P. Frölke, "Walking ability and quality of life in subjects with transfemoral amputation: A comparison of osseointegration with socket prostheses," *Archives of Physical Medicine and Rehabilitation*, vol. 94, no. 11, pp. 2174–2178, 2013. [Online]. Available: http://www.osseointegration.eu/wp-content/uploads/2015/10/PDF-van-ILP-artikel.pdf

[5] K. Hagberg, E. Hansson, and R. Brånemark, "Outcome of Percutaneous Osseointegrated Prostheses for Patients With Unilateral Transfemoral Amputation at Two-Year Follow-Up," *Archives of Physical Medicine and Rehabilitation*, vol. 95, no. 11, pp. 2120–2127, nov 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0003999314005036

[6] C. F. van Eck and R. L. McGough, "Clinical outcome of osseointegrated prostheses for lower extremity amputations," *Current Orthopaedic Practice*, vol. 26, no. 4, pp. 349–357, 2015. [Online]. Available: http://content.wkhealth.com/linkback/openurl?sid=WKPTLP:landingpage{&}an=01337441-201507000-00006

[7] R. A. Leijendekkers, J. B. Staal, G. van Hinte, J. P. Frölke, H. van de Meent, F. Atsma, M. W. G. Nijhuis-van der Sanden, and T. J. Hoogeboom, "Long-term outcomes following lower extremity press-fit bone-anchored prosthesis surgery: a 5-year longitudinal study protocol," *BMC Musculoskeletal Disorders*, vol. 17, no. 1, p. 484, dec 2016. [Online]. Available: http://bmcmusculoskeletdisord.biomedcentral.com/articles/10.1186/s12891-016-1341-z

[8] R. Tranberg, R. Zügner, and J. Kärrholm, "Improvements in hip- and pelvic motion for patients with osseointegrated trans-femoral prostheses," *Gait and Posture*, vol. 33, no. 2, pp. 165–168, feb 2011. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0966636210003929

[9] R. Atallah, R. A. Leijendekkers, T. J. Hoogeboom, and J. P. Frölke, "Complications of bone-anchored prostheses for individuals with an extremity amputation: A systematic review," *PLoS ONE*, vol. 13, no. 8, p. e0201821, aug 2018. [Online]. Available: https://dx.plos.org/10.1371/journal.pone.0201821

[10] M. A. Muderis, A. Khemka, S. J. Lord, H. Van De Meent, and J. P. M. Frolke, "Safety of osseointegrated implants for transfemoral amputees: A two-center prospective cohort study," *Journal of Bone and Joint Surgery - American Volume*, vol. 98, no. 11, pp. 900–909, jun 2016. [Online]. Available: http://insights.ovid.com/crossref?an=00004623-201606010-00003

[11] K. Hagberg and R. Brånemark, "One hundred patients treated with osseointegrated transfemoral amputation prostheses—Rehabilitation perspective," *The Journal of Rehabilitation Research and Development*, vol. 46, no. 3, p. 331, 2009. [Online]. Available: http://www.rehab.research.va.gov/jour/09/46/3/hagberg.html

[12] H. H. Aschoff, R. E. Kennon, J. M. Keggi, and L. E. Rubin, "Transcutaneous, Distal Femoral, Intramedullary Attachment for Above-the-Knee Prostheses: An Endo-Exo Device," *The Journal of Bone and Joint Surgery-American Volume*, vol. 92, no. Suppl 2, pp. 180–186, dec 2010. [Online]. Available: http://content.wkhealth.com/linkback/openurl?sid=WKPTLP:landingpage{&}an=00004623-201012002-00018

[13] D.-L. Juhnke, J. P. Beck, S. Jeyapalina, and H. H. Aschoff, "Fifteen years of experience with Integral-Leg-Prosthesis: Cohort study of artificial limb attachment system," *JRRD*, vol. 52, no. 4, 2015. [Online]. Available: https://search.proquest.com/openview/10e11a4034af91ad94168d8a2f8408b2/1?pq-origsite=gscholar{&}cbl=48772

[14] M. Al Muderis, H. H. Aschoff, B. Bosley, G. Raz, L. Gerdesmeyer, and B. Burkett, "Direct skeletal attachment prosthesis for the amputee athlete: the unknown potential," *Sports Engineering*, vol. 19, no. 3, pp. 141–145, sep 2016. [Online]. Available: http://link.springer.com/10.1007/s12283-016-0196-8

[15] M. Al Muderis, W. Lu, and J. J. Li, "Osseointegrated Prosthetic Limb for the treatment of lower limb amputations : Experience and outcomes." *Der Unfallchirurg*, vol. 120, no. 4, pp. 306–311, 2017. [Online]. Available: https://link.springer.com/content/pdf/10.1007{%}2Fs00113-016-0296-8.pdfhttp://link.springer.com/10.1007/s00113-016-0296-8{%}5Cnhttp://www.ncbi.nlm.nih.gov/pubmed/28070628

[16] J. P. M. Frölke, R. A. Leijendekkers, and H. van de Meent, "Osseointegrated prosthesis for patients with an amputation," *Der Unfallchirurg*, vol. 120, no. 4, pp. 293–299, apr 2017. [Online]. Available: http://link.springer.com/10.1007/s00113-016-0302-1

[17] R. J. Lilly, D. M. Koueiter, K. C. Graner, G. P. Nowinski, J. Sadowski, and K. D. Grant, "Computer-assisted navigation for intramedullary nail fixation of intertrochanteric femur fractures: A randomized, controlled trial," *Injury*, dec 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0020138317308598?via{%}3Dihub

[18] G. Whatling and L. Nokes, "Literature review of current techniques for the insertion of distal screws into intramedullary locking nails," *Injury*, vol. 37, no. 2, pp. 109–119, feb 2006. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0020138305003803?via{%}3Dihub

[19] K. S. Leung, N. Tang, L. W. H. Cheung, and E. Ng, "Image-guided navigation in orthopaedic trauma." *The Journal of bone and joint surgery. British volume*, vol. 92, no. 10, pp. 1332–7, oct 2010. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/20884967

[20] D. Grimwood and J. Harvey-Lloyd, "Reducing intraoperative duration and ionising radiation exposure during the insertion of distal locking screws of intramedullary nails: a small-scale study comparing the current fluoroscopic method against radiation-free, electromagnetic navigation," *European Journal of Orthopaedic Surgery and Traumatology*, vol. 26, no. 8, pp. 867–876, dec 2016. [Online]. Available: http://link.springer.com/10.1007/s00590-016-1835-2

[21] B. Diotte, P. Fallavollita, L. Wang, S. Weidert, E. Euler, P. Thaller, and N. Navab, "Multi-modal intra-operative navigation during distal locking of intramedullary nails," *IEEE Transactions on Medical Imaging*, vol. 34, no. 2, pp. 487–495, feb 2015. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6915708

[22] T. Leloup, W. El Kazzi, F. Schuind, and N. Warzée, "A Novel Technique for Distal Locking of Intramedullary Nail Based on Two Non-constrained Fluoroscopic Images and Navigation," *IEEE Transactions on Medical Imaging*, vol. 27, no. 9, pp. 1202–1212, sep 2008. [Online]. Available: http://ieeexplore.ieee.org/document/4471973/

[23] R. Londei, M. Esposito, B. Diotte, S. Weidert, E. Euler, P. Thaller, N. Navab, and P. Fallavollita, "The 'augmented' circles: A video-guided solution for the down-the-beam positioning of im nail holes," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Springer, Cham, 2014, vol. 8498 LNCS, pp. 100–107. [Online]. Available: http://link.springer.com/10.1007/978-3-319-07521-1{_}11

[24] ——, "Intra-operative augmented reality in distal locking," *International Journal of Computer Assisted Radiology and Surgery*, vol. 10, no. 9, pp. 1395–1403, sep 2015. [Online]. Available: http://link.springer.com/10.1007/s11548-015-1169-2

[25] J. Choi, J. Kim, J. Y. Hwang, M. Je, J. Y. Kim, and S. Y. Kim, "A novel smart navigation system for intramedullary nailing in orthopedic surgery," *PLoS ONE*, vol. 12, no. 4, p. e0174407, apr 2017. [Online]. Available: http://dx.plos.org/10.1371/journal.pone.0174407

[26] Y. Zhu, H. Chang, Y. Yu, W. Chen, S. Liu, and Y. Zhang, "Meta-analysis suggests that the electromagnetic technique is better than the free-hand method for the distal locking during intramedullary nailing procedures," *International Orthopaedics*, vol. 41, no. 5, pp. 1041–1048, may 2017. [Online]. Available: http://link.springer.com/10.1007/s00264-016-3230-3

[27] H. Andruszkow, M. Frink, C. Frömke, A. Matityahu, C. Zeckey, P. Mommsen, S. Suntardjo, C. Krettek, and F. Hildebrand, "Tip apex distance, hip screw placement, and neck shaft angle as potential risk factors for cut-out failure of hip screws after surgical treatment of intertrochanteric fractures," *International Orthopaedics*, vol. 36, no. 11, pp. 2347–2354, nov 2012. [Online]. Available: http://link.springer.com/10.1007/s00264-012-1636-0

[28] X. Chen, L. Xu, H. Wang, F. Wang, Q. Wang, and R. Kikinis, "Development of a surgical navigation system based on 3D Slicer for intraoperative implant placement surgery," *Medical Engineering & Physics*, vol. 41, pp. 81–89, mar 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1350453317300061

[29] T. M. Ecker, M. Tannast, and M. Puls, "Computer-Assisted Orthopedic Surgery," in *Intraoperative Imaging and Image-Guided Therapy*. New York, NY: Springer New York, 2014, pp. 661–675. [Online]. Available: http://link.springer.com/10.1007/978-1-4614-7657-3{_}50

[30] T. M. Peters and C. A. Linte, "Image-guided interventions and computer-integrated therapy: Quo vadis?" *Medical Image Analysis*, vol. 33, pp. 56–63, oct 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1361841516300780?via{%}3Dihub

[31] D. M. Kahler, "Image Guidance: Fluoroscopic Navigation," *Clinical Orthopaedics and Related Research*, vol. 421, pp. 70–76, apr 2004. [Online]. Available: http://content.wkhealth.com/linkback/openurl?sid=WKPTLP:landingpage{&}an=00003086-200404000-00014

[32] Brainlab, "Curve - Image Guided Surgery," p. 18, 2018. [Online]. Available: https://www.brainlab.com/wp-content/uploads/2014/01/Brochure-Curve.pdf

[33] PS-Tech, "PST Base Specifications," 2018. [Online]. Available: http://www.ps-tech.com/optical-trackers/optical-tracker-pst-base/specifications

[34] Sales Oceanz, "How to design parts for SLS 3D Printing." [Online]. Available: https://www.3dhubs.com/knowledge-base/how-design-parts-sls-3d-printing{#}author

[35] M. A. Seif, R. Umeda, and H. Higa, "An attempt to control a 3D object in medical training system using leap motion," in *ICIIBMS 2017 - 2nd International Conference on Intelligent Informatics and Biomedical Sciences*, vol. 2018-Janua. IEEE, nov 2018, pp. 159–162. [Online]. Available: http://ieeexplore.ieee.org/document/8279705/

[36] R. Wang, J. Yao, L. Wang, X. Liu, H. Wang, and L. Zheng, "A surgical training system for four medical punctures based on virtual reality and haptic feedback," in *2017 IEEE Symposium on 3D User Interfaces, 3DUI 2017 - Proceedings*. IEEE, 2017, pp. 215–216. [Online]. Available: http://ieeexplore.ieee.org/document/7893348/

[37] J. W. Meulstee, J. Nijsink, R. Schreurs, L. M. Verhamme, T. Xi, H. H. K. Delye, W. A. Borstlap, and T. J. J. Maal, "Toward Holographic-Guided Surgery," *Surgical Innovation*, p. 155335061879955, sep 2018. [Online]. Available: http://journals.sagepub.com/doi/10.1177/1553350618799552

[38] R. M. Barbosa, L. Serrador, B. Santos, M. V. Silva, E. De Momi, and C. Santos, "3DSlicer module to perform registration: An intraoperative situation," in *2017 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*. IEEE, apr 2017, pp. 55–60. [Online]. Available: http://ieeexplore.ieee.org/document/7964052/

[39] J. M. Goffin, P. Pankaj, and A. H. Simpson, "The importance of lag screw position for the stabilization of trochanteric fractures with a sliding hip screw: A subject-specific finite element study," *Journal of Orthopaedic Research*, vol. 31, no. 4, pp. 596–600, apr 2013. [Online]. Available: http://doi.wiley.com/10.1002/jor.22266

[40] Unity Technologies, "Unity - Manual: Glossary," 2018. [Online]. Available: https://docs.unity3d.com/Manual/Glossary.html