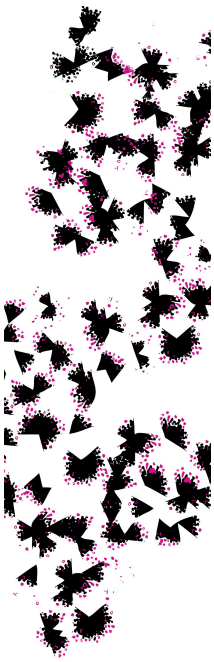

Design of an Ultrasound Guided Breast Biopsy End-effector

R.J.F. (Ruud) Spoor

MSc Report

Committee:
Prof.dr.ir. S. Stramigioli
Dr. F.J. Siepel
Prof.dr.ir. D.M. Brouwer
V. Groenhuis, MSc

November 2016

047RAM2016
Robotics and Mechatronics
EE-Math-CS
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

Summary

In this thesis, the design and evaluation of a robotic end-effector are described. This thesis is part of the MURAB project in which the possibility to increase efficiency, accuracy and reliability of ultrasound guided breast biopsy, by merging MRI data, is investigated. The designed end-effector, to be mounted on a commercial robotic arm, contains an ultrasound probe, a mechatronic needle guide and a camera. Quantitative requirements set up in the problem definition are evaluated with the help of a physical prototype designed and build during the thesis. Recommendations for future iterations will be given based on the evaluated prototype. Safety is taken into account while designing the prototype, as future devices will be used in a medical environment.

Contents

1 Project description **1**
1.1 Introduction . . . 1
1.2 MURAB project . . . 1
1.3 Project aim . . . 2
1.4 Report outline . . . 2

2 Background **3**
2.1 Robot assisted image guided breast biopsy . . . 3
2.2 Safety of robot manipulators . . . 4
2.3 Workflow needle biopsy using ultrasound . . . 4

3 Design of end-effector **6**
3.1 Proposed workflow of robot guided breast biopsy using ultrasound . . . 6
3.2 End-effector requirements . . . 7
3.3 Design overview . . . 8
3.4 Detailed description . . . 10
3.5 Prototype construction . . . 15

4 Evaluation of design **19**
4.1 Safety evaluation . . . 19
4.2 End-effector interference . . . 20
4.3 Workspace reachability . . . 20
4.4 Force disturbance . . . 22
4.5 Positional accuracy . . . 23
4.6 Manipulation by robot . . . 23
4.7 Required components . . . 25

5 Conclusion **26**
5.1 Conclusion . . . 26
5.2 Further work . . . 26

A Appendix 1 **27**
A.1 Stepper motor details . . . 27
A.2 Stepper drive electronics . . . 27
A.3 Mechanism driving software . . . 27

B Appendix 2 **28**
B.1 Mechanism inverse velocity kinematics . . . 28

B.2 Workspace analysis 30
B.3 Implementation code mechanism elastostatics 34

C Appendix 3 **39**
C.1 Python 39
C.2 Freecad 39
C.3 Mbed 39

Bibliography **40**

1 Project description

1.1 Introduction

Breast cancer is the second leading cause of cancer death for women. The American Cancer Society's estimates for the United States stated in 2016: "246,660 new cases of invasive breast cancer and 40,450 women will die from breast cancer"(ref: American Cancer Society 2016). First signs of breast cancer are often found during the screening phase. Mammography (x-ray) can show suspicious regions in the breast. Additional imaging techniques like Magnetic Resonance Imaging (MRI) and Ultrasound (US) are used for further investigation. Compared to mammography, ultrasound can detect changes in the breast that can be felt by palpation in real time, but not seen on mammography images. Ultrasound is also capable of making a distinction between fluid filled cysts and solid masses.

MRI provides images with higher resolution compared to mammography and ultrasound. MRI and mammography imaging are for instance used to screen high risk patients or confirming known tumors before surgery. MRI imaging is seen as the most sensitive technique with respect to breast cancer. Confirmation about the nature of a lesion found in a breast can however only be provided by biopsy.

In the optimal case, biopsy is performed by a hollow needle under MRI due to the high quality of images. MRI is however not real-time and it is difficult to manually position a biopsy needle in the confined space of an MRI bore. Needle positioning devices are difficult to implement as MRI makes use of huge magnets and magnetic sensors. Use of magnetic materials like steel and copper could therefore be dangerous and furthermore distort the magnetic resonance images. Finally, MRI is one of the most expensive piece of medical equipment. MRI visits should therefore be kept to a minimum during the whole diagnostics workflow.

On the other hand, ultrasound can be used as an alternative for image guided needle biopsy. Like MRI, it is harmless to the human body, and besides this, ultrasound is also real-time and relatively cheap. Use of magnetic materials like steel are allowed in the vicinity of the device and there is space around the ultrasound probe and patient to maneuverer a biopsy needle. Disadvantage of ultrasound compared to MRI is its lower image resolution. Smaller lesions and other information which can be seen on the MRI, cannot be seen with ultrasound.

1.2 MURAB project

The MURAB project (MRI Ultrasound Robotic Assisted Biopsy) investigates the combination of both MRI and ultrasound imaging with the use of robotics in the procedure of image guided biopsy. Instead of bringing a biopsy needle in the MRI bore, MRI data and patient are taken out of the machine. A robotic arm will then assists in the biopsy procedure outside of the MRI room. MRI data is merged with echography and locations of lesions invisible on ultrasound are determined based on MRI data. An off-the-shelf robotic arm from KUKA will be used to steer an ultrasound probe. After localization of the targeted lesion, a needle guide is brought into position with the use of a mechatronic end-effector. Verification of data and insertion of the biopsy needle into the human breast will be performed by a radiologist. It is expected that this kind of image guided biopsy will be performed in the future with higher accuracy, efficiency and increased reliability due to the MURAB project.

Seven partners in MURAB project are KUKA, Siemens, University of Verona, Medical University of Vienna, RadboudUMC, Stichting Ziekenhuisgroep Twente and University of Twente.

1.3 Project aim

The University of Twente, the university this part of the research is performed, is responsible for designing and implementing a robotic system performing scanning and biopsy of a breast with a robotic arm. **This study will focus on the prototype design and evaluation of an end-effector to be connected on a robotic arm which contains an ultrasound probe for scanning and a mechatronic needle guide for image guided breast biopsy**. An acoustically transparent force array for elastography and a camera for marker registration would have to be implemented in a future design these components are however outside the scope of this particular research project.

1.4 Report outline

This report consists out of five chapters. First, the problem is described. Then background information on the problem is given in the second chapter. In the third chapter, quantitative requirements are set up. Latest prototype design, based on the requirements, is described with informed about the basic features and their intended use. Details about specific design features can be found in the appendix, as a reader does not have to read these details to understand the design. Fourth chapter covers the evaluation of the design. A physical prototype is build and used as a tool for evaluation. As the requirements from the third part are given quantitatively, measurements can be used for an objective evaluation. Conclusions and recommendations can be found in the Fifth and last chapter of this report.

2 Background

2.1 Robot assisted image guided breast biopsy

Robot assisted ultrasound and image guided biopsy is not a new research field and several studies have explored the possibilities. The advantage of robotics in echography is increased accuracy and dexterity. Some ultrasound probe guiding robots are used to reduce neck and shoulder injury which are common among ultrasound operators. A feasibility study on the design of a guided ultrasound probe by robot manipulator to prevent strain injury is done by Salcudean S.E. (1999).

Other research is focused on creating better 3D models of internal parts of the human body using ultrasound transducers manipulated by robotic arms. Compared to manual operation of ultrasound probes in the creation of 3D models, robotic assisted methods perform better. In Pierrot F. (1999) a robot guided probe is used for 3D artery reconstruction. Pierrot initially used an commercial robot for a feasibility study, but uses a custom designed robot for the final implementation. Pierrot used a custom robot, as it was expected that safety requirements were more easily met with a custom design. Another example of a custom-made ultrasound guiding robot is designed by Smith-Guerin N. (2003). The robotic system allows remote control by expert ultrasound operators. Their concept is based on medical user specifications. Off the shelf robotic manipulators are also used for probe guidance in for instance Mathiassen K. (2016) to reduce cost of a complete robotic system. A UR5 industrial robot from universal robotics is used for tele-operation and relieve repetitive strain injuries which are common among ultrasound specialists.

Figure 2.1: Ultrasound robot created by Salcudean et all.

In the MURAB project, the project this thesis contributes to, it is chosen to make use of a “LWR 7 r800” robot (KUKA,Munich,Germany). An industrial robot manipulator specifically designed for safe human robot interaction. It is expected that a standard industrial robot would be more cost effective than a custom designed robot. This section covers only a small piece of the research in the field of ultrasound robotics. To get a full view of ultrasound robotics used in the medical sector, one is referred to Priester M. (2013) for a full review over the last two decades.

The combination of ultrasound with mechatronic needle guidance is less common than ultrasound probe manipulation alone.

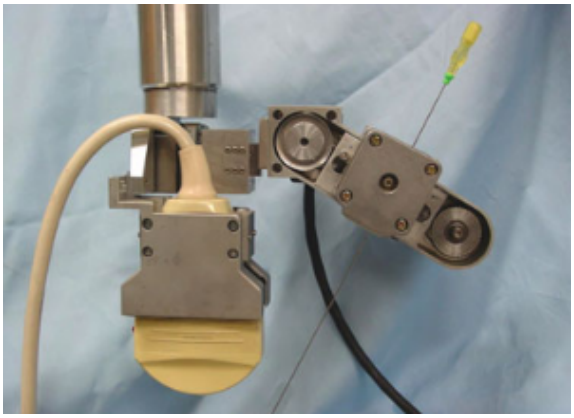

Figure 2.2: Needle steering end-effector by J. Hong

In Hong J. (2004), a simple 2 degrees of freedom steering mechanism to guide a needle tip into a gall bladder is presented. Ultrasound is used for visual servoing the biopsy needle. Measurement results show positional accuracies of 2.3 mm to 5.2 mm with target movements of 10 mm/s representing respiration. Delay due to visual processing is stated to be the main source of positioning error. Chatelain P. (2015) is using two robotic arms to steer a flexible needle with the help of 3D ultrasound. Visual servoing is again used as feedback method and results in positional errors of 1mm. Respiration and tissue deformation are however not taken into account in experiments performed by Chatelain P. (2015)

2.2 Safety of robot manipulators

Robots use high power motors which drive metal limbs with high velocity. Collisions with human bodies can cause serious injuries and even death. Moreover, output forces of even small robots can be high enough to lethally crush a human body. To ensure safety, it is common to put industrial robots inside safety cages and lock the cages with safety doors. It is however more and more desired nowadays to let humans and robots collaborate with each other. Safety in such settings is paramount. ISO-10218, which defines new collaborative operation requirements for industrial robots is set up as guideline to ensure safe human robot interaction. However, some find the standards in ISO-10218 too restrictive. Research is performed on the impact of robots colliding into humans by Haddadin S. (2007). Using crash test dummies at a car crash test facility, various types of robots and collisions are investigated. The authors where surprised by the lack of damage resulting from specific collisions. Yamada Y. (1996) uses the pain tolerance of humans as a measure to limit maximum end-effector velocity in case of a collision and ensure human robot interaction safety that way. The LWR 7 r800 robot to be used in the MURAB project is certified and specifically designed to work with and around humans. KUKA uses extensive safety measures in both hardware and software to ensure safe collaboration between humans and its robots.

2.3 Workflow needle biopsy using ultrasound

Before the actual design of an image guided breast biopsy robot, knowledge on the current manual method of biopsy using ultrasound guidance is needed. A visit to ZGT hospital gave insight in the process. A global description of the procedure is given below. For a full detailed description of core-needle biopsy using ultrasound, see Rocha R. (2013). Following current manual procedure, a new biopsy procedure using robotics is described.

An ultrasound expert is looking at overall mammography (x-ray based) scan data and is requested to investigate specific regions of a breast. The mammogram images give an overview of the internals of a human breast. Ultrasound is used to inspect areas of interest with more detail. An ultrasound scan is made around the area of interest and suspicious tissue is inspected for specific criteria. When a radiologist is not sure about the nature of specific tissue, it can be decided to perform a biopsy. Biopsy is needed as definitive answer about the nature of specific tissue can only be determined with samples in a lab. The location of probe and suspicious tissue are marked with a pen as a line on the skin. Anesthetics are applied with a syringe (without

ultrasound probe) through the pre-planned path of the biopsy needle and around the suspicious lesion.

While the anesthetics do their work, tools for the biopsy procedure are prepared. Biopsy needle and trigger mechanism are placed on a table together with a small scalpel. The ultrasound probe is put in a plastic sleeve and secured with rubber bands. Finally, the reason for biopsy and the procedure to follow are explained to the patient. Biopsy needle and trigger mechanism are shown to the patient and triggered once, so the patients knows what to expect during the procedure. A small incision is made on the skin with a scalpel to allow access for the biopsy needle. Point of entry is chosen in such a way that the biopsy needle has the shortest path through tissue to a targeted lesion, but remains about perpendicular to the chest wall. Using gauze pads, blood is cleaned of the skin. The plastic sleeve around the ultrasound probe prevents cleaning of the hardware after biopsy procedure. Actual biopsy starts by placing the ultrasound probe back on the skin and suspicious tissue will be found by making use of the marking line. While keeping the suspicious tissue insight with the ultrasound probe, biopsy needle is brought in close to the probe through the incision. The medical professional performing biopsy pays close attention that the needle tip is always in view of the ultrasound probe. Biopsy needle is steered toward the lesion and final position of the needle tip is validated. By rotating the ultrasound probe in a perpendicular plane with respect to the needle, sideways insertion is also validated. If placement is correct, the patient is informed that the biopsy needle is about to trigger. Medical professional pushes the trigger and needle tip shoots forward, grabbing a small piece of suspicious tissue. The biopsy needle is taken out of the body and tissue is put in a container for inspection in the laboratory. The process is repeated three to five times to make sure suspicious tissue is actually grabbed. The whole procedure takes about 15 minutes

Figure 2.3: Arrangement of materials required for the performance of core biopsy of breast. Fenestrated surgical drape, scalpel blade, 18-gauge to 22-gauge needles, 10 ml syringe, vial with 10% formalin, sterile gloves and gauze pads, automated core biopsy device and 14-gauge core biopsy needle. (copied from ...)

3 Design of end-effector

3.1 Proposed workflow of robot guided breast biopsy using ultrasound

Compared to current manual biopsy procedure using ultrasound, robots can improve the handling of ultrasound probe and biopsy needle. Compared to humans, robots are more accurate at placing object in space and can hold that accuracy for prolonged time. Robots in general are able to track a desired path with greater precision and perform it in a consistent way. Furthermore, they can apply constant amounts of force with accuracies superior to human beings. Consistent application of pressure and constant movement are factors determining the quality of ultrasound images and their reconstruction into 3D. It is proven that use of robots can improve the quality of ultrasound due to the improved application of force and tracking of desired paths Pierrot F. (1999). The main expected benefit of a robotic manipulator for biopsy is the increase in accuracy and reliability of biopsy compared to manual procedures. As a biopsy needle can be steered with greater precision, it is more likely the targeted tissue is actually grabbed for testing instead of healthy tissue. However the main reason of using the robotic arm in the MURAB project is to grab specific tissue, visible on MRI, but not on ultrasound. Typical are lesions smaller than 5mm which are hard to detect using ultrasound.

A robotic arm is used to position ultrasound probe and needle guide in 3D space. After the medical professional has reviewed previously made images of MRI and/or ultrasound, he or she orders the robot to scan the full breast. A full ultrasound scan of the breast is needed as input for an algorithm creating a mechanical deformation model. Input from an acoustically transparent force array sensor placed underneath the ultrasound probe is also used as data for a mechanical model. The breast deformation model will later on be used to merge MRI, elastography (force sensor data) and ultrasound data to obtain extensive information about the scanned breast.

When scanning is performed, a plan is made of how to proceed. What path should the biopsy needle follow to obtain the desired tissue and are there obstructions or limitations? Several path options generated based on different criteria could be shown to the operator. It is the operators responsibility to choose the optimal path. Anesthetics are applied similar to the manual procedure by using a syringe and small diameter needle.

The radiologist makes a small incision through the skin with a scalpel to allow access for the biopsy needle. If needed, gauze pads are used to clean blood from the incision. A needle guiding mechanism mounted on the end-effector is positioned in place and actual biopsy can begin.

The Ultrasound operator guides a biopsy needle through the needle guide and through the incision. While feeling the resistance of the needle, he/she advances the biopsy needle further into the breast until either a depth stop is reached or the medical professional is satisfied with the placement of the needle with respect to the suspicious tissue. Robot arm and needle guide make sure the biopsy needle is following the preplanned path to the tissue. Placement of the needle is evaluated by locating the tip in the perpendicular planes. If satisfied, medical professional triggers the needle and the tip shoots forward. A small piece of tissue is grabbed. The biopsy needle is retracted by the medical operator.

As robotic assisted biopsy is expected to target lesions more accurately compared to the manual procedure, only one or two tissue samples are taken from the suspicious lump, using the same incision as point of entry so least amount of tissue damage is left behind.

3.2 End-effector requirements

Before design of an end-effector can can be performed, requirements and design guidelines have to be described. Safety is the main requirement in the MURAB project as the system is expected to be used in a medical setting. Safety has to be taken into account right from the start of the design process, as it is much more difficult to make rigorous design changes due to safety concerns later on in the process.

This thesis focuses the end-effector of a robotic system for ultrasound guided needle biopsy. It is assumed that the KUKA arm, to be used in the project, is already safe. Therefore, only the safety of the end-effector is evaluated in this report.

Apart from safety, there are also functional requirements for the end-effector. These requirements are also stated. An ultrasound probe to be placed on the human body must have full spatial freedom to allow scanning of the whole breast. Preferably, the probe should also be able to scan other parts of the body. It has to be taken into account that the end-effector does not interfere with the patient or robotic arm.

Actual biopsy is performed by a hollow core needle. The needle must be guided into position with help of mechatronics. A guiding mechanism must have three degrees of freedom in the planar field of view of the ultrasound probe to reach all possible locations of suspicious lesions in a human breast. The needle guide mechanism cannot enter the body of the patient, but must have a center of rotation for the needle which can be placed anywhere on the patient's skin. Such a center of rotation allows for the smallest incision in the skin and therefore the least amount of scar tissue after the procedure.

The end-effector should accommodate multiple probes. An assumed field of view for an ultrasound probe is set as specified in figure 3.1 Linear Transducers with 7 to 12 Mhz are often used due to their higher resolution at lower depth. The assumed field of view is based on such a transducer. Figure 3.1 also shows minimal and maximal breast sizes. The variation in breast size, based on work from Huang S.Y. (2011), determines the workspace specification of the needle guide mechanism. The center of rotation of a biopsy needle must be placed on the skin of the patient to minimize scar tissue. The needle must be able to rotate at least from horizontally with the chest wall, to 30 degrees in upward direction.

As a medial expert will insert the biopsy needle inside the patient's body, the mechanism must be able to cope with force disturbances of about 10N caused by the medical professional. The value of 10 N is a rough estimate. The needle guide mechanism must be stiff enough to reduce deviations of the needle tip caused by manual insertion.

Lesions of about 5mm or higher are targeted for biopsy. A needle tip therefore needs to be placed with an accuracy which is at least smaller than +-2.5mm. A maximum deviation of +-1mm is preferred however as image resolution of ultrasound probes are expected to be improved in the coming years.

To merge MRI and Ultrasound data, a mechanical deformation model of the breast is most probably needed. The end-effector must therefore have an force array sensor to obtain data for this deformation model. The acoustically transparent sensor must be placed underneath the ultrasound probe to allow easy integration of all data.

To check if a needle is actually inside the targeted lesion before actual biopsy, Probe and force sensor should be rotated around an axis normal to the skin surface in such a way that targeted lesion is still in view, but the needle is now perpendicular to the field of view of the ultrasound probe. This is needed to check if a needle is actually going to pierce a lesion when triggered.

Last requirement which must be met, is the mechanical interface between robot and end-effector. The end-effector must in some way be mounted on the robotic arm its flange. The

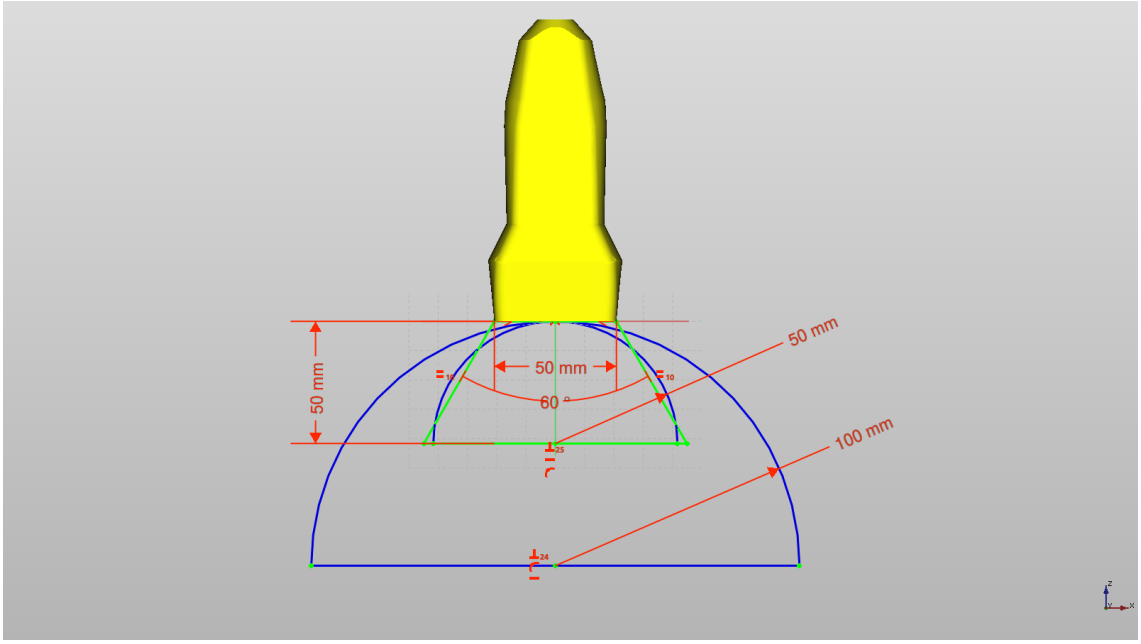

Figure 3.1: Specified workspace for needle and needle guide. In green is the outline of the field of view of the probe. In blue is shown the minimal and maximum outline of breasts.

robot arm can handle a weight of about 7 Kg. The end-effector must therefore not exceed this weight limit. A lower weight is preferred due to safety concerns.

Besides the hard requirements as described above, there are also some additional desired requirements which should be met. Orientation of the probe with respect to the arm is not fully clear at the moment. Several mounting options with respect to orientation should therefore be possible. Furthermore, it is desired to have a camera mounted on the end-effector which is capable of detecting markers on the humand skin as well as showing the insertion point of the needle. Again, it is not fully clear what camera type is going to be used. It would therefore be desired, at least for a prototype, to allow space for a camera module. Finally, it is desired to have a probe holder which accepts a variety of ultrasound probes and allows easy replacement of these probes. This also means there should be some way of registering a new ultrasound probe to the needle and needle guide. All requirements are summarized in table 3.1 for later reference.

3.3 Design overview

The end-effector, based on the requirements, must contains 5 basic components. An ultrasound probe for scanning internals of the human body. A force array sensor for elastography to be merged with ultrasound data, a camera for visual registration and a mechanism for guiding a biopsy needle. Part of the end-effector also comprises some component which handles

Requirement	Ideal value	Unit
Safety: risk of identified hazardous scenarios	level ≤ 5	risk level
End-effector, robot arm and patient do not interfere during procedure	true	boolean (true/false)
Needle and needle guide can completely reach defined workspaces	true	boolean (true/false)
Needle mechanism can handle force disturbances	10	Newton (N)
Positional accuracy of needle tip	±2.5	millimeter (mm)
KUKA arm can manipulate (mass) end-effector	<7	kilogram (Kg)
End-effector contains all required components	true	boolean (true/false)

Table 3.1: Quantitative summary of main requirements

the interface to a Kuka arm. Based on the requirements, global architecture of the end-effector is basically fixed. The force array sensor must be placed coaxially underneath the ultrasound probe and must therefore be acoustically transmissible. This force Array sensor also requires contact with the human body in order to perform elastography. Needle guide mechanism and camera can therefore only be placed on the side or top of the ultrasound probe depending on placement of the interface between robot arm and end-effector. The camera needs a clear field of view and must not be obstructed by any of the other components.

When focusing on the needle guide mechanism, there are two general options. A parallel actuated configuration or a serial actuated configuration. Serial mechanisms are relatively simpler in design and have in general a larger workspace compared to parallel actuated mechanisms. Parallel mechanisms are more accurate compared to serial manipulators and in general stiffer. It is decided to use a parallel mechanism for the needle guide due to its stiffness and accuracy characteristics. Figure 3.2 show a computer model of the complete end-effector. Detailed explanations for the components are described in the next section.

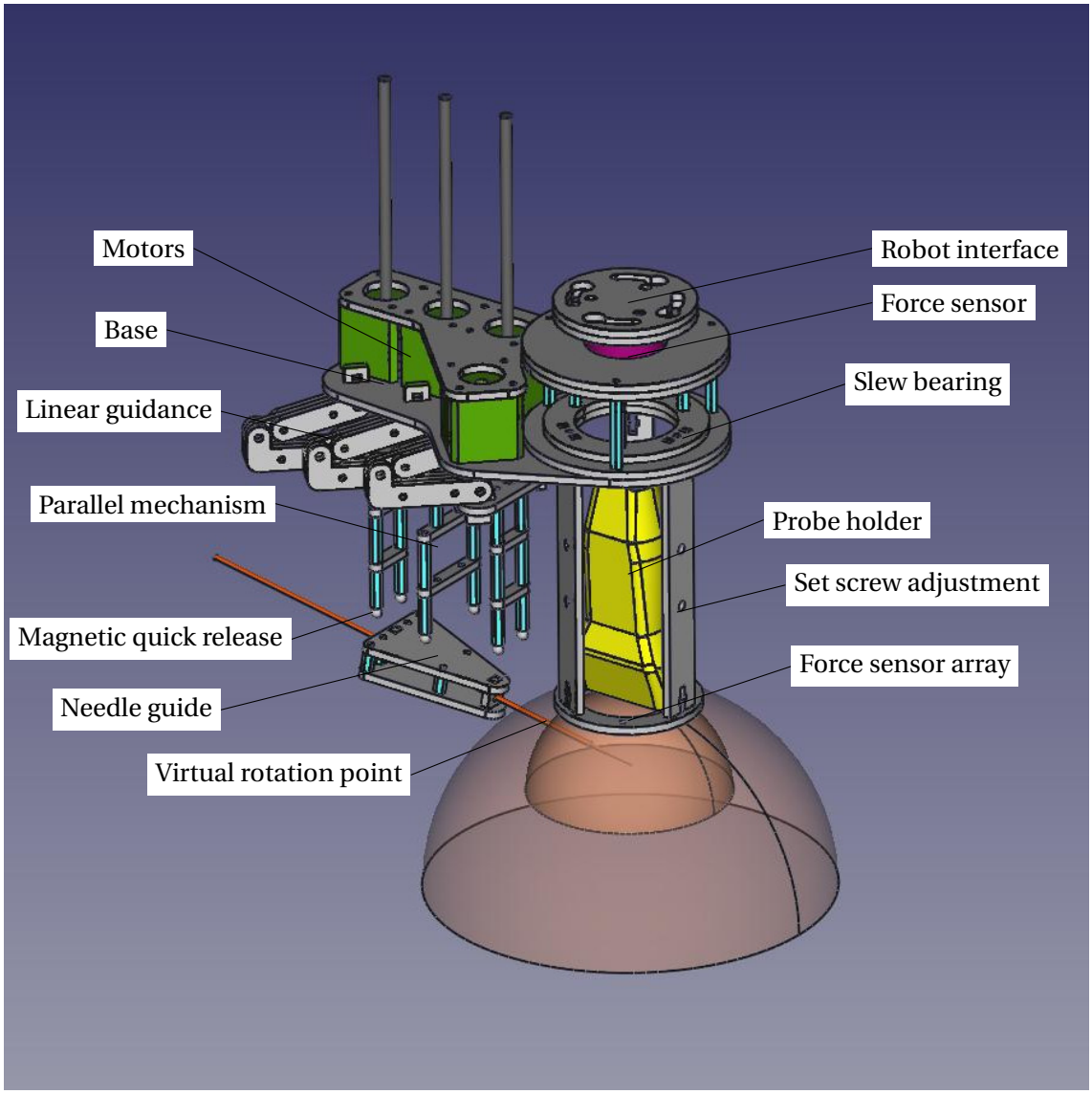

Figure 3.2: Design overview of end-effector.
Digital version: click annotation for specific section

3.4 Detailed description

In this section, detailed descriptions of the components, shown in figure 3.2, are given.

Base

The base of the end-effector interconnects motors, probe holder and robot interface to each other. It is the main structural components of the end-effector. M3 screws and standoffs are used to connects parts of the base in the vertical direction. An additional plate connects three linear stepper motors together on the backside. This backplate increases the stiffness of the base structure.

Note that the upper slew bearing and motor backplate are not made out of one piece. Although this would be possible and stiffen the base, during the design phase it was not sure which size of motor was going to be used. It is chosen to make use of smaller separate components to accommodate easy redesign.

As a safety precaution all corners of the base are rounded to minimize damage in case of a collision.

Slew bearing

The functions of the slew bearings are to suspend ultrasound probe and force sensor underneath the base plate and allow rotation of the whole probe holder with respect to the base. By allowing rotation of the ultrasound probe, position of the needle tip can be viewed by the ultrasound probe in perpendicular planes. In this way, it can be checked from all directions that a biopsy needle would actually puncture targeted lesion.

Loads and speeds on the bearings are expected to be low in the design. Therefore it is chosen to create the bearings for a prototype out of 4mm delrin with the help of a laser cutter and 4mm plastic balls as rolling elements. Two bearings can be found in the design. One at the top just underneath the robot interface and 6 axis force sensor. The other one is positioned on the same base plate the motors are mounted on. PCB standoffs are used to connect both bearings. Washers allow for a small clearance gap to allow bearing balls to roll. Figure 3.3 shows a detailed computer model of the bearing.

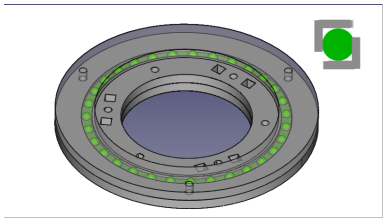

Figure 3.3: Laser cut bearing

Motors

Stepper motors are used in the first prototype to manipulate a needle guiding mechanism. Stepper motors are controlled by powering sets of coils in a particular sequence. The motors can be position controlled without feedback which makes them easy to implement. The used stepper motors are of the non-captive linear type. Non-captive means there is a threaded nut inside driven by the motor. This nut in turn drives a freely rotating spindle up and down. Rotation of the spindles must be constrained in order for the spindles to move up and down. Due to clearance in the driving nut of the motors, play can be noted in sideways directions on the spindles which was found out during prototype construction. A linear motion guide is needed for proper actuation.

By using non-captive stepper motors and a parallel mechanism for the needle guide, one can place the relatively heavy motors on the base of the end-effector. In this way, the motors do not have to drive their own weight. Less torque is needed to drive the needle guiding mechanism. Therefore, smaller and lighter motors can be used. Fixed spindle motors are an alternative option to non-captive spindle motors. However, the non-captive design allows complete retraction of the needle guide mechanism. This is advantageous for scanning of the human body when biopsy needle guidance is not needed.

Positional accuracy of the spindles is estimated to be 0.02mm. The motors needs 200 steps to perform one revolution. Spindles have a pitch of 2mm. Therefore positional resolution is 0.01mm. Tolerances of the spindles are also estimated around 0.01mm. Therefore positional accuracy of the spindles is estimated to be 0.02mm. This is about $\frac{1}{100}^{th}$ of the accuracy required for the tip of the needle. Assuming proper joints and materials, it is expected that the positional accuracy requirement of the needle tip can be fulfilled.

An additional rotary motor will be needed to actuate rotation of the probe holder. The requirement of being able to rotate the probe holder during scanning and biopsy procedures came later in the project. Driving electronics where already ordered and adding additional motor drives is not possible with current electronic setup. It is therefore chosen to temporarily exclude the motor for driving rotation of the probe holder in the prototype design.

For details about specific motors, driving software and electronics the reader is referred to appendix A

Linear guidance

Purpose of the linear guidance mechanism is to prevent out of plane motion of the needle mechanism and to take up clearance which is inherent to linear spindles. It is preferred to use rotary joints for the linear guidance as they are easier to manufacture and have tolerances which are less tight. Expected lifetime of rotary bearings is also considered longer. It is decided to make use of a Sarrus linkage as linear guidance. A Sarrus linkage gives true linear motion in the mathematical sense and is not to complex in design. Each arm of the linkage has three revolute joints which allows full planar motion. By connecting the two arms perpendicular to each other, resulting motion is given by the intersection of both motion spaces which is shown in figure 3.4. Intersection of two perpendicular planes gives a straight line and so the only allowed motion by the mechanism is a straight line. The Sarrus linkage allows a stroke of 90mm.

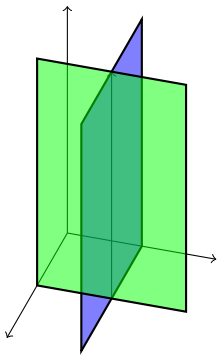

Figure 3.4: Intersection of planar motion spaces, indicating the resulting motion when connecting two arms perpendicular to each other

Robot interface

It is expected that a Kuka lbr iiwa 7 r800 robot is going to be used in the Murab project to manipulate the end-effector around a patient. Biopsy end-effector, to be mounted on the robotic arm, must therefore interface with the flange of the Kuka arm. Four M6 screw bolts are used to connect the end-effector with the Kuka arm. To allow easy attachment and detachment of the end-effector, one only has to loosen the bolts and rotate the end-effector before sliding it off (fig: 3.5).

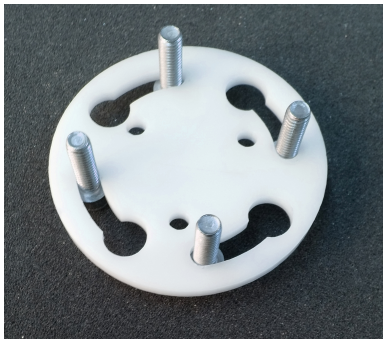

Figure 3.5: Interface of end-effector with Kuka arm. By rotating the interface place, the bolts connecting the arm, fit through the holes for easy attachment.

It is at the moment not decided from which angle the robot arm should interface with the end-effector. The author proposes mechanically interface robot and end-effector coaxially with the probe holder's actuated axis of rotation. In this way, additional component mounted on the base plate, like a generic needle guide, can be rotated out of the way during scanning phase to prevent interference with human or robot arm.

Force sensor

A six degrees of freedom force sensor is going to be mounted between the kuka robot and end-effector. The purpose of the force sensor is to feedback forces so an ultrasound probe can be placed perpendicular against a human with constant force. Applying constant normal force on the ultrasound probe is important for its functioning. Internal torque sensors of the Kuka arm cannot be used for measurements as they are not accurate enough.[1]

Needle guide

As the name suggests, a needle guide has the function of guiding a needle for biopsy in the desired direction. Several sizes of needles are used in biopsy procedures, so each needle size would need its own guide. As the guide is simple in design and might be injection modeled, it could be used as a disposable part. Costs of injection molded components are generally low. Disposable parts can be thrown away after a biopsy procedure, while a renewable part must be cleaned each and every time. [2]

Virtual rotation point

The needle guide and its parallel mechanism are designed in such a way, that the rotation point of the biopsy needle can be virtually placed exactly on the skin of a patient. In this way, biopsy needle can always be rotated around the point where the needle goes through the skin of the patient. Only a small incision is needed to allow access of a needle into the patient's body. A

[1] This is the case for an older orange Kuka. The new white robot (lwr iiwa 7 r800) has better build in torque sensors. It is however not known if these sensors are accurate enough to replace the force sensor on the end-effector.

[2] Disposable parts are also beneficial from a business point of view. As is could give a sustainable income for a company.

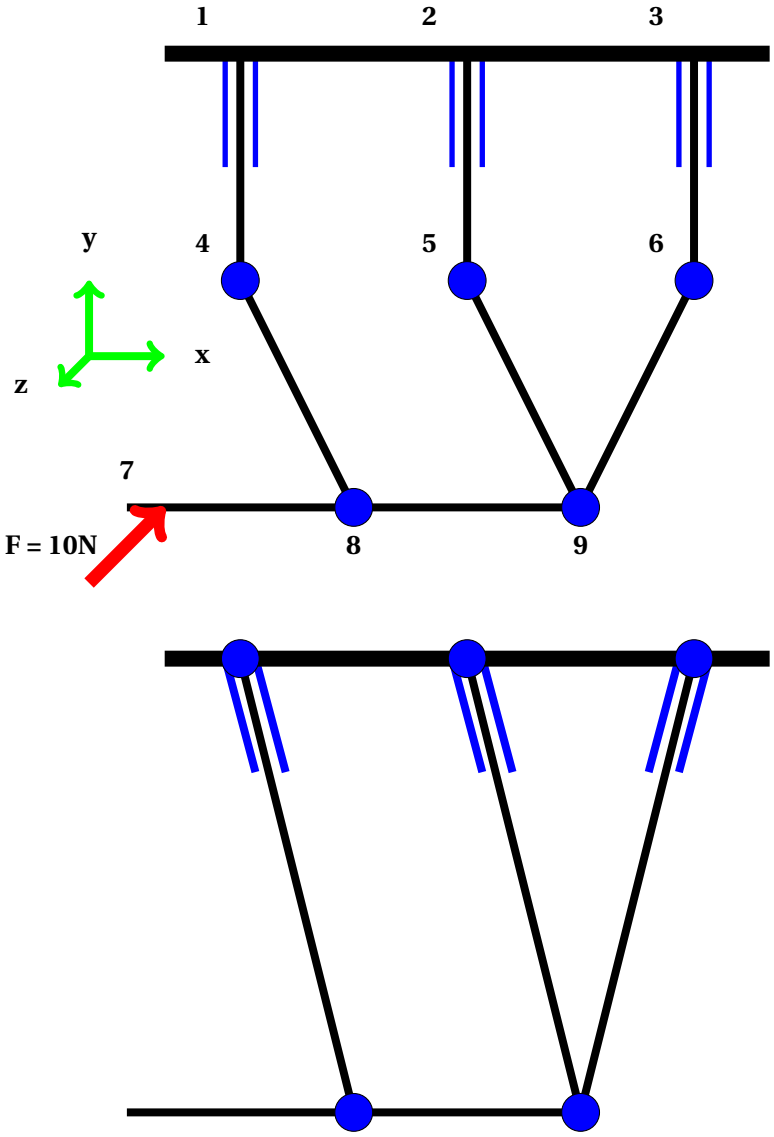

Figure 3.6: Topology of parallel mechanism. Blue lines indicate linear actuators. At the top is the topology implemented in the end-effector. At the bottom the initially investigated one. Kinematic and elastostatic analyses can be found in appendices B.1 and B.3.

small incision gives the least amount of scar tissue which is beneficial for the patient undergoing a biopsy procedure. Forward position analysis is performed to make sure biopsy needle and virtual rotation point are able to reach the full workspace as defined in the requirements. The workspace analysis can be found in appendix B.2 as well as inverse kinematics for the virtual rotation point(appendix: B.1)

Parallel mechanism

A parallel mechanism is used to move the needle guide. Parallel manipulators have higher accuracy and stiffness characteristics relative to serial robots. Simple links are used to connect motor spindles to the needle guide (fig 3.6 top). Another parallel topology, initially investigated, is possible where the linear motor spindles would be connected directly to the needle guide (fig: 3.6 bottom). However, this would require complex joints connecting the motors to the base of the robot.

As a radiologist inserts a needle into the needle guide to perform biopsy, it is expected that disturbance forces will be applied to the mechanism. In the requirements it was estimated that there will be a maximum disturbance of about 10 N. The parallel mechanism must be stiff enough to prevent deviation of the needle tip. An elastostatic analysis is performed during the design phase, to make sure a mechanism can be made stiff enough to handle the disturbances. Five millimeter round steel is assumed as link and spindle material. Compliance and play in joint is furthermore excluded.

To get an idea of the sideways deflections in case of a disturbance force. Python is used to perform elastostatic calculations. Code can be found in appendix B.3.

In elastostatics, it is assumed that forces on nodes are resulting from deviations of node positions.

$$[F] = [K][U] \tag{3.1}$$

Where F represents nodal forces and U represents node deviations. K represents a global stiffness matrix which is build up by summing smaller element matrices. Element matrices are 12x12 matrices representing the stiffnesses of elements between nodes.

To solve above equation, a partition is made between free and supported nodes.

$$\begin{bmatrix} F_f \\ F_s \end{bmatrix} = \begin{bmatrix} K_{ff} & K_{fs} \\ K_{sf} & K_{ss} \end{bmatrix} \begin{bmatrix} U_f \\ U_s \end{bmatrix} \tag{3.2}$$

Nodes which are supported cannot deviate $U_s = 0$. The reaction forces supplied by the supported nodes are however not known. Free nodes are the complement of supported nodes. It is not known how free nodes deviate, but the forces applied on them are known as they are defined by the elastostatic problem.

Based on the assumptions, the system of equations can be solved by simply rearranging terms.

The global stiffness matrix in the program is created by suppling node indices and dimensions to a function which creates a specific element stiffness matrix from a generic beam element. The element stiffness matrices are then added to the global stiffness matrix to define the complete system (Meinders T. (2014)).

$$[U_f] = \left[K_{ff}^{-1}\right][F_f] \tag{3.3}$$

$$[F_s] = [K_{sf}][U_f] \tag{3.4}$$

Table 3.2 shows the results of the analysis. A sideways force of 10 N is applied on the tip of the needle guide. It is shown that deflection is not a problem in theory when joints are stiff enough and 5mm round steel is used. Note that linear guidance for the motors is ignored in the implementation of the analysis as calculations were performed before the idea of using linear guidance. The guidance is expected to have positive influence on the stiffness. Therefore calculations are not repeated.

Magnetic quick release

Magnetic balls are used as joint for the connections between needle guide and spindles. The magnetic balls, allow smooth backlash free motion. As the holding force of the magnets is limited, the joints also act as quick release mechanism when to much force is applied on the needle holder. The needle holder will simply snap off (fig: 3.7). Magnetic balls on the upper side of the connection links are one size bigger with respect to the lower magnetic balls, so the

Node	Δx	Δy	Δz	$\Delta\theta_x$	$\Delta\theta_y$	$\Delta\theta_z$
4	0.	0.	$2.07e^{-6}$	$7.42e^{-6}$	$-4.28e^{-7}$	0.
5	0.	0.	$2.10e^{-6}$	$1.13e^{-5}$	$-1.02e^{-6}$	0.
6	0.	0.	$2.56e^{-6}$	$3.25e^{-5}$	$9.05e^{-7}$	0.
7	0.	0.	$4.58e^{-6}$	$-2.10e^{-5}$	$-2.80e^{-5}$	0.
8	0.	0.	$3.96e^{-6}$	$-2.10e^{-5}$	$-5.49e^{-6}$	0.
9	0.	0.	$3.71e^{-6}$	$-2.09e^{-5}$	$-7.28e^{-6}$	0.

Node	F_x	F_y	F_z	τ_x	τ_y	τ_z
1	0.	0.	-4.88	$-5.26e^{-5}$	$-2.06e^{-5}$	0.
2	0.	0.	-3.67	$-4.96e^{-5}$	$4.93e^{-5}$	0.
3	0.	0.	-2.23	$-4.24e^{-5}$	$-4.34e^{-5}$	0.

Table 3.2: Result of the elastostatic analysis. A force of 10N is applied on $node7$ in z-direction. Δ is the deflection in meters. $\Delta\theta$ is angular deflection in radians

weakest point will be at the needle holder. Bottom joints will therefore break first. By making use of magnets as joints, one can completely seal the linear guidance mechanism and rest of the end-effector in a disposable plastic cover. In the current manual biopsy procedure, a disposable plastic sleeve is wrapped around the ultrasound probe so the probe itself does not have to be cleaned after each biopsy procedure. As the end-effector under design will be an even bigger piece of equipment, it would be beneficial to use an equivalent of a plastic sleeve.

Probe holder

The probe holder is designed in such a way that interference with the human body during scanning is minimized. Interference of the end-effector with the human body is further minimized due to the fact that both the wrist of the kuka arm and ultrasound probe holder can rotate around the same axis of rotation. This makes it possible to rotate the needle mechanism out of the way while scanning a patient's body.

Force sensor array

An acoustically transparent force array sensor should be placed coaxially underneath the ultrasound probe. The intended use of the force sensor is to do elastography. Elastography data is used to create a mechanical deformation model of a scanned breast which is needed for fusing MRI and ultrasound images.

Set screw adjustment

An of the shelf ultrasound probe is initially going to be used for the ultrasound guided biopsy procedure. The ultrasound probe is going to be fixed by six screws. It is of great importance that the needle to be used in the biopsy is visible on the ultrasound images at all time. The six screws holding the ultrasound probe are used to adjust the alignment of needle and probe, so the needle is always visible on ultrasound footage (fig 3.8).

3.5 Prototype construction

A physical prototype of the design is realized by exporting digital designs of the custom parts to a laser cutter. 4mm thick white Delrin is used as material. 4mm is considered to be a suitable thickness, as it is rigid due to its thickness, but thin enough so a chamfer on the edges due to the laser cutting process does not cause trouble during assembly. Laser cutting is preferred above 3D printing as the cutting process is more accurate and much faster. Laser cut parts are furthermore cut out of solid sheets of material resulting in stronger parts compared to semi-hollow 3D printed parts. Figure 3.9 show the physical prototype.

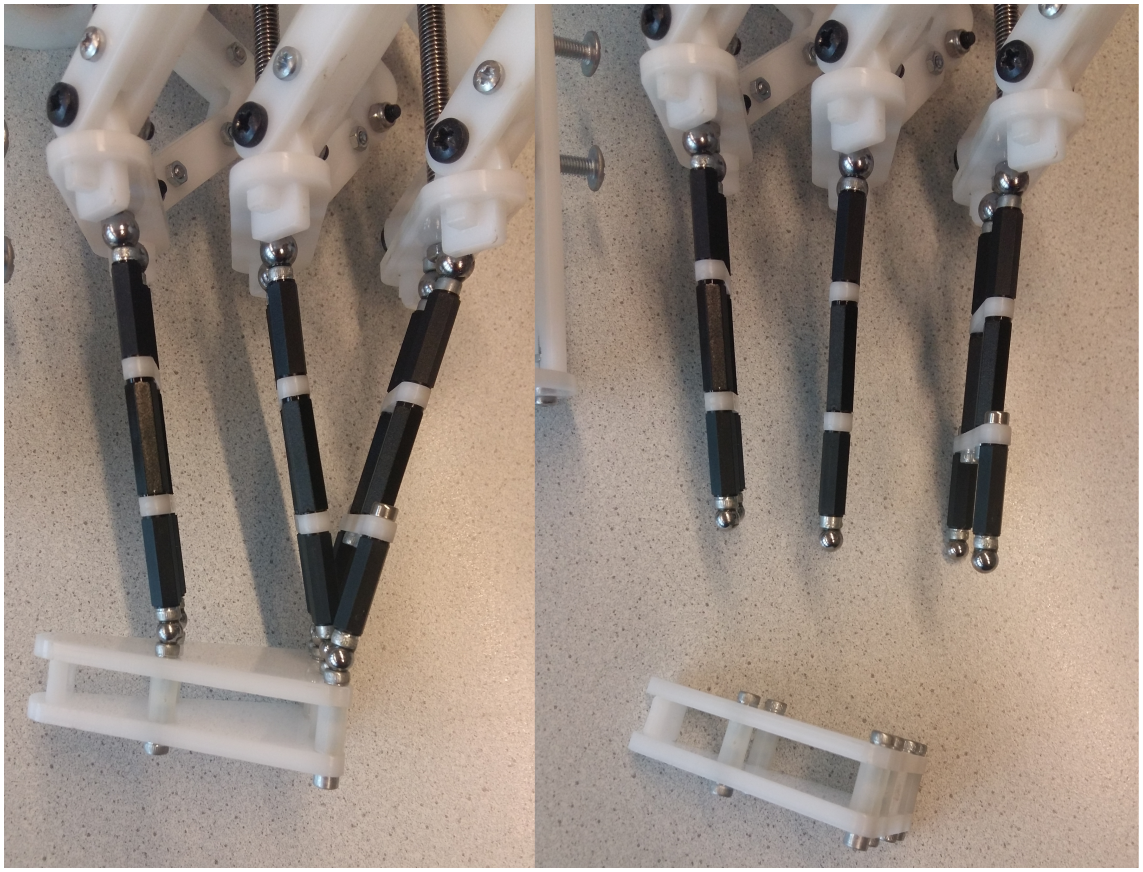

Figure 3.7: Magnetic ball joints. The needle guide releases from the parallel mechanism when excess force is applied.

The prototype can be controlled by directly controlling combinations of motors using buttons on the LCD display shield. Although inverse kinematic relations are derived in appendix B.1, they are not implemented yet on the microcontroller. Table 3.3 shows the functions of currently implemented manual control.

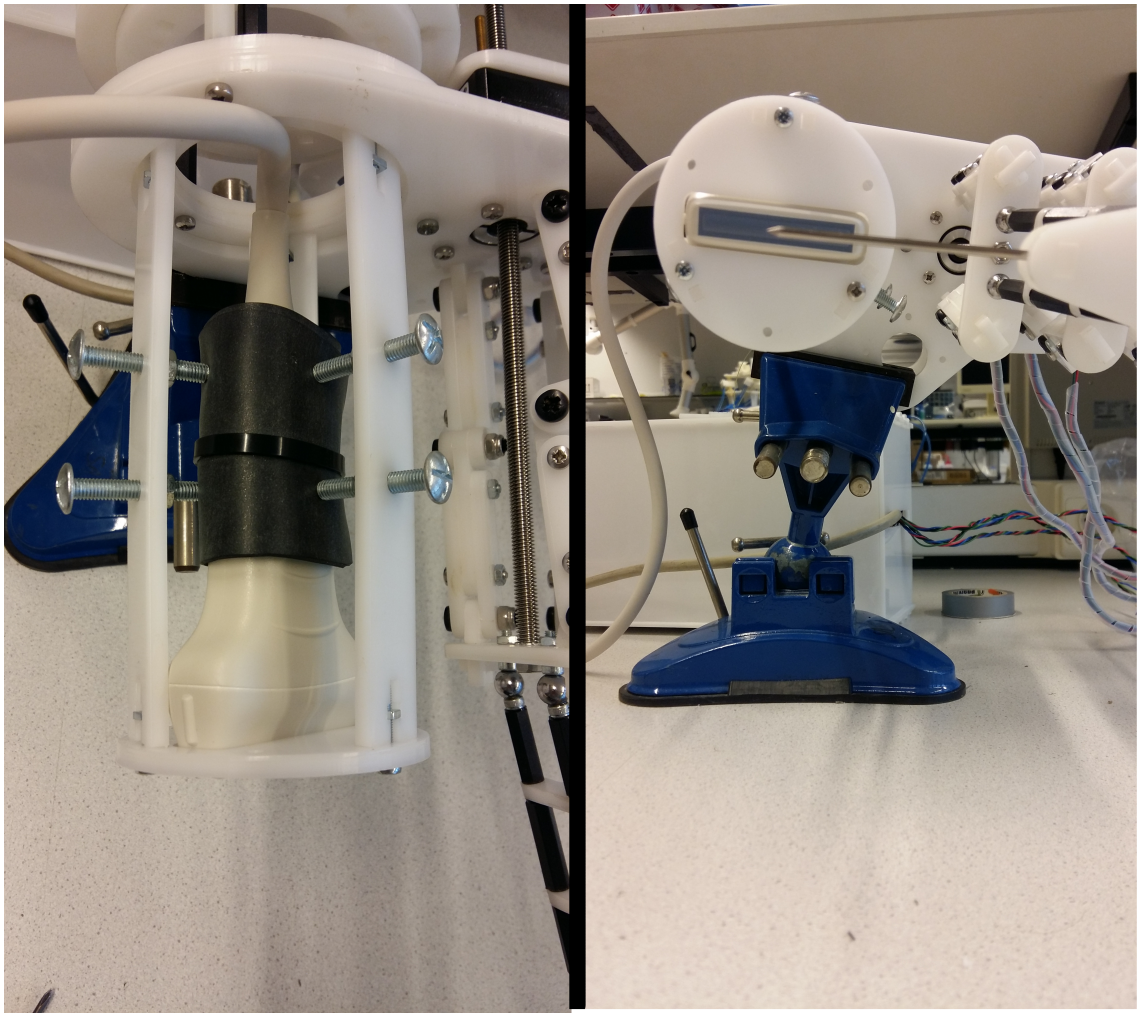

Figure 3.8: On the left one can see an actual ultrasound probe clamped in six set screws. On the right, probe and needle are aligned.

Button	Function
up	move motor(s) up
down	move motor(s) down
left	decrement mode
right	increment mode
select	stop motors (drivers in high impedance state)
Mode	Motor combination
1	1
2	2
3	3
4	1, 2
5	2, 3
6	1, 3
7	1, 2, 3

Table 3.3: functions of the buttons and combinations of motors belonging to selectable modes

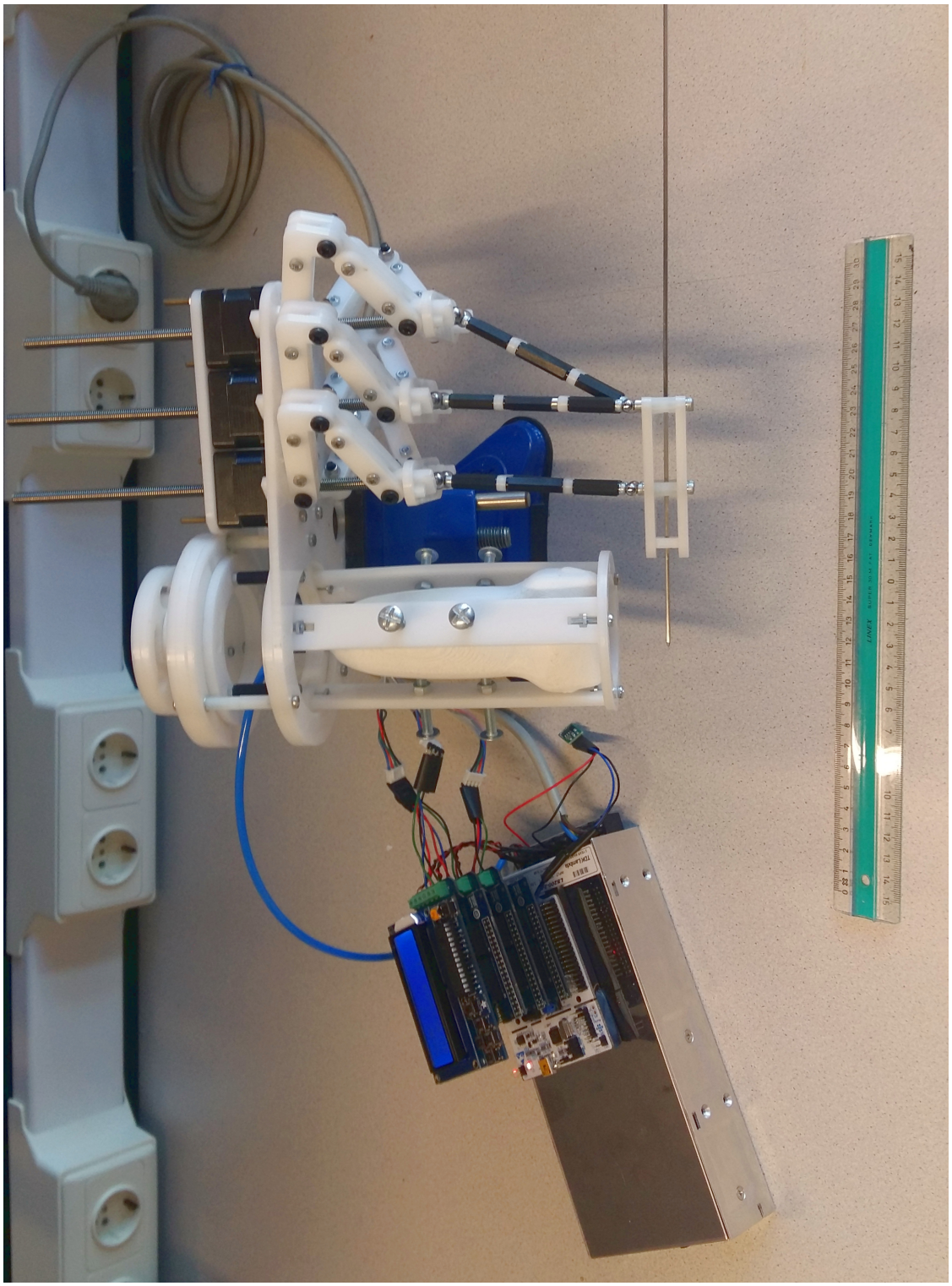

Figure 3.9: Image of actual end-effector and drive electronics. Clamped probe is a 3d scanned and printed model of an actual ultrasound probe.

4 Evaluation of design

A physical prototype is build based on the design developed in chapter 3. The prototype is evaluated with the help of stated requirements. Based on the evaluation, new insight on the design is obtained and conclusions can be formulated.

Table 3.1, stating the requirements setup in chapter 3, is used to evaluate the physical prototype. Due to the fact that all requirements are stated quantitatively, measurements are used for evaluation. Each subsection evaluates a specific requirement from table 3.1.

4.1 Safety evaluation

A simple preliminary hazard analysis is performed to identify potential risks with respect to safety in the prototype design. As the end-effector is going to be used in a medical setting, safety analysis and risk management must be performed. It is strongly advised to start as early as possible with safety analysis and risk management. This prevents complex and expensive changes of design in the later stages of development (Palanichamy (2010)). Safety with respect to the KUKA arm is not taken into account at the moment. The robotic arm is already designed specifically for safe human robot interaction.

Safety risk of the end-effector is analyzed with the help of a preliminary hazard analysis. A preliminary hazard analysis consists basically out of three steps. First potential hazards are identified. Second step is to describe consequences and frequencies of occurrence of all identified potential hazards. Multiplying severity of consequence and frequency of occurrence gives the risk of an identified potential hazard.

All calculated risks must in the third step be ranked from high to low. A trade off must then be made about the risk of some hazard and the added value of the prototype feature. When a risk is considered to high, subsequent measures must be taken to reduce the risk to an acceptable level.

Common sources of potential hazards are paths through which energy can flow. Think of moving mechanical components and flow of electrical currents other sources are software bug and human errors. Major hazards identified at the moment as potential risks are:

- The 220V connection from mains to switching power supply. Screw terminals connecting the wires are not perfectly shielded. (table 4.1: power sup.)
- The motors, linear guidance and parallel mechanism provide regions where parts of the human body can be pinched. Pinching is identified as another potential hazard. (table 4.1: pinching)
- Software and incorrect handling by humans is a safety concern. Unexpected motions of robot and end-effector could result from software faults or human errors. (table 4.1: software)

Consequences of the identified hazards are now described. Contact with 220V AC straight out of a mains connection can be lethal and is rated as a level 4 for severity. Frequency of occurrence is estimated as level 3 according to table 4.1.

Consequences of pinching body parts is minor in the prototype design. This corresponds to a level 1 on the severity scale (fig 4.1). Motors current is limited and therefore the torque capabilities of the motors. Experiments with the author's fingers show that pinching is clearly noticeable, but does not result in any damage. Occurrence of potential hazard is estimated as "will probably occur" level 4.

0 – 5 = Low Risk 6 – 10 = Moderate Risk 11 – 15 = High Risk 16 – 25 = extremely high unacceptable risk		Severity of the potential injury/damage				
		Insignificant damage to Property, Equipment or Minor Injury **1**	Non-Reportable Injury, minor loss of Process or slight damage to Property **2**	Reportable Injury moderate loss of Process or limited damage to Property **3**	Major Injury, Single Fatality critical loss of Process/damage to Property **4**	Multiple Fatalities Catastrophic Loss of Business **5**
Likelihood of the hazard happening	Almost Certain **5**	5	10	15	20	25
	Will probably occur **4**	4	8	12	16	20
	Possible occur **3**	3	6	9	12	15
	Remote possibility **2**	2	4	6	8	10
	Extremely Unlikely **1**	1	2	3	4	5

Figure 4.1: General risk assessment matrix. Colors indicate the acceptance of a certain risk level. Frequency of occurrence of a certain event is divided in 5 levels on the right. At the top is the severity of a certain event also divided in 5 levels. Multiplication of both give the amount of risk involved in a certain event. (source: (Palanichamy (2010)) Basics of risk management for medical device design)

Hazard	Severity level	Occurrence level	Risk level
software	4	4	16
power sup.	4	3	12
pinching	1	4	4

Table 4.1: Result of the preliminary hazard analysis.

Bugs in software could result in unexpected movement of the needle guide. Unexpected movement can result in major injuries (level 4) even if magnetic ball joints do break. This is especially true when a sharp biopsy needle is inserted into a human body. Occurrence of the hazard is estimated as possible (level 4). Human errors due to wrong handling could also result in unexpected movement of the needle guide. The consequences are therefore graded equally.

Ranking risk levels results high risk for the mains power connection (12), an extreme high risk level for software and handling errors and low risk for pinching body parts. High and extremely high risk levels are unacceptable. To mitigate power supply and drive electronics risks, the power supply is enclosed to prevent a conductive path from a human to mains power connection. Thorough testing of the software and human interface to the end-effector must be performed to prevent hazardous situations with needles moving unexpectedly. Results of the analysis are summarized in table 4.1.

4.2 End-effector interference

Designed end-effector should not interfere with any object. The design includes features to prevent interference. The base of the robot is able to rotate independent of robot and ultrasound probe. The base plate and connected motors can therefore move out of the way. Furthermore, links connecting needle guide and linear guidance can be removed, so more space is available around the probe during scanning phase (fig 4.2 and 4.3).

4.3 Workspace reachability

A forward kinematic analysis is performed to determine the reachable workspace of the mechanism. It is chosen to theoretically determine the workspace of the end-effector as it is difficult to perform measurements in real practice. Symbolic solutions for forward position kinematics

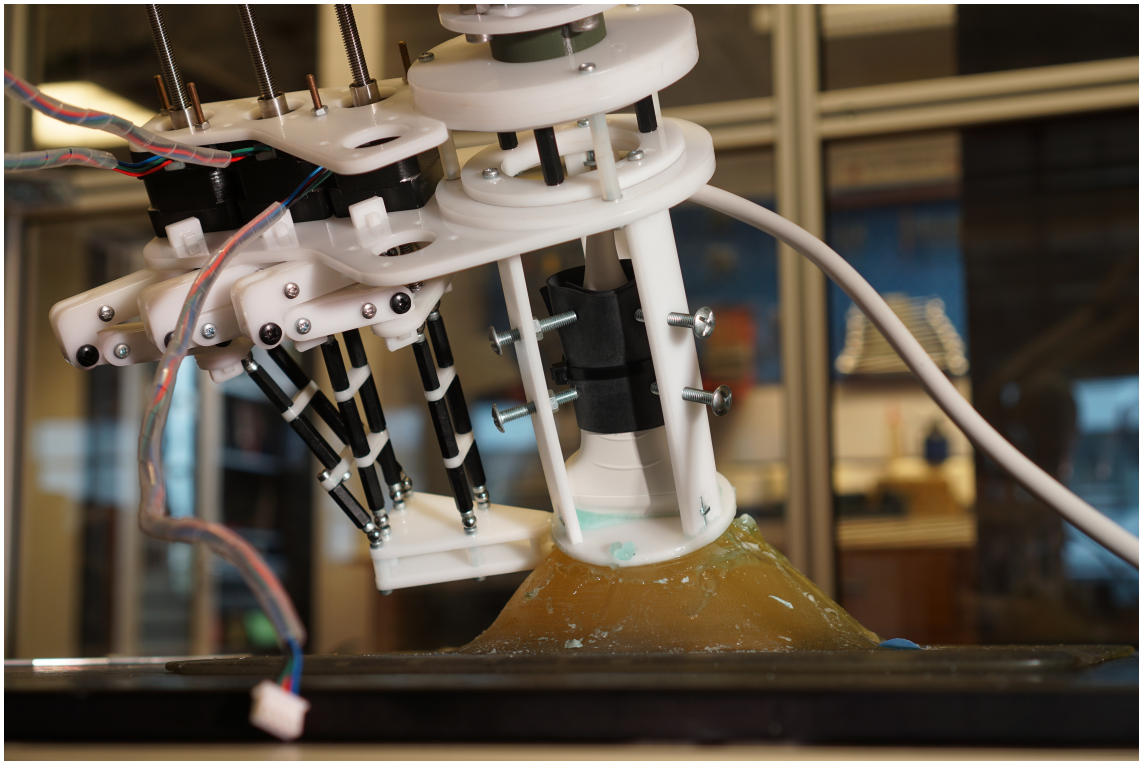

Figure 4.2: End-effector connected to robotic arm. The end-effector is able to move around a phantom breast, without interfering with phantom patient and robotic arm.

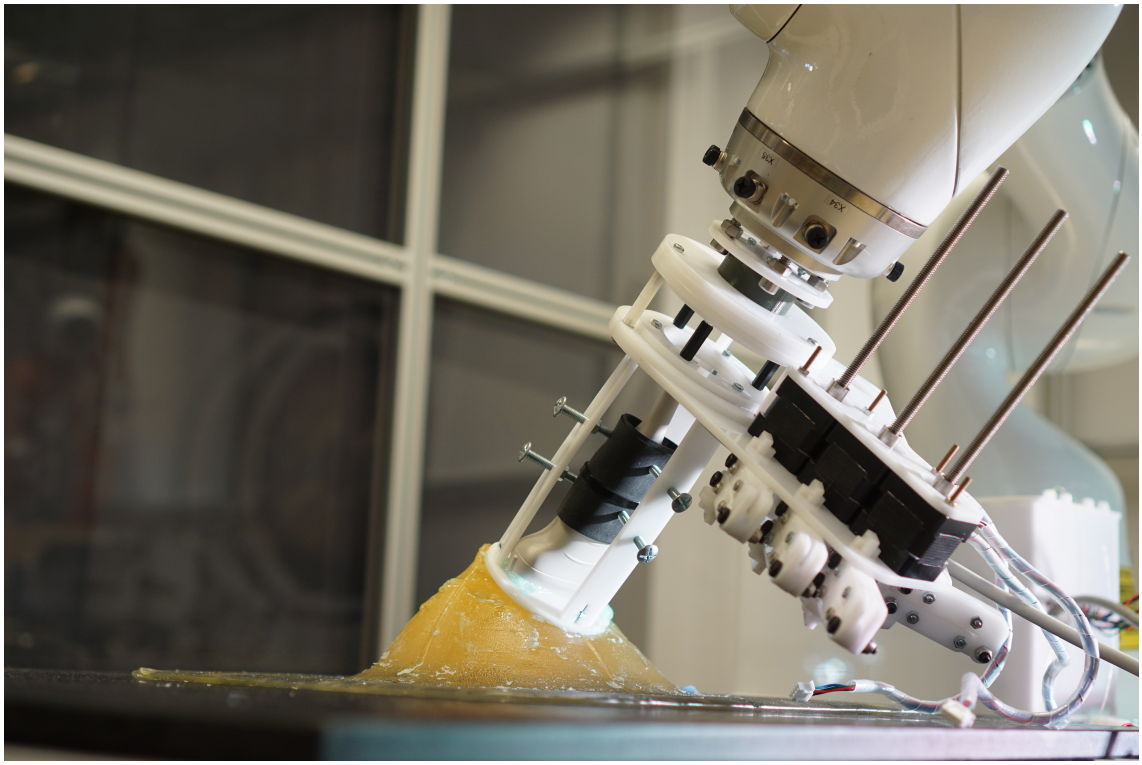

Figure 4.3: By removing the parallel mechanism using the magnetic joints, interference with a patient can be prevented during scanning phase.

are hard to find for parallel manipulators. Therefore, Newton-Rhapson method is used to numerically solve a system of equations. Python is used as a scripting language to do the analysis. Details about the script can be found in appendix B.2.

Figure 4.4 shows the result of the workspace analysis.

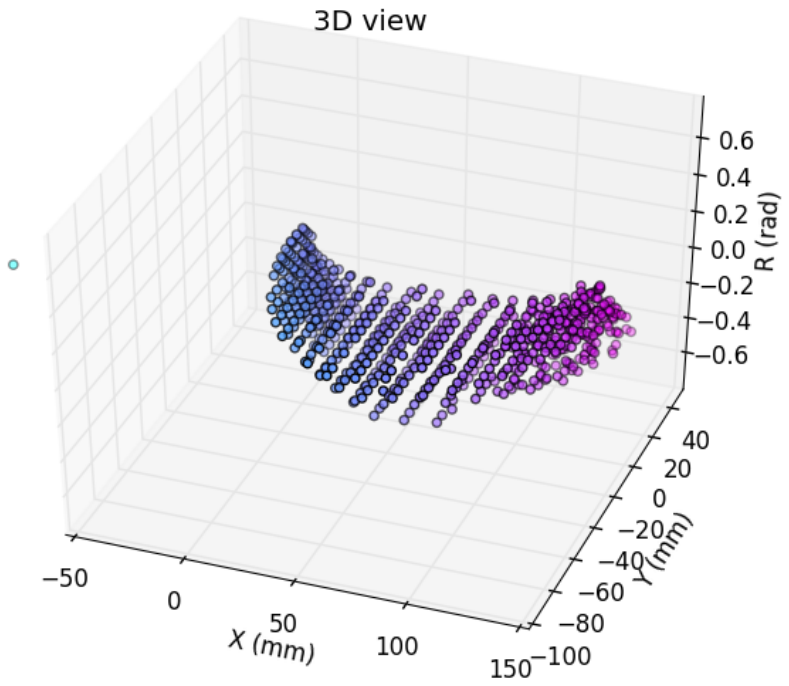

Figure 4.4: Theoretic workspace of needle guide. When looking to the side of the parallel mechanism, +y is in the upward direction. +x is to the left and +R is counter-clockwise rotation in the plane.

Although it is a bit difficult to visualize the result, desired workspace as defined in the requirements section is contained within the calculated workspace. The virtual center of rotation can be placed anywhere between minimal breast sizes with a radius of 50mm and maximum breast sizes with a radius of 100mm. The needle can furthermore rotate around this center with 0 to -30 degrees with respect to the chest wall. Therefore, the workspace requirements is fulfilled.

4.4 Force disturbance

A medical professional is responsible for inserting a needle through the needle holder into the human body to perform the actual biopsy. In theory a needle is pushed through the guide perfectly and no forces are applied on the needle guide. In reality disturbance forces will be present on the needle guide due to insertion of the needle. It was estimated that a person inserting a needle would apply a maximum force of 10 N on the needle guide. The needle guide must be able to handle these disturbance forces.

Force measurements were done with the help of a spring scale to determine the holding force of the magnetic spheres. The spring scale was hooked onto the needle guide and pulled by hand. Torques are measured by hooking the spring scale on the needle at 10 cm from the needle guide. Results are stated in table 4.2.

Direction	Force,Torque (N,Nm)
$+\hat{y}$	*
$-\hat{y}$	6
$\hat{z}$	3
$\hat{\theta_y}$	0.3
$\hat{\theta_z}$	0.1

Table 4.2: Holding forces and torques of magnetic joints.

The requirement of withstanding 10 N of disturbance forces is not met. However, 10 N of disturbance force applied on the needle guide could be an over estimate as a medical professional would only apply such forces on purpose when inserting a needle.

It was expected that the magnetic spheres would not be able to hold 10 N of force. The magnetic joints are not perfectly constructed. Hex head screws are used as sockets for the ball magnets. The magnetic balls are furthermore quite small and their attractive force is limited to about 3 N per magnet. However, the needle guide does break of when excessive force is applied on the needle guide.

The backlash in the magnet joints is minimal. Although the 10 N requirement is not met, the concept with magnetic joints does work.

4.5 Positional accuracy

The needle and guide must have an accuracy of 1 mm. Inverse kinematic position analysis is performed to verify this accuracy is achievable in theory. A 1mm grid is created in the desired workspace. For each grid coordinate, spindle lengths are calculated for each motor using inverse kinematics. Neighboring coordinates in the desired workspace correspond to "neighboring" spindle coordinates for each motor. By taking the difference between neighboring required spindle lengths, the minimum required accuracy of the motor spindles can be determined. The actual positional accuracy of the spindles must be lower than this minimal difference. Results from the python script show that the accuracy that is needed is about 18 times higher than the accuracy that can be achieved by the motors and spindles. Theoretical accuracy is therefore easily met.

However, compliance in the end-effector can be observed. The compliance causes deflections at the needle tip. The compliance is mainly caused by the lower stiffness of laser cut Delrin and backlash in the joints. Especially play in the joints is a source of error, due to the use of bolts instead of proper bearings.

It is however expected that the positional requirements can be achieved by implementing support bearings for the rotary joints.

4.6 Manipulation by robot

The end-effector is going to be maneuvered by a robot arm. There must therefore be a valid interface between robot arm and end-effector. The KUKA arm to be used in the MURAB project can handle a maximum weight of 7 Kg. The weight of the end-effector is measured at 1.376 Kg. Weight requirement for the end-effector is therefore easily fulfilled. By simply bolting the end-effector on the KUKA arm as can be seen in figure 4.5, it is proven that the end-effector can be mounted on the KUKA arm. The robotic arm is able to manipulate the end-effector and therefore the requirement is fulfilled.

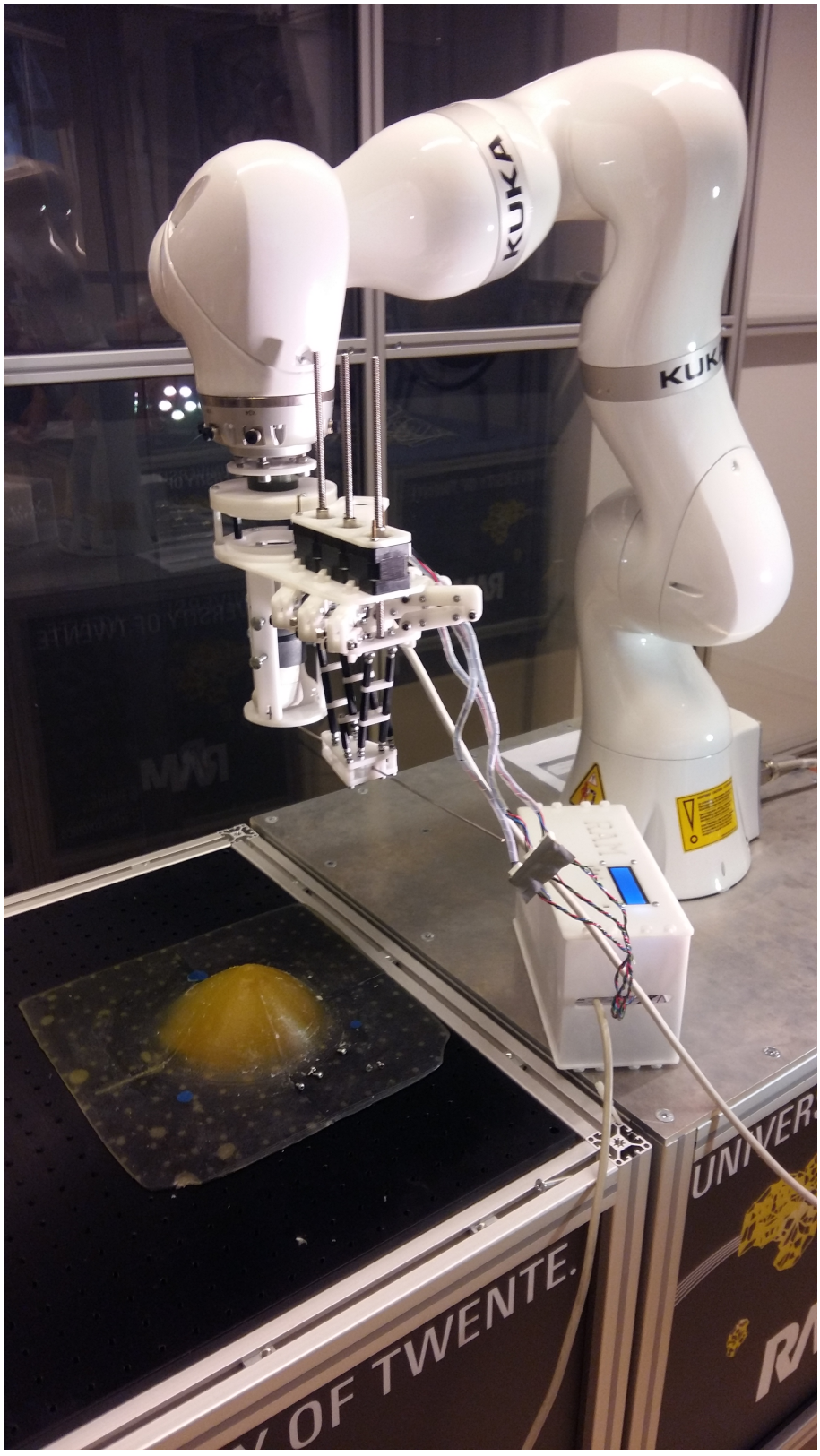

Figure 4.5: end-effector mounted to kuka arm

4.7 Required components

In the requirements, it was stated that the end-effector should contain a camera and an acoustically transparent force array for elastography. It was chosen to exclude the camera and force array sensor from the design, as it was outside the scope of this project. However, empty space can be found underneath the ultrasound probe where a force array sensor could be placed. A camera module could be placed in the empty space between probe holder and motors.

5 Conclusion

5.1 Conclusion

In this study, the design, creation and evaluation of an image guided breast biopsy end-effector prototype is performed. Conclusions about the design are drawn based on the evaluation. A simple preliminary hazard analysis shows that a safe end-effector can be constructed as long as one pays close attention to the prevention of software bug and shielding of electrical components. Evaluation of performance on the mechatronic needle steering shows that the required workspace for the needle guide is within the reachable workspace of the end-effector. Furthermore, analysis shows that positional accuracy can be achieved in theory. Force disturbances created by insertion of a biopsy needle by a radiologist are causing deflections in the mechanism which do causes deviations at the needle tip. Magnetic joints will also brake due to these estimated disturbance forces. All flaws discovered during the evaluation phase are expected to be solvable. Therefore the conclusion can be made that current design is a suitable starting point for further development. It is expected that a future version of this end-effector can improve the accuracy of needle punctures during image guided biopsy procedures.

5.2 Further work

Although the design is functional, it can be improved. Stiffness of the overall device is low. Not only is this caused by the material properties, structurally it can be improved as well by connecting the stepper motors at the top to the robot arm interface. The linear guidance of the spindles can also be improved. The arm lengths of the linear guidance are to short and limit the full stroke of the motor spindles. Although it was stated that rotary joints would have advantages over prismatic guides, it is advised to look into prismatic joints for linear guidance of the spindles, as it is expected that differences between prismatic and rotary joints would be minimal. Forces on the mechanism are small and accuracy requirements are in addition not to tight. Therefore, it is expected that prismatic joints would also work perfectly.

A Appendix 1

A.1 Stepper motor details

Motors used on the end-effector are non-captive linear stepper motors from Nanotec Electronics. Their size is standard NEMA 14 with a motor length of 34 mm. 200 steps are needed for one full revolution of the drive nut. Spindles are 6mm in diameter and have a pitch of 2mm. A lathe is used to give the lower end of the spindles M4 thread. Rotation of the spindles is prevented by lock-washers.

A.2 Stepper drive electronics

The brain of mechatronic drive chain is an Arduino like microcontroller board called Nucleo-F401re. Nucleo has the same pin interface as Arduino plus additional pins. Nucleo is also faster with its 32bit processor and 84Mhz clock compared to the 16Mhz Arduino has. At the same time, Nucleo is half the price when ordered from RS components. The board is MBED enabled, which means code can be created and compiled with an online c++ IDE running in any modern browser. Programs can be downloaded to the board independent of operating system or browser. Also ordered from RS are three stepper motor drivers (X-Nucleo-IHM01A1) which stack on the microcontroller board using the Arduino interface. A maximum of three stepper drivers can stack on top of each other due to the limiting Arduino pin interface. On top of the stack is an Adafruit 16x2 LCD screen with keypad connected through I2C to the microcontroller. The LCD keypad is used to allow control of the motors without connecting a computer to the microcontroller. All electronic components are powered by a 24v 200w switching power supply. A small buck converter is used to externally power the microcontroller and stepper driver logic. Pay attention that for external power, Jumper J5 (below black reset button) has to be in the **E**5V position.

A.3 Mechanism driving software

Note: for programming the microcontroller, jumper J5 (below black reset button) has to be in the **U**5V position! Microcontroller can be connected to the pc by a USB->USB-B-mini cable.

Software for the microcontroller is written in c++ using the online MBED compiler. Standard driver libraries for the stepper driver L6474 and LCD display are used. The software can be accessed on `https://developer.mbed.org/accounts/login/`, logging in with user name: **s1096729** and password: **ramutwente**

Software for driving the motors is quite simple. A couple of interrupts are initialized and motor settings are applied. An infinite while loop does the actual control of the motors based on user input. While running through the infinite loop, states of the buttons on the LCD shield are polled. A function from the LCD library is used for reading these states. Then, a switch statement changes control variables according to pressed buttons. Another switch statement commands motor actions in case some control variable is changed. Text on the LCD display is also updated when a control variable changes.

B Appendix 2

B.1 Mechanism inverse velocity kinematics

Kinematics of the mechanism is determined with the help of a python script. Although screw theory was initially used to obtain complete forward and inverse velocity kinematics, it was decided in the end to make use of finite elements and geometric transfer functions for the inverse kinematics as it is easier to understand for the interested reader. Inverse kinematics is most important for stepper motor control.

Nodes are defined for the mechanism and these nodes are interconnected by rigid elements. Simple loop equations are set up for each element and equated to zero, meaning that deformations of the elements are not allowed. For the analysis, only nodes 4,5,6,8 and 9 are taken into account. Nodes 8 and 9 will later be expressed in terms of node 7, which is defined as virtual center of rotation and rigidly connected to nodes 8 and 9. Numbers assigned to nodes can be seen in the code at the end of this section or in figure 3.6.

The nodal coordinates are partitioned into a position vector.

$$\vec{x} = [x^{(o)}|x^{(c)}|x^{(m)}] \tag{B.1}$$

$$\vec{x} = [x_4 x_5 x_6 | y_4 y_5 y_6 | x_8 y_8 x_9 y_9] \tag{B.2}$$

$x^{(o)}$ represents constrained coordinates. Coordinates which do not change with time. $x^{(c)}$ are dependent coordinates. These dependent coordinates do change with time and depend on independent coordinates $x^{(m)}$. It is desired to express the dependent coordinates in terms of the independent coordinates. $x^{(c)} = F(x^{(m)})$. The nodes in the parallel mechanism are interconnected by rigid truss elements. The rigid element can mathematically be described by loop equations and the result is a non-linear system of equations $D(\vec{x}) = \vec{0}$.

$$\begin{aligned}
\sqrt{(x_8 - x_4)^2 + (y_8 - y_4)^2} - L_{48} = 0 \\
\sqrt{(x_9 - x_5)^2 + (y_9 - y_5)^2} - L_{59} = 0 \\
\sqrt{(x_9 - x_6)^2 + (y_9 - y_6)^2} - L_{69} = 0
\end{aligned} \tag{B.3}$$

These non-linear systems of equations are in general difficult to solve symbolically, but can be solve numerically by for instance Newton-Rhapson method. Dependent velocities can however be solve symbolically. By taking the derivative of the loop equations, one obtains $\frac{dD(x)}{dx}\dot{x} = \vec{0}$. By applying the same partitioning to the Jacobian and velocity vector as was done to the position vector, one obtains

$$\left[D_{x^{(o)}}|D_{x^{(c)}}|D_{x^{(m)}}\right]\begin{bmatrix}\dot{x^{(o)}}\\ \dot{x^{(c)}}\\ \dot{x^{(m)}}\end{bmatrix} = \left[.|D_{x^{(c)}}|D_{x^{(m)}}\right]\begin{bmatrix}\vec{0}\\ \dot{x}^{(c)}\\ \dot{x}^{(m)}\end{bmatrix} = \vec{0} \tag{B.4}$$

$\dot{x^{(o)}} = 0$ as the constraints do not depend on time. By rearranging the matrix, one can express the dependent velocities in term of independent velocities. $\dot{x}^{(c)} = -D^{-1}_{x^{(c)}} D_{x^{(m)}} \dot{x}^{(m)}$.

$$\begin{bmatrix}\dot{y_4}\\ \dot{y_5}\\ \dot{y_6}\end{bmatrix} = \begin{bmatrix}\frac{x_4 - x_8}{y_4 - y_8} & 1 & 0 & 0\\ 0 & 0 & \frac{x_5 - x_9}{y_5 - y_9} & 1\\ 0 & 0 & \frac{x_6 - x_9}{y_6 - y_9} & 1\end{bmatrix}\begin{bmatrix}\dot{x_8}\\ \dot{y_8}\\ \dot{x_9}\\ \dot{y_9}\end{bmatrix} \tag{B.5}$$

The velocities $\dot{x}^{(m)}$ are dependent on node 7. As node 8 and 9 are rigidly connected to node 7, they can simply be expressed by

$$\begin{aligned} x_8 &= x_7 + l_{78}cos(\theta_7) \\ y_8 &= x_7 + l_{78}sin(\theta_7) \\ x_9 &= x_7 + l_{79}cos(\theta_7) \\ y_9 &= x_7 + l_{79}sin(\theta_7) \end{aligned} \tag{B.6}$$

taking the time derivative of these equations, gives the velocities of node 8 and 9 in terms of node 7. In matrix form these derivatives become

$$\begin{bmatrix} \dot{x}_8 \\ \dot{y}_8 \\ \dot{x}_9 \\ \dot{y}_9 \end{bmatrix} = \begin{bmatrix} 1 & 0 & -l_{78}sin(\theta 7) \\ 0 & 1 & l_{78}cos(\theta 7) \\ 1 & 0 & -l_{79}sin(\theta 7) \\ 0 & 1 & l_{79}cos(\theta 7) \end{bmatrix} \begin{bmatrix} \dot{x}_7 \\ \dot{y}_7 \\ \dot{\theta}_7 \end{bmatrix} \tag{B.7}$$

The python script below shows the derivation of the geometric transfer function. Jonker J.B. (2015)

```
import sympy as sp #symbolic library for python

# PARALLEL MECHANISM
#
# 1 o  2 o   3 o
#   |    |     |
#   |    |     |
# 4 o  5 o   6 o
#    \    \   /
#     \    \ /       +y
#    8 o  9 o        |
# 7 o--+----+-       o-- +x

#DEFINE SYMBOLS
x4,y4,x5,y5,x6,y6,x8,y8,x9,y9,q8 = sp.symbols("x4,y4,x5,y5,x6,y6,x8,y8,x9,y9,q8")
x7,y7,q7,x78,y78,x79,y79 = sp.symbols("x7 y7 q7 x78 y78 x79 y79")
vx7,vy7,w7 = sp.symbols("vx7 vy7 w7");
L48,L59,L69,L89 = sp.symbols("L48,L59,L69,L89")

#DEFINE AND PARTITION POSITION VECTOR [constrained (o), dependent (c), independent
    (m)]
x = sp.Matrix([x4,x5,x6, y4,y5,y6, x8,y8,x9,y9])
nxo = 3; nxc = 6; nx = 10; # partition indices

# LOOP EQUATIONS PER ELEMENT D(x) = 0;
F = sp.Matrix([sp.sqrt((x8-x4)**2 + (y8-y4)**2) - L48 ,\
               sp.sqrt((x9-x5)**2 + (y9-y5)**2) - L59 ,\
               sp.sqrt((x9-x6)**2 + (y9-y6)**2) - L69]);

# SYMBOLIC JACOBIAN
Dx = sp.simplify(F.jacobian(x))

# 0 = diff(D(x))*diff(x)
# x was partitioned, so dx will always be
#[dxo,dxc,dxm] = [0,0,0,dxc,dxc,dxc,1,1,1,1]
# diff(D(x)) will be 0 = [[Dx(o)],[Dx(c)],[Dx(m)]]
# Dx(c)dxc + Dx(m)dxm
# dxc = -Dx(c)^-1*Dx(m)*dxm -> Fqc = -Dx(c)^-1*Dx(m) which is...
# called first order geometric transfer function.
# Dx(c) will always be square if...
# the system is perfecly constrained (not under or over constrained),
```

```
# and therefore always invertible
Dc = Dx[:,nxo:nxc]; #Dx(c) part
DcInv = Dc.inv();
Dm = Dx[:,nxc:nx];
Fqc = -DcInv*Dm

print""
print "geometric transfer function Dx8,Dy8,Dx9,Dy9 --> Dy4,Dy5,Dy6 in terms of x8,
    y8, x9, y9"
print str(x[nxo:nxc]) + "T =";
print""
sp.pprint(sp.simplify(Fqc))

#
    ##############################################################################

# calculate positions and velocities of nodes 8 and 9 in terms of node 7

x8 = x7 + L78*sp.cos(q7); Dx8 = Dx7 -L78*sin(q7)*Dq7;
y8 = y7 + L78*sp.sin(q7); Dy8 = Dy7 +L78*cos(q7)*Dq7;
x9 = x7 + L79*sp.cos(q7); Dx9 = Dx7 -L79*sin(q7)*Dq7;
y9 = y7 + L79*sp.sin(q7); Dy9 = Dy7 +L79*cos(q7)*Dq7;
```

B.2 Workspace analysis

(A. (1966)) To determine the reachable positions and orientations of the needle guide mechanism, a forward position analysis is performed. Non-linear loop equations stated in previous section define relations between nodes of the parallel mechanism. Newton-Rhapson method is then used to numerically solve the non-linear system for motor positions. By iterating through all possible motor positions reachable workspace is obtained.

Newton-Rhapson method needs an initial guess for its variables to converge to a solution. Solutions from previous motor positions is used as initial guess for solving next iteration. This can be done under the assumption that node positions of the mechanism would not change to much when motor configuration is changed by a small step. The assumption is true most of the time when iterating through all motor configurations. There is however one exception to this assumption.

Three nested for-loops are used to represent iteration through all motor configurations. When one of the for loops finishes, the loop will restart counting from the beginning. Due to the discontinuity, the assumption of small motor steps is not valid anymore, and the initial guess for next iteration is far off. To solve the special case. Some of the solutions of specific motor configurations are stored and used as initial guess when a for-loops restarts.

When the program has iterated through all motor configurations, a 3D scatter plot is made with on the axes: x_7−position, y_7−position and needle guide orientation

```
import numpy as np
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt

# initialize array (one cannot concatenate arrays to empty array in python)
workspace = np.zeros(3)

# define constants
x4 = 60.0; x5 = 95.0; x6 = 130.0;
L48 = L59 = L69 = 85.0; L89 = 35.0; L78 = 30.0;

# create motor positions
#                    x8    y8    x9     y9
xquess = np.array([x4+15.0,11.0,x5+15.0,11.0])
xstore6 = xstore56 = xquess

begin = 100; end = 20; step = -5;
```

```
    for y4 in range(begin,end,step):
        for y5 in range(begin,end,step):
            for y6 in range(begin,end,step):

                y4 = float(y4)
                y5 = float(y5)
                y6 = float(y6)

                iterations = 0;
                ebound = 0.001*np.ones(4)

                xold = xquess
                while True:
                    validNumber = True
                    iterations = iterations + 1
                    if iterations > 10: # discard solution if more than 10 iterations
                        are performed
                        validNumber = False
                        break

                    try:
                        F = np.array([np.sqrt((xold[0]-x4)**2 + (xold[1]-y4)**2) - L48,
                             np.sqrt((xold[2]-x5)**2 + (xold[3]-y5)**2) - L59, np.sqrt
                            ((xold[2]-x6)**2 + (xold[3]-y6)**2) - L69, np.sqrt((xold
                            [2]-xold[0])**2 + (xold[3]-xold[1])**2) - L89 ])
                    except ArithmeticError: # ignore result if there is a math error
                        validNumber = False
                        break
                    try:
                        Jinv = np.array([
                                        [(-xold[1] + xold[3])*np.sqrt((x4 - xold[0])**2
                                             + (y4 - xold[1])**2)/((x4 - xold[0])*(
                                            xold[1] - xold[3]) - (xold[0] - xold[2])*(
                                            y4 - xold[1])), (y4 - xold[1])*(-(x6 -
                                            xold[2])*(xold[1] - xold[3]) + (xold[0] -
                                            xold[2])*(y6 - xold[3]))*np.sqrt((x5 -
                                            xold[2])**2 + (y5 - xold[3])**2)/(((x4 -
                                            xold[0])*(xold[1] - xold[3]) - (xold[0] -
                                            xold[2])*(y4 - xold[1]))*((x5 - xold[2])*(
                                            y6 - xold[3]) - (x6 - xold[2])*(y5 - xold
                                            [3]))),  (y4 - xold[1])*((x5 - xold[2])*((
                                            x6 - xold[2])*(xold[1] - xold[3]) - (xold
                                            [0] - xold[2])*(y6 - xold[3])) + (xold[0]
                                            - xold[2])*((x5 - xold[2])*(y6 - xold[3])
                                            - (x6 - xold[2])*(y5 - xold[3])))*np.sqrt
                                            ((x6 - xold[2])**2 + (y6 - xold[3])**2)/((
                                            x6 - xold[2])*((x4 - xold[0])*(xold[1] -
                                            xold[3]) - (xold[0] - xold[2])*(y4 - xold
                                            [1]))*((x5 - xold[2])*(y6 - xold[3]) - (x6
                                             - xold[2])*(y5 - xold[3]))), (-y4 + xold
                                            [1])*np.sqrt((xold[0] - xold[2])**2 + (
                                            xold[1] - xold[3])**2)/((x4 - xold[0])*(
                                            xold[1] - xold[3]) - (xold[0] - xold[2])*(
                                            y4 - xold[1]))],
                                        [ (xold[0] - xold[2])*np.sqrt((x4 - xold[0])**2
                                             + (y4 - xold[1])**2)/((x4 - xold[0])*(
                                            xold[1] - xold[3]) - (xold[0] - xold[2])*(
                                            y4 - xold[1])),  (x4 - xold[0])*((x6 -
                                            xold[2])*(xold[1] - xold[3]) - (xold[0] -
                                            xold[2])*(y6 - xold[3]))*np.sqrt((x5 -
                                            xold[2])**2 + (y5 - xold[3])**2)/(((x4 -
                                            xold[0])*(xold[1] - xold[3]) - (xold[0] -
                                            xold[2])*(y4 - xold[1]))*((x5 - xold[2])*(
```

```
                                    y6 - xold[3]) - (x6 - xold[2])*(y5 - xold
                                    [3]))), -(x4 - xold[0])*((x5 - xold[2])*((
                                    x6 - xold[2])*(xold[1] - xold[3]) - (xold
                                    [0] - xold[2])*(y6 - xold[3])) + (xold[0]
                                    - xold[2])*((x5 - xold[2])*(y6 - xold[3])
                                    - (x6 - xold[2])*(y5 - xold[3])))*np.sqrt
                                    ((x6 - xold[2])**2 + (y6 - xold[3])**2)/((
                                    x6 - xold[2])*((x4 - xold[0])*(xold[1] -
                                    xold[3]) - (xold[0] - xold[2])*(y4 - xold
                                    [1]))*((x5 - xold[2])*(y6 - xold[3]) - (x6
                                     - xold[2])*(y5 - xold[3]))), (x4 - xold
                                    [0])*np.sqrt((xold[0] - xold[2])**2 + (
                                    xold[1] - xold[3])**2)/((x4 - xold[0])*(
                                    xold[1] - xold[3]) - (xold[0] - xold[2])*(
                                    y4 - xold[1]))],
                                [

                                    0,

                                    (-y6 + xold[3])*np.sqrt((x5 - xold[2])**2
                                    + (y5 - xold[3])**2)/((x5 - xold[2])*(y6 -
                                     xold[3]) - (x6 - xold[2])*(y5 - xold[3]))
                                    ,

                                    (y5 - xold[3])*np.sqrt((x6 - xold[2])**2 +
                                     (y6 - xold[3])**2)/((x5 - xold[2])*(y6 -
                                    xold[3]) - (x6 - xold[2])*(y5 - xold[3])),


                                    0],
                                [

                                    0,

                                    (x6 - xold[2])*np.sqrt((x5 - xold[2])**2 +
                                     (y5 - xold[3])**2)/((x5 - xold[2])*(y6 -
                                    xold[3]) - (x6 - xold[2])*(y5 - xold[3])),


                                    (-x5 + xold[2])*np.sqrt((x6 - xold[2])**2
                                    + (y6 - xold[3])**2)/((x5 - xold[2])*(y6 -
                                     xold[3]) - (x6 - xold[2])*(y5 - xold[3]))
                                    ,

                                    0]])


            except ArithmeticError: # ignore result if there is a math error
                validNumber = False
                break
            e = Jinv.dot(F) # calculate error
            xnew = xold - e # calculate new estimate
            xold = xnew

            if np.all(np.less(e,ebound)): # stop iterating if error is less
                than lum
                validNumber = True
                break

        # end while

        xquess = xold;
        if y6 == begin:
```

```
                xstore6 = xquess;
            if y5 == begin:
                xtore56 = xquess;

            # check if solution could be valid
            if validNumber:

                q7 = np.arctan2((xnew[3]-xnew[1]),(xnew[2]-xnew[0]))
                x7 = xnew[0] - L78*np.cos(q7)
                y7 = xnew[1] - L78*np.sin(q7)
                x = np.array([x7,y7,q7])
                # check if solution lies within certain bounds
                if np.all(np.isreal(x)) and -np.pi/6.0 <= q7 <= np.pi/6.0 and y6 >
                    xold[3] and y5 > xold[3] and y4 > xold[1]:
                    workspace = np.vstack((workspace,x)) #concatenate solutions
                    print x

        # end for
        xquess = xstore6 #use stored quess when for-loop restarts
    # end for
    xquess = xstore56 #use stored quess when for-loop restarts
# end for

# scatterplot solutions. X,Y is end-effector position Z and color is orientation.
    only orientations withing +-30 degrees are plotted

workspace = np.delete(workspace,0,0)

fig = plt.figure()

ax1 = fig.add_subplot(2,2,1, projection='3d')
ax1.scatter(workspace[:,0], workspace[:,1], workspace[:,2],c = workspace[:,2], cmap
    ="cool", marker="o")
ax1.set_xlim([-50,250])
ax1.set_ylim([-150,100])
ax1.set_zlim([-np.pi/4,np.pi/4])
ax1.set_title("3D view")
ax1.set_xlabel('X (mm)')
ax1.set_ylabel('Y (mm)')
ax1.set_zlabel('R (rad)')
ax1.autoscale_view(True,True,True)

ax2 = fig.add_subplot(2,2,2, projection='3d')
ax2.scatter(workspace[:,0], workspace[:,1], workspace[:,2],c = workspace[:,2], cmap
    ="cool", marker="o")
ax2.set_xlim([-50,250])
ax2.set_ylim([-150,100])
ax2.set_zlim([-np.pi/4,np.pi/4])
ax2.set_title("XR view")
ax2.set_xlabel('X (mm)')
ax2.set_ylabel('Y (mm)')
ax2.set_zlabel('R (rad)')
ax2.view_init(0,-90)
ax2.autoscale_view(True,True,True)

ax3 = fig.add_subplot(2,2,3, projection='3d')
ax3.scatter(workspace[:,0], workspace[:,1], workspace[:,2],c = workspace[:,2], cmap
    ="cool", marker="o")
ax3.set_xlim([-50,250])
ax3.set_ylim([-150,100])
ax3.set_zlim([-np.pi/4,np.pi/4])
ax3.set_title("YR view")
ax3.set_xlabel('X (mm)')
```

```
ax3.set_ylabel('Y (mm)')
ax3.set_zlabel('R (rad)')
ax3.view_init(0,0)
ax3.autoscale_view(True,True,True)

ax4 = fig.add_subplot(2,2,4, projection='3d')
ax4.scatter(workspace[:,0], workspace[:,1], workspace[:,2],c = workspace[:,2], cmap
    ="cool", marker="o")
ax4.set_xlim([-50,250])
ax4.set_ylim([-150,100])
ax4.set_zlim([-np.pi/4,np.pi/4])
ax4.set_title("XY view")
ax4.set_xlabel('X (mm)')
ax4.set_ylabel('Y (mm)')
ax4.set_zlabel('R (rad)')
ax4.view_init(90,-90)
ax4.autoscale_view(True,True,True)

plt.show()
```

B.3 Implementation code mechanism elastostatics

Implementation in python can be found below. Results are printed to console.

```
import numpy as np
np.set_printoptions(precision=4)
np.set_printoptions(threshold=60)
# based on: introduction to finite element method (reader)
# author reader: T. Meinders, A.H. van den Boogaard (2014)
# author scripy: Ruud Spoor (2016)
#
#   B: beam
#
#
#    1*  2*  3*
#     |   |   |
#  (1)|(2)|(3)|
#     |   |   |
#    4*  5*  6*
#   (4)\(5)\ /(6)
#   *---*---*
# 7 (7) 8 (8) 9
#
#    y
#    |
#    |
#    o------ x
#   /
#  z

# nodes 1,4,7: fully supported (Us) | nodes 2,3,5,6,8,9,10: free nodes (Uf)
# applied forces (Ff) | reaction forces (Fs)
# Theory:
# [Ff,Fs]^T = [[Kff,Kfs],[Ksf,Kss]] * [Uf,Us]^T
# Uf = Kff*Uf + Kfs*Us
# Us = 0 by definition: supports do not allow displacement
# Uf = inv(Kff)*Ff
# Fs = Ksf*Uf
#
# u = translational | r = rotational
#
# index|nodeID       index | nodeID        index | nodeID
#   0  |   ux1         6   |  ux2            12   |  ux3
```

```
#   1  |   uy1       7   |  uy2         13   |  uy3
#   2  |   uz1       8   |  uz2         14   |  uz3
#   3  |   rx1       9   |  rx2         15   |  rx3
#   4  |   ry1      10   |  ry2         16   |  ry3
#   5  |   rz1      11   |  rz2         17   |  rz3

# index|nodeID     index | nodeID      index | nodeID
#  18  |   ux4      24   |  ux5         30   |  ux6
#  19  |   uy4      25   |  uy5         31   |  uy6
#  20  |   uz4      26   |  uz5         32   |  uz6
#  21  |   rx4      27   |  rx5         33   |  rx6
#  22  |   ry4      28   |  ry5         34   |  ry6
#  23  |   rz4      29   |  rz5         35   |  rz6

# index|nodeID     index | nodeID      index | nodeID
#  36  |   ux7      42   |  ux8         48   |  ux9
#  37  |   uy7      43   |  uy8         49   |  uy9
#  38  |   uz7      44   |  uz8         50   |  uz9
#  39  |   rx7      45   |  rx8         51   |  rx9
#  40  |   ry7      46   |  ry8         52   |  ry9
#  41  |   rz7      47   |  rz8         53   |  rz9

numberOfNodes = 9;
node1 = np.array([0,1,2,3,4,5]); node2 = np.array([6,7,8,9,10,11]); node3 = np.
    array([12,13,14,15,16,17]);
node4 = np.array([18,19,20,21,22,23]); node5 = np.array([24,25,26,27,28,29]); node6
     = np.array([30,31,32,33,34,35]);
node7 = np.array([36,37,38,39,40,41]); node8 = np.array([42,43,44,45,46,47]); node9
     = np.array([48,49,50,51,52,53]);

####################
# HELPER FUNCTIONS #
####################

def addBeam(sysMat, NodeP, NodeQ, L, Rmat):
    # addBeam: add beam with specific paramters to global system matrix sysMat
    # sysMat: global system matrix
    # NodeP: 1x6 array, indexing P node elements in system matrix. NodeP = np.array
        ([uxp,uyp,uzp,rxp,ryp,rzp])
    # NodeQ: 1x6 array, indexing Q node elements in system matrix. NodeQ = np.array
        ([uxq,uyq,uzq,rxq,ryq,rzq])
    # L: length of beam element
    # Rmat: 3x3 rotation matrix. Rmat = np.aray([[1,0,0],[0,1,0],[0,0,1]])

    # 1) axial
    # 2) torsional
    # 3) bending y
    # 4) shear y
    # 5) bending z
    # 6) shear z

    # 5mm circular beam with length of 100mm
    A = 2*10**(-5);
    I = 1*10**(-9);
    J = 6*10**(-11);
    E = 2*10**(11);
    G = 8*10**(10);

    A = (E*A)/L; B = (E*I)/(L**3); C = (E*I)/(L**3); D = (G*J)/(L);

    #               uxp uyp uzp rxp ryp rzp uxq uyq uzq rxq ryq rzq

    Kel = np.array([[A,  0,  0,  0 , 0 , 0, -A,  0,  0,  0,  0,  0 ],\
```

```
                    [0, 12*B,0,  0,  0,6*L*B,0,-12*B,0,  0,  0,6*L*B],\
                    [0,  0, 12*C,0,6*L*C,0,  0,  0,-12*C,0,6*L*C,0],\
                    [0,  0,  0,  D,  0,  0,  0,  0,  0, -D,  0,  0,],\
                    [0,  0,6*L*D,0,4*L*L*D,0,0,0,-6*L*D,0,2*L*L*D,0,],\
                    [0,6*L*B,0,0,0,4*L*L*B,0,-6*L*B,0,0,0,2*L*L*B,],\
                   [-A,  0,  0,  0,  0,  0, A,  0,  0,  0,  0,  0,],\
                    [0,-12*B, 0,  0,  0,-6*L*B,0,12*B,0,0,0,-6*L*B,],\
                    [0,  0,-12*C,  0,-6*L*C,0,  0,  0,12*C,0,-6*L*C,  0,],\
                    [0,  0,  0, -D,  0,  0,  0,  0,  0,  D,  0,  0,],\
                    [0,  0,6*L*D,0,2*L*L*D,0,0,0,-6*L*D,0,4*L*L*D,0,],\
                    [0,6*L*B,0,0,0,2*L*L*B,0,-6*L*B,0,0,0,4*L*L*B]])

    R = np.zeros([12,12])
    R[0:3,0:3] = Rmat; R[3:6,3:6] = Rmat; R[6:9,6:9] = Rmat; R[9:12,9:12] = Rmat
    Kg = np.transpose(R).dot(Kel.dot(R))
    idx = np.concatenate((NodeP,NodeQ))
    for r in range(0,Kel.shape[0]):
        for c in range(0,Kel.shape[1]):
            sysMat[idx[r],idx[c]] = sysMat[idx[r],idx[c]] + Kg[r,c]
    return sysMat

#########################################################################
#                                MAIN                                   #
#########################################################################

# define global system matrix

globalStiffnessMatrix = np.zeros((6*numberOfNodes,6*numberOfNodes));

# Beam 1
theta = -np.pi/2.0;
NodeP = node1; NodeQ = node4; L = 0.1; Rmat = np.array([[np.cos(theta),np.sin(theta
    ),0],[-np.sin(theta),np.cos(theta),0],[0,0,1]])
globalStiffnessMatrix = addBeam(globalStiffnessMatrix,NodeP,NodeQ,L,Rmat)

# Beam 2
theta = -np.pi/2.0
NodeP = node2; NodeQ = node5; L = 0.1; Rmat = np.array([[np.cos(theta),np.sin(theta
    ),0],[-np.sin(theta),np.cos(theta),0],[0,0,1]])
globalStiffnessMatrix = addBeam(globalStiffnessMatrix,NodeP,NodeQ,L,Rmat)

# Beam 3
theta = -np.pi/2.0
NodeP = node3; NodeQ = node6; L = 0.1; Rmat = np.array([[np.cos(theta),np.sin(theta
    ),0],[-np.sin(theta),np.cos(theta),0],[0,0,1]])
globalStiffnessMatrix = addBeam(globalStiffnessMatrix,NodeP,NodeQ,L,Rmat)

# Beam 4
theta = np.arccos(15.0/-90.0)
NodeP = node4; NodeQ = node8; L = 0.09; Rmat = np.array([[np.cos(theta),np.sin(
    theta),0],[-np.sin(theta),np.cos(theta),0],[0,0,1]])
globalStiffnessMatrix = addBeam(globalStiffnessMatrix,NodeP,NodeQ,L,Rmat)

# Beam 5
theta = np.arccos(15.0/-90.0)
NodeP = node5; NodeQ = node9; L = 0.09; Rmat = np.array([[np.cos(theta),np.sin(
    theta),0],[-np.sin(theta),np.cos(theta),0],[0,0,1]])
globalStiffnessMatrix = addBeam(globalStiffnessMatrix,NodeP,NodeQ,L,Rmat)

# Beam 6
theta = np.arccos(-15.0/90.0)
```

```
NodeP = node9; NodeQ = node6; L = 0.09; Rmat = np.array([[np.cos(theta),np.sin(
    theta),0],[-np.sin(theta),np.cos(theta),0],[0,0,1]]) # pay attention to node P
     and Q
globalStiffnessMatrix = addBeam(globalStiffnessMatrix,NodeP,NodeQ,L,Rmat)

# Beam 7
theta = 0.0
NodeP = node7; NodeQ = node8; L = 0.03; Rmat = np.array([[np.cos(theta),np.sin(
    theta),0],[-np.sin(theta),np.cos(theta),0],[0,0,1]])
globalStiffnessMatrix = addBeam(globalStiffnessMatrix,NodeP,NodeQ,L,Rmat)

# Beam 8
theta = 0.0
NodeP = node8; NodeQ = node9; L = 0.03; Rmat = np.array([[np.cos(theta),np.sin(
    theta),0],[-np.sin(theta),np.cos(theta),0],[0,0,1]])
globalStiffnessMatrix = addBeam(globalStiffnessMatrix,NodeP,NodeQ,L,Rmat)

# define applied forces
F = np.zeros((6*numberOfNodes,1));
F[node7] = np.array([[0],[0],[10.0], [0],[0],[0]])
print "applied␣forces:␣node7␣[0␣0␣10␣0␣0␣0]"

# calculate nodal displacements free nodes
Kff= np.delete(globalStiffnessMatrix,np.append(node1,[node2,node3]),1);
Kff= np.delete(Kff,np.append(node1,[node2,node3]),0);
Ff = np.delete(F,np.append(node1,[node2,node3]),0);
Kffinv  =np.linalg.inv(Kff)
Uf = Kffinv.dot(Ff)

# calculate reaction forces supported nodes
Ksf = globalStiffnessMatrix[np.append(node1,[node2,node3]),:];
Ksf = np.delete(Ksf,np.append(node1,[node2,node3]),1);
Fs = Ksf.dot(Uf);

print ""
print "deviations:␣"
for i in range(0,36,6):
    print("␣ux" + str(i/6+4) + ":␣" + str(Uf[i+0]) +  "␣uy" + str(i/6+4) + ":␣" +
        str(Uf[i+1]) +  "␣uz" + str(i/6+4) + ":␣" + str(Uf[i+2])+  "␣rx" + str(i
        /6+4) + ":␣" + str(Uf[i+3]) +  "␣ry" + str(i/6+4) +":␣" + str(Uf[i+4]) +
        "␣rz" + str(i/6+4) + ":␣" + str(Uf[i+5]));

print ""
print "reaction␣forces:␣"
for i in range(0,18,6):
    print("␣Fx" + str(i/6 + 1) + ":␣" + str(Fs[i + 0]) + "␣Fy" + str(i/6 + 1) + ":␣
        " + str(Fs[i + 1]) +  "␣Fz" + str(i/6 + 1) + ":␣" + str(Fs[i + 2]) +  "␣Tx
        " + str(i/6 + 1) + ":␣" + str(Fs[i + 3]) +  "␣Ty" + str(i/6 + 1) + ":␣" +
        str(Fs[i + 4]) +  "␣Tz" + str(i/6 + 1) + ":␣" + str(Fs[i + 5]))

print ""
print "reference␣drawing:␣"
print "␣␣␣1*␣␣2*␣␣3*"
print "␣␣␣|␣␣␣|␣␣␣|"
print "␣(1)|(2)|(3)|"
print "␣␣␣|␣␣␣|␣␣␣|"
print "␣␣␣4*␣␣5*␣␣6*"
print "␣␣(4)\(5)\␣/(6)"
print "␣␣*---*---*"
print "7␣(7)␣8␣(8)␣9"
print ""
print "␣␣␣y"
print "␣␣␣|␣"
```

```
    print "␣␣␣|"
    print "␣␣␣o————␣x"
    print "␣␣/"
    print "␣z␣"
```

C Appendix 3

As a small side experiment, the whole project described in this report was done with the use of free and open source software. The reader can find some information on the used software in the next sections.

C.1 Python

Python is a high level interpreted programming language. It is easy to learn and can be used to quickly develop software. In this project, python is used as a substitute for the expensive matlab environment. Compared to matlab, python is free, open source and cross platform. Although matlab has allot more build in functions, huge variety of libraries can easily be imported in a python script. Examples are numeric, symbolic or graph plotting toolboxes designed to work similar to matlab.

Coding a python script takes about the same time as creating a matlab script. Command line programming is also possible just like matlab's command line.

Based on this project, python is seen as a suitable substitute for matlab. Furthermore, as the fast majority of companies do not have a matlab license, it is more efficient to master python as it can be used all the time.

C.2 Freecad

FreeCAD is a graphical 3D parametric modeler. It is again free, open-source and cross-platform. To the author's knowledge, it is the only CAD software which is truly cross-platform with similar functionality as programs like Solidworks or Pro-engineer. Although Solidworks has a more user friendly interface, FreeCAD is just as easy to use.

C.3 Mbed

Mbed is an online compiler used to program the microcontroller driving the stepper motors. Mbed is similar to the arduino IDE, but can be accessed from any browser. Code is created and compiled online and send back as a binary download. Mbed enabled microcontrollers show themselves to a pc as mass storage devices. One only has to place the downloaded binary in the microcontroller's folder to upload code to the microcontroller. Many of the mbed enabled microcontrollers like nucleo-board have the same pin interface as arduino shields. Mbed has furthermore similar driver libraries just like the arduino IDE.

Bibliography

A., B.-I. (1966), A Newton-Raphson method for the solution of systems of equations, in *Journal of Mathematical analysis and applications, 15(2),*, pp. pp.243–252.

Chatelain P., Krupa A., N. N. (2015), 3D ultrasound-guided robotic steering of a flexible needle via visual servoing, in *2015 IEEE International Conference on Robotics and Automation*, pp. pp. 2250–2255.

G., P. (2010), Basic principles of risk management for medical device design.

Haddadin S., Albu-Schaffer A., H. G. (2007), Safety Evaluation of Physical Human-Robot Interaction via Crash-Testing, in *In Robotics: Science and Systems Vol. 3*, pp. 217–224.

Hong J., Dohi T., H. M. K. K. H. N. (2004), An ultrasound-driven needle-insertion robot for percutaneous cholecystostomy.

Huang S.Y., Boone J.M., Y. K. P. N. M. S. P. N. L. K. Y. M. (2011), The characterization of breast anatomical metrics using dedicated breast CT., in *Medical physics, 38(4)*, pp. pp.2180–2191.

Jonker J.B., Aarts R.G.K.M., M. J. (2015), *Flexible multibody dynamics for (control) design purposes*.

Mathiassen K., Fjellin J.E., G. K. H. P. E. O. (2016), An Ultrasound Robotic System Using the Commercial Robot UR5, in *Frontiers in Robotics and AI, 3*, p. p.1.

Meinders T., v. d. B. A. (2014), *An Introduction to the Finite Element Method*.

Pierrot F., Dombre E., D. E. U. L. C. P. B. S. G. J. M. J. (1999), Hippocrate: a safe robot arm for medical applications with force feedback, Medical Image Analysis 3(3), pp. 285–300.

Priester M., Natarajan S., C. M. (2013), Robotic Ultrasound Systems in Medicine, in *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, pp. 507–523.

Rocha R., Pinto R., T. D. G. C. (2013), Step-by-step of ultrasound-guided core-needle biopsy of the breast: review and technique.

Salcudean S.E., Bell G., B. S. Z. W. A. P. L. P. (1999), Robot-assisted diagnostic ultrasoundâĂŞdesign and feasibility experiments, in *In Medical Image Computing and Computer-Assisted InterventionâĂŞMICCAI*, Springer Berlin Heidelberg, pp. 1062–1071.

Smith-Guerin N., Bassit L.A., P. G. D. C. A. P. V. P. (2003), Clinical validation of a mobile patient-expert tele-echography system using ISDN lines, in *In Information Technology Applications in Biomedicine, 2003. 4th International IEEE EMBS Special Topic Conference on (pp. 23-26). IEEE.*

Yamada Y., Hirasawa Y., H. S. U. Y. (1996), Fail-safe human/robot contact in the safety space., in *In Robot and Human Communication, 1996., 5th IEEE International Workshop*, IEEE, pp. 59–64.