Mälardalen University
School of Innovation Design and Engineering
Västerås, Sweden

Robotics - Project Course 30 ECTS
Advanced Embedded Systems - Project Course 30 ECTS

PRECIS POSITIONING AND SCANNING FOR MICROWAVE IMAGING

Victor Aziz
vaz17001@student.mdu.se

Aref Bahtiti
abi18002@student.mdu.se

Ingrid Heien Bjonge
ihe21001@student.mdu.se

Emanuel Bjurhager
ebr17003@student.mdu.se

Jonathan Kenth Holm
jhm18006@student.mdu.se

Amanda Rautio
aro18001@student.mdu.se

2023-01-13

Abstract

Emanuel & Ingrid

Cancer has long been one of the leading causes of death in humans, with breast cancer being the most frequent type of cancer in women. There are several techniques for screening for breast cancer, the most common of which is Mammography. However, Mammography has disadvantages and side effects, making a new screening method desirable. A research group at MDU is developing a new screening technique based on microwave tomography. It uses non-ionizing microwaves to distinguish between healthy tissue and irregularities. To accomplish this, a transmitter and receiver antenna must be placed on the object's surface, with the antenna requiring precise positioning. This project aims to build a system that can perform an automatic microwave tomography scan in collaboration with the MDU research group. The solution uses two robots: one to place the transmitter and the other to place the receiver and scan the object with a laser. The breast is first laser scanned to generate a point cloud, which is then used to generate a 3D reconstruction of the object with Power Crust. The 3D reconstruction provides the system with information about the shape and location of the Object Under Study (OUS). A GUI is used to configure the region and density for the microwave tomography scan. The results show that, with a few exceptions, the system can automatically place the receiver antennas at the desired locations for a microwave tomography scan. The results are promising and show that this system has the potential to be used for microwave tomography in the future, but some improvements are needed.

Contents

List of Figures

List of Tables

1. Introduction - *Amanda*

Breast cancer is one of the most common types of cancer and predominantly affects women. Around 30% of the diagnosed cancer among women is breast cancer, and in Sweden alone, according to the Swedish Cancer Society, approximately 20 women get diagnosed with it every day [1]. Often, the cancer is close to symptom less in its preliminary stages, making it hard to detect. But if found early, the likelihood of eliminating the cancer is high. Commonly, breast cancer is found with mammography [2], where X-ray images are taken of the breast. This process requires the breast to be compressed, which may feel uncomfortable or cause pain for the patient. The X-ray used in mammography exposes the patient to ionising radiation, which, while being in tiny doses, still pose as a potential risk for the patient. Mammography has limited capacity in identifying tumours without a characteristic mass, often seen in lobular breast cancer or tumours without calcification [3]. Other solutions, such as MRI and ultrasound, also have limitations. Therefore, there is a need for novel solutions that can detect breast cancer early.

Microwave tomography (MWT) is a biomedical imaging technique where microwaves are used to illuminate a target from multiple angles and where the collected data is used to create cross-sectional images. Microwave imaging makes it possible to detect cancerous tumours as healthy and cancerous tissue respond differently to electrical fields in the microwave spectrum, making them distinguishable from one another in the produced images. MWT poses as a potential complement to mammography, as it is non-intrusive, comfortable and without ionising radiation, making it suitable for both detection and recurrent monitoring during treatment [3].

Innovation and development in the field of MWT is the focus of an ongoing project at Mälardalens University. Previously in the project, a robot-controlled data acquisition system for microwave imaging was developed and validated [4]. The system consisted of a robotic arm, two monopole antennas, a water tank in which the object to be scanned and the antennas were placed, and a MATLAB interface for controlling the data acquisition. During the scanning, both the transmitting and receiving antennas would be immersed in the water tank, where the transmitting antenna was in a fixed position while the receiving antenna was fixed to the robotic arm, which could move along the surface of the object in a pre-programmed pattern. Because the receiving sensor only moved in a pre-programmed pattern with no consideration for the object's placement, the object itself had to be positioned in relation to the robot's coordinate system with high accuracy. When validated through comparison between measured and simulated data, the system showed promising results.

Since then, different prototypes of arm designs have been developed and tested, and improvements to the transmitters have been made, making them suitable for a dry set-up. These transmitters are now a microwave sender that is meant to be placed under the object, and a receiver that should be moved over the object, measuring the microwave signals at different parts of the object. To achieve this, a robotic system that can move the transmitters over the object at a high degree of freedom and place them at specified distances from the object with high precision is needed, as the characteristics of the microwave signal change greatly depending on the distance to the object. Additionally, a 3D model of the object, accompanied with a graphical user interface needs to be created to aid with the positioning and user interaction to control scanning area, distance and patterns.

In this project, a prototype system for the precise positioning of the microwave antennas has been developed. The system consist of a ABB SAY collaborative robot that positions the receiving antenna and laser sensor at different positions around the object, a custom made robotic arm that positions the sending antenna under the object, a laser sensor that measures distance and is used to scan the object to determine its shape, scanning and object reconstruction algorithms, a GUI that controls all measuring sequences and a frame on which the object is mounted. Results when verifying the system shows that there are some issues when it comes to the positioning accuracy of the system. This is credited to poor calibration of the SAY robotic arm, which was not in the best condition. Therefore, the desired accuracy with under one millilitre precision has not been

achieved. However, the system still showed good repeatability in positions, and can successfully scan, reconstruct and control the positions of both receiving and sending antenna arms, thereby posing as a good base for future improvements and development.

In this report, a in-depth description of the system and its components is made, as well as the reasoning behind them. Relevant background and related work is presented and the verification method to get the results is explained. Lastly, results of the verification is presented and the result and future work is discussed.

1.1 Problem Formulation - *Amanda*

Previously at Mälardalen University, researchers had developed microwave imaging antennas with the purpose of detecting breast cancer [4]. Unlike many other microwave imaging antennas, these are intended to be used in a dry setup with no contact to the object being studied. To test these antennas and further advance the research, a positioning and measurement system that can place the antennas at various distances from the object with a high degree of freedom is needed. Because of the properties of the antennas, this positioning system must have high accuracy and distance control, preferably under a millimetre. Furthermore, the antennas should be perpendicular to the object's surface. However, since the exact shape and size of the object is unknown at the start of a measuring session, the surface of the object first needs to be defined to achieve such precision. Therefore, the measurement session can be divided into two phases: the initial scanning phase to gain knowledge of the object, and the microwave scanning phase, which uses the previously gained knowledge to achieve better positioning of the antennas. The hardware used for the system also plays a crucial role in the accuracy that can be achieved. For this project, ABB has lent out a SAY collaborative robot, which will be used to position the receiving antenna. Hence, a part of the project is focused on investigating the suitability of the SAY for this purpose.

The purpose of this project is to develop a robotic system designed for microwave imaging, which can achieve precise positioning of the microwave antennas, relative to the object under study, at a high degree of freedom.

To achieve this purpose, the following research questions and goals have been defined:

1.1.1 Research questions

1. How suitable is the SAY for the purpose of microwave imaging?
2. At what accuracy can the system position the sensors?
3. With what accuracy can the system replicate the object's surface?
4. What limitations does the solution have?

1.1.2 Goals

1. Develop a robotic system that can move at least one applicator around an object under study with all the required degrees of freedom.
2. Implement a surface scanning and/or positioning system to support the exact placement of the antenna at a required distance d away from the object under study ($0 < d < 10$mm).
3. Develop the necessary control software to move the transmitter to an arbitrary position and to run a pre-defined measurement sequence.
4. Develop a graphical user interface to control the measurement system.

2. Background

2.1 Moving object in 3D space - *Victor*

Robotic manipulators, mechanical devices made up of links connected by actuators (joints), are frequently used to move objects in three-dimensional space. The base forms one end of the link series, and the end-effector, at the other end, is the moving end of the manipulator.

There are two types of joints: prismatic and revolute joints, which allow the link to move in a straight line or rotate about an axis. Even though prismatic joints are easier to understand and work with, the most common type is the revolute joint. Each joint can only impose a position in a single dimension on a robotic manipulator's space. For example, the robotic manipulator needs three joints to move an object to a specific place (x,y,z) in 3D space. The robotic manipulator typically has six joints; the first three are for positioning the object, and the last three are for orienting the object in space [5].

The position of the end effector is controlled by adjusting the angles of the joints. Forward kinematics is the relationship between the joints' angles and the end effector's position. The inverse kinematic relation must be solved to find the joint value needed to force a specific position (x, y, and z). Some (x, y, z) values in the space have no solution for the inverse kinematics, making the point out of reach for the robotic arm. Other points might have more than one solution for inverse kinematics, so the robotic manipulator could be placed differently to reach the same point in space. This makes it easier for the robotic manipulator to avoid collisions because it can choose the best way to move from several options. Multiple solutions often occur when the number of joints on the robotic manipulator exceeds the required degree of freedom [6].

Kinematics modelling is a straightforward depiction of how an automatic manipulator functions; in the real world, control is carried out through intricate motion control schemes that consider system dynamics and actuator noise. Moving an object requires motion over time, so from another angle, the robot's motion must be anticipated by mapping out its course and creating a trajectory. Because the robotic system is complicated and uses a lot of computing power, it needs different tools, numerical methods, and algorithms. Different coordinate systems can be used in robotics applications to specify an object's or a robot's position and orientation. Position and orientation are modelled as rotation and translation in computation. The homogenous transformation matrix, which combines rotation and translation into one matrix, is a helpful way to represent a pose. Axis-angle representation, which describes the rotation by a scalar rotation around a fixed axis defined by a vector, can also describe the rotation in 3-D space. The Euler angles representation, which uses three scalars to represent a rotation around a specific coordinate frame axis (such as "ZYZ," "XYX,"...), is a more condensed way to represent rotation or orientation. If the coordinate frame axis order is "ZYX," the Euler angles representation can also be written as "Roll, Pitch, Yaw" [6]. The robotics system toolbox is a helpful MATLAB tool for designing, simulating, and testing robotics applications [7].It offers a coordinate transformation besides manipulator design algorithms.

Robotic manipulators are a standard tool that has recently gained popularity and accessibility for individuals and businesses. The models on the market range from the less expensive, less precise models to the high-priced, high-precision models. One of the top companies in the world for creating robots with high-precision standards is ABB. Their robotic manipulators come in a variety of sizes and functions. Aside from being highly precise, some manipulators are certified to be used near people and in the medical setting to administer medication to patients. Most low-cost models have stepper motors and a small single-board computer. On the other hand, high-precision models have servo motors and computers with more features. Besides the more affordable models on the market, there are many open-source and public projects for constructing a robotic arm [8].

2.2 3D surface reconstruction - *Jonathan and Victor*

Three-dimensional (3D) reconstruction is a mathematical representation of a real-world object. 3D scanning and reconstruction are becoming more common in many industries, including medicine, where it is being used to improve treatment [9]. The method for 3D reconstruction is often done by first collecting data by using a sensor to determine where the object/objects are. These sensors are available in both contact and contactless configurations [10]. Using non-contact sensor technology (a distance sensor is an option; one example could be LIDAR), the 3D model can be created without any physical contact with the object under study. The discrete points are collected into a point cloud. A point cloud is a collection of discrete data points arranged in 2D or 3D space. The cloud is then combined with some algorithm that processes the points together to finalise a surface.

When working with data collected from practical applications, there will certainly be noise in the data. Reconstruction of objects digitally is no exception and can make a representation look nothing like the actual object that was scanned if it is not dealt with [11]. Speckle noise and reflections are some of the largest contributors to the noise in image reconstruction [12]-[13]. Many proposed solutions to the problem have been tried, ranging from using filters on the signal to extract the desired data to smoothing data using different mathematical functions [14].

3D reconstruction is widely used in multiple fields, which creates a wide variety of algorithms. Each algorithm has its own pros, cons, and prior assumptions about the scannable object and the data [15]-[16] . In summary, two aspects characterize the solutions, the prior knowledge about the scannable object and the information collected in the acquisition phase [17]. A solution, as in [18] requires information about the sampling process while another solution [19] needs surface normal and sample point neighbour information [20]. Many 3D reconstruction algorithms use basic methods from computational geometry, such as Voronoi diagrams, Delaunay triangulation, and the medial axis. In the next section, a short description will explain some terms, while better descriptions and mathematical proofs can be found in [21].

The Voronoi diagram shown in figure 1 is a map of a surface created from sample points called sites that have specific coordinates. The sites are separated by edges which create different regions or cells for every site. A vertex is when 3 Voronoi cells meet at a point. The important purpose of the Voronoi diagram is that in a specific site region, all other coordinates are closest to the region site than to any other site.

On the boundary of the Voronoi diagram representation, there is 2 types of regions, the bounded and unbounded regions. The bounded regions are where the edges of that region converge to an intersection while unbounded regions' edges do not intersect. A contour shape is created by joining all the unbounded regions. The contour is called a convex hull.

A Delaunay triangulation is also a surface map representation that instead of dividing the surface into regions, the convex hull is divided into triangles see figure 2. The unique property of Delaunay triangulation is that every Circumscribed circle around each triangle on the surface is empty e,g The circle does not contain any other point from the points set.

Finally the medial axis shown in figure 3 is a set of points that represents the skeleton of the shape.In a sufficiently sampled dense set of point, the Voronoi vertices approximate the medial axis of a shape.

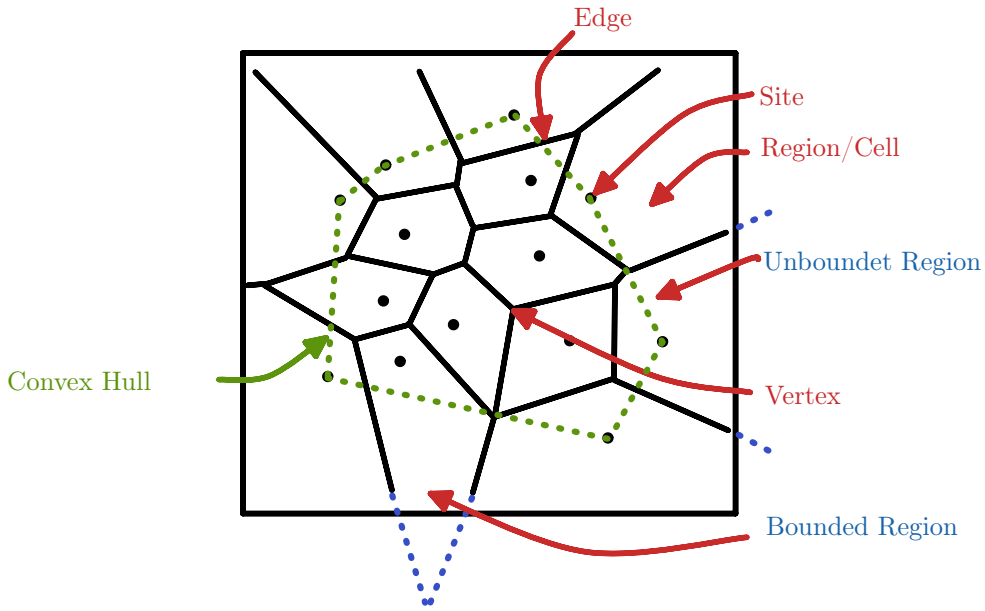


Figure 1: The Voronoi diagram termonology

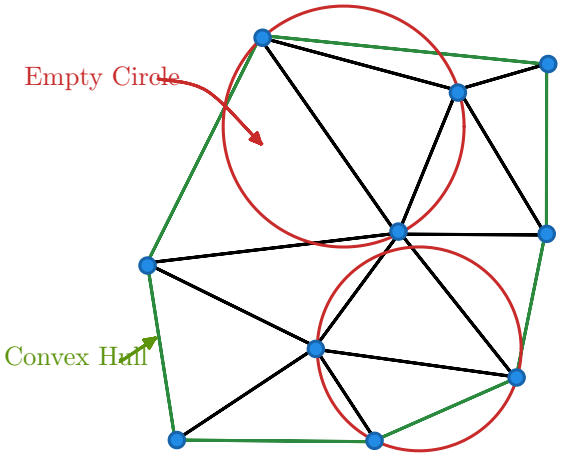


Figure 2: The delaunay triangulation that devide the green convex hull in black triangles and the circumscribed empty circle in red of 2 triangles that does not contain any other blue points.

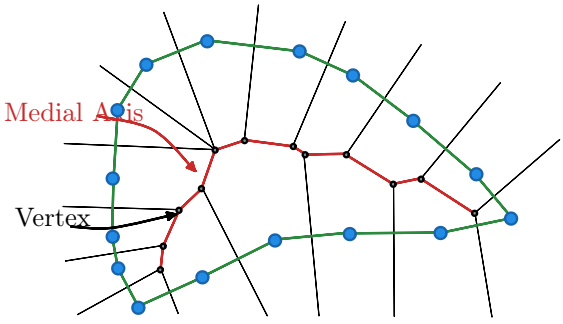


Figure 3: The medial Axis in red is created by connecting the black vertices of the voronoi diagram.

The crust and alpha shape algorithms are popular 3D reconstruction techniques. Both only require a set of points, but the alpha shape also uses an additional tolerance parameter. In an alpha shape, the tolerance parameter shows how close the surface should be to the set of points. The power crust does not use tolerance parameters because it is proven, in theory, to recreate the exact surface[22]. The Alfa shape algorithm is built into MATLAB and is available in the Computational Geometry library[23]. The MATLAB code for power crust can be found on GitHut[24]. It takes a set of points and makes a triangulated surface. The alpha-shape algorithm makes an alpha-shape object that represents the surface and has different functions, like checking if a point is inside the alpha-shape.

2.2.1 α-shape

One popular algorithm for 3D reconstruction is α-shape. The algorithm uses Voronoi diagram and Deluney triangulation to minimize a volume in a space R and "carve out" everything around the points S of the surface being reconstructed. As described by the creators of the α-shape algorithm; one can think of α-shape as a rock stuck in sand. The sand being a space Rd and the rock being described by the point set called S. With a spherical shovel of radius α, sand is removed from the space without touching the points in S of the rock. Which will finalize in a shape described by the space left in the sand [25]. Negative properties with the method are that the data cant be non-uniform and little to no noise present. This will may cause it to be hard or even impossible to find an α [26].

2.2.2 The Power Crust algorithm

A popular algorithm; Power Crust[26] is a 3D-reconstruction algorithm. Benefits with the Power crust algorithm is that the data can be unorganized and requires no additional information about the surface normal or the sampling process. Given a priory knowledge of smooth surface and high-density sampling, The Power Crust algorithm is theoretically proven to produce a surface that is topologically equivalent to the real surface[22]. The crust algorithm does not have any problem with some noise or outlier in the data, but it cannot handle data that has large signal-to-noise ratio or sharp edges. It also requires a post-processing stage to get rid of invalid parts of the surface[26].

3. Related Work - *Jonathan and Amanda*

A multitude of different hardware and software setups for microwave imaging have been developed since the technology first emerged. In the early 2000s Meaney et al. [27] were one of the first to develop a clinical prototype for microwave imaging of a breast. The system comprised 16 monopole antennas arranged in a circular array configuration, where the antennas were submerged in a saline bath, and the patient was laid on a table above the tank with the breast hanging into the fluid. Since then, a large quantity of systems have emerged with similar designs. Using multiple transmitters in a circular array could still be regarded as state-of-the art today [28], since it is still used by the Wavelia system.

The Wavelia system [29] is a microwave breast imaging device developed by MVG Industries and currently under clinical investigation. Similarly to the system developed by Meaney et al., the Wavelia system also comprises multiple antennas arranged in a circular array, where the patient lies prone on a table with the breast hanging into the coupling fluid. But the new system differentiates itself by using 18 equally spaced wideband Vivaldi-type probes, placed outside of the fluid filled cylinder. These probes are rotated vertically around the breast, illuminating it in 5 mm intervals. The microwave breast imaging system is also paired with an optical breast contour detection (OBCD) system. The OBCD system allows the external surface of the breast to be reconstructed and the breast volume to be calculated with the help of a Three-dimensional (3D) stereoscopic camera, placed below the examination table. As of now, the system has shown good results, both when detecting dielectric contrast between tumour phantoms and synthetic breast models, and in clinical studies with human subjects. However, the current generation of the system cannot detect lesions less than 10mm, and is unsuitable for patients with smaller breasts, as the breast must have enough mass to create a pendulous reach when lying prone. The system is also limited in detecting abnormalities near the underlying chest muscles, as there is 2-4 centimetre between the uppermost scanning position and the chest wall. Using coupling fluid is also not preferred because it is can be seen as unhygienic and requires the patient to clean their breasts after examination.

Circular transmitter arrays are however not the only solution when it comes to reconstruction of breast surfaces and microwave scanning. A research group based at the University of Calgary [30]; was able to recreate a breast surface through point laser scanning and microwave imaging. The laser scanning was performed with a laser with an accuracy lower than 400 μm. Tilting the laser at a 15 °angle was also tested to improve the curve estimation of the breast. The laser point data was later fed into the PowerCrust algorithm to get a surface estimation. Both methods were performed with the breast submerged in a liquid. The paper concluded with a result that showed laser were superior to microwave imaging.

The same research group from the same university [31] also successfully created a method for automatically scanning for breast cancer. Their 3D reconstruction was done using a laser sensor mounted on a four-degrees-of-freedom robotic arm, which measured the breast in the horizontal plane. The 3D scanning protocol was based on scanning the breasts at different heights in the Z-plane to get a representation of the contour at different angles in the X, and Y plane. Creating a point cloud representing the contour of the breast at different angles. The data collected was interpolated and then used to get an image of the entire breast. The reconstruction was then used as a reference when controlling the robotic arm, which had a sensor for detecting anomalies in the breast tissue. The paper concluded that their surface reconstruction using laser triangulation was quick and successful.

Another solution to get a accurate 3D-estimation of a object was conducted by a group based in the Technical University of Braunschweig [32]. By combining the information of both laser and a camera the error could be reduced down to 1mm off from the actual model. The laser was sweept over the object while a color camera is used to detect the laser strip on the object the frames was then interpolated to get surface estimation.

Another research group [33], also worked with breast cancer detection, but in contrast to the group based in Calgary, this 3D reconstruction technique used a webcam to estimate the surface of the breast. The camera took 12 pictures of the breast in the azimuth plane; ranging from 0 to 165 °angles. By using edge detection, the breast contour could be approximated. This procedure was repeated around the Z-axis in 0.5mm intervals in the X, Y-plane. By combining all the breast contour estimations, a 3D surface could be estimated. The greatest surface error was 2mm compared to the real breast. However, the Z-axis was elongated in the estimation, which caused the position of the tumor to be off by around 10mm in the z-direction.

Currently, no evidence has been found that microwave tomography is used in practice, as it is still a method under research. The Wavelia system mentioned above is one of the systems closest to a finished product, currently undergoing clinical trials. Therefore, in the area of breast tumor detection, mammography is the state of practice.

4. Method - *Victor*

The system is built by linking two subsystems: the positioning and scanning systems. The positioning system's role is to place the measurement tools in 3D space with a specific orientation. The scanning system uses the positioning system to scan the OUS with a scanning device and reconstruct the OUS surface. A power supply then powers the entire system, and the two subsystems communicate over a network. Finally, a GUI made in MATLAB with many features is used to control the whole system.The following section will summarize the parts of the two subsystems and explain the motivation behind the chosen solutions. The method used to design an automatic microwave measurement system considers a time constraint of 20 weeks and a budget of 20,000 SEK.

Three solutions for positioning in 3D space have been discovered. The first solution is to use an industrial manipulator, the second is to buy a low-cost 3D-printed robotic arm, and the third is to design and build a 3D-printed robotic arm. Unfortunately, the cost of the industrial manipulator is more than the budget for the project, and the cheaper alternative does not give the project the precision it needs. Finally, because of the project's time constraints, it is impossible to build a robotic manipulator. In collaboration with ABB, the company provides a single Arm Yumi (SAY) prototype for testing its use case in the application at no cost. The SAY is used to position the receiver antenna and the scanning device. In contrast, the sender antenna positioning robot is designed and built within the project's scope. Even though SAY can pick tools with a gripper at the end effector, cables complicate the operation. So, a new SAY end-effector is being designed and printed with 3D printing to hold the receiver antenna and the scanning device on SAY. However, because the antennas are fragile, 3D-printed copies are used for the evaluation. Section 6.1 explains the details of how the positioning system works. It includes information about the sender robot, SAY, 3D-printed copies of the antennas, and the new end-effector.

The scanning system has two primary functions: scanning and reconstructing the surface from the scanning data. The method is tested using two 3D-printed OUS models, one symmetrical and one asymmetrical. The OUS models are mounted on a System Installation Bench (SIB) designed to be mounted on an ABB SAY table. The symmetrical model's purpose is to make it easier to identify errors in the system because the actual surface of the symmetrical model is easier to predict. Four main types of devices are suitable for contactless measurements. The devices include a 3D scanner available at MDU, a camera, microwave technology, and a laser distance sensor. The 3D Scanner is eliminated because another person has already reserved it; however, the 3D Scanner size may have difficulties reaching all points around the OUS, and SAY may not hold its weight. According to a literature review, a triangulation laser is the most accurate solution. Another solution emerged, which is the combination of the laser and the camera. However, this solution was ruled out because of the project's time constraints and the need to create another positioning system for the camera. A laser distance scanning sensor is chosen as a scanning device. While there are various types of laser distance sensors, the best models used in the industry have strip light beams but are more expensive than the project budget. The scanning device used for this project is a laser single-beam sensor from an earlier project that was already available at MDU.

The method used to scan the OUS and reconstruct its surface is based on the journal [34]. To better meet the project's requirements, adding a new step may improve the method described in the journal. Initially, the laser measures the distance to the OUS surface from various positions. Then a linear mapping of the laser value and position produces 3D points $P_i = (x_i, y_i, z_i)$, which are the Cartesian coordinates of the measured point Pi on the OUS surface. The next step is to fit non-linear curves to the measured point set and re-sample at a predetermined interval to produce a uniformly dense point set. The density requirements of the surface reconstruction algorithm determine the re-sampling interval. Notably, the method used in the journal [34] shows a 0.5 mm error in the OUS's middle and chest wall regions. However, the error in the nipple region is approximately 1 mm, which is greater than the minimum error required for this project. Because the nipple region's OUS curvature increases, the higher error in that region is most likely because of the large angle between the laser beam and the vector normal to the surface. A

new step is added to the base method to generate a more accurate point set before fitting and re-sampling the data. The new step uses the initial scanned point set to determine the normal surface orientation and creates new scanning positions that orient the laser beam perpendicular to the OUS surface. Finally, the OUS surface is re-scanned with the new positions and orientations.

Section 6.2 goes into greater detail about the scanning system. The section contains information about each stage of the implementation. Section 6.3 contains information about the entire system's power, network, and GUI. The evaluation of the method includes evaluating each part of the system as well as the entire system performance.

4.1 Implementation Tools

4.1.1 Hardware Tools - *Aref & Ingrid*

Single-Arm YuMi - *Ingrid*
The robot used in this project is provided by ABB and will be a Single-Arm YuMi (SAY). It is currently one of ABB's most agile and compact collaborative robots [35]. In contrast to typical industrial robots that need to be protected with various safety measures, collaborative robots, also known as cobots, are safe to use when in contact with humans. This makes SAY suitable for this application. The SAY has seven different joints, as opposed to the six different joints found in regular industrial robots made by ABB. The six joints give the robot freedom to move to all coordinates within reach with different orientations, but the seventh joint allows the robot to move to all the same coordinates, but also with different joint configurations. The seventh joint provides redundancy, resulting in freedom to perform more complex movements than a six-joint robot. This makes it easier to avoid contact with the environment. The arm has a reach of approximately 56 cm, mimicking a human arm, with pose repeatability of 0.02 mm [35].

Arduino UNO - *Aref*
The Arduino Uno is an open-source microcontroller board created by Arduino. cc and first made available in 2010. It is based on the Microchip ATmega328P microprocessor. A variety of expansion boards (shields) and other circuits can be interfaced with the board's digital and analogue input/output (I/O) pins. The board features six analogue I/O pins, and 14 digital I/O pins, six of which may output PWM. This microcontroller will be utilised to operate and communicate with the sender's robotic arm and the laser sensor[36].

OptoNCDT 1302-200 Laser Distance Sensor - *Aref*
optoNCDT 1302-200 form Micro-epsilon is a highly accurate distance laser sensor which can measure the range of 200 mm starting from 60 mm far from the sensor lens. The sensor employs the optical triangulation concept, which involves projecting a visible, modulated point of light onto the intended surface. A visible/red semiconductor laser with a wavelength of 670 nm powers the sensors. A steady beam is emitted by the laser. One mW is the most remarkable optical power. The sensors fall under Laser Class 2 classification and weigh approximately 83 g. The sensor can communicate with a controller in analogue and digital(RS422) modes. In this project, this laser sensor will be used because it was tested in previous iterations, and it is available to use. Therefore, the RS422 communication protocol has been used [37].

Cables and connectors - *Aref*
The types of cables and connections used in this project will depend on its requirements and behavior, but as a general rule, it will require a few Ethernet cables for the robot arm and the communication with the microcontrollers, as well as a few cables and connectors for the power connection. It's also important to take into account the microwave antenna cables and the laser sensor cable(M12-12pins).

Computer - *Aref*
The computer is the central control device that will link all of the parts—sensors, microcontroller, and robot hand—together. The computer ought to meet the needs that each application and system have.

Power supply - *Aref*
Power is required for the robot arm, sensors, microcontrollers, and computer. The laser sensor requires a 24 volt power source, the microcontroller a 5 volt supply, and some of the components a 230 volt supply. To properly power all components, we therefore require some voltage regulators and step-down circuits.

3D reference models - *Aref*
The OUS has to be created using average human breast size as a guide. Two models will be created, one of which is a symmetrical design with a spherical and cylindrical shape. In addition, a different one will be created to mimic the intricate contour of a real woman's breast. These models ought to be fastened to a stationary holder in the robot arm's work area.

Original Prusa I3 MK3 3D-printer - *Aref*
Prusa I3 MK3 3D-printer has been designed by Josef Prusa. It is the replacement for the well-liked Prusa I3 MK2S. A removable build plate, an auto-leveling sensor, and a Bowden extruder are a few of the printer features. This printer's workspace measures 25 x 21 x 21 cm and can only use 1.75 mm PLA/ABS filament [38]. The initial prototypes required for creating the project's hardware will be produced with this printer.

4.1.2 Software tools - *Jonathan, Aref & Ingrid*

MATLAB - *Jonathan*
For easy implementation of the scanning protocol, surface reconstruction, computation of trajectory path points and plotting of the result, MATLAB [39] was the chosen method. MATLAB has an intuitive programming style and supports state-of-the-art algorithms using toolboxes. To minimize the time needed before the implantation could start, a familiar language was chosen for the software team.

RobotStudio & RAPID - *Ingrid*
RobotStudio is a program developed by ABB that features offline programming, modeling, and simulation of robot cells. RAPID is a high-level programming language that is used as the basic programming language in RobotStudio, and is used to control ABB industrial robots. All robot movement will be programmed using RAPID, and will be tested in a simulation using RobotStudio.

SOLIDWORKS - *Aref*
SOLIDWORKS CAD is a mechanical design automation tool, enables designers to swiftly sketch out concepts, play with features and measurements, and create models and thorough drawings. When it comes to motion simulation and aerodynamics simulations, it is a highly useful tool [40]. This tool will be used in this project to design the end-effector, which contains all of the sensors and measuring components.

Prusa slicer - *Aref*
A software program called Prusa Slicer transforms SOLIDWORKS STL files into gcode files that the 3D printer can understand. In addition to being open-source and simple to use, it is the original software for the Prusa 3D printer[38].

Visual Studio code - *Aref*
Visual Studio code is a Linux, Windows, and macOS-compatible standalone source code editor.This compiler can be used by web developers and it supports almost any programming language [41]. This tool will be used to program the microcontroller.

5. Ethical and Societal Considerations - *Amanda*

While the system might have a human subject as the object under study in the far future, no human subjects are used in this stage of the research. Therefore, little to no consideration has to be made to ensure confidentiality and/or comfort of future users. However, because the objects under study in this research are modeled to generalize the shape of a human breast, this could be sensitive or offensive to some people. Therefore, efforts have been made to keep the research professional and appropriate by differentiating the objects from that of real breasts by avoiding certain colorations and too high levels of details, without affecting the results of the project. The results of this project will help advance the research of the microwave antennas and accompanying systems, which in turn could improve cancer detection and monitoring in the future. The research is not aimed at replacing current cancer detection methods. Instead, the results of the research will work as a compliment to existing methods, possibly offering a more comfortable and less harmful method for detecting breast cancer. If the results of the research would be developed into a commercially available product in the future, this could result in more women being willing to get themselves screened for cancer, increasing the chances for early detection of cancerous tissue.

Part of this project is to investigate the suitability of the single-armed YuMi collaborative robot for microwave imaging. Because this robotic arm was lent to the project for free from ABB, there is a risk that there might be some biased when judging the results of the robot, as future collaboration between the research and ABB might be desired. Because of this, extra consideration must be taken to judge and deliver the results honestly, so as not to let the biased affect them.

6. Implementation

Each phase of the implementation process will be covered in detail and explained in this section.

6.1 Positioning Systems

This section will discuss all of the implementation processes that the positioning system presents, including each system component.

6.1.1 The Sender Robot - *Emanuel*

In this section, the hardware and software for the sender robot are described. All parts that were 3D printed, was made of PLA.

The actuator of the Sender Robot is one stepper motor. The stepper motor is rotating a threaded rod, that causes a cart to move up and down. An end effector is connected to this cart by linear rods. By moving the cart, the end effector moves. See Figure 4. The following paragraphs will describe the Sender Robot in more detail.

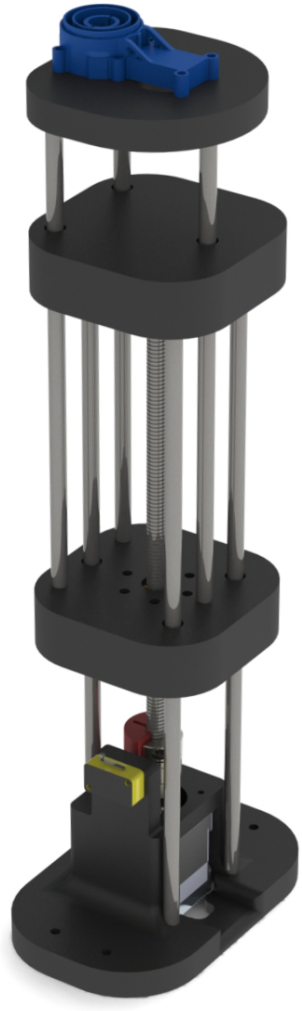


Figure 4: The Sender Robot.

The Base Figure 5 display a render of the base of the Sender Robot. The 3D printed base is wide enough such that the Sender Robot can stand upright without support. It also has holes such that

it can be securely mounted using screws. It has a mount for a NEMA 17 stepper motor. Here, a Wantai 42BYGHM809 was used. It has a mount for the micro switch (XGK2-88-S20Z1) and three linear rods (Bosch Rexroth R100000800,400mm). Three linear rods were chosen because it is the minimum amount of points needed to fully define a plane. Fewer rods will cause rotational forces that might cause bending, and more rods will over-define the plane. The stepper and micro-switch were mounted using M3 nuts and bolts. The linear rods are held in place by friction.

Figure 5: The base of the Sender Robot.

The Cart In Figure 6, the 3D printed cart can be seen as the black part at the bottom. Everything related to the base, the stabiliser and the bottom linear rods are hidden in the render to increase visibility.

The purpose of the cart is to mount the threaded rod nut to the upper linear rods. The cart contains three linear bearings (RJUM-01-08), the threaded rod nut (RS PRO 862-5326) and mounts for the upper linear rods. The threaded rod nut is mounted using M5 nuts and bolts. The linear bearings and linear rods are held in place by friction.

(a) The cart and the end-effector of the Sender Robot as seen from above.

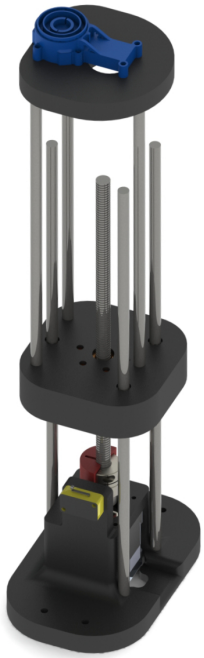
(b) The cart and the end-effector of the Sender Robot as seen from below.

Figure 6: The cart of the Sender Robot.

The Stabiliser Figure 7a display the stabiliser seen as the bottom black part. The stabiliser is 3D printed and its purpose is to stabilise all linear rods, as well as protect the linear rods and threaded rod. The stabiliser contains three linear bearings (RJUM-01-08) as well as mounts for the bottom linear rods. It has a large hole for the threaded rod, but it never has contact with the threaded rod. Both the bearings and linear rods are mounted by friction.

The Sender Robot could function without the stabiliser, therefore it has no vital function. In Figure 7b, the Sender Robot can be seen without the stabiliser. In this configuration, the bottom linear rods are only mounted at the base. The top linear rods that connect to the end-effector could more easily bend sideways, since the linear rods would only be stabilised from lateral forces by the linear rod mounts. Therefore, the stabiliser help with stabilising the Sender Robot.

(a) The stabiliser and the end-effector of the Sender Robot.



(b) The Sender Robot without the stabiliser.

Figure 7: The stabiliser of the Sender Robot.

The End-Effector Figure 8 displays the end-effector of the Sender Robot, the sender and the top linear rods. Everything else is hidden in the render. The end-effector is 3D printed and has holes to mount the sender antenna. It also has one large hole for the antenna cable, as well as three mounts for the top linear rods. The linear rods are held in place by friction and the sender antenna by M3 nuts and bolts.

(a) The end-effector of the Sender Robot as seen from above.

(b) The end-effector of the Sender Robot as seen from below.

Figure 8: The end-effector of the Sender Robot.

The Coupler Figure 9 display a close up render of the mechanism used to couple the shaft of the stepper motor to the threaded rod. The stepper motor has a shaft diameter of 5mm and the threaded rod has a diameter of 10mm. Because no 5mm/10mm coupling was available for purchase, two coupling hubs were used instead (MJC25-10-A and MJC19-5-A). The two coupling hubs were not made to fit together, therefore a 3D printed part (coloured black) was made that act as an adapter for the coupling hubs. The two coupling hubs and the 3D printed part slides together and are rigid for rotational forces caused by the stepper, but it has a tendency to slide apart. To prevent the coupling mechanism from sliding apart, a coupling cover was 3D printed (the red part).

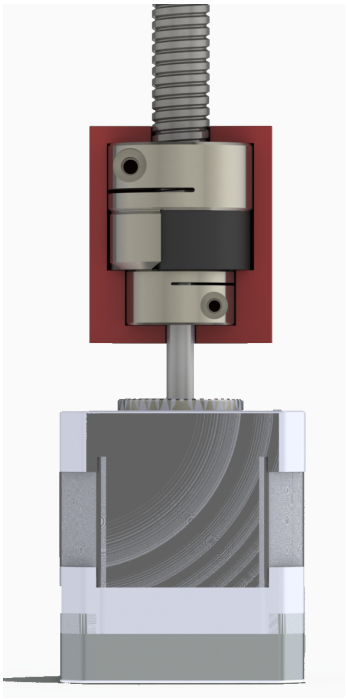


Figure 9: The coupling mechanism of the Sender Robot.

Actuation Figure 10 display an overview of the Sender Robot actuation. It can be seen that the stepper motor rotates a threaded rod, which is connected to the cart through the brass nut. Therefore, when the threaded rod rotates, the carts move up and down. The cart is prevented from rotating because of the bottom linear rods (not shown in Figure 10). Three linear rods are mounted to the top of the cart and are therefore pushed up and down as the cart moves. The three linear rods are also mounted to the end-effector, which causes it to move up and down. Note that the bottom linear rods, base and micro-switch are hidden in the render for increased visibility.

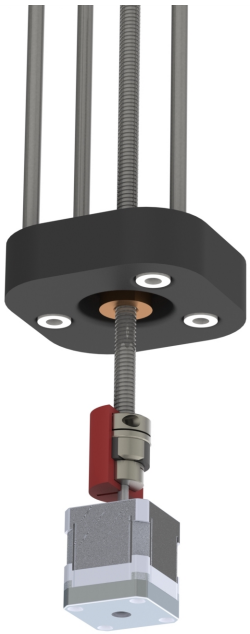


Figure 10: Overview of the actuation of the Sender Robot.

Figure 11 display an overview of the Sender Robot electronics. It uses the micro-switch along with a debouncing circuit to find its zero coordinate. A stepper motor driver (TB6600) is fed with 24V and powers the stepper motor, and has micro-stepping capabilities. The debouncing circuit was made from two resistors and a capacitor. The micro-switch is pulling the debouncing circuit output to ground when pressed. The Arduino used is an Arduino Uno R3, and it has a DFRobot DFR0272 Ethernet shield for Ethernet communication.

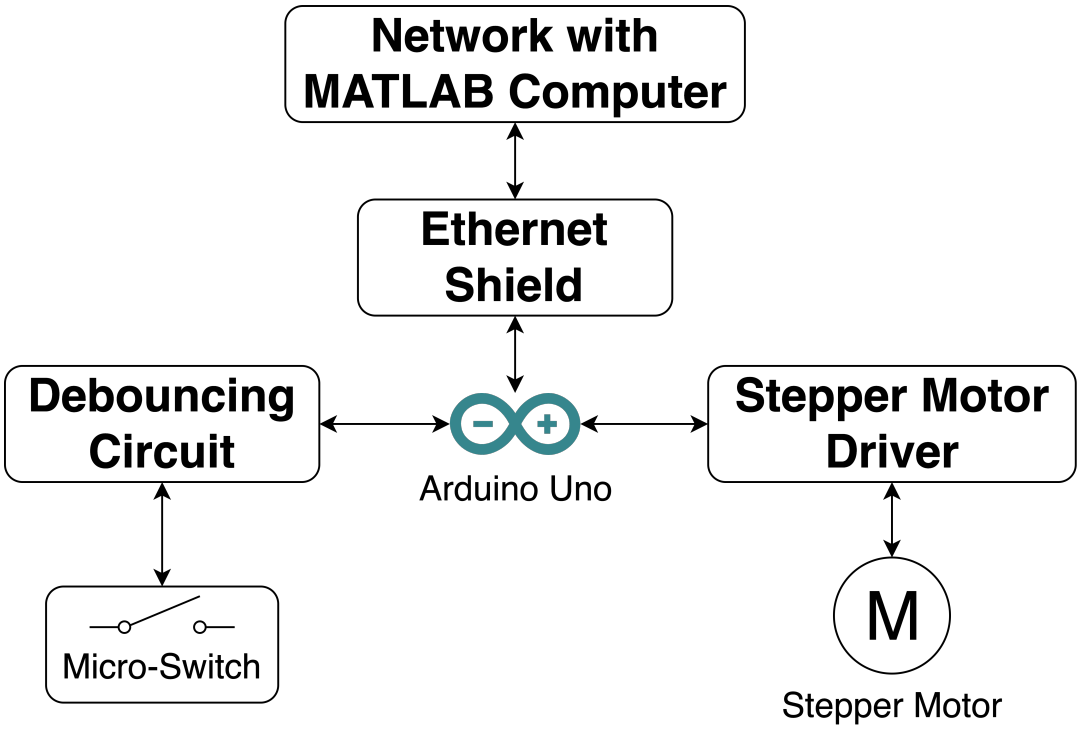


Figure 11: Overview of the electronic design of the Sender Robot.

First Algorithm 1 will initialise everything that needs to be initialised, and also do a calibration of the Sender Robot. It will then listen for incoming messages. An example of an incoming message could be "goTo:100", this means go 100mm above the 0 point. Another example could be "moveUp:6400". This would mean move 6400 steps up. The main function will also return the current height above the 0 point after executing its command. The Ethernet communication is described further in Section 6.3.3.

Algorithm 2 will first move the sender downwards quickly until the micro-switch is triggered. It will then rise approximately 150 steps above the micro-switch trigger point. It will then move down slowly until the micro-switch is triggered. This position is the 0 point.

Algorithm 3 will take an input for how many steps to move. It will then move the sender down that number of steps.

Algorithm 4 will take an input for how many steps to move. It will then move the sender up that number of steps.

Algorithm 5 will take an input for how many millimetres above the 0 point to move to, and then use Algorithm 4 or Algorithm 3 to move to that position. It will also make sure to not move above the highest allowed height, or below the 0 point.

6.1.2 Single-Arm Yumi - *Ingrid*

The ABB robot SAY is used during both the laser scanning and microwave scanning phases, and all code used to control the robot is written in RAPID. During the laser scanning phase, SAY is used to collect data points of where the OUS surface is. SAY moves around the OUS with a laser as its tool, aiming it at the OUS. The position the laser is in and the collected data points will later be used to create a 3D-model of the OUS. During the microwave scanning phase, SAY will move the receiver on the surface of the OUS at the position chosen by the operator. The position of the receiver and the result from the scanning can then be used to map the position of the irregularity.

User Frame The user Frame is a Cartesian coordinate system that defines the workspace relative to the world frame of SAY, and is based on the position of the OUS (see figure 12). Every coordinate used in this application is represented by the user frame. The rotation of each coordinate is based on the frame rotation from the user frame, and is rotated according to the ZYX Euler angles.

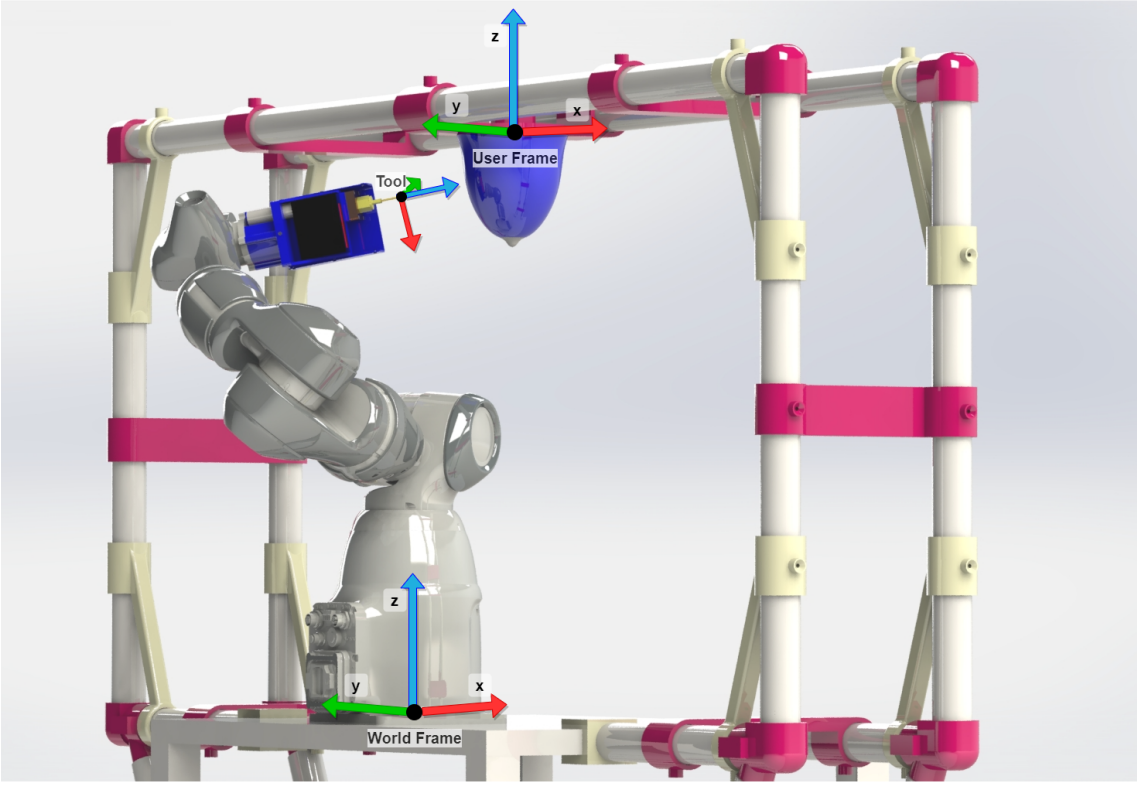


Figure 12: User Frame

Connection To control the robot movements, a TCP/IP connection between MATLAB and the robot is presented. The connection is used to send commands to the robot and obtain information from the robot's back.

MATLAB sends a 3-by-3 matrix (see figure 14a) with different parameters, providing the robot with the information it needs to move to a position. The first three parameters are the x-, y-, and z-coordinates of a point in the User Frame where the robot should place its tool. The next three parameters provide the robot information regarding the rotation of the tool at that position. The third row, with parameters, indicates the robot which tool it should use and the speed at which it should move. The robot has two different tools to choose between, the receiver and the laser (see figure 13), which are used during the two different phases. The speed parameter provides the robot speed in [mm/s] at which it should move between points. The third row also contains a parameter called "extra". This is an additional parameter that has not been used yet.

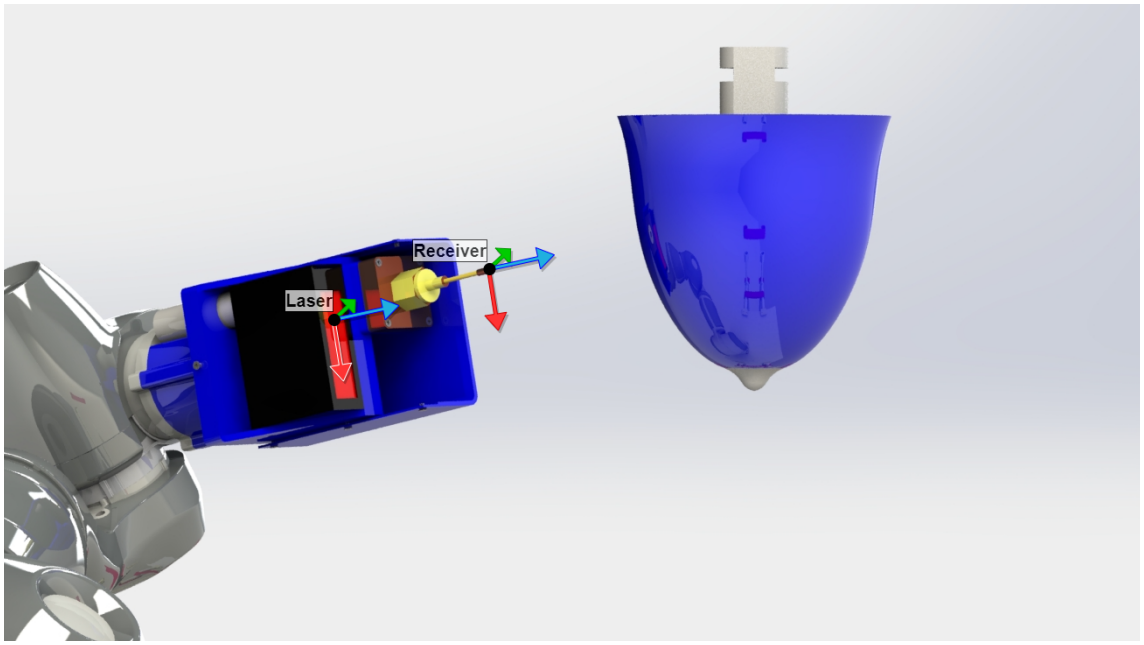


Figure 13: Tool

pos x	pos y	pos z
rot x	rot y	rot z
tool	speed	extra

(a) Parameters sent from MATLAB to robot

pos x	pos y	pos z
rot x	rot y	rot z

(b) Parameters sent from robot to MATLAB

Figure 14: Communication between MATLAB and SAY

The robot will move to the position decided by MATLAB, but small differences can occur. When the robot is stationary in the goal position, the real current position and rotation is sent back to MATLAB via the TCP/IP connection (see figure 14b).

Configuration calculation The robot should be able to reach the different positions without collisions between its arm and the environment or the OUS. One of the advantages of SAY is that it has seven joints. The seventh joint provide the SAY with the possibility of reaching a position with different configurations on its joints. By default, the robot will choose the best configuration for itself, but since the environment has to be taken into consideration, a configuration has to be calculated for the robot at the end position, and for the movement to reach that position.

Every position on the left side of the OUS will have the arm on the left side of the OUS, and every position on the right side will have the arm on the right side. To do this without making big changes to all the joints, the first joint will make the biggest change its configuration. This causes the robot to rotate around the OUS without changing its configuration on the whole robot. This is important because the robot has cables attached to its outside. Depending on how much the tool position is in degrees according to the spherical coordinate system (see figure 15), joint one will change its configuration in reference to that.

Because the OUS is placed in front of the SAY, the distance from the base of the SAY and the tool position in the x-direction (see fig 12 for the x-direction), will have an impact on the angle of the seventh joint. The change in that joint will change the angle of the "elbow" of the robot, and will therefore prevent the arm of the SAY from collision.

Path planning For industrial robots, path planning entails moving the robot tool from point A to point B while preventing collisions. It is important that the robot does not collide with the OUS phantom or environment to reach its destination. All ABB robots can calculate their own trajectory plan from one position to another; however, because the goal is not only to move to the correct position, but also to move to that position without colliding with the OUS or the environment, some extra path planning has to be included.

Added path planning is based on the tool position in degrees in the spherical coordinate system around the OUS (see figure 15) of the tool position in the User Frame. The first step is to calculate the degrees and z-coordinates for the current tool position and goal tool position. If the difference in degrees and/or the z-coordinate is large for the robot to move there in a straight line, a path around the OUS has to be calculated for the movement between does points.

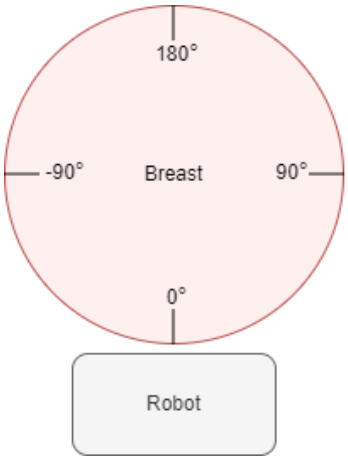


Figure 15: Spherical coordinates

The path around the OUS will be calculated, with a constant distance, in a circle around the OUS. The first position in the circle is at the same angle in the spherical coordinate system as the start position, and the last position is at the same angle as the goal position (See figure16). While the robot tool is circling the OUS, the robot will always point its tool to the center of the OUS.

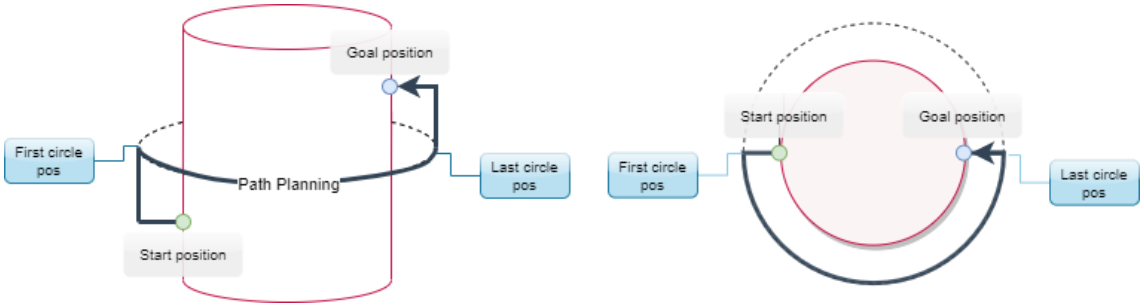


Figure 16: Path Planning

6.1.3 Sender and Receiver Antenna Replicas

Two plastic 3D models have been created to simulate the sender and receiver's proportions following earlier designs of the sender and receiver(see figure 17). These models were developed in order to gauge the robot workspace's usage and prevent damage to the real transmitter and receiver during the testing and scanning phases.

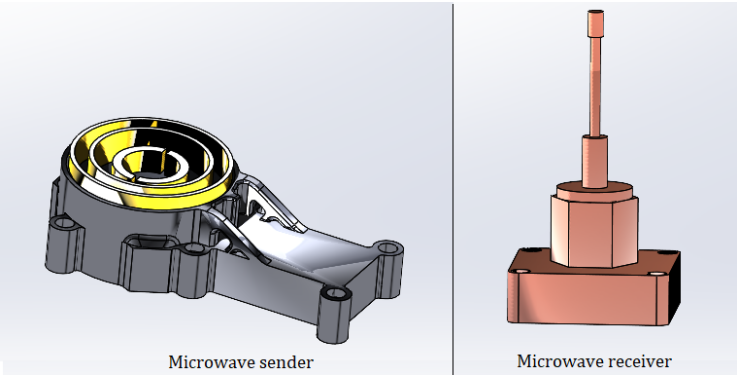


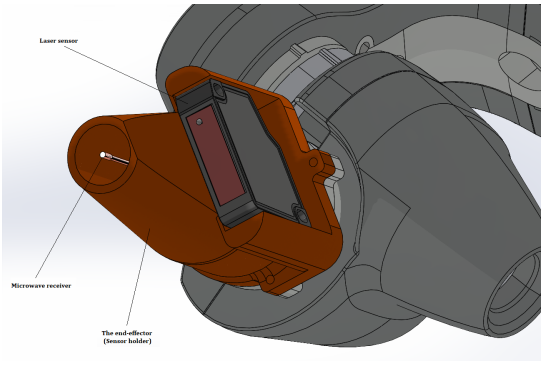
Figure 17: the sender and receiver test design to place them instead of the real components.

6.1.4 The 3D model of the receiver end-effector - *Aref*

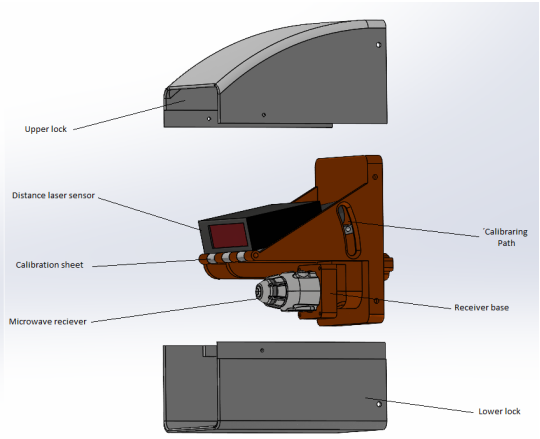
Version 1 The main requirement for this part is to hold the microwave receiver and the distance laser sensor. To prevent the inaccuracy caused by the length of the end-radius effector when it rotates, this section should be as small and as short in length as possible. The distance laser sensor is located in this type on the upper side of the sensor holder. The laser sensor's and the microwave receiver's dimensions and shapes determine how the end-effector is made. The sensor analysis was made, and the conclusion was to employ a Baumer OM30-L0550 distance laser sensor [42]. The investor was then asked for the shape and size of the microwave receiver. The receiver design has been rebuilt to roughly match the size and form of the genuine microwave receiver because the sensor's size and shape were ambiguous and undetermined. In around four hours, the design was completed using the SOLIDWORKS software [40]. Constructing the holder's base first so it matches the mounting section on the robot arm then construct the microwave receiver's home. Last but not least, the functionality of the microwave receiver and the measuring range of the laser sensor were taken into account when designing the distance laser sensor holder. This version wasn't produced via 3D printing. As a cuboid, this design's ultimate measurements are roughly 65x65x138 mm(see figure (a) 18).

Version 2 The entire design had to be rebuilt because the sensor had to be changed. OptoNCDT 1302-200 [37], a new distance laser sensor from MICRO-EPSILON, is used. this sensor has been used in the previous version of this project (previous iteration of this project). The end-effector needs to be larger and wider in order to accommodate the new Distance laser sensor. Additionally, a new feature that enables users to calibrate the laser sensor's angle has been introduced to the end-effector. This angle can be calibrated between 0 degrees and 22 degrees. This causes the design to be heavier and wider. That caused some problems with the YuMi movement that increases the collision between the end-effector and the YuMi robotics arm. In this version, a house for a camera has been placed. The 3D model has been 3D printed using Prusa Orginal [38] with PLA material. but some problems have been faced with the 3D printer. It needed some calibrations, therefore the 3D model was badly produced and easy to destroy. This version was produced via 3D printing. As a cuboid, this design's ultimate measurements are roughly 75x100x135 mm (see figure (b) 18).

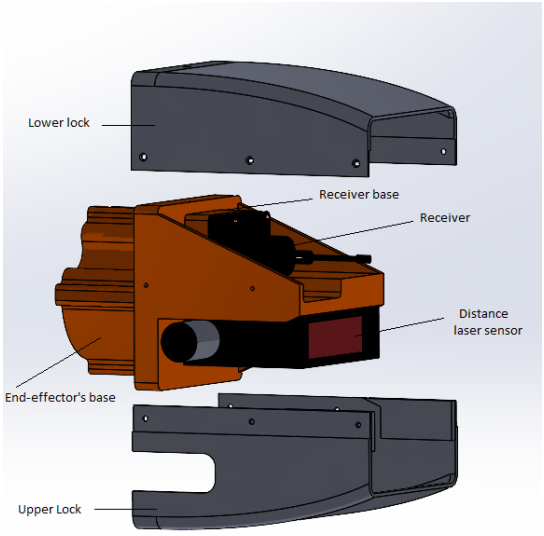
Version 3 Version 2 has some drawbacks, such as the possibility of Link 6's movement being limited if the end effector's base is too small. The breadth of the end-effector, on the other hand, might impose restrictions on the rotation between links 5 and 6. effector's The mechanism (angle calibrator) between the receiver normal line and the distance laser sensor is not necessary for the



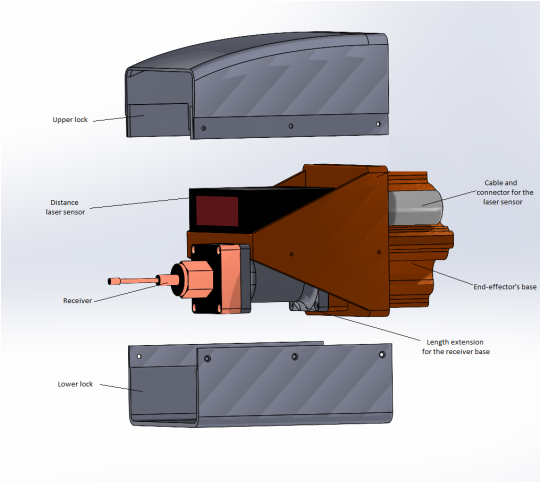
(a) Version 1: The initial design of the End-effector consists of the Microwave receiver, the distance laser sensor, and The end-effector (sensor holder-/housing)



(b) Version 2: In this design, the OPTO NCDT 1302-200 laser sensor has been used with the old version of the microwave receiver. The end-effector has been designed to allow the user to calibrate the sensor angle using the calibration sheet.



(c) Version 3: In this design the OPTO NCDT 1302-200 laser sensor has been used with a 3D model that fulfill the estimated diminutions of the receiver. The end-effector has got longer base and the connector of the laser sensor has placed on the upper side of the laser sensor.



(d) Version 4. In this design, the OPTO NCDT 1302-200 laser sensor has been used with a 3D model that fulfills the estimated diminutions of the receiver. The end-effector has got a more extended base and the connector of the laser sensor has been placed on the tail of the laser sensor. A length extension has been added to the receiver base to put the top of the receiver 60 mm far from the lines of the laser sensor.

Figure 18: the receiver end-effector was being developed during the project's duration.

initial prototype because the pre-scanning method is sufficient. Following system testing, the angle mechanism can be implemented into the design. The base length in this design is 35 mm to prevent contact between the end-effector and link 5 of the YuMi robot arm. The end effector has had a maximum width of 74 mm. The length has gotten 25 mm longer due to the base's larger length. To verify its suitability for this usage and to make attaching to it easier, the 12-pin connector for the sensor has been moved to the top side of the device. This version was produced via 3D printing. As a cuboid, this design's ultimate measurements are roughly 75x75x135 mm(see figure (c) 18).

Version 4 After some testing on Version 3, it was determined that positioning the 12-pin connector so that it would hit the patient's bed was not the greatest idea. Additionally, it will reduce the robot's scanning range and make handling small things too challenging. Another issue is that the sensor's working range is between 60 mm and 200 mm, necessitating positioning the top of the reception antenna inside that range. In this situation, the top of the receiver needs to be 60 mm away from the lines of the distance laser sensor. To do this, a component extension has been created to bring the top of the microwave receiver within the range of the distance laser sensor. To avoid the issue that was reported in the previous version, the laser cable/connector hole has been placed on the rear side of the sensor. The end-locks effectors have undergone some modifications to accommodate the new developments. This version was produced via 3D printing. As a cuboid, this design's ultimate measurements are roughly 75x75x195 mm (see figure (d) 18).

6.2 Scanning system

6.2.1 OUS 3D models - *Aref*

For usage in the test scanning phase, two different OUS 3D models have been created. The OUS shape is fully symmetrical at every point in the first one, which is a symmetric design. The other is an asymmetric OUS that mimics the real-shaped OUS of a medium-sized female in Sweden(see figure 19). As a first test, it will be simpler to use a 3D model that has a symmetric shape. In more complex system tests, the asymmetric OUS will be employed. The dimensions of the symmetric model are 120 mm in diameter and 120 mm in height. The asymmetric type has a diameter of around 150 mm and a height of about 120 mm[43]. Both variants have a similar fixture/Footprint with dimensions of 30 mm in length and 30 mm in diameter.

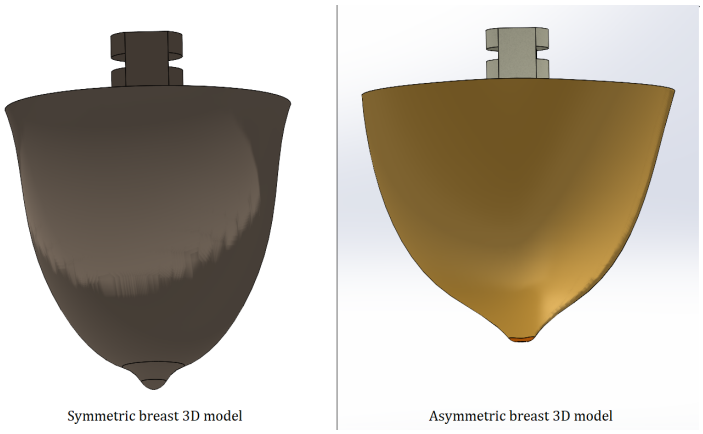


Figure 19: Symmetric and asymmetric OUSs which will be use in the scanning phase.

6.2.2 System Installation Bench(SIB) - *Aref*

The system installation bench was designed using SOLIDWORKS program(See figure **??**). The System Installation Bench (SIB) has been made of plastic to reduce the noise that metal generates and that could affect the microwave system. The OUS 3D model will be hanged using ten PVC pipe pieces, and some PETG/PLA 3D printed parts will be used to join the PVC pipes and gives the structure the stability needs. The parts will be mounted on the robot table using clamps (see figure **??**). The SIB diminutions depends on the Work space of the SAY robot arm. The height of the SIB is 661 mm from the lowest point on the robot to the highest point on the OUS model. The width of the SIB is 1400 mm [35].The reason why the metal should not be used in the microwave measuring space is that the flow of electrons will heat the metal by resistive heating unless it is a superconductor. However, because metallic conductors will mostly reflect microwaves which will affect the microwaves measurement [44].

To build the SIB, a 3D printed parts was printed using a Prusa MK3s[38]. Some parts have been separated into sub-parts due to the limited printing area in order to fit them there. Many issues

were encountered during the printing process that increased the delivery time and will be covered in more detail in the following sections. The printing procedure took three weeks to complete rather than one week as expected. A standard saw and a file were used to manually cut the PVC pipes at the same time. Due to human considerations, the cutting procedure was too complicated to be accurate, yet it was as precise as possible. Around 1 mm worth of cutting inaccuracy was estimated.

Some instability problems have been discovered after the SIB was constructed and tested. The SIB vertical pipes that were attached to the robot table were curved downward. In order to hold the SIB from the bottom corners, four side supports have been created as shown in figure **??** .

6.2.3 The Distance Measurement System - *Emanuel*

In this section, the hardware and software for the Distance Measurement System is described.

The laser sensor is mounted in the end-effector of the receiver robot. This is further described in Section 6.1.4.

Figure 20 display an overview of the Distance Measurement System electronics. The Distance Measurement System uses a laser sensor (Micro-Epsilon optoNCDT 1302) to read the distance. It communicates over Serial RS422 to the Arduino Uno R3 through a Serial shield (DFRobot DFR0259). The Arduino Uno also has an Ethernet shield (DFRobot DFR0272) for the Ethernet communication.

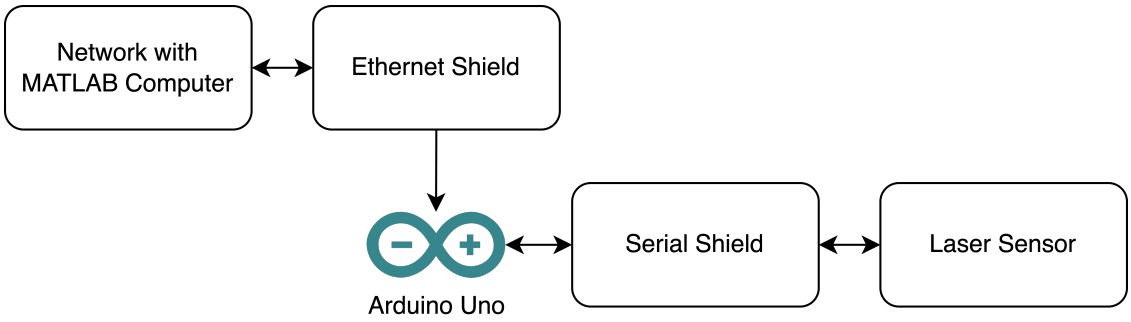


Figure 20: Overview of the electronic design of the Distance Measurement System.

Laser Communication The laser communicates over the serial protocol RS422. It will send 750 measurements per second over serial. The sending rate could be controlled by setting the baud rate. The sensor sends values between 0 and 16383, where values between 16370 and 16383 are used for error codes. The other values need to be converted into millimetres by the formula $(((measurement \cdot \frac{1.02}{4096}) - 0.01) \cdot 200) + 60$.

Because this sensor has been used for other projects in the past, the internal configuration of the sensor had been changed and did not suit this project. The biggest issue was that the range had been limited to 12-18cm. One of the first things tried was a factory reset procedure described in the sensor manual [45], by pressing the select button in a special sequence. The sequence was tried multiple times, but it did not work. Therefore, the next step would be to connect the sensor to the computer and try to configure the sensor that way instead. Paragraph 'Laser and Computer Communication' describes how the sensor could communicate with the computer and what attempts were made to configure it using a computer. Paragraph 'Laser and Arduino Communication' describes how the sensor communicates with the Arduino Uno after the sensor had been reset.

Laser and Computer Communication To edit the internal configuration of the sensor, an USB to Serial adapter (Exsys EX-1309-T) was used to connect the sensor to a computer. The dip-switches of the adapter were set for RS422 communication. A program called sensorTOOL V1.9.0 was downloaded [46]. This program could communicate with the sensor and read some

representation of a distance, but it was off by many centimetres. Unfortunately, this sensor could not be configured using this program.

The next attempt was to push commands to the sensor over serial. The sensor manual mentions that it is possible to send the command "GET_INFO" and "0x20490002" to read the configuration of the sensor. These commands were sent to the sensor both in ASCII mode and HEX mode using PuTTY [47] and Realterm [48]. All attempts were unsuccessful.

There was one last thing to try. The manual also mentions a C++ library called MEDAQLib [49]. This library is used to build advanced applications for the sensor. Therefore, a C++ development environment was setup, a hello world program was tried to make sure everything is compiling and executing properly and then the different example codes for MEDAQLib as provided by the sensor manual was tried but all of them returned a sensor communication error.

It had been noted that the raw data returned by the sensor was looking strange. Both when reading with PuTTY and Realterm. For example, if a value of "839" was expected, the sensor might return "8", "39", "98" or some other value that comprised the correct digits, but not the correct value. At this point, the sensor seemed broken, but because of delivery issues with other products, a new sensor would probably not arrive in time. Therefore, one last attempt was made at resetting the sensor by the select button sequence, repeatedly for many minutes, when suddenly the LED flashes to indicate a successful reset of the sensor. After this, the sensor still sent the wrong digits, but it had the full range of 6-26cm.

Laser and Arduino Communication There are multiple different function available for the Arduino to read serial data. Below are the functions tried:

- Serial.parseInt()
  This function reads the next valid integer in the incoming serial port and returns a long. This long is then converted to a char array using sprintf and sent to the computer. At a constant distance of about 10cm, the returned value is changing rapidly between 8 and 893892. The true value should be 893. Because both 8 and 890 are valid outputs, it is hard to know which readings are true, and which are incorrect.

- Serial.read()
  This function reads one char. This means that the range is 0 to 9, which is not enough for this application.

- Serial.readString()
  This function freezes the Arduino and it never returns.

- Serial.readBytes() reading 6 chars
  Reads a fixed number of characters from the serial port. A value of 16376 + terminator should fit in an array of 6 chars. When reading 6 chars using this function at a fixed distance of about 10cm, it returns various things that are hard to read. For example '39\r 8', '839 8', '39 \r39\r' or ' 8 8'. It would be impossible to make any use of this.

- Serial.readBytes() reading 20 chars
  Because the sensor returns a lot of trash data, if we try to read a lot more than we actually need, for example this is the result: ' \r 829\r 829\r 829' or '29\r829\r 829\r 829\r 8'. We still get a lot of bad readings, but at least each reading has the true number somewhere between two \r.

As Serial.readBytes() seemed to work best, it was tried with all reading lengths between 6 and 20 chars. 16 chars worked best because fewer chars made the reading not have the true value between two \r sometimes, and more chars sometimes made the true reading appear twice. This does not matter, but there is no point in reading more than necessary. This made it possible to find the true value.

Arduino Software This paragraph describes the software that is run on the Arduino, except for the Ethernet communication. That is described in Section 6.3.3.

To clean the data, the first \r was found. This \r and everything before it is replaced with spaces. Then the next \r was found and it, and everything after it is replaced with spaces. Now the result is for example, ‘ 812 ‘ or ‘ 812 ‘. The important thing was that each reading has the true value. To test this, 500 measurements were done. Every single one of them contained the true reading surrounded by a various amount of spaces. After this, a for loop was written that removes all spaces and only keeps the numbers. This was then tested on 3000 readings at various distances covering the whole range of the sensor and every single reading was correct.

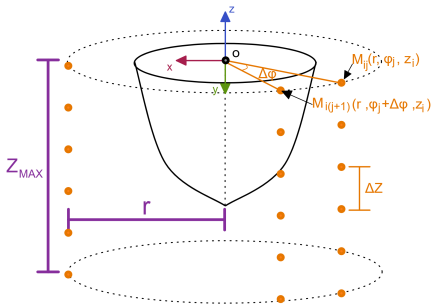
The system works by the computer sending an UDP package to the Arduino with the message “Get” on port 1000 and the Arduino returns the current distance from the sensor to the computer. Algorithm 6 describes how this was done with pseudo-code and how to listen for incoming messages. The only valid incoming message is “Get”. That tells the Arduino to take a reading from the laser sensor. Note that the reading sent from the sensor is not in millimetres.

Computer Software The MATLAB computer is running a script that communicates with the Arduino, does error handling, filtering and converts the values into millimetres. The script was written in Python 3.10 [50]. Algorithm 7 describes the Python script. The script is called from MATLAB, and the Python script returns to MATLAB. The algorithm works by receiving 8 readings from the sensor with a delay of 0.01 seconds in between. If all 8 reading are the same, then the sensor has stabilised. This is needed because the value of the sensor oscillates after movements, therefore, this acts as a filter. It then returns the error, or the value, into millimetres. This process can be repeated at most 20 times before it returns. This is because if the sensor is looking at a bright source, for example, a light, it cannot determine if there is an object or an error, therefore the value returned by the laser oscillates over a wide range and could create an infinite loop. The 20*8 reading limit prevents that from happening.

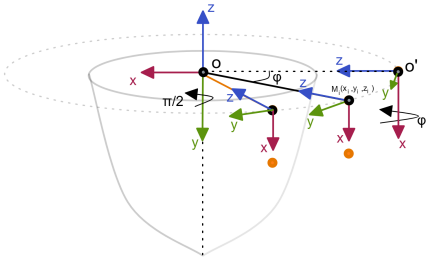
6.2.4 Scanning protocol-interpolation - *Victor*

The laser is placed in specific positions and orientations from the OUS to measure a point on the OUS surface. The laser measurement position is a 6-dimensional vector. The first three values represent the laser position in Cartesian space (x_i, y_i, z_i), and the last three represent the laser orientation in that position as Euler angles. The SAY put the laser in scanning positions following a cylindrical shape around the OUS and the laser beam oriented to the centre. The user defines the cylindrical boundaries of the scanning as a maximum height Z_{max} and a radius r. The points are distributed uniformly on the cylindrical boundaries, and the distribution is defined in cylindrical coordinate by a difference in height ΔZ and azimuth angle $\Delta\phi$, as shown in figure 21a. A homogeneous transformation (HT) matrix consisting of a translation and a rotation defines the position and orientation of a measuring point M_i. The HT matrix represent the transformation of the base coordinate system o to the laser coordinate system which is considered placed at the beginning of the laser beam pointing the Z-axis in the laser beam direction. The translation component is the coordinate of $OM_i = (x_i, y_i, z_i)$, where o is the User Frame's centre in the middle of the OUS hole.(x_i, y_i, z_i) are calculated by converting the cylindrical coordinates $M_{ij} = (r, \phi_j, z_i)$ to Cartesian coordinates. The orientation part of the HT matrix is made of two rotation matrices. First is a 90° rotation around the y-axis, followed by a rotation around the x-axis by the point M_{ij} azimuth angle ϕ. Figure 21b shows a geometrical representation of the transformation.The Robotics system toolbox in MATLAB [7] is used to generate HT and rotation matrices. It is also used to convert the HT matrix to a 6 Dimensional position vector, which is then sent to the SAY.

A point set from the surface of the OUS $P_i = (z_i, y_i, z_i)$ is calculated using the SAY position and sensor values. Multiple curves are fitted to the data using the Curve fitting toolbox in MATLAB [51] and re-sampled at a uniform interval to create uniformly distributed point set. The method employs three one-dimensional curve functions to fit the three-dimensional data. The first two functions, $Q(x)$ and $R(x)$, respectively, fit the relationship of z_i with x_i and y_i, so that



(a) The figure shows the scanning region bounded by Z_{max} and r and the scanning points distribution defined by ΔZ and $\Delta\phi$.

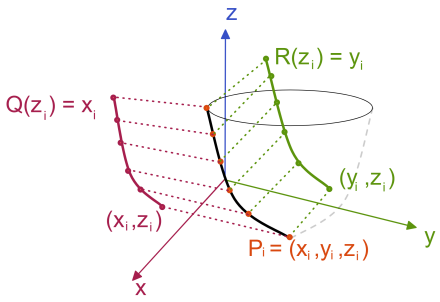


(b) The figure shows the orientation and translation from base coordinate o to M_i. The rotation from o to M_i include 2 rotations, first around y_0 by $\frac{\pi}{2}$ Then a rotation around $x_{o'}$ by the angle ϕ coresponding to M_i

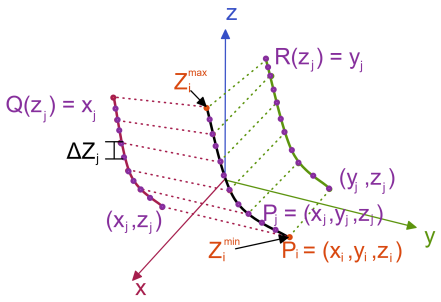
Figure 21: The figures give a geometrical representation for the scanning points and transformation from the base coordinate to the scanning point coordinate.

$Q(Zi) = x_i$ and $R(Zi) = y_i$. A smooth spline curve is used to approximate $Q(Zi)$ and $R(Zi)$ and a graphical representation is shown in figure 22a. Two curves $Q(x)$ and $R(x)$ are approximated for each azimuth angle of the point set. Then a new equidistant vector Z_j is created To resample the spline curves functions. The size and maximum length of z_j are based on the desired height interval δZ_j and the lowest Z_i value of a point P_i in the original point set. The Z coordinate of the new resampled surface point $P_j = (x_j, y_j, z_j)$ is represented by z_j. $P(z_j)$ and$F(z_j)$ are used to calculate the values of X_j and y_j, respectively. The graphical representation of the splines resampling is shows in figure 22b. Resampling the splines curves produces a new point set P_j, with equidistant heights and the same z_j values across all azimuth angles. For each height Z_j, a Fourier series is used to fit the data of x_j and y_j. First, the coordinates (x_j, y_j) are converted to cylindrical coordinates (ρ_j, θ_j). Next, a Fourier series is fitted to the periodic relationship between ρ_j and θ_j using the curve fitting app in MATLAB. A graphical representation of the fourie curves is shown in figure 22c. Then the Fourie series are resampled using an equidistant vector of azimuth angle $\Delta(\theta_k)$ between 0° and 360°. The resulting polar coordinate points (ρ_k, θ_k, z_j) from $F(\theta_k) = \rho_k$ are converted back to cartesian coordinates. The graphical interpetation of the resampling of the Fourier serie is shown in figure 22d. Finally, this method creates a uniformly distributed point set whose density and uniformity can be controlled using the azimuth interval $\Delta(\theta_k)$ and the height interval Δz_j.

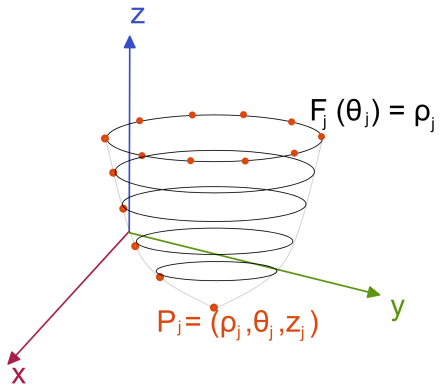
This step is added to the original method to reduce the error caused by the high angle between the normal vector to the surface and the laser beam. Given an initial point set P_i, this method finds the normal to the surface N_i at P_i. Then it generates new scanning points M_i' that place the laser at a distance d from the surface point P_i and orient the laser beam parallel to the normal N_i. The initial point set $P_i = (x_i, y_i, z_i)$ that represents the coordinate of a measured point P_i on the OUS surface is created using the first cylindrical scan mentioned above. The unity normal vectors to the surface N_i at each P_i are calculated using a set of close neighbours [52]. The new 6 D laser scanning positions are derived by converting an HT matrix to a 6-dimensional vector. The first three values of the vector represent the position of the new measuring point M_i', and the last three values represent the orientation given as Euler angles. The vector $\overrightarrow{OM_i'}$ represents the translation part of the HT matrix, where M_i' is the new scanning point for the surface point P_i. The coordinates of $\overrightarrow{OM_i'}$ are computed by $\overrightarrow{OP_i} + d.\overrightarrow{N_i} = \overrightarrow{OM_i'}$. The rotation part of the HT matrix represents the rotation that orients the basis vector Z_o into the orientation of N_i, as the laser beam is considered to be pointing in the Z direction. The rotation is calculated by the angle axis representation and converted to a rotation matrix. $AX = \overrightarrow{Z} \times \overrightarrow{N_i}$ gives the axis of rotation (AX) that is normal to both the Z axis and the surface normal N_i. Then, the angle ang between the Z axis and the surface normal is calculated using $ang = acos(\overrightarrow{Z} \cdot \overrightarrow{N_i})$. The robotics system toolbox in MATLAB is used to implement the HT and conversion between representations[7]. The OUS is



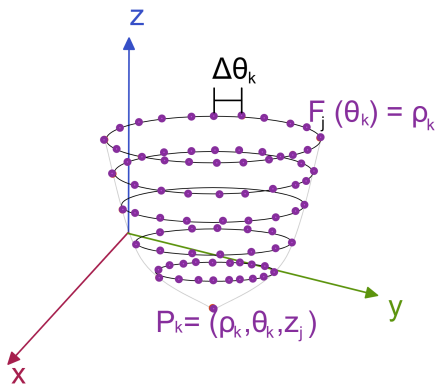
(a) The figure shows the smooth splines fitted to the point set of the scanned data



(b) The figure shows the resampling of the smooth spline by an interval of ΔZ_j



(c) The figure shows the Fourie series fitted to each height of the resampled splines



(d) The figure shows the resampling of the Fourie series by an interval of

Figure 22: The graphical representation of the curve fitting and resampling method.

re-scanned using the new scanning positions and orientations and then uniformly re-sampled using smooth splines and Fourier series.

6.2.5 Surface reconstruction - *Jonathan*

After comparison, as stated in the background 2., and looking at the state-of-the art solutions of similar systems [31]. The chosen algorithm for the method of 3D-reconstruction is PowerCrust. This is due to the algorithm's demonstrated ability to produce a surface that resembles the surface that was discretely sampled using a singular laser, being the same method path as chosen by the group. The method by which PowerCrust works can be seen in its components in 23. The algorithm employs Voronoi diagrams in conjunction with polar balls and Delaunay triangulation to generate an estimated surface.

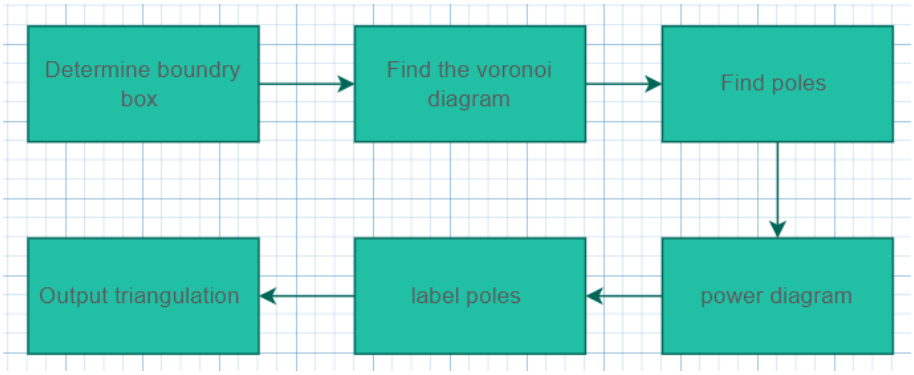


Figure 23: Images that depict the entire powercrust algorithm. Beginning with the containment of the point cloud by a boundary box, then continuing with the Voronoi diagram. When the two previous steps have been completed, the algorithm continues with finding the poles and then the power diagram. Lastly the poles is labelled then a surface is created.

The algorithm starts by taking in a point cloud in 2D or 3D in the form of a nxm matrix; where n is the number of samples and m is the dimension number. From the matrix, a Voronoi diagram is created. To avoid creating infinitely large Voronoi cells, which would result in some vertices being produced far away from the object in 3D space. As a result of the following, the model will have a spiky appearance in the end. The algorithm creates a boundary box around the point-cloud to minimize the problem as much as possible. The boundary box is created by taking the maximum sample in every dimension and multiplying these coordinates by a predefined factor of 5 (this gave the best result). This will result in a rectangular box surrounding the entire point cloud. When the boundary points are determined, the Voronoi diagram is computed using Matlab's built-in function VoronoiDiagram. The function outputs the vertices and which cell is bounded by which specific vertices, as can be seen in 24.

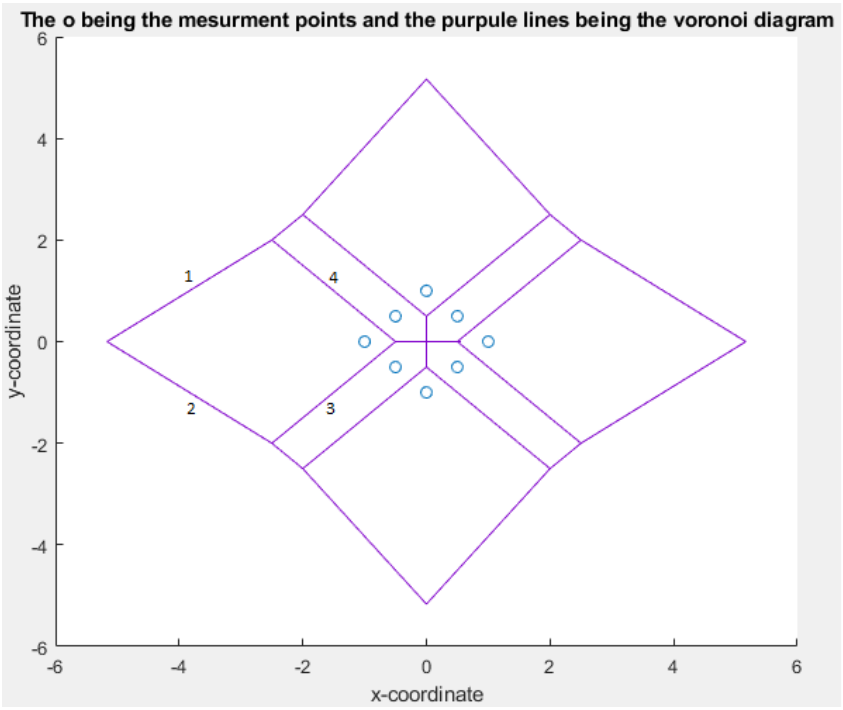


Figure 24: A graph depicting a 2D point cloud plotted as blue circles purple line representing the Voronoi diagram, the first cell (furthest to the left) the vertices index is also stated.

After the Voronoi diagram has been determined. Every Voronoi vertex is referred to as a pole.

By iterating through all cells and selecting the sample point that is in that specific cell. Two poles are found for each sample in the cloud. The pole that is farthest away from the sample is saved as pole number one for this sample. When the distance to the first pole has been calculated, the search for the second pole is conducted. By using the distance to the first pole, the algorithm checks for the pole that has a negative dot product to find the second pole of that sample point. When all cells in the point cloud have been checked, the array of poles is passed through a function to generate a power diagram. This is done by computing another Voronoi diagram, but this time using the poles as the input. The input is given a weighted region around each pole. A region is a ball with a radius corresponding to the distance from the pole to the closest sample in the sample cloud.

When the power diagram has been determined, the algorithm expands the balls in the center of the poles, creating balls on both the inside and outside of the point cloud (as there are vertices on the outside of the point cloud). To locate the point cloud's crust, a labeling process was carried out in which the poles were labeled as being on the inside or outside. If the pole is inside the point cloud, the pole is given a flag of 1, and if it is on outside the cloud, a flag of -1 is set. When the code has iterated through all the poles, the construction of the crust is conducted. Being the triangulation of the poles on the inside of the point-cloud.

6.3 Other Systems

6.3.1 Network - *Emanuel*

Figure 25 display an overview of the network architecture used to connect all devices. The MATLAB computer is running the software that controls the entire system. All other devices receive commands from the MATLAB computer and acts accordingly. Because the IP addressing is done by network equipment owned by MDU, we could not choose the IP range, and we also shared the network with other students. Therefore, we had to pick from the few IPs that were available. All our IPs are static.

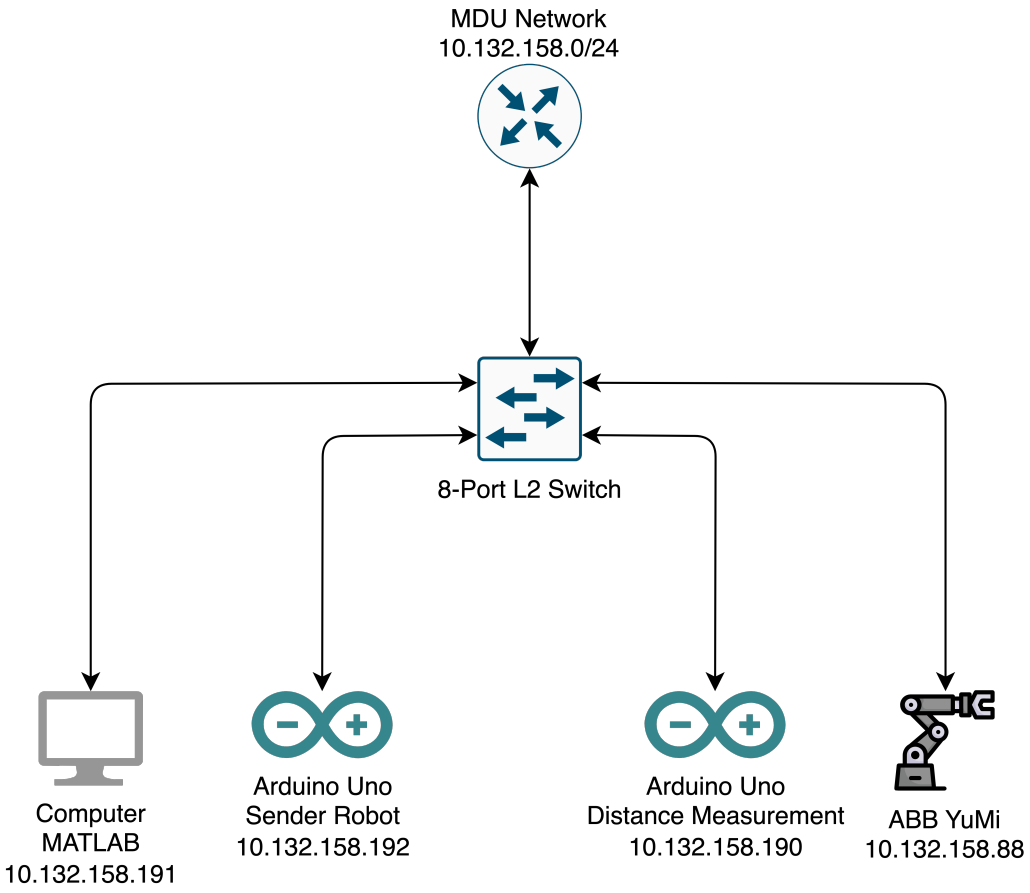


Figure 25: Overview of the network architecture.

6.3.2 Power Supply- *Emanuel*

This section only addresses the power supplies used for powering everything related to the Sender Robot (sec. 6.1.1) and the Distance Measurement System (sec. 6.2.3).

An overview of the power system described can be seen in Figure 26. The system is powered using one 230V AC to 24V DC power supply (Mean Well LRS-150-24). The laser sensor and stepper motor are powered using 24V while the rest of the system is powered using 5V provided by a step down circuit. The functionality of the step down circuit is provided by a switched DC/DC regulator (MP-K78L05-1000R3). The circuit has an 8 connection screw terminal block used for inputs and outputs.

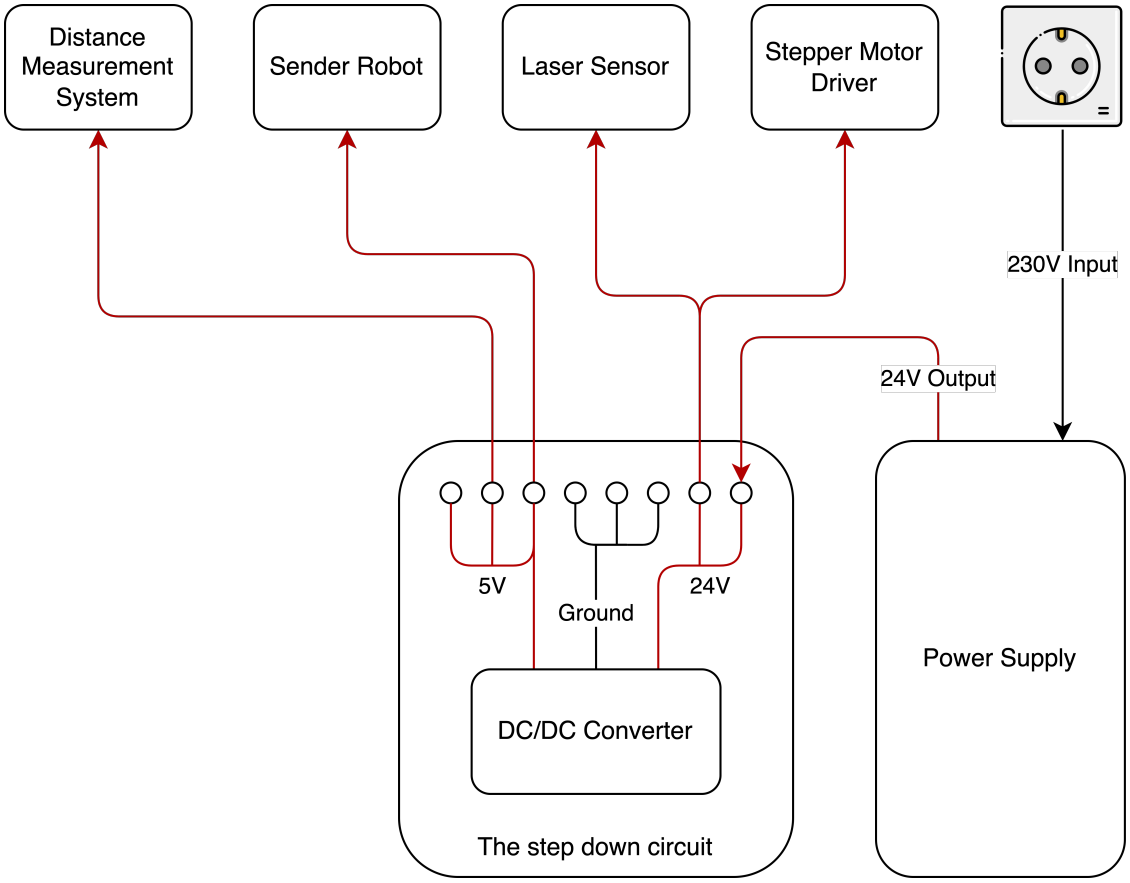


Figure 26: An overview of how the different components are powered. Note that no ground cables are connected to aid with readability.

6.3.3 Arduino Ethernet Communication - *Emanuel*

A library called Ethernet_Generic[53] was used to interface with the Ethernet shield. Algorithm 8 describes how this library was used to send and receive Ethernet packages.

6.3.4 GUI Layout - *Amanda*

The Graphical user interface (GUI) is designed to ease the interaction between human and robot by providing multiple different options of robot control aimed at laser scanning and microwave scanning, replacing the standard robot GUI. The layout is designed to be intuitive, easy to use and easy to navigate. The GUI is made in the MATLAB App Designer. The GUI layout consists of three main panels; the automatic control panel, the visual representation panel and the manual control panel, seen in figure 27. Additionally, the GUI contain safety features such as a emergency stop button and a button which drives the robot away from the object. The GUI also features a log dialog window, where updates of the processes and program execution and error descriptions are given.

The automatic control panel allows the user to configure different scanning parameters, where the program finds the required movement. The panel is further split into three parts, being the 3D-model reconstruction, microwave scanning settings and empty area for future VNA options. In the 3D-model reconstruction part, the user can choose a scanning protocol from which the model should be built, the option to connect to the robot through Ethernet tcp and the option to start the 3D scan, resulting in a visual representation of the object. In the microwave scanning settings, the user can choose one out of three scanning modes, being full object scan, area scan or selected points scan. Furthermore, the user choose the distance to be held to the object, how dense the

scanning points should be placed in full object scan mode and area scan mode, boundary points of area in area scan mode and scanning point locations in the selected points scan mode. From the panel, the user also has the option to preview the generated scanning point locations, and to start the microwave scan.

The visual representation panel is a 3D-figure plot, where the resulting 3D model of the object from the object reconstruction and the robot end effector coordinate location and rotation is shown, as a red circle and arrow. The plot is intractable where the user can zoom, rotate and pan to view the scene in different perspectives. When the end effector moves, the movement is visually represented in the figure.

In the manual control panel the user is able to control the end effector's location in the 3D-space by incrementing or decrementing the values of its current coordinates with buttons. The user can select to manipulate the location in a Cartesian, cylindrical or spherical coordinate system. The user can also specify the step size for each movement, by how large value the coordinate should be changed, with the help of a value spinner, and the minimum distance to the object, meaning how close to the object the end effector is allowed to be, with another value spinner.

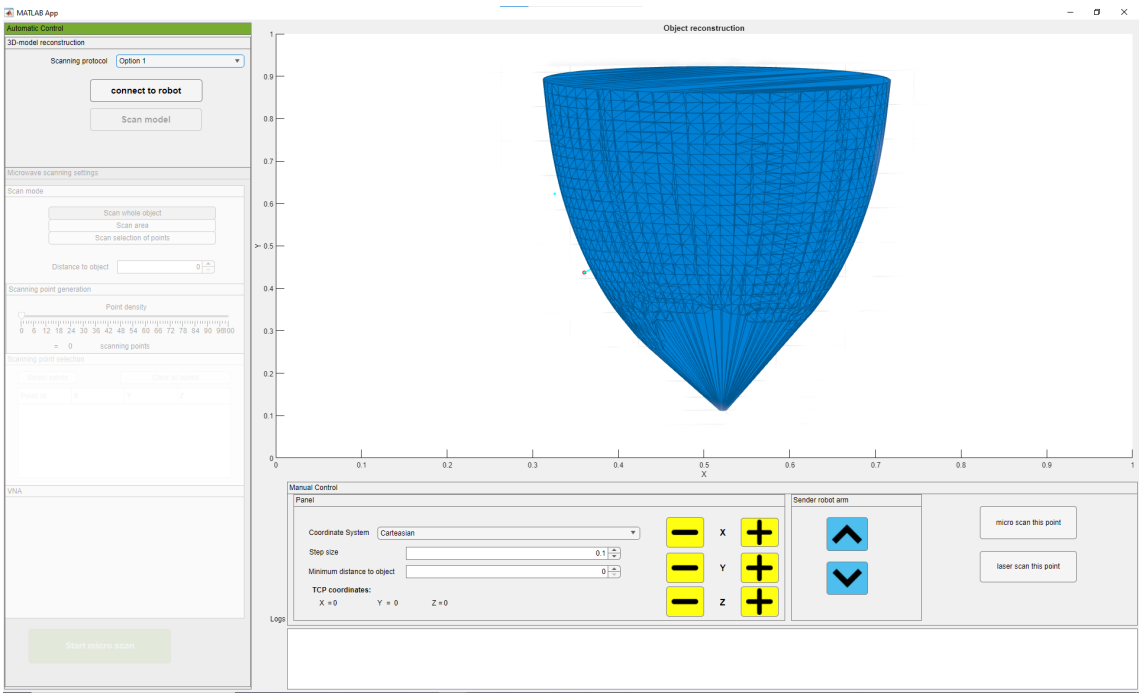


Figure 27: A screenshot of the app layout. In the app, the user can choose scanning protocols and scanning options, interactively see the model representation of the object and the end effector coordinates and angle in the 3D-space, manually control the end effector location in different coordinate systems and view progress and error logs.

6.3.5 GUI features - *Amanda*

The location of the robot's end effector can be manually controlled with the provided plus and minus buttons. Each button either increase or decrease one of the three parts of the coordinate's value. The end effector location is always represented internally in the system and when communication with the robot as Cartesian coordinates, but the user can manipulate the coordinates in spherical and cylindrical coordinate systems with simple transformations functions between the coordinate systems provided by MATLAB. The user control the step size of each coordinate change with a stepper input.

Before each step in manual control mode, the step is controlled for collision with the object.

This function prevents the user from driving the robot end effector into, or too close to the object, adding additional safety for the users. To test for collisions, a vector from the future position of the end effector, pointing towards the center of the object and with the length of the minimum distance to object spinner value, is created, see figure 28. The point of the vector is tested for intersection with the object's alpha shape, using the inshape() function of the alpha-shape class featured in MATLAB. If the point of the vector would be within the shape, the movement step will be denied and a error message is printed in the log window for the user. If the the step is allowed, the end effector will send the new coordinate and rotation to the robot which will execute the movement.

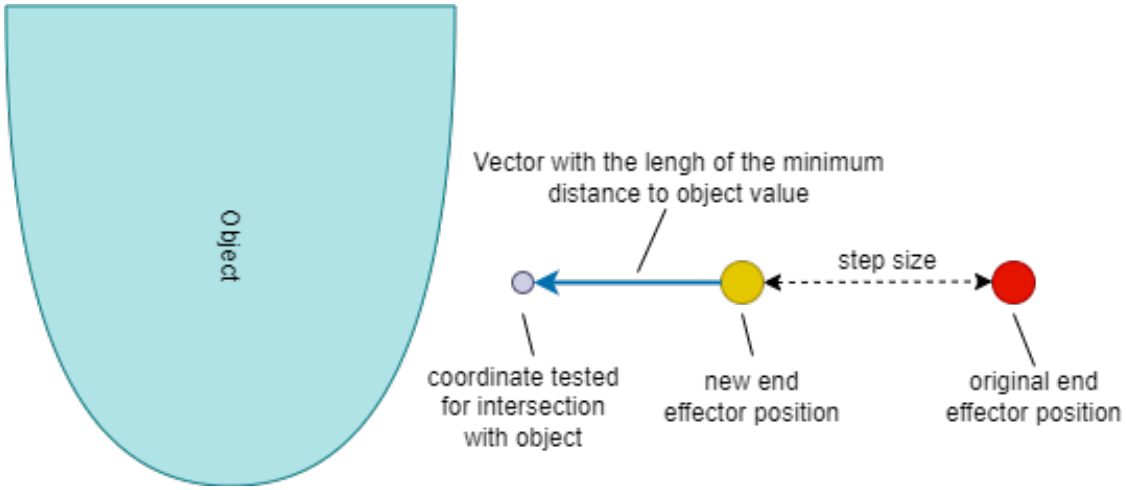


Figure 28: Figure describing the collision detection of the GUI. The future position is tested before a step is made to avoid collision with the object.

When the robot end effector is moved, in both manual and automatic mode, the rotation of the end effector is configured so that its always angled normal to the object's surface, pointing to the inside of the object. This rotation is achieved by calculating the center points and the inverted surface normals of the boundary triangles, constituting the object surface:

$$\vec{n} = -\frac{(p_1 - p_2) \times (p_1 - p_3)}{\left\|(p_1 - p_2) \times (p_1 - p_3)\right\|} \tag{1}$$

where p represents the vertices of the triangle. The surface normal's are inverted to point inwards, to represent the direction of the end effector. When a TPC movement is preformed, a search is done to find the closest boundary triangle center point to the new end effector coordinate in the 3d space, from which the correlated surface normal is chosen. From the inverted surface normal a yaw Eq.(2), pitch Eq.(3) and roll angle Eq.(4) is calculated in degrees and communicated to the robot control system together with the new coordinate position.

$$\alpha_y = \tan^{-1}(\vec{n_y}, \vec{n_x}) \cdot \frac{180}{\pi} \tag{2}$$

$$\alpha_p = \sin^{-1}(\vec{n_z}) \cdot \frac{180}{\pi} \tag{3}$$

$$\alpha_r = \tan^{-1}\left(\frac{\vec{w_0} \cdot \vec{u}}{\|\vec{w_0}\|}, \frac{\vec{u_0} \cdot \vec{u}}{\|\vec{u_0}\|}\right) \cdot \frac{180}{\pi} \tag{4}$$

Where $\vec{n}$ is the inverted normal vector for the selected boundary triangle, $\vec{u} = [0, 0, 1]$ representing the up direction in the space, $\vec{w_0} = [-\vec{n_y}, \vec{n_x}, 0]$ and $\vec{u_0} = \vec{w_0} \times \vec{n}$.

One feature of the GUI is the ability to select microwave scanning points and scanning areas by clicking directly on the object's surface in the GUI plot window. To achieve this, MATLAB datacursormode is used with a custom data tip function, which is called when the surface of the

object is clicked. The custom data tip function replaces the standard data tip functionality by saving the coordinates of the point and drawing it on the object. If a point already exist at the clicked point, it will be removed. Furthermore, if the scanning mode is in the scan area mode, three or more points marked on the object will trigger the scanning point generation function, which will generate points within the marked area. In "Scan selection of points" scan mode the user clicks on the model to produce scanning points

6.3.6 Microwave scanning point generation - *Amanda*

The generateScanningPoints function generates almost evenly spaced points along the whole object's surface. The spacing between the points are controlled by the point density slider in the automatic control panel. A high value of the slider will result in tightly placed points, while a lower value will result in the opposite. Points are generated along the object's surface by programically stepping around the object, from the top to the bottom of the object in a cylindrical coordinate system, trying to place points on the surface at the distance given by the density slider.

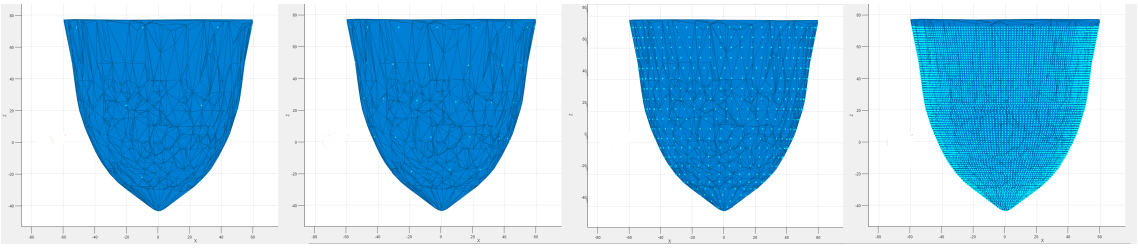


Figure 29: Scanning points (cyan) generated at different density values. From left to right: 0% density resulting in 17 scanning points for this object, 55% density resulting in 64 scanning points, 80% density resulting in 272 scanning points and 100% resulting in 13080 scanning points.

A cylindrical coordinate system is used for navigation in the space because of the innate shape of the objects that will be represented in the system. Because the objects are always somewhat round, aligned with the Z axis in it's length, and have its mass centered around the middle, the theta value of the cylindrical coordinate system allows for movement on a circle around the object, while the Z value is the same as in a Cartesian coordinate system, allowing for positioning up and down on the object, along it's length. The R value of the cylindrical coordinate system, i.e. how far away from the center of the object the surface lay at that theta and z, or the radius, is interpolated by giving the theta value and the Z value to a scattered interpolation function. The interpolation function is trained on previous vertex and center point coordinates of the object model, translated to the cylindrical coordinate system. Because of the possibility of lacking information of the radius in some areas of the model, making the interpolation inaccurate in those areas especially on unsymmetrical shapes, an additional function attempts to place the point more exactly on the object's surface by adjusting the R value given by the interpolation, first reducing the value so that the point is found inside of the object, collision controlled by using the inShape function, then increasing the r value until the point is outside of the object's surface. This results in a point that lays extremely close to the outer boundary of the object's surface, essentially being on the surface of the object.

The spacing between the generated points are, as previously stated, based on the density slider. However, in many cases it will be impossible to maintain that exact spacing. Therefore, a tolerance value was introduced, adding a interval where the point's position is accepted if its within the interval. The distance between the points are adjusted iteratively, by adjusting the theta and z value in small incremental or decremental steps until the distance to previous and above points are in the accepted interval. To avoid getting stuck in infinite adjustments to a point, where the desired interval is always missed, the tolerance value is increased after a thousand adjustments done to a point's position, increasing the chance of acceptance. The distance's are always measured to the

closest above point and the closest previous point in the same row. The adjustments are done in such a way that the theta and z can increase and decrease depending on if the distance is too big or to small, but the point is never allowed to pass back over the previous or above point. When a whole round around the object is completed, Z is decreased and a new row is started below, until the lowest point of the object is reached. If the "scan area" scan mode is selected, a selection will be done on the generated points, so that only the points within the marked area are saved and displayed.

7. Results

7.1 Positioning System

7.1.1 Grid test - *Aref*

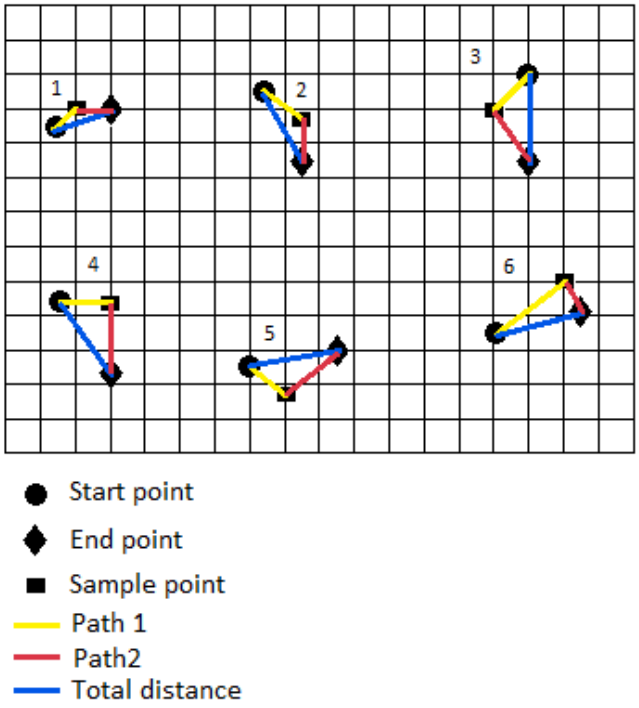


Figure 30: In this test the robotics arm´s links has been moved without the end-effector that shows that the end-effector is not stationary.

The precision of the robot arm while the end-effector is stationary in one location and the other links are moving around the end-effector are defined using a paper grid, hence the name "grid test." The laser distance sensor has been mounted on the end-effector. By using the laser beam as a pointer, three samples have been taken (Start point, sample point and end point) and these points created(Path1, Path2 and Total distance) (See figure 30).An A4 paper sheet that has a printed 4x4 mm grid has been attached on the surface of the robot's table . To test any potential arm-related errors, the paper has been attached by both sides of the robot. The testing procedure began when the robotic arm was moved to a specific spot and used the distance sensor laser beam to point on a specific point on the grid. by shifting every link while maintaining the same position for the laser beam. The outcome fell short of expectations, and the beam's position has been adjusted because of some calibration error in the SAY arm, as , shown in 1.

Samples	Path1	Path2	Total distance
1	4.5 mm	5 mm	8 mm
2	8 mm	6.5 mm	11.5 mm
3	8 mm	9 mm	12.5 mm
4	7.5 mm	10.5 mm	12.5 mm
5	6.5 mm	10 mm	13 mm
6	12 mm	5 mm	12.5 mm

Table 1: The error of the robot arm while the end-effector is stationary in one location and the other links are moving around

7.1.2 Rotation test - *Aref*

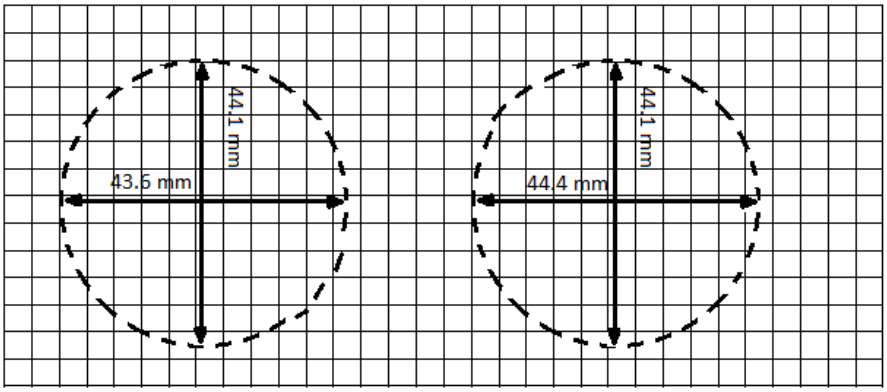


Figure 31: The circles are similar to each other that gives that the error in the 3D printed parts the error is minor

Only link 6 has been used in this test to identify the 3D-printed end-effector accuracy. The plan was to rotate link 6 and fix all links in a fixed position. The laser beam should move in a circular pattern since the link will spin around its axis and the laser beam will move around the link axes. by incrementally rotating the axis and marking the laser beam with a pen. Ultimately, a circle has been painted. The diameter was measured using a caliper to confirm that the shape is a perfect cycle (See figure 31). The diameter was measured using a caliper to confirm that the shape is a perfect cycle. That shows that a minor errors comes from the end-effector 3D model.

7.1.3 3D printed models tests - *Aref*

In this test, the design dimensions in SOLIDWORKS were compared to the dimensions of the 3D-printed end-effector. The test was simple and involved comparing the dimensions of the 3D-printed end-effector to those of the original SOLIDWORKS design using a caliper and an accurate roller.

The end-effector has no faults when it comes to the degree of inclination compared to the SOLIDWORKS design, as seen in the table from the dimension comparison and angle measurement. On a disc with an inclination of y = -0.4 and x = 0.8 degrees, the end effector has been placed perpendicularly. After that, the spirit bubble was utilised to gauge how far the end-effector's point was inclined.The outcomes in this test are shown 2.

Sample	Real Value	SOLIDWORKS Value
1	59.70 mm	60 mm
2	70.40 mm	71 mm
3	70 mm	70 mm
4	34.60 mm	35 mm
5	31.50 mm	32 mm
6	52.80 mm	53 mm
7	0.0°	0.0°

Table 2: Results of distance measurements in a comparison between the length of the SOLIDWORKS 3D model and the length of real-life 3D printed parts.

7.1.4 OUT-IN test - *Aref*

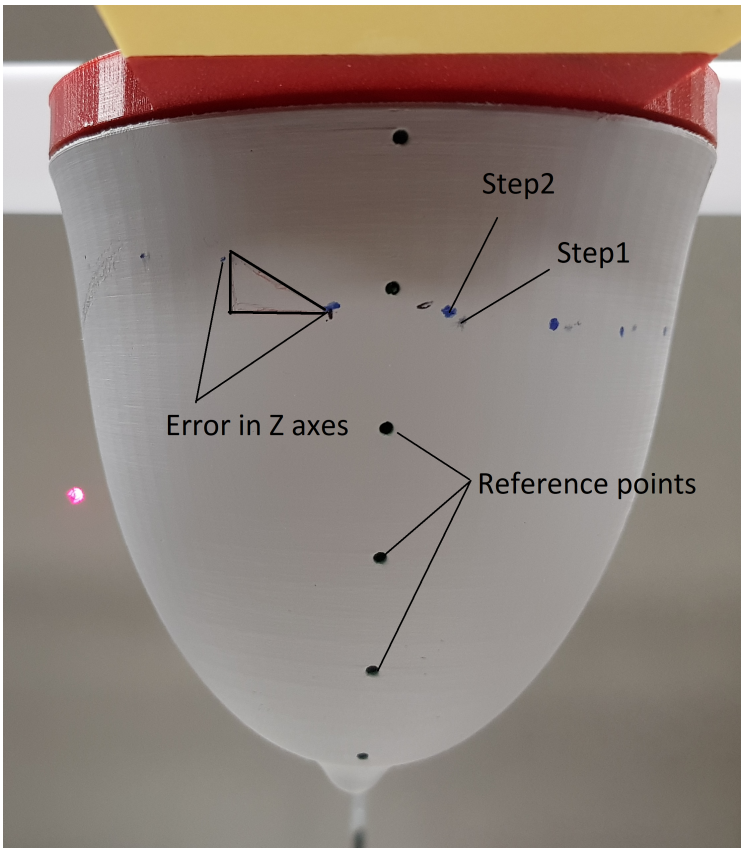


Figure 32: This picture has been taking during the in out test. the SAY robotics arm has been moved around the OUS model as steps on the same plan in circular trajectory.The gray points show the laser beam position when the robot is around 10 mm from the OUS model "Step1" and the blue points are the position of the laser beam when the distance with the OUS model increase to 80 mm at exactly the same point "Step2". The picture shows that even the height of the beam changes significantly between the first and the last point in the trajectory.

In the OUT-IN test the SAY robotics arm moved around the physical 3D OUS at the same level there the laser beam pointing at the OUS at a certain point. The OUS model has been designed with some reference points to positioned of four vertical splines. between each point and the others, 20mm in height and splines are 90°from each other. As Step1 the robotics arm go to the start point which is one of the reference points on the OUS. The laser beam pointing on a certain point on the OUS 10 mm far from the OUS surface. thereafter, as Step2, the SAY moves back to increase

the distance to 80 mm at the same point according to SAY positioning (See figure 32).

This test shows a big difference in height between the first and the last sample points in the horizontal circle, the error in z axis was around 9 mm. As well as the difference in distance between the position of the laser beam when the arm moving between step1 and step2 that was between 2mm up to 5mm (See figure 32).

7.1.5 Sender Robot - *Emanuel*

Overall, the Sender Robot worked well. It was tested by measuring the height of the Sender Robot end-effector at different heights with the Distance Measurement System stationary above the Sender Robot. The distance between the laser and the Sender Robot end-effector when at height 0 was subtracted from the measurements. Then the Sender Robot was moved to a higher position and a new measurement was taken. This was repeated over the range 0-140mm. Then the Sender Robot was lowered and new measurement were taken on the way down to measure if there is a difference moving upwards or downwards. Table 3 and Table 4 display the measured data.

Sender Robot Height *(mm)*	**Raw Measurement** *(mm)*	**Subtracted Measurement** *(mm)*	**Error** *(mm)*
0	221.060546875	0	0
5	216.677734375	4.3828125	-0.6171875
10	211.59765625	9.462890625	-0.537109375
15	205.8203125	15.240234375	0.240234375
20	201.23828125	19.822265625	-0.177734375
30	191.875	29.185546875	-0.814453125
40	181.4658203125	39.5947265625	-0.4052734375
50	170.8076171875	50.2529296875	0.2529296875
60	160.298828125	60.76171875	0.76171875
70	150.3876953125	70.6728515625	0.6728515625
80	140.3271484375	80.7333984375	0.7333984375
90	130.1669921875	90.8935546875	0.8935546875
100	120.7041015625	100.3564453125	0.3564453125
110	110.59375	110.466796875	0.466796875
120	100.6328125	120.427734375	0.427734375
130	90.6220703125	130.4384765625	0.438476562
140	80.3125	140.748046875	0.748046875

Table 3: Measurements of Sender Robot accuracy test moving upwards.

Before the project presentation, a similar test to this test was made, but with fewer measurements. In that test, the error was always less than 0.5mm.

Sender Robot Height *(mm)*	**Raw Measurement** *(mm)*	**Subtracted Measurement** *(mm)*	**Error** *(mm)*
0	221.0107421875	0.0498046875	0.0498046875
50	170.658203125	50.40234375	0.40234375
100	120.75390625	100.306640625	0.306640625

Table 4: Measurements of Sender Robot accuracy test moving downwards.

It can be seen that the error in Table 3 almost seems to oscillate. In Figure 33 the error is plotted to further show how the error varies with height. The average error is 0.2150268554mm.

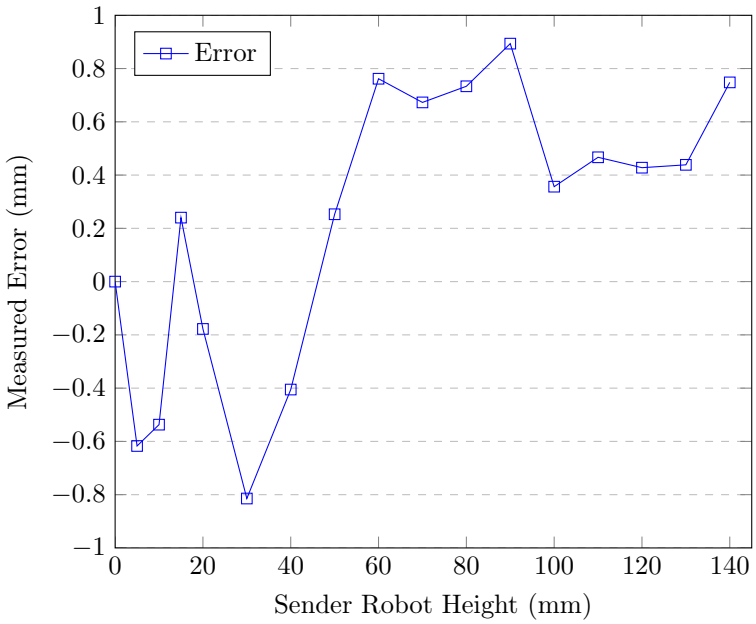


Figure 33: Accuracy error of Sender Robot

As a last test, the calibration function was run multiple times and a height measurement was taken between each calibration. The purpose of this test was to estimate how well the system can find the same zero point repeatedly. Table 5 display the measurements taken during this test.

Sender Robot Calibration	**Raw Measurement *(mm)***	**Error Compared to Previous Measurement *(mm)***
First	221.060546875	-
Second	221.1103515625	0.0498046875
Third	221.16015625	0.0498046875
Fourth	221.16015625	0
Fifth	221.16015625	0

Table 5: Measurements of Sender Robot height doing repeated calibration.

Physical Design Overall, the Sender Robot design proved to be successful and provide a foundation for future iterations of this project to build upon.

- **The Base:** The base had to be redesigned a few times because the components had to change due to of supply issues at different suppliers. Once the components arrived and everything could be properly measured, everything worked as expected. The base is more stable than expected.
- **The Cart:** There were no issues related to the cart. It worked as expected.
- **The Stabiliser:** There were no issues related to the stabiliser. It worked as expected.
- **The sender End-Effector:** There were no issues related to the end-effector. It worked as expected.
- **The Coupler:** Because no 5mm to 10mm couplers were available for purchase, an adapter had to be 3D printed. This works, but the coupler appears to be not perfectly straight and can also be bent slightly.
- **Actuation:** The actuation works, but both the threaded rod and the coupler appear to be slightly bent.

Electronic Design The electronic design worked as expected.

Arduino Software The Arduino software does what it is supposed to do. A few bugs were found, but all could be fixed. Commands such as “moveUp:9999999999”,"moveDown:9999999999", “goTo:-100” and “goTo:1000” were tested but it could handle all such cases tried without issues.

7.1.6 Robot system setup test - *Ingrid*

To determine whether the robotic system can move the receiver on the surface of the OUS with all necessary degrees of freedom, a test was conducted. To prepare the test, the SAY first laser scanned the OUS to create a digital twin. The digital twin was then used to create a scanning pattern on the whole surface (see figure 29). The test was then executed by letting the SAY attempt to reach all the points created on the surface with the receiver as its tool.

The result from the test showed that the SAY could not reach all points that were created on the surface. The points behind the OUS were inaccessible owing to collision between the SAY and the sender robot, while points that were close to the base of the robot and had an angle that caused the tool to point upwards were inaccessible due to collision between the tool and the base of the SAY.

7.2 Scanning system

This section will discuss all of the results that the scanning system presents, including every system component.

7.2.1 Distance Measurement System - *Emanuel*

In the end, the Distance Measurement System worked well, but there were various issues, most of them related to the laser sensor being difficult to reset, but also the laser sensor sending invalid data and oscillating after movement. These issues could be fixed, but the fix is more temporary than proper. The accuracy of the Distance Measurement System was tested by facing the laser sensor at the table, taking a measurement, then sending a command to the Receiver Robot to move straight up 50mm from the current location, and then taking a new measurement. This was repeated over the range $\approx$ 70mm to $\approx$ 220mm. All measurements were made using both the Distance Measurement System and a ruler with a millimetre scale. Table 6 displays the results of this test.

Receiver Robot Relative Height	**Raw Measurement Distance Measurement System *(mm)***	**Raw Measurement Ruler *(mm)***	**Difference Receiver Robot and Distance Measurement System**	**Difference Ruler and Distance Measurement System**
0	71.9453125	72	-	0.0546875
50	121.7998046875	122	0.1455078125	0.2001953125
100	171.953125	171	0.0078125	0.953125
150	222.0068359375	221	0.0615234375	1.0068359375

Table 6: Measurements of Distance Measurement System accuracy test.

One measurement takes 0.2 seconds to conduct, from the time a packet is sent to the Arduino, to the time a measurement in millimetre can be read into a variable in MATLAB.

Electronic Design The electronic design worked well and there were no issues related to this.

Laser Communication The laser communication worked out, but it still does not work properly. Software fixes had to be implemented to get rid of the communication issues. In the end, a factory reset was possible to enable the full range of the sensor, but it did not solve the issues with incorrect values.

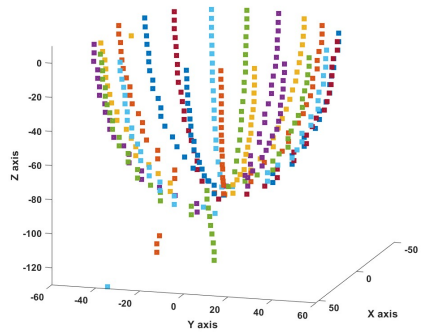
- **Laser and Computer Communication:** The laser could communicate with the computer using the USB to Serial adapter, but the measurements were incorrect and it could not be configured as initially thought.
- **Laser and Arduino Communication:** The communication between the laser sensor and the Arduino worked, but suffered from the same issues as when communicating with the computer. Through trial and error, it was discovered that using the "Serial.readBytes" function with a fixed length of 16 chars, a valid measurement was always sent between two \r which made it possible to clean up the Serial data in software.

Arduino Software The Arduino Software worked out great and did what it was supposed to do.

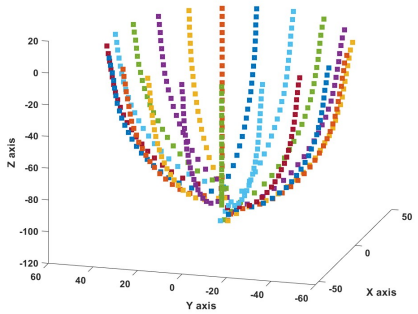
Computer Software The Computer Software worked out great and did what it was supposed to do.

7.2.2 Scanning and Reconstruction algorithm - *Jonathan and Victor*

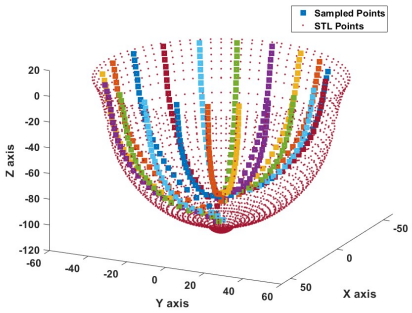
The first scan revealed some outliers in the point set. Outliers are laser-measured data points with no objects in their paths. Aside from the outliers, there were some missing data points in the nipple region. Covering the SIB with a black textile cover removes outliers, and missing data is reduced by tilting the laser 10° away from the normal to the Z-axis direction. Figure 34a and 34b shows the difference in the data when the SIB is covered and uncovered. As seen in figure 34c the measured points are not symmetrical , the data measured on the right side have higher Z coordinates then the left side. The difference in height between 2 opposite azimuth angles parallel with the Y- axis is $\approx 16mm$ which is shown in figure 34d. On the other hand the difference between the opposite azimuth angles parallel with the X-axis is $\approx 3mm$.



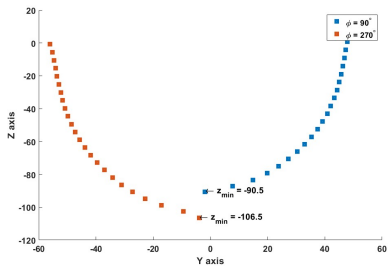
(a) The data points measured with the SIB uncovered and Laser beam normal to the Z axis



(b) The data points measured after covering the SIB with a black textile cover and tilting the laser beam 10° from the normal to the Z-axis



(c) The sampled points compared to the true values from the STL-file.



(d) The data points in 2D from 2 opposite azimuth angles $\phi = 90°$ and $\phi = 270°$. The difference in the minimum height $\Delta z_{min} \approx 16mm$

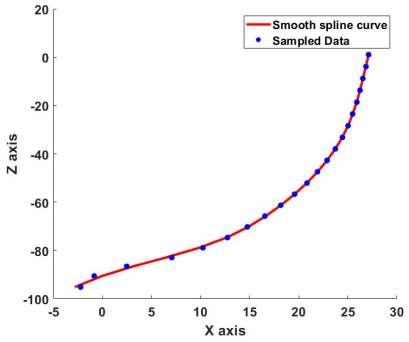
Figure 34: The figures shows the difference between the data point when the SIB is covered and uncovered.It shows also the true points and the sampled points, additionally the sampled data is plotted in 2D for azimuth angle $\phi = 90°$ and $\phi = 270°$.

The symmetrical OUS is scanned cylindrically with different azimuth intervals $\Delta\phi$ and height intervals ΔZ. The scanned data is fed into the power crust algorithm, which generates the surface directly from the scanned data without resampling. Section 9. in 41 shows the test results comparing different $\Delta\phi$ and ΔZ. The algorithm converged and the result became more accurate with a smaller $\Delta\phi$ than ΔZ. The scanning time is recorded for each configuration. The results of various scanning times are shown in Table 7. After 40 minutes of scanning time, the best results are obtained from a cylindrical scan with sampling intervals of $\Delta\phi = 5°$ and $\Delta Z = 5mm$. The configuration chosen is $\Delta\phi = 18°$ and $\Delta Z = 5mm$. It generates a set of points sorted by azimuth angle ϕ.

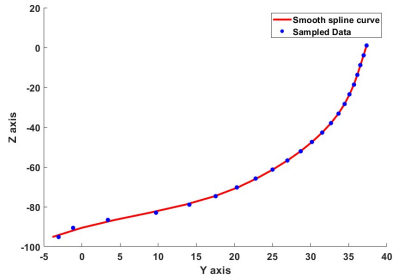
	90 deg	45 deg	22.5 deg	11.25 deg	5.625 deg
20 mm step-size	1.32	2.35	4.11	7.96	16.00
10 mm step-size	1.80	3.5	6.34	12.45	24.23
5 mm step-size	2.87	5.24	10.07	20.31	40

Table 7: The time it took for each of the scans in varying degree-steps and size-steps in height. The time displayed is in minutes.

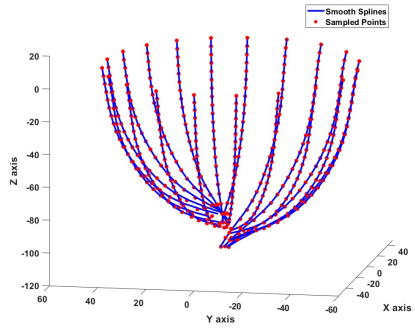
The smooth splines functions of the MATLAB curve fitting toolbox [51] are used to fit the data point coordinates. In addition, the app allows the change of the smoothness and roughness parameters. A smoother curve surface prioritises smoothness and fits the data points less accurately. In contrast, a rougher curve surface fits the data more accurately but may lose smoothness and become wobbly. After approximating the curves the smooth splines are sampled at a $\Delta Z = 2mm$. The splines fit in the x dimension is shown in figure 36a and the y dimension in figure 36b. The 3D plot with the sampled data is shown in figure 36c and the resampled splines are shown in figure 36d. The average root-mean-square error (RMSE) for the x and y dimension is $0.47mm$. The outcome of the resampling is a new point set equidistant in the z dimension with an intervall of $\Delta Z = 2mm$.



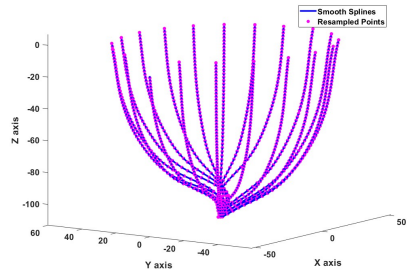
(a) The smooth spline fitted curve to the sampled data for x coordinates



(b) The smooth spline fitted curve to the sampled data for y coordinates



(c) The approximated splines in 3D with the sampled point from the surface.

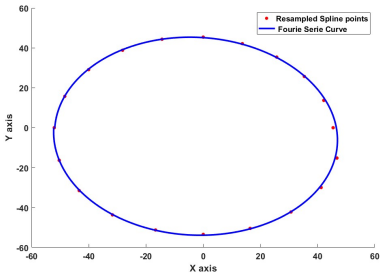


(d) The splines fitted on the sampled data are resampled at an interval of $\Delta Z = 2mm$.

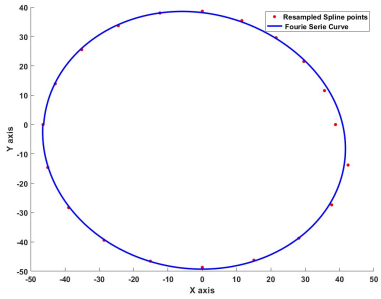
Figure 35: The results of the spline fitting and resampling, The average RMSE in x and y coordinate is $0.47mm$.

For each height z the x and y coordinates of the new resampled point set are converted to polar coordinates (r, ϕ). A second-degree Fourier Series function $Q(x)$ is fitted to the relationship between r and ϕ using the curve fitting app in MATLAB, such that $Q(\phi) = r$. The fitting results are shown in figure 36, The error varies depending off the OUS region. Near the chest wall the average RMSE is in average $0.5mm$ while in the middle region the RMSE average is $0.8mm$ and finally around the nipple region the average RMSE is $3.3mm$. Finally, the Fourier series is re sampled with a 3°

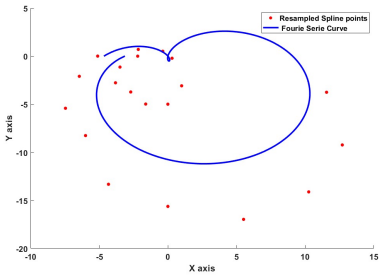
sampling interval between 0° and 0°. As a result, a dense set of uniformly distributed points has been generated and used as the power crust input.



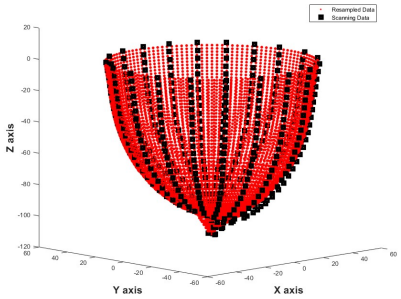
(a) Fourier Series fitting the sampled data from the height close to the chest wall region.



(b) Fourier Series fitting the sampled data from the height in the middle region of the OUS

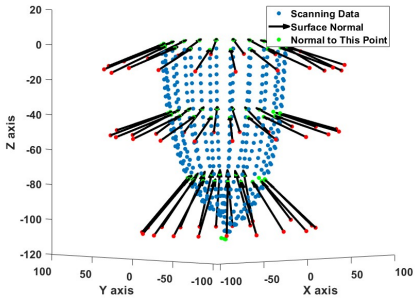


(c) Fourier Series fitting the sampled data from the height close to the nipple region. The sample points in the middle are from the shifted splines shown in figure34d.

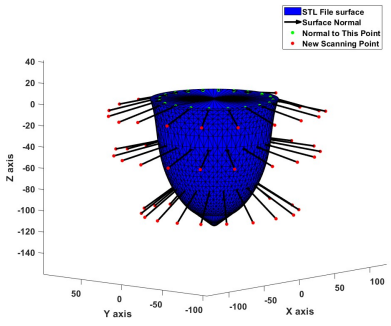


(d) The complete resampling of the scanned data, The new data set is more dense and approximately uniformly distributed.

Figure 36: The results of the Fourie series fitting to each height of the new data resampled from the splines curves. The Fourier series curve fitting error varies between different height regions. Resampling the Fourier series curves produce a complete resampled OUS.



(a) The 3D plot of the normal to the surface , scanning data points and the new scanning position.



(b) The 3D plot of the normal to the surface, the true surface from the STL file and the new scanning position.

Figure 37: Shows the results of the implementation of finding the surface normals and placing the laser normal to the surface at a certain distance.

The new step of orienting the laser normally to the surface is implemented and tested on the sampled data, but the surface was not rescanned. Figure 37 shows the new scanning points with

the normal orientation to the surface. The evaluation of the system couldn't be tested on the unsymmetrical OUS.

The result from the reconstruction algorithm fitted the data well as can be seen in picture 38. Due to the error from the SAY which resulted in that the interpolated data to be off. The reconstruction is off the actual data from the CAD file, this is depicted in 39. The reconstructed model is off from the actual point cloud imported.

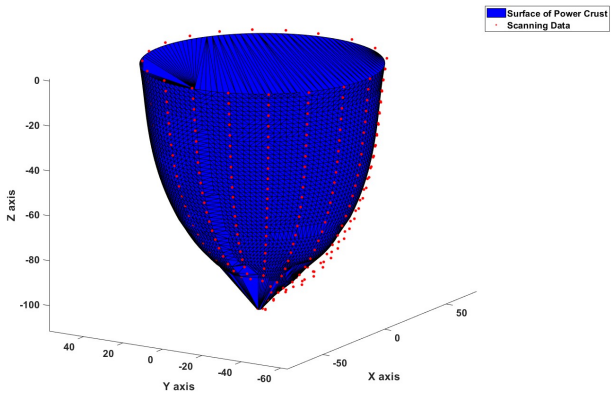


Figure 38: The sample points in red dots are compared to the reconstruction of these point which is depicted in blue. In the picture the results shows that the reconstruction algorithm can fit the input data well.

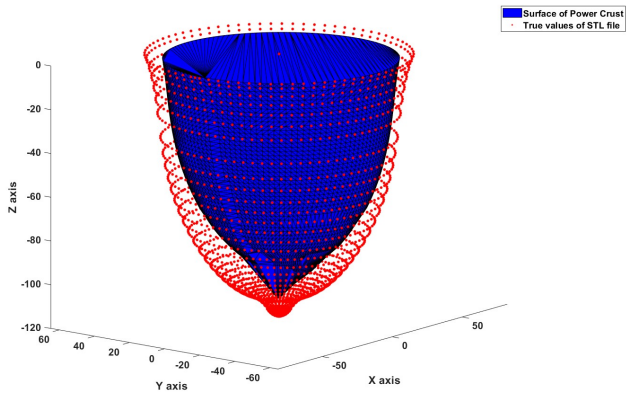


Figure 39: The true points from STL file in red dots are compared to the reconstruction of the scanned points which is depicted in blue. The reconstructed surface is off from the model, especially on one side of the model.

7.3 Other Systems

7.3.1 Network - *Emanuel*

After all devices were connected to the network and assigned a static IP, the devices were sent pings to make sure they can communicate. Some issues related to IP conflict were then discovered, which was easily corrected by assigning a different static IP to the device with a conflict. After all devices could ping each other, all other communication worked.

7.3.2 Power Supply - *Emanuel*

There were no issues related to the power supply. It worked as expected.

7.3.3 Arduino Ethernet Communication - *Emanuel*

The Ethernet communication worked great using this library. There were no issues related to the Ethernet communication.

7.3.4 System Installation bench(SIB) - *Aref*

The SIB has been created and produced to hold the OUS 3D model while the system is in operation, and it serves the intended purpose. There are a few minor instability difficulties, but they don't have a big impact on the scanning and measuring processes of the microwave. The SIB's inclination was measured using a spirit bubble, and the results show that it is x=-0.4 and y=-0.6, which is similar to the robot's table inclination, indicating that there are no severe inclining difficulties. Due to the elasticity of the plastic utilised, the SIB can move when pressure forces are applied to its sides.

7.3.5 The receiver End-effector - *Aref*

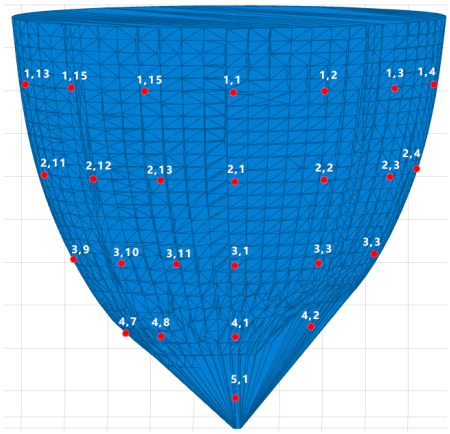
Additionally serving the intended purpose, the end-effector houses the microwave receiver and the distance laser sensor and fastens them to the SAY robotics arm. The 3D models have an average error of 0.3 mm and an inclining error of almost 0.0 degrees, as stated in the test of the 3D printed models. However, this considerably improves the end-effector' suitability while having no noticeable adverse effects on the system.

7.3.6 Microwave scanning point distance experiment - *Amanda*

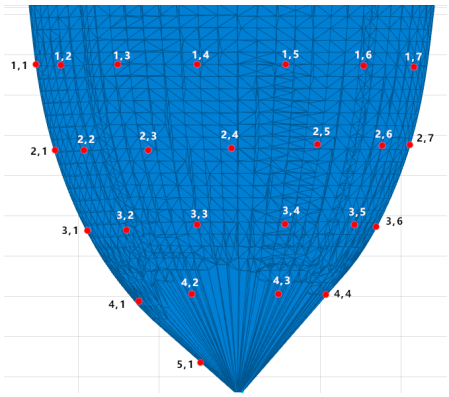
An experiment was done to test the accuracy of the reconstructed object model compared to the OUS by using the microwave scanning point generation feature of the GUI. First, the OUS was scanned with the laser while the object and system were covered, making the measuring area dark to mitigate influence from the environment. Based on the coordinate values from the laser scanning phase, the object was reconstructed and displayed in the GUI. In the automatic control panel, the "Scan whole model" mode was selected, and the "Point density" slider was moved to the value of 60. This results in scanning points generated over the digital object's surface. All scanning points were labelled to easier to identify and to facilitate analysis of the results. The experiment was done in four iterations. The experiment's first iteration was conducted by setting the "Distance to object" value to 0 and then pressing the "Start micro scan" button. At each scanning point the distance to the object from the tip of the receiver was measured with a caliper, and noted, until the whole microwave scanning phase was completed. This was repeated for the "Distance to object" value 4, 9 and 15, all together resulting in table 8, 9, 10, and 11, respectively.

After scanning the object and setting the "Point density" slider to a value of 60, the micro-scanning points seen in figure 40 were generated. Because the object and scanning points are displayed in 3D, the four subfigures show four different sides of the object.

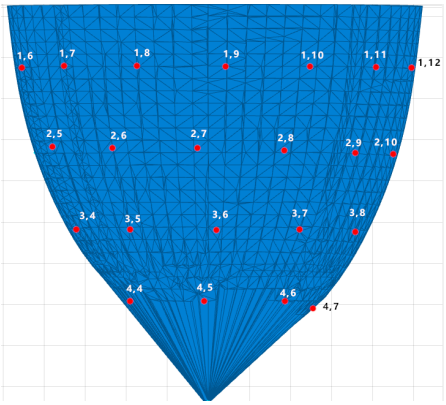
When setting the "Distance to object" value to 0 millimeter, the following distance values between the tip of the receiver and the object, seen in table 8 was measured. The results show that when set to a desired distance of 0mm, the average distance around the object was 5.6 millimeter, and the largest distance was 12 millimeter at scanning point (3, 1), found on the lower parts of the object, and the lowest distance 0 millimiter at scanning points (1,9), (1,10), (1,11), (2, 8) and (2,9), see figure 40. All planned scanning points in the experiment could not be completed due to issues with the robot configuration, where the robot could not reach one or more scanning points and execution of experiment was aborted. In the following tables in this section, the "-" sign signifies that the above issue occurred in the scanning sequence and that execution of the experiment was aborted. When setting the desired distance from the object to 4 millimeter the values in table 9 were measured. The table shows that the average distance to the object was 10.3 millimeter,



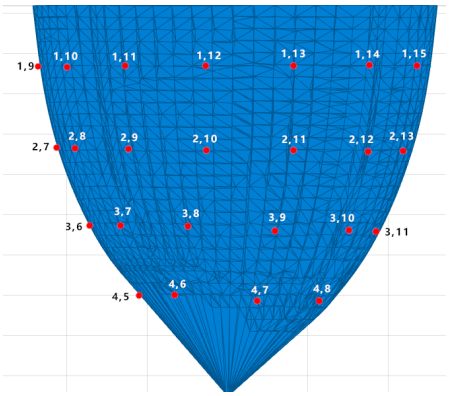
(a) Side one, scanning points seen from the perspective of negative y-axis and positive z-axis



(b) Side two, scanning points seen from the perspective of a negative x-axis and positive z-axis



(c) Side three, scanning points seen from the perspective of a positive y-axis and positive z-axis



(d) Side four, scanning points seen from the perspective of a positive x-axis and positive z-axis

Figure 40: Scanning points generated over the object's surface at a "Point density" value of 60, seen in four different perspectives.

with big individual variations depending on where the scanning point was located on the object. Generally, individual distances increased the further down on the object the scanning was made. The largest measured distance was 18 millimiters at point (4,1) and the scanning points closest to the desired distance was point (1,10) and (1,11). Similarly, table 10 and 11 shows the measured distance when the desired distance was 9 millimeter and 15 millimeter, respectively. Both tables show an average error of around 5.8-5.9 millimeters from the desired distance, and large variations in the distance of individual points depending on where on the object the points were located, but that the distance generally increased when the scan was made further down on the object.

Row	Measured distances (mm)	Average distance (mm)	Average error (mm)
1	8, 8, 7, 6, 5, 5, 3, 1, 0, 0, 0, 1, 3, 4, 6	3.8	3.8
2	10, 10, 8, 5, 4, 3, 1, 0, 0, 2, 4, 5	4.7	4.7
3	12, 10, 8, 6, 5, -	8.2	8.2
4	-	-	-
5	-	-	-
All		5.6	5.6

Table 8: Results of distance measurements when distance to object was set to 0mm. The measuring point's locations on the object corresponds to those in figure 40.

Row	Measured distances (mm)	Average distance (mm)	Average error (mm)
1	10, 12, 11, 10, 9, 8, 7, 5, 5, 4, 4, 5, 6, 7, 8.5	7.4	3.4
2	14, 16, 13, 11, 10, 9, 8, 6.5, 6, 5, 6, 7, 9	9.2	5.2
3	16, 15, 12, 12, 10, 7, 6.5, 6, 6, 9, 10	10	6
4	18, 15, 11 , -	14.6	10.6
5	-	-	-
All		10.3	6.3

Table 9: Results of distance measurements when distance to object was set to 4mm. The measuring point's locations on the object corresponds to those in figure 40.

Row	Measured distances (mm)	Average distance (mm)	Average error (mm)
1	16, 16, 15.5, 16, 15, 15, 14, 10, 9, 9, 10, 11, 13, 12, 13	13	4
2	20, 18, 17, 16, 15, 14, 13, 11, 9, 10, 10, 12, 14	13.8	4.8
3	20, 19, 17.5, 16, 15, 15, 11, 11, 12, 13.5, 14	14.9	5.9
4	21, 17, 16 , -	18	9
5	-	-	-
All		14.9	5.9

Table 10: Results of distance measurements when distance to object was set to 9mm. The measuring point's locations on the object corresponds to those in figure 40.

Row	Measured distances (mm)	Average distance (mm)	Average error (mm)
1	20, 22, 23, 22, 20.5, 19, 18, 16.5, 16, 15, 16.5, 16.5, 17, 18, 20	19.5	4.5
2	25, 24.5, 25, 23.5, 22, 20, 18.5, 17, 16.5, 16, 17, 18, 20	20.2	5.2
3	26, 24, 23, 22.5, 20, 19, 19, 17, 17, 18.5, 20	20.5	5.5
4	27.5, 21.5, 20 , -	23	8
5	-	-	-
All		20.8	5.8

Table 11: Results of distance measurements when distance to object was set to 15mm. The measuring point's locations on the object corresponds to those in figure 40.

8. Discussion - *Everyone*

The purpose of this project was to develop an automated measurement system for microwave imagining. From this purpose, four research questions were specified to evaluate the developed system on suitability, limitations, and accuracy. Four goals were also specified to steer the project in the right direction. At the end of this project all four goals were achieved. This has resulted in a completed measurement system for microwave imaging that can position the sensors around the OUS at high degrees of freedom and at different distances from the object. The scanning phases of the system was divided into a laser scanning phase and micro scanning phase. The laser scanning phase measured the distance to the object with a laser sensor for the object's surface to be reconstructed digitally. The microwave scanning phase used the digitally reconstructed surface to position the sensors more accurately around the object. Additionally, the system was accompanied by a GUI that enabled the user to control the scanning phases and scanning parameters, a system installation bench on which the OUS was mounted and a robotic arm that could move the sender.

The research questions were attempted to be answered through various tests of the system and its accuracy. These tests showed that the accuracy of several parts of the system were not as expected, and that the system was subjected to some limitations. One limitation was the robot system setup. During the laser scanning phase, there were no problems with the current setup of the system; however, in the microwave scanning phase, issues arose. These issues can be divided into two sections: the first is the collision between the SAY and the sender robot when the SAY has to stretch its arm to reach the surface on the other side of the OUS, and the second is when the SAY attempts to reach positions close to its base, with an angle, creating a collision between the tool and the robot base.

The first issue stems from the placement of the OUS in relation to the SAY, as well as the placement and size of the sender robot. Because the sender robot is floor-mounted and must be placed beneath the OUS, the OUS must be placed in front of the SAY rather than directly above it. This limits the reach on the side of the OUS that is furthest away from the SAY's base. The reach is even more limited because the SAY needs to bend its arm around the sender robot. The second issue is also caused by how the SAY is positioned in reference to the SAY. The points on the side of the OUS that are closest to the robot base will be close to straight above the base of the robot, and the tool can collide with its own base.

Countermeasures that could be taken as the first step toward resolving these two issues are divided into two categories. The first path entails moving the OUS directly above the SAY and designing a new sender robot that is mounted from above. The sender robot must be able to move around the OUS while avoiding collisions with the SAY. The second option is to use a different robot than the SAY, which can rotate its "base" in a circle around the OUS. The new robot can reach all positions without having to bend its arm around the sender robot. This information accompanied with the results of the OUT-IN test (see 7.1.4) and the grid test (see 7.1.1) which showed troubling accuracies of the SAY's position leads to the conclusion that the SAY is not suitable in the current setup of the system. It is not possible for SAY to reach all desired positions on the surface of the OUS with this setup, and due to it being an old prototype seems to have degradation of joints and motors caused by long time wear as well as calibration issues making it misunderstand its actual location.

Because the SAY misunderstand its location due to calibration issues, these errors in position propagated through the system when scanning the object, causing errors in the reconstruction, and lowering positioning accuracy as it alters the location of the datapoints. Regretfully, it was not possible to plan a SAY calibration with ABB due to the time constraints of this project, and the fact that the SAY lent to the project was an old prototype. To continue the project, solving these calibration issues will be a crucial step to increase the overall system accuracy.

When measuring the reconstruction accuracy by comparing the PLY file to the re-sampled data and reconstructed object errors became apparent. In general, the reconstructed object is off the

OUS when the PLY file is used to represent the OUS as seen section 7.2.2. The reconstruction algorithm fits the input data well, like stated in 38. But due to a propagating error from the SAY, the input data is off, and this causes the model to be off as well in the end. This error is not consistent, making it harder to find a way to filter the data. The accuracy of the input data has a mean error of 9.7mm on one side and 17.5mm on the other.

Furthermore, when the data has a high signal-to-noise ratio, further problems arose in the form of uneven surfaces and inaccurate datapoints. Because of the surface's spiky appearance, the model became inaccurate and unusable for the GUI, resulting in inaccuracies when moving the receiver around the object. The noise in the data was reduced by having a blanket over the setup, protecting the system from influence of the surrounding. This way, many of the outliers in the point cloud were eliminated. This, in conjunction with sufficient sampling density, resulted in more consistent surface estimation.

The importance of the sample density varied in height compared to the x and y-plane. Having a small step size when sampling in the x and y - plane, gave better results than having a small step size in z-height. This is demonstrated in 41; regardless of the step size in z-height, the rotational angle in x and y can compensate to a certain degree. In this project, no time requirement was put on the system, so this aspect is something to consider if the time of the reconstruction is too long. On top of that, the scanning protocol had trouble determining how big of a step size the height should be in the z-plane to optimize the scanning point locations. An improvement would be to first scan one height line to determine the minimum and maximum height and then determine the necessary step size in height based on that. If the step size is too large, the lower regions of the object will have an unscanned region resulting in higher inaccuracies. Another approach would involve scanning a single vertical line, fitting a smooth spline to it, and then creating simulated splines by rotating the measured spline at various azimuth angles. As a result, a point set that roughly approximates the OUS surface is created. This point set can also be used with the new step to determine the surface normal and produce a better scanning path that can precisely detect surface points. Because only one vertical line of points is scanned, this method also speeds up subsequent scans, which focus on the precise points needed for interpolation.

This study [54] shows a more efficient method for interpolating non-equidistant periodic data than the Fourier series. In future work, calibration and evaluation should take precedence to determine whether particular techniques are enhancing the system. The evaluation can also help identify which system components should be improved to lower the overall error and thus enhance the system as a whole.

When considering the positioning accuracy of the system, it was evident from the microwave scanning point distance test (see 7.3.6), that there was an large error when comparing the desired distance to the object to the actual distance, which generally increased the further down on the object the sensor was placed. The test showed that there were large variations in the measured distances depending on which side of the object the sensor was placed, but these variations stayed fairly consistent among the different iterations of the test, meaning that some sides of the object seemed to be more accurately reconstructed than others, and thereby the measured distance were more accurate at these sides. The largest error measured throughout the test was up to fourteen millimetres, while the overall average error was around 5.9 millimetres. The measured distance was always larger than the desired distance, never less, meaning that some parts seem to be reconstructed as larger, or further out, than they actually are. However, further testing would be needed to confirm this, as this contrasts the results in 39 when comparing the reconstructed object to the PLY-file. This contrast could be caused by the propagated error from the SAY. Regrettably, the lower parts of the object could not be measured as the SAY were not able to reach these points without colliding with the sender arm or itself. Therefore, the result could be different if these parts were also measured.

Because this experiment evaluates the complete system, the reason for these inconsistencies in distances are hard to pinpoint. However, considering the results from the tests done on the SAY's

accuracy, this could likely be attributed to previously mentioned issues with the SAY, laser distance senor value errors, and/or a possibly unidentified error in the scanning procedure or reconstruction. The accuracy of sensor positioning during the microwave scan is greatly dependent on the accuracy of the object reconstruction, improving the accuracy of these should also improve the accuracy of the sensor placement.

When considering the accuracy of the Sender Robot, the accuracy was worse than initially believed. Measurements to get an estimate of the accuracy had been conducted before the project presentation and those tests had an error of less than 0.5mm. However, it must be noted that this test has many sources of errors such as the accuracy of the laser, that the laser was held by the SAY during the test which could be affected by movements, that the threaded rod seems bent and/or that the coupler seems bent.

The greatest source of error is probably the coupler and threaded rod, because when the Sender Robot is moving up and down, it is clearly visible that the threaded rod is moving back and forth. It is unclear if both the threaded rod and the coupler are bent, or only the coupler. As shown in Table 3 the accuracy seems to oscillate, this is believed to be caused by the bent rod and/or coupler. The threaded rod is solid and should not bend while mounted on the Sender Robot, but the coupler is not rigid and could be bend while mounted. We are confident that a new proper coupler would improve the accuracy, but with an average accuracy of $\approx 0.2mm$ the Sender Robot is still good enough for a first prototype. This coupler was used because no 5mm to 10mm coupler was in stock, this coupler was never meant as a long-term solution.

Regarding the Distance Measurement System, it now works as intended, but there were various issues related to the laser sensor during the development and testing. At the time of the test, the SAY was thought to have good accuracy. Because the test relied heavily on the SAY, and an inaccurate ruler, the test only really shows that the accuracy of the laser sensor is less than $\approx 1mm$. However, because the Serial communication is digital and the datasheet of the laser sensor provides the formula needed to convert between the sensor value and mm, the accuracy of the system should match the claimed accuracy of the laser sensor in the datasheet. Furthermore, there were issues with the laser communication. The source of the is not known. The sensor is thought to be faulty, but even though there were so many issues related to the laser sensor, a workaround could be accomplished using software. The only known downside of this for now is that a measurement is slower because more processing must be done in software.

Although the chosen method for implementing network communication works, it is far from the best method. Because all IPs are static, and must be static for the system to work properly, no changes can be made. For example, in this case, the system was connected to the MDU Network that has an address of 10.132.158.0/24, if this system would be connected to a network with a different address, it might not work. Therefore, a better approach would be, for example, to use a router with dynamic IP at WAN and port forwarding, with all internal devices using static IPs and the MATLAB computer not being on the internal network. In that case, the system could be connected to any network, the router would get an IP automatically and sending packets to the correct ports of the router would make sure that the packets are forwarded to the correct device. It might even be possible to use the hostname of the router instead of the IP. In that case, the system would work effortlessly on any network. Initially, this was the plan, but because of the chip shortage, no proper routers were in stock at any store at a reasonable price. The cheapest routers in stock were around 700-1000€. Routers in the price range of 100-300€ could be delivered at the earliest after the project deadline and therefore a switch (switches were in stock) had to be used instead.

If future work is done on this project it is recommended to replace the laser sensor. Micro-Epsilon has sponsored the project with a new laser that hopefully should solve most issues. For further improvements it should also be investigated if it is possible to use a more flexible cable with a shorter connector to make the Receiver Robot end-effector smaller. Another approach would be to use two lasers, one with short range to get close and one with longer range for when the object is too far away for the short range laser.

Further limitations were found through the system installation bench and the OUS model. During the development of the system installation bench and OUS model, issues with 3D printer supplies and printer malfunctions made the process tedious and time consuming. The job was eventually completed on schedule despite the delay. However, because of the limited 3D printer supplies, the plastic used for the system's construction was less than ideal. The plastic's elasticity led to numerous oscillations and vibrations throughout the entire bench when the SAY moved. Because the laser sensor needs to have a stable value before going to the next point, these vibration lead to even longer scanning times. Regarding the OUS models, both asymmetric and symmetric variants are available. The dimensions of these models are good, and the symmetric 3D OUS model was useful in many of the tests. Nevertheless, the asymmetric OUS 3D model was slightly larger than anticipated and has not been assessed because of shifted focus on the inaccuracies of the SAY. Additionally, one issue to solve in future work is how to minimise the unreachable area of the OUS that is close to (the SIB/the bed). One suggestion is to positioning the tip of the receiver in the highest point of the end-effector. For the same reason, the laser beam should likewise be at the higher part of the end-effector.

Finally, the network solution for communication between system components in this project has not been ideal. Although the chosen method for implementing network communication works, it is far from the best method. Because all IPs are static and must be static for the system to work properly, no changes can be made. For example, in this case, the system was connected to the MDU Network that has an address of 10.132.158.0/24, if this system would be connected to a network with a different address, it might not work. Therefore, a better approach would be, for example, to use a router with dynamic IP at WAN and port forwarding, with all internal devices using static IPs and the MATLAB computer not being on the internal network. In that case, the system could be connected to any network, the router would get an IP automatically and sending packets to the correct ports of the router would make sure that the packets are forwarded to the correct device. It might even be possible to use the hostname of the router instead of the IP. In that case, the system would work effortlessly on any network. Initially, this was the plan, but because of the chip shortage, no proper routers were in stock at any store at a reasonable price. Routers in a reasonable price range could be delivered at the earliest after the project deadline and therefore a switch had to be used instead.

9. Conclusions

In conclusion, all goals for the project were achieved resulting in a complete system for microwave imaging that could scan and reconstruct the surface of an object and preform automated microwave scanning sequences based on the reconstructed object surface. Through testing and validation of the system it was found that this specific individual of SAY was not suitable for this project because of its inaccuracies regarding calibration and because of the limitations of the system setup. Its issues are thought to have propagated throughout the entire system, causing lower than desired positioning accuracy for both the receiver and sender antenna, showing an average error of 5.9 mm for the receiver and 0.2 mm for the sender. The positioning accuracy of the receiver during the microwave scanning phase was also directly dependent on the accuracy of the surface reconstruction. In turn, the accuracy of the surface reconstruction was also lower than desired with a mean error up to 17.5 mm, making it difficult to evaluate chosen methods for the reconstruction. The system was affected by many limitations such as the physical system setup, the material used for the system installation bench and the networking solution. While this project has developed a functioning system for microwave imaging, the system is far from perfect and does not reach the submillimetre placement precision goal that was envisioned at the start of the project. Suggested future work for the system includes solving the issues with the SAY, improving the system setup, preforming more evaluation on the system, improving scanning protocols and algorithms, and changing certain parts of the system hardware for more suitable alternatives.

References

[1] A. Helena Torsler and N. Loman, *Bröstcancer*, Cancerfonden, [Online] Available: https://www.cancerfonden.se/om-cancer/cancersjukdomar/brostcancer, 2021.

[2] S. Susanna, *Mammografi*, 1177, [Online] Available: https://www.1177.se/Vastmanland/behandling--hjalpmedel/undersokningar-och-provtagning/bildundersokningar-och-rontgen/mammografi/, 2020.

[3] T. Rydholm, 'Experimental Evaluation of a Microwave Tomography System for Breast Cancer Detection,' En, Ph.D. dissertation, Chalmers University of Technology, Gothenburg, 2018. [Online]. Available: https://research.chalmers.se/publication/505529/file/505529_Fulltext.pdf (visited on 07/09/2022).

[4] N. Petrovic, T. Gunnarsson, N. Joachimowicz and M. Otterskog, 'Robot controlled data acquisition system for microwave imaging,' p. 5,

[5] B. Siciliano, *Robotics: modelling, planning and control*. Springer, 2009.

[6] P. I. Corke and O. Khatib, *Robotics, vision and control: fundamental algorithms in MATLAB*. Springer, 2011, vol. 73.

[7] MathWorks, *Robotics system toolbox*, https://se.mathworks.com/products/robotics.html, Accessed 19-12-2022.

[8] Z. Lu, A. Chauhan, F. Silva and L. S. Lopes, 'A brief survey of commercial robotic arms for research on manipulation,' in *2012 IEEE Symposium on Robotics and Applications (ISRA)*, Kuala Lumpur, Malaysia: IEEE, Jun. 2012, pp. 986–991, ISBN: 978-1-4673-2207-2 978-1-4673-2205-8 978-1-4673-2206-5. DOI: 10.1109/ISRA.2012.6219361. [Online]. Available: http://ieeexplore.ieee.org/document/6219361/ (visited on 11/01/2023).

[9] A. Haleem, '3D scanning applications in medical field_ A literature-based review,' en, p. 13,

[10] E. Smith, R. Calandra, A. Romero *et al.*, '3d shape reconstruction from vision and touch,' Jul. 2020.

[11] Y. Cui, S. Schuon, S. Thrun, D. Stricker and C. Theobalt, 'Algorithms for 3d shape scanning with a depth camera,' *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 5, pp. 1039–1050, 2013. DOI: 10.1109/TPAMI.2012.190.

[12] C. K. Rekha, K. Manjunathachari and G. V. S. Rao, 'Speckle noise reduction in 3d ultrasound images — a review,' in *2015 International Conference on Signal Processing and Communication Engineering Systems*, 2015, pp. 257–259. DOI: 10.1109/SPACES.2015.7058260.

[13] V. Kumar, A. Kumar Dubey, M. Gupta, V. Singh, A. Butola and D. Singh Mehta, 'Speckle noise reduction strategies in laser-based projection imaging, fluorescence microscopy, and digital holography with uniform illumination, improved image sharpness, and resolution,' *Optics & Laser Technology*, vol. 141, p. 107 079, 2021, ISSN: 0030-3992. DOI: https://doi.org/10.1016/j.optlastec.2021.107079. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0030399221001675.

[14] P. Craven and G. Wahba, 'Smoothing noisy data with spline functions,' *Numerische mathematik*, vol. 31, no. 4, pp. 377–403, 1978.

[15] M. Berger, A. Tagliasacchi, L. M. Seversky *et al.*, 'A survey of surface reconstruction from point clouds,' *Computer Graphics Forum*, vol. 36, no. 1, pp. 301–329, Jan. 2017, ISSN: 0167-7055, 1467-8659. DOI: 10.1111/cgf.12802. [Online]. Available: https://onlinelibrary.wiley.com/doi/10.1111/cgf.12802 (visited on 28/09/2022).

[16] A. Khatamian and H. R. Arabnia, 'Survey on 3D Surface Reconstruction,' en, *Journal of Information Processing Systems*, vol. 12, no. 3, pp. 338–357, Sep. 2016. DOI: 10.3745/JIPS.01.0010. [Online]. Available: https://doi.org/10.3745/JIPS.01.0010 (visited on 06/09/2022).

[17] F. Bellocchio, N. A. Borghese, S. Ferrari and V. Piuri, *3D Surface Reconstruction*. New York, NY: Springer New York, 2013, ISBN: 978-1-4614-5632-2. DOI: 10.1007/978-1-4614-5632-2. [Online]. Available: http://link.springer.com/10.1007/978-1-4614-5632-2 (visited on 06/09/2022).

[18] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald and W. Stuetzle, 'Surface reconstruction from unorganized points,' p. 8,

[19] P. Crossno and E. Angel, 'Spiraling edge: Fast surface reconstruction from partially organized sample points,' in *Proceedings Visualization '99 (Cat. No.99CB37067)*, San Francisco, CA, USA: IEEE, 1999, pp. 317–538, ISBN: 978-0-7803-5897-3. DOI: 10.1109/VISUAL.1999.809903. [Online]. Available: http://ieeexplore.ieee.org/document/809903/ (visited on 30/09/2022).

[20] A. Alqudah, 'Survey of surface reconstruction algorithms,' *Journal of Signal and Information Processing*, vol. 05, no. 3, pp. 63–79, 2014, ISSN: 2159-4465, 2159-4481. DOI: 10.4236/jsip.2014.53009. [Online]. Available: http://www.scirp.org/journal/doi.aspx?DOI=10.4236/jsip.2014.53009 (visited on 27/09/2022).

[21] J. o'Rourke *et al.*, *Computational geometry in C*. Cambridge university press, 1998.

[22] N. Am and M. Bernt, 'Surface reconstruction by voronoi filtering,' p. 10,

[23] MathWorks, *Alphashape*, https://se.mathworks.com/help/matlab/ref/alphashape.html, Accessed 19-12-2022.

[24] Jakub Konka, *Powercrust*, https://github.com/kubkon/powercrust, Accessed 19-12-2022.

[25] H. Edelsbrunner and E. Mucke, 'Three-dimensional alpha shapes,' *ACM Transactions on Graphics*, vol. 13, Oct. 1994. DOI: 10.1145/147130.147153.

[26] N. Amenta, M. Bern and M. Kamvysselis, 'A new voronoi-based surface reconstruction algorithm,' in *Proceedings of the 25th annual conference on Computer graphics and interactive techniques - SIGGRAPH '98*, Not Known: ACM Press, 1998, pp. 415–421, ISBN: 978-0-89791-999-9. DOI: 10.1145/280814.280947. [Online]. Available: http://portal.acm.org/citation.cfm?doid=280814.280947 (visited on 29/09/2022).

[27] K. Paulsen, S. Poplack, Dun Li, M. Fanning and P. Meaney, 'A clinical prototype for active microwave imaging of the breast,' *IEEE Transactions on Microwave Theory and Techniques*, vol. 48, no. 11, pp. 1841–1853, Nov. 2000, ISSN: 00189480. DOI: 10.1109/22.883861. [Online]. Available: http://ieeexplore.ieee.org/document/883861/ (visited on 06/09/2022).

[28] S. Lee and S. Kwon, *Recent Advances in Microwave Imaging for Breast Cancer Detection*, English, 2016. [Online]. Available: https://www.researchgate.net/publication/312244309_Recent_Advances_in_Microwave_Imaging_for_Breast_Cancer_Detection (visited on 05/09/2022).

[29] B. M. Moloney, P. F. McAnena, S. M. Abd Elwahab *et al.*, 'Microwave Imaging in Breast Cancer – Results from the First-In-Human Clinical Investigation of the Wavelia System,' en, *Academic Radiology*, vol. 29, S211–S222, Jan. 2022, ISSN: 10766332. DOI: 10.1016/j.acra.2021.06.012. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1076633221002993 (visited on 07/09/2022).

[30] T. C. Williams, J. Bourqui, T. R. Cameron, M. Okoniewski and E. C. Fear, 'Laser surface estimation for microwave breast imaging systems,' *IEEE Transactions on Biomedical Engineering*, vol. 58, no. 5, pp. 1193–1199, 2011. DOI: 10.1109/TBME.2010.2098406.

[31] D. Kurrant, J. Bourqui and E. Fear, 'Techniques for breast surface reconstruction with applications,' in *2016 17th International Symposium on Antenna Technology and Applied Electromagnetics (ANTEM)*, 2016, pp. 1–3. DOI: 10.1109/ANTEM.2016.7550205.

[32] S. Winkelbach, S. Molkenstruck and F. Wahl, 'Low-cost laser range scanner and fast surface registration approach,' vol. 4174, Sep. 2006, pp. 718–728, ISBN: 978-3-540-44412-1. DOI: 10.1007/11861898_72.

[33] J. M. Felicio, J. M. Bioucas-Dias, J. R. Costa and C. A. Fernandes, 'Microwave breast imaging using a dry setup,' *IEEE Transactions on Computational Imaging*, vol. 6, pp. 167–180, 2020, ISSN: 2333-9403, 2334-0118, 2573-0436. DOI: 10.1109/TCI.2019.2931079. [Online]. Available: https://ieeexplore.ieee.org/document/8777154/ (visited on 05/09/2022).

[34] D. Kurrant, J. Bourqui and E. Fear, 'Surface estimation for microwave imaging,' *Sensors*, vol. 17, no. 7, p. 1658, 19th Jul. 2017, ISSN: 1424-8220. DOI: 10.3390/s17071658. [Online]. Available: https://www.mdpi.com/1424-8220/17/7/1658 (visited on 06/09/2022).

[35] *Yumi irb 14050*, ABB, Mar. 2021.

[36] *Arduino uno*. [Online]. Available: https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf.

[37] *Optoncdt 1302*. [Online]. Available: http://www.micro-epsilon.pl/download/man--optoncdt-1302--en.pdf.

[38] *Prusa i3 mk3*. [Online]. Available: https://www.electronicsdatasheets.com/manufacturers/prusa/parts/original-prusa-i3-mk3#datasheet.

[39] *Matlab*. [Online]. Available: https://se.mathworks.com/products/matlab.html.

[40] *Solidworks*. [Online]. Available: https://www.solidworks.com/.

[41] *Visual studio code*. [Online]. Available: https://code.visualstudio.com/.

[42] Baumer, *Baumer om30-l0550*. [Online]. Available: https://www.elfa.se/Web/Downloads/_t/ds/OM30-L0550.HV.YIN_eng_tds.pdf.

[43] *"the perfect breast": Measuring cosmetic outcomes after breast-conserving therapy*. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405857220300966.

[44] N. R. COUNCIL, *MICROWAVE PROCESSING OF MATERIALS*, 2nd ed. National Academies Press Washington, DC, 2008, p. 27.

[45] Micro-Epsilon, *Instruction manual optoncdt 1302*, Micro-Epsilon. [Online]. Available: http://www.micro-epsilon.pl/download/man--optoncdt-1302--en.pdf.

[46] Micro-Epsilon, *Download for micro-epsilon products*, Micro-Epsilon. [Online]. Available: https://www.micro-epsilon.com/download/software/sensorTool.exe.

[47] Putty, *Download putty*, PuTTY. [Online]. Available: https://putty.org/.

[48] Realterm, *Realterm: Serial terminal*, sourceforge. [Online]. Available: https://realterm.sourceforge.io/.

[49] Micro-Epsilon, *Download for micro-epsilon products*, Micro-Epsilon. [Online]. Available: https://www.micro-epsilon.com/download/software/MEDAQLib.zip.

[50] python, *Python is a programming language that lets you work quickly and integrate systems more effectively.* python. [Online]. Available: https://www.python.org/.

[51] MathWorks, *Curve fitting toolbox*, https://se.mathworks.com/products/curvefitting.html, Accessed 19-12-2022.

[52] Zachary Taylor, *Find 3d normals and curvature*, https://se.mathworks.com/matlabcentral/fileexchange/48111-find-3d-normals-and-curvature, Accessed 19-12-2022.

[53] K. Hoang, *Ethernet_generic*, GitHub. [Online]. Available: https://github.com/khoih-prog/Ethernet_Generic.

[54] T. Strohmer, T. Binder and M. Sussner, 'How to recover smooth object boundaries in noisy medical images,' in *Proceedings of 3rd IEEE International Conference on Image Processing*, vol. 1, Lausanne, Switzerland: IEEE, 1996, pp. 331–334, ISBN: 978-0-7803-3259-1. DOI: 10.1109/ICIP.1996.559500. [Online]. Available: http://ieeexplore.ieee.org/document/559500/ (visited on 02/11/2022).

[55] *Experimental comparison of selected triangulation and tof optical distance sensors*. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-031-03502-9_29.

[56] life.augmented, *Vl6180*. [Online]. Available: https://docs.rs-online.com/aa8b/A700000008302836.pdf.

[57] life.augmented, *Vl53l0*. [Online]. Available: https://www.techtarget.com/searchnetworking/definition/TCP.

[58] SHARP, *Sharp gp2y0e02b*. [Online]. Available: http://www.socle-tech.com/doc/IC%5C%20Channel%5C%20Product/Sensors/Distance%5C%20Measuring%5C%20Sensor/Analog%5C%20Out(Including%5C%20I2C%5C%20output)/GP2Y0E02B_SPEC.pdf.

[59] SICK, *Sick wtb16p*. [Online]. Available: https://cdn.sick.com/media/pdf/9/39/239/dataSheet_DT20-P224B_1040405_en.pdf.

[60] Creality, *Creality ender-2 pro*. [Online]. Available: https://filament2print.com/es/index.php?controller=attachment&id_attachment=1100.

[61] Aratron, *Rk lightunit*. [Online]. Available: https://www.aratron.se/wp-content/uploads/2015/05/Aratron_linjarenheter_rk-lightunit.pdf.

[62] VEVOR, *Linear actuator*. [Online]. Available: https://eur.vevor.com/linear-actuators-c_11633/38-61cm-electric-recliner-motor-replacement-kit-lift-chair-dc-motor-actuator-p_010497223871?v_tag=cdc5c770-9014-11ed-9ea4-73f528fd86d3.1.

[63] SNAPMAKER, *A350- linear module*. [Online]. Available: https://www.3dprima.com/spare-parts-accessories/manufacturer/snapmaker/snapmaker-linear-module-a350_26354.

Appendix

Algorithm 1 The main function

```
MaxHeight ← 195
init()                                   ▷ Arduino function for starting timer, ADC etc.;
Enable inputs and outputs
Serial baud rate ← 115200
calibrate()                              ▷ This function is described in Algorithm 2
IP address ← 10.132.158.192
MAC address ← DE − AD − BE − EF − FE − 0B
Port ← 1001
Start Ethernet communication
while True do
    if Received Ethernet package then
        Split message at ':' into array
        if array[0] == "goTo" then
            goTo(array[1])               ▷ This function is described in Algorithm 5
            return CurrentHeight in millimetres to message IP and port 1885
        else if array[0] == "moveDown" then
            moveDown(array[1])           ▷ This function is described in Algorithm 3
            return CurrentHeight in millimetres to message IP and port 1885
        else if array[0] == "moveUp" then
            moveUp(array[1])             ▷ This function is described in Algorithm 4
            return CurrentHeight in millimetres to message IP and port 1885
        else
            Invalid command, return "Invalid command" to message IP and port 1885
        end if
    end if
    Wait for serial to finish
end while
```

Algorithm 2 The calibrate function

```
while Micro-Switch == 0 do                          ▷ Go to bottom
    moveDown(1)
end while
while Micro-Switch == 1 do          ▷ Go up at least 150 steps above switch trigger point
    i ← 0
    while i < 150 do
        moveUp(1)
        i++
    end while
end while
while Micro-Switch == 0 do                          ▷ Go to bottom slowly
    moveDown(1)
    delay 1 ms
end while
CurrentHeight ← 0
```

Algorithm 3 The moveDown function

Input: StepsToMove
Set stepper motor direction
$i \leftarrow 0$
while i < StepsToMove **do**
    Move one step
    Update CurrentHeight
    i++
end while

Algorithm 4 The moveUp function

Input: StepsToMove
Set stepper motor direction
$i \leftarrow 0$
while i < StepsToMove **do**
    Move one step
    Update CurrentHeight
    i++
end while

Algorithm 5 The goTo function

Input: desiredHeight
while CurrentHeight < desiredHeight **AND** CurrentHeight <= MaxHeight **do**
    moveUp(1)
end while
while CurrentHeight > desiredHeight **AND** CurrentHeight >= 0 **do**
    moveDown(1)
end while

Algorithm 6 The main function

```
Enable inputs and outputs
Serial baud rate ← 115200
IP address ← 10.132.158.190
MAC address ← DE − AD − BE − EF − BE − 0A
Port ← 1000
Start Ethernet communication
while True do
    if Received Ethernet package then
        if message == "Get" then
            laserReading ← Serial.readBytes
            Add null at end of laserReading

            i ← 0
            while i < length of laserReading do    ▷ Replace all chars before and \r with spaces
                if laserReading[i] == '\r' then
                    j ← 0
                    while j <= i do
                        laserReading[j] ← ' '
                        j++
                    end while
                    break loop
                end if
                i++
            end while

            i ← 0
            while i < length of laserReading do      ▷ Replace all chars after and \r with spaces
                if laserReading[i] == '\r' then
                    j ← i
                    while j < length of laserReading do
                        laserReading[j] ← ' '
                        j++
                    end while
                    break loop
                end if
                i++
            end while

            i ← 0
            j ← 0
            while i < length of laserReading do              ▷ Remove all spaces by shifting left
                if laserReading[i] != ' ' then
                    laserReading[j] ← laserReading[i]
                    j++
                end if
                i++
            end while
            laserReading[j] ← null

            return laserReading to message IP and port 1884
        end if
    end if
end while
```

Algorithm 7 The main function of the Python script

```
Create sockets for receiving and sending to Arduino
attempts ← 0
while True do
    allReadings ← empty array
    j ← 0
    while j < 8 do
        Wait 0.01 seconds
        Send "Get" to Arduino
        append data from Arduino into allReadings
        attempts++
        if attempts == 20 then                          ▷ This prevents infinite loops
            Return "no object detected"
        end if
        if all values of allReadings is the same then
            if allReadings == 16370 then
                Return "no object detected"
            else if allReadings == 16372 then
                Return "too close to sensor"
            else if allReadings == 16374 then
                Return "too far from sensor"
            else if allReadings == 16376 then
                "target can not be evaluated"
            else
                Return ((((allReadings*(1.02/4096))-0.01)*200)+60)    ▷ This returns the reading
into millimetres
            end if
        end if
        j++
    end while
end while
```

Algorithm 8 Example of how to send and receive Ethernet data using Ethernet_Generic

```
Enable Ethernet pins
Init Ethernet
Set MAC address
Set IP address
Set Port
Push MAC and IP to shield
Create socket at Port
Read packet on socket
Begin packet to return IP and return Port
Write data to packet
Send packet
```

Bill Of Materials (BOM) - Aref

Name	Discription	Total Price	Quantity	Manufacture
PTEG plastic	Printer filament	- SEK	1.5Kg	Prusa
PLA plastic	Printer filament	- SEK	2.5kg	Prusa
PP-pipes	40mmx2mmx2m white	399.6 SEK	4	Biltema
Screws	M3x18mm "box"	65 SEK	1	HECO
Loctite 406	Super Glue 20 g	506 SEK	1	Loctite
YuMi single arm	Robotics arm	- SEK	1	ABB
OmniCore	Robot controller	- SEK	1	ABB
HP Z	Stationary computer	- SEK	1	HP
LRS-100	Power Supply,Ac to 24V dc	514 SEK	1	MW
MP-K78Lxx-1000R3	DC to DC converter 24 to 5v	25.11 SEK	1	MulticompPro
MP-K78Lxx-1000R3	DC to DC converter 24 to 12v	25.11 SEK	1	MulticompPro
Pawer cable	C13 to EU plug	- SEK	1	-
Power train cables	Sigle power cables 2mm	- SEK	5m	-
Power train cables	Jumper weirs 0.5mm	- SEK	15	-
OPTON-CDT130	Distance laser sensor	- SEK	1	Micro-Epsilon
12 Pins cable	Sensors cable	- SEK	1	Micro-Epsilon
DFR0259	RS482 communication shield	114 SEK	1	DFROBOT
DFR0125	Ethernet shield	338 SEK	2	DFROBOT
A000066	Microcontroller Arduino UNO	531.9 SEK	2	Arduino
DRI0043	Stepper Motor Driver	908 SEK	1	DFROBOT
TL-SG108	8-portars gigabit-switch	349 SEK	1	TPlink
Ethernet cables	2 to 5 m Ethernet cables	- SEK	4	-
Round Nut	Round Nut For Lead Screw, 10mm	565 SEK	1	RS Pro
Lead Screw	Lead Screw, 10mm	148.73 SEK	1	RS Pro
Long Steel Closed Bush Shaft	Long Steel Closed Bush Shaft, 8mm	779.4 SEK	6	Bosch Rexroth
RJUM-01-08	Bearing with 16mm Outside Diameter	947.46 SEK	6	Igus
Coupling	Coupler 19.1mm Outside Diameter	144.44 SEK	1	Ruland Jaw
Coupling	Coupler 25.4mm Outside Diameter	182.31 SEK	1	Ruland Jaw
Flexible Coupling	Coupling Flector 19.1mm	54.63SEK	1	Ruland Jaw
Flexible Coupling	Coupling Flector 25.4mm	60.385 SEK	1	Ruland Jaw

Table 12: The project's bil of materials. (-) means that the product was given for free.

Distance sensor Study-Aref

The distance sensor was selected based on a few factors. The accuracy of the sensor, the working range, the measuring speed, and the cost were the project's most crucial variables. These six sensors were found by looking through the available distance sensors on the market and were a match for the project. In this project, a high-accuracy distance sensor with a sizable working range and high measuring-speed was required. Because the Triangulation method is more useful for measuring a small range with high accuracy, it was more practical to use it for this project than the sensors that use the TOF measuring method. TOF can measure farther away objects with less accuracy. [55]

Name	Accuracy	Price	Working rang	Measuring speed	Method
Life-augmented VL6180x	3%	53.32 SEK	0 to 620 mm	200 ms	Time of Flight
Life-augmented VL5310x	3%	22.02 SEK	0 to 1200 mm	200 ms	Time of Flight
SHARP GP2Y0E02B	0.3 mm	107 SEK	40 to 500 mm	40 ms	Triangulation
SICK DT20 Hi	>0.125 mm	12151 SEK	50 to 1000 mm	2.5 ms	Triangulation
Baumer OM30-L0550	2 to 86 um	8072 SEK	50 to 550 mm	0.4 ms	Triangulation
Micro-Epsilon OPTON-CDT1302	40 to 100 um	8000 SEK	60 to 260 mm	0.2 s	Triangulation

Table 13: The distance laser sensor study table

Baumer OM30-L0550 was the greatest option on this list, but the cost was a bit expensive. Instead, the supervisor recommended and provided **Micro-Epsilon OPTON-CDTI1302**[37] for free. The **Micro-Epsilon OPTON-CDTI1302**[37] distance sensor was employed in this research since it is sufficiently accurate and has a working range of 200 mm. **VL6180x** [56], **VL5310x**[57], and **SHARP GP2Y0E02B**[58] were rejected because their accuracy was below that needed for this project, which was >0.1 mm. Due to its high cost, the **SICK DT20 Hi** [59] sensor was also rejected.

Sender arm Study-Aref

Name	Stroke lengh	Extented legth	Accuracy	Price
CREALITY ENDER-2 PRO	400mm	600 mm	High	1995 SEK
aratron RK LightUnit	300mm	520 mm	High	Unknown
VEVOR Linear Actuator Replace-ment Kit	230 mm	610mm	Low	650.65 SEK
SNAPMA-KER LINEAR MODULE - A350	360 mm	670 mm	High	1629 SEK
Self-designed 4	395	630	High	2882.6 SEK

Table 14: The sender robotics arm study table

On the basis of a few criteria, the Sender robotics arm (SRA) was chosen. The most crucial elements were size, precision, consistency, and arm cost. **The CREALITY ENDER-2 PRO** [60] was intended to be modified into a 3-axis robot arm that could move on the x, y, and z axes, however the modification procedure took a lengthy time, causing the project's deadline to be missed. In order to prevent collisions between the YuMi robotics arm and the SRA, the height of the 3D printer is higher than what is required in this project, Therefore this suggestion has been rejected.
To be utilized as a Sender robotics arm, **the Aratron linear RK LightUnit** [61] needs to be upgraded, and an actuator must be attached to the system. It will take more time and effort to modify this rotation-to-linear transformer axis than to design a robotic arm from scratch. In addition to not having an available mechanical connector between the actuator and the axis and the cost of this component is also uncertain. Consequently, this suggestion has been rejected.
VEVOR Linear Actuator Replacement Kit [62] is a linear servo with poor accuracy, making it possibly the worst choice for holding a microwave transmitter. Consequently, this option has been disregarded.
SNAPMA-KER LINEAR MODULE - A350 [63] is one of the greatest suggestions since it is simple to use, an integrated actuator complete linear module, and simple to mount. The length of the stroke is longer than what the system requires, which is the only factor keeping it from being accepted. **Test of the parameters**

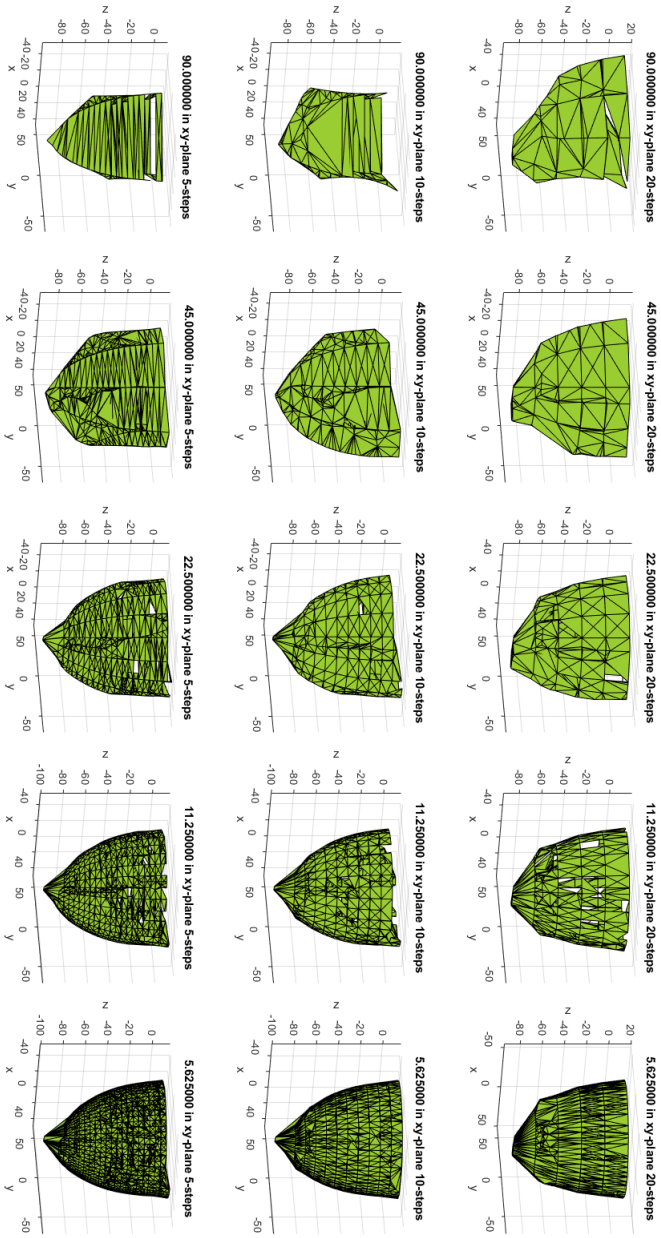


Figure 41: Picture depicting; the result from having the robot scan in different azimuth angle steps and elevation steps. On the sample points, powercrust conducted. The upper left describes the step size of 90 ° and continues down with decreasing the value with half for every picture along the row. For every angle the step-size in change to 20, 10 and five in height (Smaller step in the elevations is equal to more samples)