Enabling Robotic Support for Novice Creators

UHasselt

Bram van Deurzen

06 June 2018

Abstract

Designing and building a 3D model requires experience with CAD applications, knowledge on the build process and the skill to assemble it. This makes it harder for novice users to create something personalised that is useful to them.

The goal of this thesis is to use a robotic arm to assist a novice user during the assembly of an object. The robotic arm holds the next part of the model in the required position, allowing the user to attach it using both hands. The robotic arm functions as a third hand for the user. To facilitate this, a construction kit is designed with which the user can build the model. It consists of plates that can be attached to each other using corner pieces, and bolts and nuts. The construction kit makes it easier to assemble the model and is optimised to be used with a robotic arm.

The novice user can design a model with the CAD design tool created for this project. It consists of a simple user interface, with which the user can place the construction kit parts into the tool to create a 3D model. When the user wants to build his model, the tool is used to calculate the position in which the robotic arm should hold each part. These positions are used to control the robotic arm during the build process with the user.

The design tool is implemented and tested with two robotic arms: the Arduino Braccio and the Commonplace Robotics Mover6. Both are used to aid a novice user during the build process.

The three hand fabrication is achieved, which empowers a novice user to design and build a 3D model. The user could design something that he needs instead of relying on a store selling it.

Acknowledgements

I would like to express my deep gratitude to Prof. Dr Kris Luyten, Prof. Dr Raf Ramakers and Dr Jan Van den Bergh, my thesis supervisors at Hasselt University, for their guidance, enthusiastic encouragement and useful critique on my thesis work. Prof. Luyten's continuous support and drive throughout the year helped shape my thesis. I could always rely on him for guidance. The insights provided by Prof. Ramakers were of great help on the design tool and the position calculation. The expertise on human robot collaboration provided by Dr. Van den Bergh was of great help for the robot control.

I would also like to thank Tom Veuskens, student at Hasselt University, and Danny Leen, Phd student at LUCA school of arts, who were involved during the UIST Student Innovation Contest. The groundwork for my thesis was created with their help.

I would also like to thank the participants of my user study. Without their participation and input, the study could not have been successfully conducted.

Finally, I must express my very profound gratitude to my parents and to my girlfriend for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis. Special thanks to my girlfriend for grammatical support on this thesis text.
This accomplishment would not have been possible without them. Thank you.

Bram van Deurzen

Contents

Chapter 1

Introduction

I start this thesis with a short introduction on human-robot collaboration. Next, I introduce three hand fabrication and my contribution. I end with an overview of this thesis.

1.1 Human-robot collaboration

When we think of robots and human workers that are building an object, we think of an industrial robot behind a safety cage that does the heavy work and a human worker that does the finishing touches. For the human worker this often means working in sub optimal conditions, which results in workplace injuries [20]. A robot is useful to complete repetitive and heavy duty tasks, but for every change in the product the robot has to be reprogrammed. A human worker offers dexterity, flexibility and problem-solving to the build process. Human-robot collaboration (HRC) tries to combine the automation benefits of robots with the flexibility and adaptive nature of humans. This by using collaborative robots, the so-called cobots. The human worker and the cobot should work together in the same workplace, instead of being caged off from each other [20].
Human-robot collaboration is not only a new trend in the research community, but also in the industry. It can be seen as part of the fourth industrial revolution. The so-called Industry 4.0 tries to combine smart and interconnected machines and sensors in a factory, although the definition of it is not well defined [12]. One of the goals of Industry 4.0 is to be adaptable to changes and variations in the products that are made. Human-robot collaboration supports this by using the flexibility of humans to adapt the process to the instructions given by the system.

Working together with a robot does not come without challenges and difficulties. The starting point is to establish a good communication protocol among the human and the robot, which the research of Makrini et al. [20] focuses on. Another study on the challenges in shared-environment human-robot collaboration by Hayes B. and Scassellati B. [10] discusses the following four challenges, which they say are still underexplored in research at the moment of writing in 2013.

- How can cobots enable and facilitate bi-directional intent recognition?
- How does a cobot know what roles to take when working with human teammates?

- How does a cobot know when to trade off task execution optimally for co-worker preferences?
- How does a cobot self-evaluate during live collaboration?

The focus of this thesis is not on researching these challenges or comparing different types of communication protocols. My goal is to test and compare the capabilities and limitations of various types of small-scale robots, which aid a novice user in designing and building a 3D model.

1.2 Three hand fabrication

The goal of my thesis is to enable robotic support for novice creators, by enabling a robotic arm to be used as a third hand. The result of this thesis is shown as a storyboard in figure 1.1, which visualises the process from idea to physical model. To enable three hand fabrication with a robotic arm I contribute the following, which make it possible for a novice user to design a 3D model and build it.

- A construction kit: the basic parts that are used to simplify the build process and design. They are created using a laser cutter.
- A software design environment: a 3D CAD environment, in which the construction kit parts can be placed and combined to design the desired model (figure 1.1 step 1).
- Automatic robot programming: the model is used to automatically create robot programs.

The 3D model the user designs is built step by step. First, the robot holds the next plate in the desired position against the model. Secondly the user fastens the plate with bolts and nuts onto the model using 3D printed corner pieces. Because the robot holds the plate, the user can use both hands to fasten it (figure 1.1 step 4). The design tool shows the current step in the build process and gives instructions to the user.
The robotic arms with which I test the three hand fabrication are the Arduino Braccio [2] and the Commonplace Robotics Mover6 [5]. The Braccio has 5 degrees of freedom (DoF), the number of joints that defines its position. The Mover6 has 6 degrees of freedom, which makes the robot more flexible and enables it to reach more positions. Both have limited reach and can only lift a small amount of weight (100-200 gram). For this reason, I created a platform on which the model that the user is building is placed. This platform can move in linear direction and rotate, which increases the flexibility of the robot.

The thesis is organised as follows: Chapter 2 gives an overview of related work. In chapter 3, I discuss the software design environment, construction kit, and the algorithm that is used to automatically program the robotic arm. In chapter 4, I give an overview of the implementation with the Braccio robot arm, the limitations and the benefits. Chapter 5 discusses the implementation with the Mover6 robot arm, the improvements made to the tool, the limitations and the benefits. Chapter 6 discusses the small formative study that was conducted as the final part of my thesis. In chapter 7, I list possible extensions that can be made to the design tool. I conclude this thesis with a retrospect on the implementations and the tool.

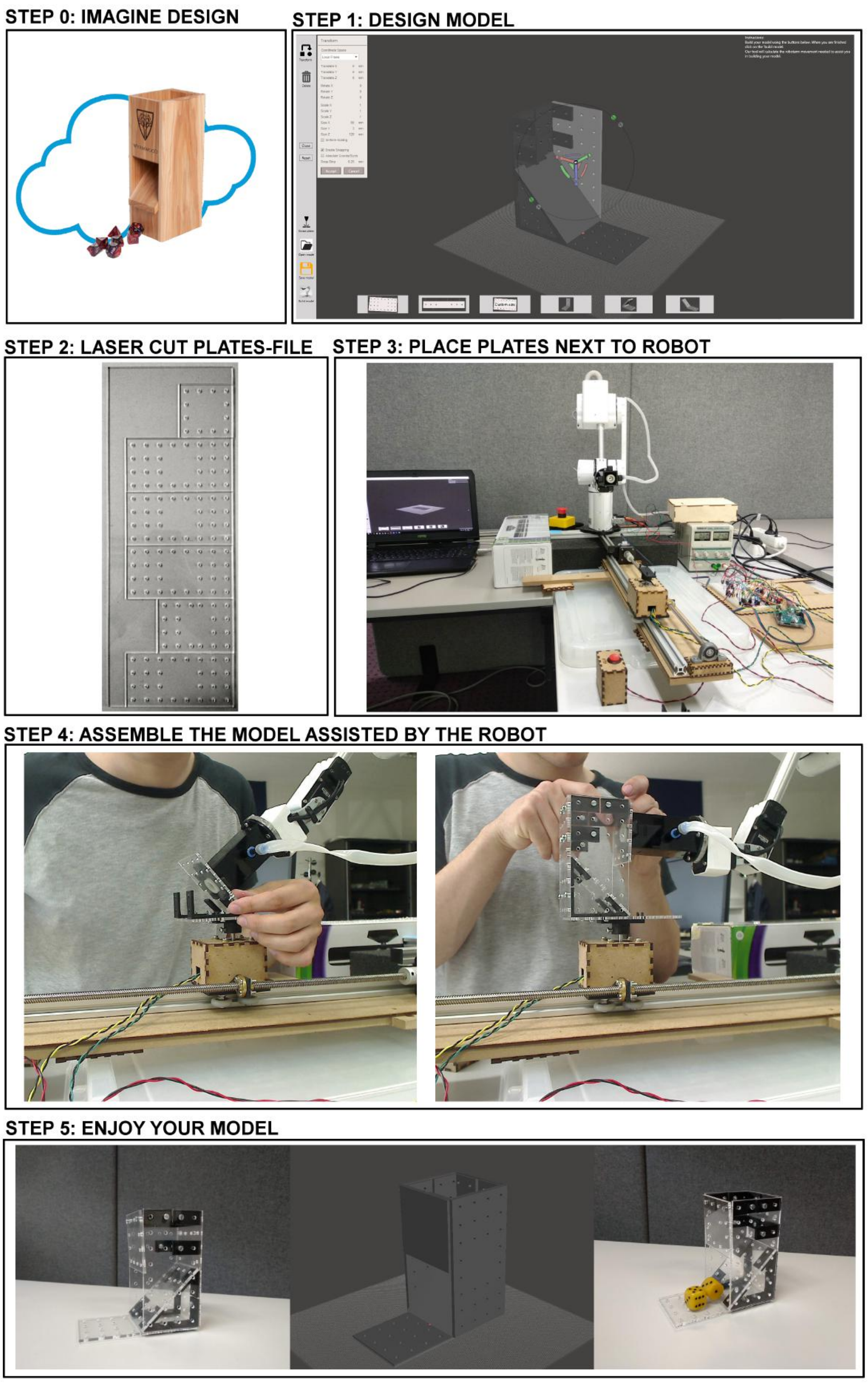

STEP 0: IMAGINE DESIGN
STEP 1: DESIGN MODEL
STEP 2: LASER CUT PLATES-FILE
STEP 3: PLACE PLATES NEXT TO ROBOT
STEP 4: ASSEMBLE THE MODEL ASSISTED BY THE ROBOT
STEP 5: ENJOY YOUR MODEL

Figure 1.1: Storyboard that shows the process of designing a dice tower with the software design environment, and building it assisted by a robotic arm.

Chapter 2

Related work

I start this chapter with an overview of different types of robots that are used in research and in the industry. Some of the research that I discuss in this chapter uses these cobots. Next I give an overview of robot programming and discuss research on robot programming based on CAD applications. I finish this chapter with a discussion of research on collaborative human-robot assembly tasks and design tools for novice users.

2.1 Different types of robots

Different types of robots are used in the industry and in research. According to ISO 8373 [17] a robot is defined by "the ability to perform intended tasks based on the current state and sensing, without human intervention". In this section, I only expand on collaborative robots and service robots because these are important for my thesis. For the sake of completeness, the definition of an industrial robot as defined by ISO 8373 [17] is: "an automatically controlled, reprogrammable, multi-purpose manipulator programmable in three or more axes, which can be either fixed in place or mobile for use in industrial automation applications". This report [14] by the International Federation of Robotics gives the definition and classification of industrial robots.

2.1.1 Collaborative robots

I discuss three different cobots that are used in both research and the industry. Both the Universal Robots UR10 and Rethink robotics Baxter are used in research related to my thesis. The KUKA LBR iiwa was the first robot developed that is approved for human-robot collaboration [18]. I briefly state the capabilities of each robot and highlight their the unique features.

The KUKA LBR iiwa cobot [18] (figure 2.1) is available in two versions with payload capacities of 7 and 14 kilograms. Both versions have 7 degrees of freedom and a reach of 800-820mm. The cobot uses torque sensors to detect contact. The joints instantly reduce their level of force and speed to prevent injuries. The torque sensors can also be used to interact with the robot through gestures, for example by touch. The housing is designed to eliminate all crushing and shearing hazards of the robot.

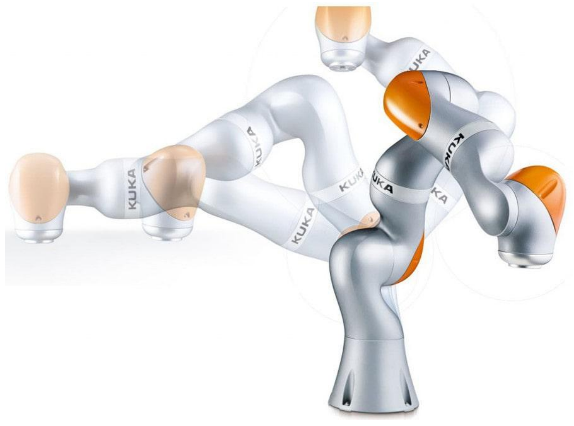


Figure 2.1: The KUKA LBR iiwa cobot

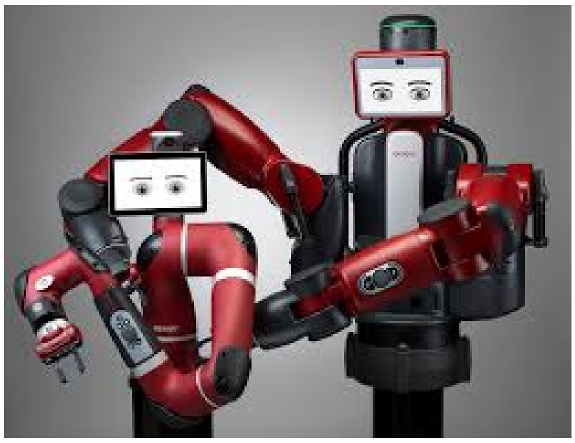


Figure 2.2: The Rethink Robotics Sawyer on the left and the Baxter on the right

Rethink robotics [28] designed two types of cobots (figure 2.2): the Baxter, a two armed robot, and Sawyer, a one armed robot which has a smaller footprint and longer reach. Both robots feature a 7 degree of freedom robot arm with built-in force sensing technologies. A unique feature of the robots is that they both have a built-in display that functions as a head for the robot.

Universal Robots [34] sells three different types of cobots with payload capacities of 3, 5 and 10 kilograms (figure 2.3). All versions have 6 degrees of freedom. The UR3 is the only version that supports infinite spin on its last joint, allowing the use for screwing applications without a special end effector. All other joints on the robots support 360-degree rotation.

2.1.2 Service Robots

A service robot is defined as ”a robot that performs useful tasks for humans or equipment excluding industrial automation applications” according to ISO 8373 [17]. It can be classified further according to the application area in which the service robot is used. The main difference is in personal/domestic use or professional use. This report [15] of the IFR highlights all the different classifications.
The two robotic arms that I use during my thesis are service robots.

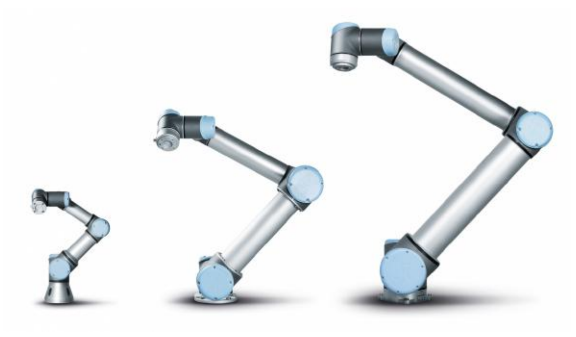


Figure 2.3: The Universal Robots UR3, UR5 and UR10

2.2 Robot programming

Robots require programming, which is time consuming and requires skilled workers. To ease this process, CAD applications are used in research as a programming aid [9]. Before I expand upon this idea, here are some definitions that are used when describing robot programming.

2.2.1 Definitions

Robotic tasks can be divided into two methods of trajectory: point-to-point and continuous point [9]. Robot programming can be done in two different ways: online or offline programming.

Point-to-point method of trajectory

The point-to-point method requires the operator to define the key points that the end effector must reach. The important issue is to ensure collision-free movement. Knowledge of the rules of motion programming is often enough to do the job. An example of this method is handling tasks, e.g. picking up a part [9].

Continuous point method of trajectory

The continuous point method requires precise definition of the path followed by the end effector. A large amount of points is required to define the path. Precise knowledge of the path is required to define it correctly. An example of this method is welding or painting [9].

Online programming

Online programming is writing a program directly on the robot system or using the robot itself to write the program. The most used way is the teaching process, through the use of the robot teach pendant [25]. A teach pendant is a controller for the robot, which can be used to move the robot to the required positions and save them as points in the program.

Offline programming

With offline programming the robot is programmed from a computer. The main advantages are that you do not need the robot to write the program, and the increased safety. The main disadvantages are the necessity for the proper software, training and the difference between the simulation and the real machine. The robot needs to be calibrated and tested in the real environment before the program can be put to use [9].

2.2.2 CAD-based robot programming

Research is done in ways on which a CAD application can be used to do offline programming with a robot. The goal is to use the CAD drawing as a base to determine characteristic points of the robot path, which makes it easier to define a continuous point of trajectory [9]. A second goal is to develop methodologies that help users with programming a robot in an intuitive way, with a high level of abstraction. The CAD-based system allows users with basic skills in CAD and robot programming to generate offline robot programs [26]. The following are research projects in which CAD-based robot programming is proposed.

A CAD-based system proposed by Neto P. et al [26][24] uses Autodesk Inventor as the CAD system, which lets the user generate a robot program by drawing the desired robot path. The end effector position and rotation have to be defined as well, by placing simplified tool models along the robot paths. Interpolation is implemented on the end effector path to smooth out the movement. The user defines areas of risk where the interpolation is applied. The information needed is automatically extracted, analysed and converted into robot programs.
Sensor data can be used to track the movement in real time and make adjustments on the fly. The proposed tool is tested with two different experiments in which offline programs are created. The results of the tests show that the system is easy to use and within minutes a user can generate a robot program.

A second platform offers an intuitive and convertible environment for designing and simulating robotised tasks, named IRoSim (Industrial Robotic Simulation Design Planning and Optimisation platform), presented by Baizid et al. [4]. The platform is based on Solidworks as the CAD system. It includes various 3D models that are essential for developing any robotised tasks including different types of robotic arms.
Besides aiding in the creation of the robotic program, the platform can be used to check the reachability of the end effector, simulate the motion and validate the trajectory to avoid possible collisions. These can be used for time optimisation and collision avoidance of the robotic task designed with the platform.

The two proposed systems still require the user to program the robot, which requires them to have a basic understanding of how robot programming works. The software design environment which I propose in this thesis however, does not require any robot programming. The 3D model that is designed by the user in the CAD system is used to program the robot. This should allow the software design environment to be used by novice users who do not have any knowledge about robot programming.

2.3 Collaborative human-robot assembly tasks

Research on collaborative human-robot assembly tasks explores techniques for humans and robots to collaborate.

The design of a collaborative architecture for human-robot assembly tasks, is studied by Makrini et al. [20]. The proposed architecture is tested by using a Baxter robot to guide a user through the assembly of a box. The robot holds the appropriate part, while the human user assembles the plate correctly on the semi-assembled box by screwing them together.
Gesture recognition is performed by a Kinect v2 camera, to detect hand waving and the thumbs up gesture. The researchers chose gesture recognition over voice recognition because a factory is too loud. Facial recognition with the Kinect is used to determine if a user is permitted to interact with the robot. During the assembly, the camera in the arm of the Baxter checks if all the screws are screwed in. Human-like robot behaviour is provided with social cues such as head nodding/shaking and eye gaze. The screen on the robot displays eyes that look at the end effector that is moving. Text displays are used to communicate or ask information from the user.

The work of Autodesk Research on crowdsourced fabrication [19], uses a robotic arm to aid volunteers with the execution of a complex filament winding procedure. Without the robotic arm it would not be possible for an untrained volunteer to create it. The result of the filament winding procedure is one module. These modules were then combined together to form a bamboo pavilion. The worker places the components into the end effector of the robotic arm (a Universal Robotics UR10), whereupon it executes the complex procedure. Each volunteer creates a part of the pavilion. The intelligent construction space guides the users, in combination with the foreman engine which coordinates the overall build process. Each module is fitted with connector nodes, which contain a RGB LED. The volunteer places the module in the correct spot, highlighted by the colour of the LED's in the modules. Each volunteer is given a smartwatch that shows non-intrusive instructions to the user. The smartwatch is connected to a phone in the tool belt that was given to each volunteer. The tool belt contains a Kontakt.io iBeacons to track the location of the workers between the stations in the workspace.

Both these tasks show the potential of human-robot collaboration.
The robotic arm that is used to test the collaborative architecture, shows how a robot can guide and assist a user through an assembly process. This decreases the chance of faults during the assembly, while the human worker can execute the tasks which would require reprogramming of the robot. For example when different sizes of boxes should be assembled.
The robotic arm used in the research on crowdsourced fabrication executes the complex filament winding procedure, which the volunteer could not do. The volunteer uses the result of the robotic task to build a module, which becomes part of the pavilion. Building the pavilion without the help of the volunteers would require mobile robots with a higher reach to attach each module to the pavilion.

The goal of my thesis is to use a robotic arm as an assistant during the assembly of a model, which the user designed using a software design environment. The robotic arm assists the user by holding parts of the model in the desired position, allowing the user to attach it using both hands.

2.4 Design tools for novice users

Interactive design tools that are created to help novice users during the design process:

A novel interactive tool for garment design that enables bidirectional editing between 2D patterns and 3D high-fidelity simulated draped forms, is presented by Umetani et al. [33]. Artists can interactively edit 2D pattern designs and immediately observe how these changes affect the 3D form. With the use of a template, a novice user can easily design a piece of clothing without needing the necessary technical background.

Plushie [23], an interactive system that allows non-professional users to design their own plush toys, is introduced by Mori Y. and Igarashi T. The user creates the plush toy model by drawing its desired silhouette and they can edit the model with a simple sketching interface. The resulting model is associated with a 2D pattern and a 3D model that is the result of physical simulations that mimic the inflation effect caused by stuffing.

A novel computation technique that can model the aerodynamics of free-flight gliders is developed by Utemani et al. [32]. The model is used to automatically optimise a glider to maximise the flight-ability. This technique is used in the design tool to optimise the wing configuration of the users' design. The design and optimisation can be quickly iterated to reach a shape that is both aesthetically appealing and aerodynamically effective. The tool enables users to create all kinds of free-flight gliders without the required knowledge about the aerodynamics of the design.

An interactive design system that allows casual users to quickly create 3D-printable robotic creatures is created by Disney Research Zurich [21]. The technical core of the framework is an efficient optimisation-based solution that generates stable motions for legged robots of arbitrary designs. The tool consists of two viewports: one for editing the structure and the motion of the robot, and one for displaying the resulting real-world behaviour as predicted by the simulation.

The tools discussed in this section are all focused on the design tool and the underlying algorithms that make the functionality of the tool possible. They all show the possibilities of allowing novice users to design various models without requiring the specific domain knowledge.
In my thesis the focus is not on the design tool, but on using the design tool to build a model with the robotic arm. Part of what makes that possible is the underlying algorithm that transforms the CAD model into commands for a robotic arm.

2.5 Conclusion

In this chapter I described previous work related to my thesis. I explained the different types of robots that are used in both the industry and research.

A brief overview on robot programming and CAD-based robot programming is given. The tool described in this thesis also uses a CAD application as basis for the robot programming. The discussed research is focused on extending a CAD application, which allows the user to define robot paths. The tool I propose, uses the CAD information to calculate robot commands. Thus requiring no robot programming by the user.

A short overview of techniques in which human-robot assembly tasks are used in other research projects is given. The research shows the potential of humans and robots working together. The focus of my tool is on using the robotic arm as an assistant for the user that can hold the parts that need to be assembled. This allows the user to atttach it using both hands.

Research on design tools aimed at aiding novice users during the design process is discussed. The main focus of these tools is on the underlying algorithms. These make it possible for novice users to do something that would otherwise require domain specific knowledge and/or experience. For my thesis, the design tool is designed to aid the user in the design process of the model by simplifying it. The main goal of the tool is to use a robotic arm to assist the user during the assembly.

Chapter 3

3D design tool for novice users

In this chapter, I discuss the design tool which aids a novice user during the design process of a 3D model. The user can design a 3D model using parts of the construction kit. The tool calculates the hold positions for each part and automatically programs the robotic arm.

3.1 Construction kit

The construction kit consists of the following parts (shown in figure 3.1):

- Plate: 120x60x3mm
- Bar: 120x20x3mm
- Corner pieces: 45, 90 and 135 degree angle.

Each part of the construction kit has 3mm holes in it, placed 15mm apart, which makes it possible to screw them together with bolts and nuts using the corner pieces. The corner pieces are 3D printed. The plate and the bar are laser cut out of MDF or Plexiglas, depending on the robot that is used.

3.2 Design tool

The design tool consists of an easy-to-use CAD environment in which the user can create 3D objects, using the primitive building blocks from the construction kit or custom size plates. The tool uses Autodesk Meshmixer [3] for the rendering and

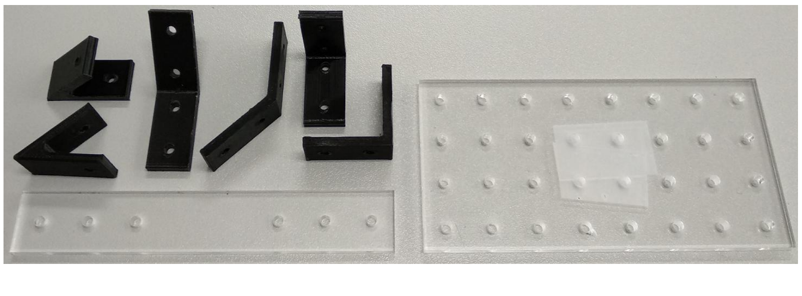


Figure 3.1: The different parts of the construction kit that are used to design and build a model.

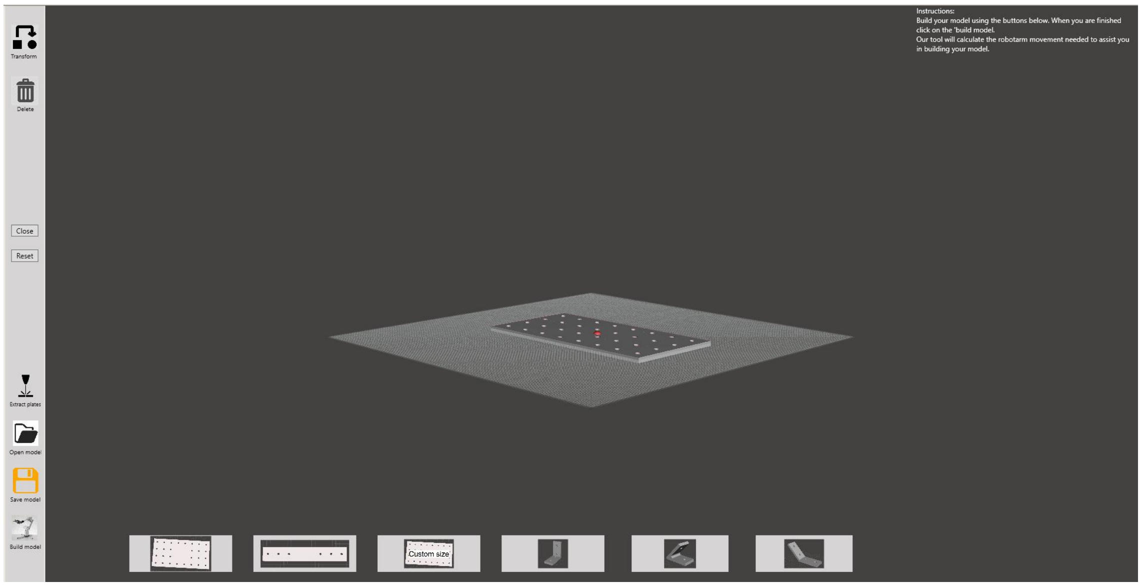

Instructions:
Build your model using the buttons below. When you are finished click on the 'build model.
Our tool will calculate the robotarm movement needed to assist you in building your model.

Figure 3.2: The UI of the design tool in which the user can create 3D objects using a CAD environment.

manipulations on the 3D model. The custom user interface (UI) of the tool is displayed on top of Meshmixer (as seen in figure 3.2). The UI limits the user to the custom actions that are supported by the tool and hides a lot of the complexity of the Meshmixer interface.

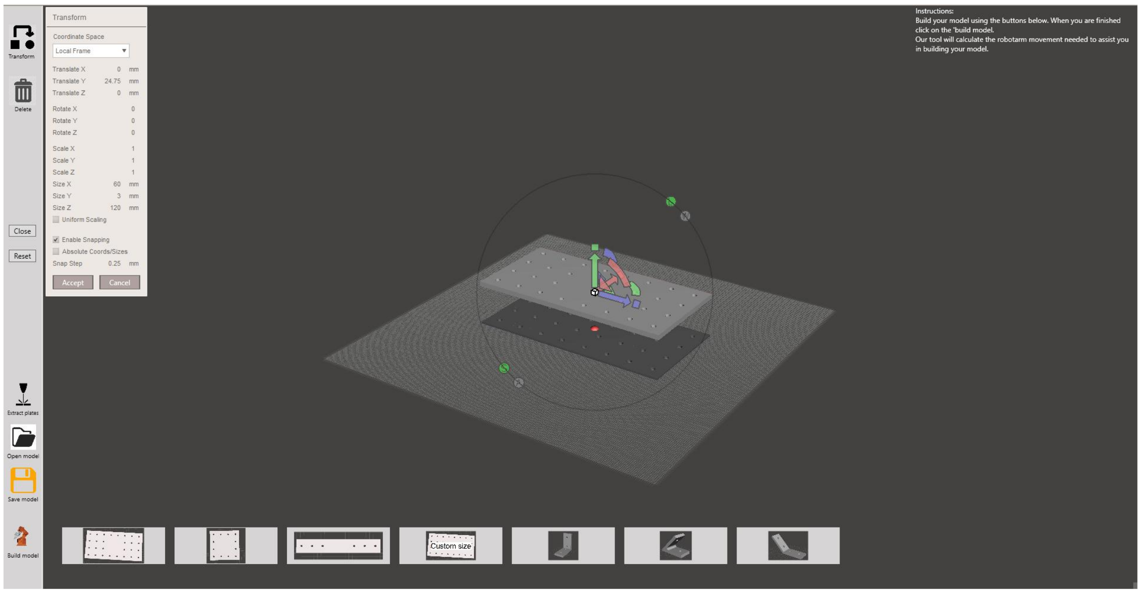
The user adds parts of the construction kit into the model by clicking on one of the buttons on the bottom of the UI. The construction kit part is added at the centre of the scene (the red dot in figure 3.2) and the Meshmixer transformation tool opens as seen in figure 3.3. Using the arrows and arcs, the part is translated and/or rotated into the position that the user wants. The UI displays its own 'Accept' button on top of the Meshmixer transformation tool, to process the transformation data into the design tool. The tool automatically snaps the corner pieces to the nearest plate when the user places a corner piece and accepts the transformation. Using the construction kit buttons, the user designs the model in the tool.

The toolbar on the left enables the user to transform and delete parts that are placed in the scene, reset the scene completely, open and save the model and start the build process.

The text in the upper right corner shows instructions to the user. The text changes according to the actions of the user. During the assembly, instructions for the user are shown that explain how to proceed. Each step in the build process is visualised to show the process so far and to highlight the next plate that has to be placed. The tool also shows any corner pieces that should be placed onto the model.

3.3 Custom size plates

The user can add a custom size plate to the scene by clicking on the 'Custom size' button. A pop-up window (see figure 3.4) opens in which the user can set the



Transform

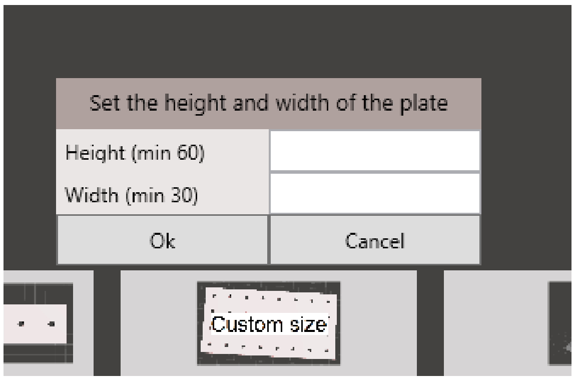
Coordinate Space: Local Frame

Translate X	0	mm
Translate Y	24.75	mm
Translate Z	0	mm
Rotate X	0	
Rotate Y	0	
Rotate Z	0	
Scale X	1	
Scale Y	1	
Scale Z	1	
Size X	60	mm
Size Y	3	mm
Size Z	120	mm

☐ Uniform Scaling
☑ Enable Snapping
☐ Absolute Coords/Sizes
Snap Step 0.25 mm

Instructions:
Build your model using the buttons below. When you are finished click on the 'build model.
Our tool will calculate the robotarm movement needed to assist you in building your model.

Figure 3.3: The Meshmixer transformation tool that is used to position the part into the scene. Translation is done using the green, blue or red arrows and rotation is done using the arcs.



Set the height and width of the plate

Height (min 60)
Width (min 30)

Figure 3.4: The pop-up window in which the user can input the dimensions of the custom plate that he wants to add to the scene.

dimensions of the plate. Clicking on 'OK' adds a plate of the given dimension to the scene, which the user can transform as usual. With this button the user can design 3D models that are not possible with the construction kit parts, for example a triangle. Before the user can start the build process, the custom size plates need to be created and fabricated first. This process is described below.

3.3.1 Extraction plates

Building a model which contains custom size plates requires, the user to create the plates before he can build the model. To do this the 'Extract plates' button is clicked, which generates a plates file. The plates file is created as a pdf file with the C# Pdfsharp [8] library, with which plates are drawn in the pdf as described below. With this file the user can laser cut the required plates and use them to build the model.

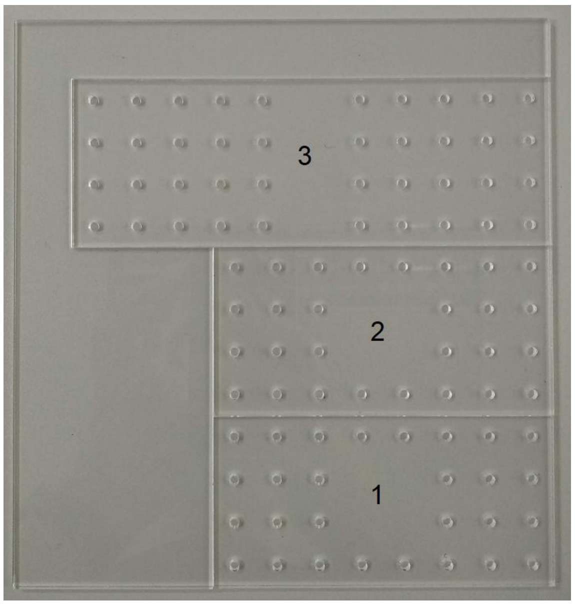


Figure 3.5: An example of the layout of a laser cut file for a model that contains three plates. The model can be seen in chapter 5.

3.3.2 Layout laser cut file

The layout of the laser cut file is as follows: the bottom right corner of the first plate is aligned with the bottom right corner of the laser cut plate (see figure 3.5, plate 1). The next plate (see figure 3.5, plate 2) is aligned with its bottom right corner against the previous plate, and so on until all plates are placed in the file. Around all the plates a rectangle is created, which ensures that all the plates are part of a whole. The left side of this rectangle has an offset of 20mm from the biggest plate and the top side of this rectangle has an offset of 20mm from the top plate. This laser cut plate is used with the robotic arm to build the model (figure 3.5 shows the entire laser cut plate).
An issue with custom size plates are the holes in the plate. They require to be 15 mm apart to fit with the other plates, and the holes should start at 7,5mm from the edge of the plate to be used with the corner pieces. The holes in a custom size plate are placed by starting from both the edges of the long side until the centre is reached. At the centre there should be at least 20mm space without holes, leaving enough space for the suction cup. This centre space also compensates for the difference in hole position from both sides of the plate.

3.3.3 Build process with custom plates

When the 'Build model' button is clicked, the design tool determines if the model contains a custom size plate. If it contains a custom size plate the build process is as follows, otherwise the robot assumes that the parts of the construction kit are in the predefined positions.

Before the user can start the build process, the entire laser cut plate should be placed next to the robot on the predefined position. The bottom-right corner of that plate is at a known offset for the robotic arm that is used. From this position

the robot arm can calculate what the pick up position of the next plate is.
The first plate should always be placed on the holder of the build platform, because it is the base plate of the model. All other plates are picked up by the robot and used during assembly.
The position of the next plate in the build process is calculated as follows. The width of the next plate determines the distance that the robot should move to pick up the next plate. The difference in height between the current and the next plate determines how much the robot should move closer or further to pick up the next plate.

3.4 Automatic position calculation

When the user clicks on the 'Build model' button, the tool calculates the desired positions in which the robot arm should hold each part of the model. In this section, I explain the algorithm and logic behind the robot position calculation. I start with a high level overview of the main idea and then I explain each part in more detail.

3.4.1 High level overview

Each time the user places a new part into the scene, the position data from Meshmixer and extra information about the part is saved in the tool. This information is required for the position calculation. For each plate that the user places into the scene, the following commands for the robot arm are calculated and saved in the build file (a JSON file).

- Take command. This indicates which type of plate the robot needs to pick up.
- Platform rotation. The amount that the platform on which the object is being built should rotate.
- Hold command. The position that the robot should hold the plate in. It contains the x, y, z coordinate, end effector angle and plate orientation.
- Release command. The position from which the robot should move away. It holds the same values as the hold command.

The order in which the user places the parts into the scene determines the build order that will be used during the build process. A possible extension to the tool is calculating the best build order from the model.

3.4.2 Take Command

When the user adds a part to his model, the type is saved by the tool. The type is used to create the right take command for the robot. The height and width of the plate are saved as well. The dimensions of the plate are used to calculate the position of a custom size plate.
The construction kit parts are defined as '4x8' or '1x8' (based on the amount of rows and columns of holes in the plate). The custom size plates are defined as 'Custom:objectId'.

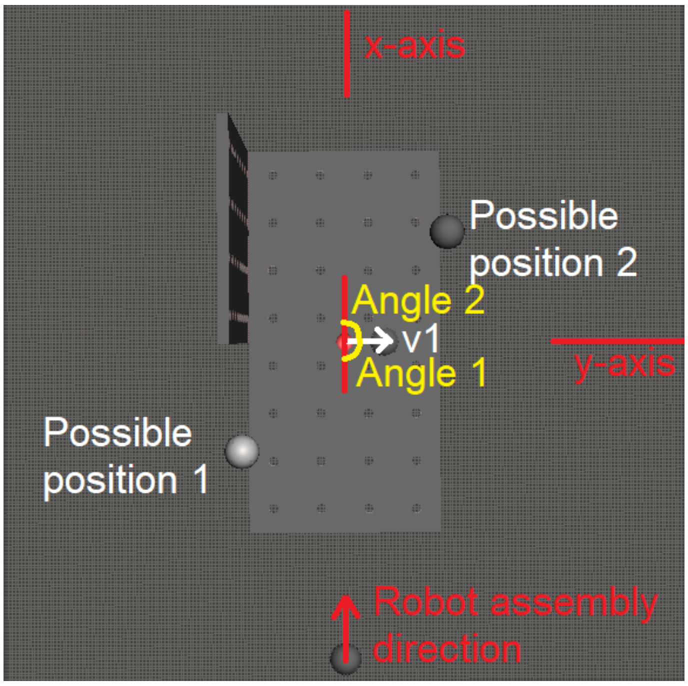


Figure 3.6: Visualisation of the platform rotation calculation. The figure shows the two possible positions that the robot can hold the plate in, the assembly direction, vector v1, angle 1 and angle 2.

3.4.3 Platform rotation

A platform is used with a robot arm to increase the reachability of each robot. The model that the user is building is placed on this platform. It can rotate and move across a linear axis. Moving closer or further away from the robot allows the robot to reach more positions. Rotating the robot allows access to positions on the opposite side of the model. The details of this platform are explained in chapter 4.
The code that controls the robot calculates the linear movement of the platform, because it is different for each type of robot that is used. The tool calculates the rotation of the platform, because this should be the same for each type of robot that is used (in chapter 4 I explain why this is not the case).
The result of the rotation calculation is saved and used in the rotation calculation of the next plate. This ensures that the platform rotates from the current position to the next position.

Rotation calculation algorithm

Two possible positions in which the robot can hold the plate are calculated. The optimal position that the algorithm selects is saved as the position that the robot brings the plate to. Figure 3.6 shows the two possible positions, the assembly direction of the robot and the possible angles.

The two possible positions are calculated as follows:

1. Create normalised vector v1 that points outward of the plate.

2. Calculate angle 1 between the robot assembly direction and v1. The supple-

mentary angle is angle 2.

3. Calculate the angle between v1 and the x-axis and v1 and the z-axis. These are the angles of the plate in the x- and z-axis.

4. Rotate the position of the plate according to angle 1 and 2 to calculate the possible positions 1 and 2. The centre point of the rotation is the zero point of the model (the red dot in figure 3.6). Visualisation of the rotation is shown in figure 3.7.

5. Test if the possible positions are on the x-axis or not.

6. If the possible positions are on the x-axis, use the angle of the plate in the x-axis to determine if the plate is angled. If the plate is angled, the furthest position is taken. If the plate is at a 90 degree angle, the closest position is taken.

7. If the possible positions are not on the x-axis, use the angle of the plate in the x- and y-axis to determine if the plate is angled. If the plate is at a 90 degree angle, the closest position is taken. If the plate is angled, test if the plate is angled above another plate, by shooting a ray down from the centre of the plate.
    - When there is a plate beneath the current plate, the robot cannot hold the plate from the bottom because there is not enough room. The best position is the position that is in front of the centre of the plate, which is the position smaller than or equal to the centre.
    - When there is no plate beneath the current plate, the furthest possible position should be taken.

8. The final position is transformed from scene coordinates to world coordinates, which determine the offset of the robot from the centre of the platform. The coordinates are saved as they are later used for the hold and release commands.

9. Return the angle that corresponds to the final position. This is the angle that the platform should rotate for this step in the build process.

3.4.4 Hold command

The hold command contains the x, y, z coordinates, the end effector angle and the plate orientation. In the previous step the x, y, z coordinates are already calculated so the next step is to calculate the end effector angle and then the plate orientation.

End effector angle

The end effector angle that is calculated is interpreted as follows: 0 degrees is the end effector pointing down, 90 degrees is the end effector pointing forward. The algorithm that calculates the angle with which the robot should hold the plate is as follows:

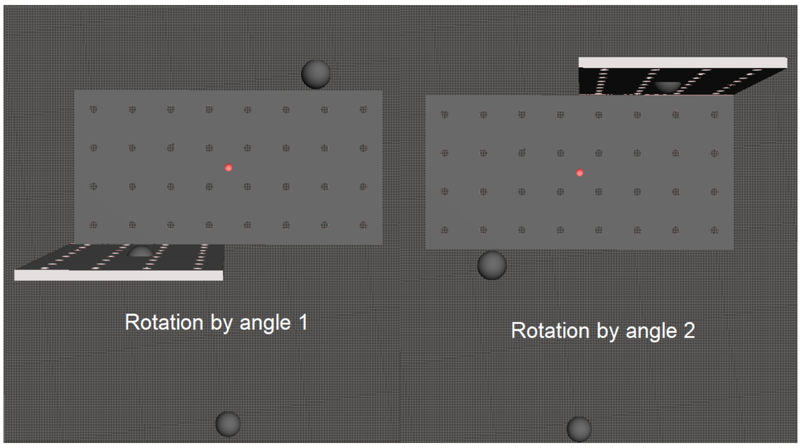


Figure 3.7: Visualisation of the platform rotation with angle 1 and angle 2 that gives the two possible positions in which the robot can hold the plate.

1. Create normalised vector v1 and v2 that point outwards of the plate. Vector v2 is the inverse of vector v1.
2. Rotate vector v1 and v2 with the platform rotation angle around the centre of the model. This way v1 and v2 are angled into the position that the robot should bring them. Figure 3.8 shows vector v1 and v2 for the plate at an angle of 45 degrees.
3. The z value of v1 and v2 determines the angle of the plate.
4. When the z value is equal, the plate is at a 90 degree angle. The angle is the same for v1 and v2, so v1 is the correct vector.
5. When the z value is not equal, the plate is angled. Test if the plate is above another plate by shooting a ray down from the centre of the plate.
    - When there is a plate beneath the current plate, the robot cannot hold the plate from the bottom because there is not enough room. The robot should hold the plate from the top, which is the vector with the highest z value.
    - When there is no plate beneath the current plate, the vector with the smallest z value is taken.
6. Return the angle between the correct vector from the previous step and the vector pointing up (displayed in figure 3.8).

Plate orientation

The plate orientation determines if a plate is placed horizontal (0 degrees), vertical (90 degrees), or any angle in between onto the model. Figure 3.9 shows the difference. This is calculated as follows:

1. Create normalised vector v1 and v2 that are pointing across the long axis of the plate (as seen in figure 3.9).

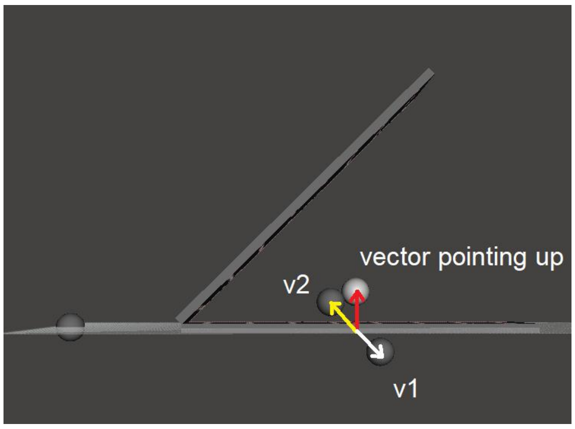


Figure 3.8: Visualisation of vector v1, v2 and the vector pointing up that are used to calculate the end effector angle.

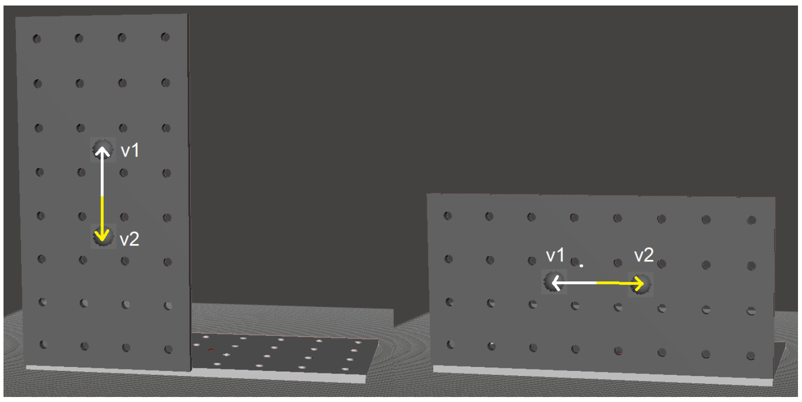


Figure 3.9: The plate on the right is in vertical orientation, the plate on the left is in horizontal orientation.

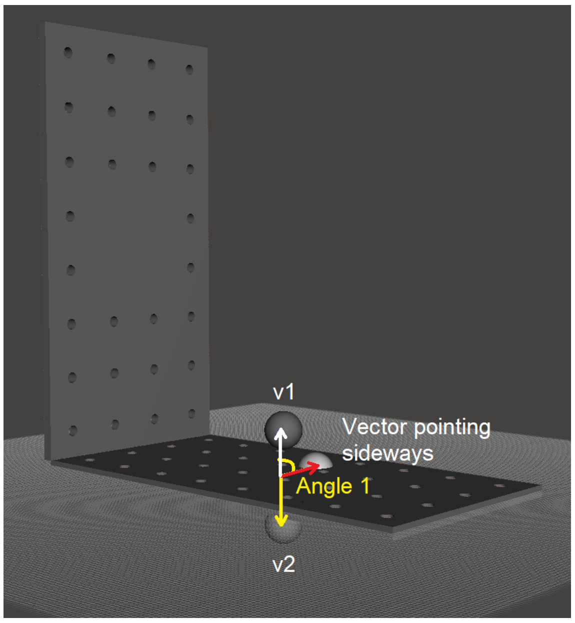


Figure 3.10: Visualisation of the vectors v1, v2 and vector pointing sideways which are used to calculate the plate orientation.

2. Rotate vector v1 and v2 with the platform rotation angle around the centre of the model. This way v1 and v2 are angled into the position that the robot should bring them.

3. Calculate the angle between v1 and the vector pointing sideways and do the same for v2 (as seen in figure 3.7). The smallest angle indicates the orientation of the plate and is returned.

3.4.5 Release command

The release command contains the same values as the hold command, so there is no calculation needed for this command. The release command is used as a build step for the robot. The values are used to calculate the movement of the robot arm to move away from the plate.

3.5 Conclusion

The design tool should make it simpler for novice users to design a 3D model in a CAD environment, by providing a basic user interface and restricting the user to a limited set of parts from a construction kit. Besides aiding with the design, the tool calculates robot positions which are used to program a robotic arm. During the assembly, the design tool changes its functionality to show instructions to the user.

Chapter 4

Primary robotic support with design tool

In this chapter, I explain how the Arduino Braccio robotic arm is used with the design tool from the previous chapter. The Braccio works with a first version of the tool, which only includes the following parts of the construction kit: the plate, bar and 90 degree corner pieces. To do this, a platform is designed and built to compensate for the missing degrees of freedom and limited reach of the robot. I end this chapter with a discussion on the implementation with the robot arm and the results.

4.1 Arduino Braccio

The Arduino Braccio is a tinkerkit, which contains the required parts to create a robotic arm that can be programmed using an Arduino and the included Braccio Shield. The Braccio is a cheap robot (€200 at the moment of writing) that is designed for DIY-projects. The robotic arm can be assembled in a multitude of ways, but for this thesis the setup that can be seen in figure 4.1 is used. This setup gives the robotic arm 5 degrees of freedom. The maximum operating distance in this configuration is 80 cm, the maximum height is 52 cm and the maximum load capacity at 32 cm is 150 g.

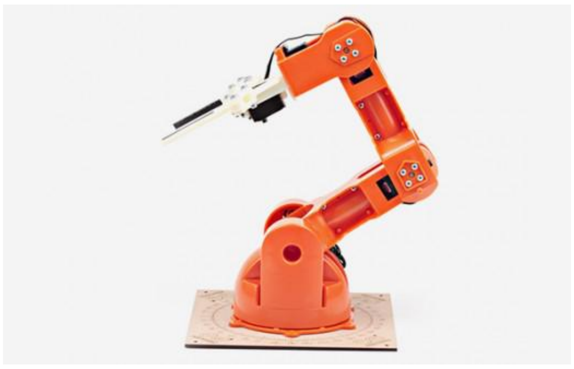


Figure 4.1: The Arduino Braccio robotic arm [2].

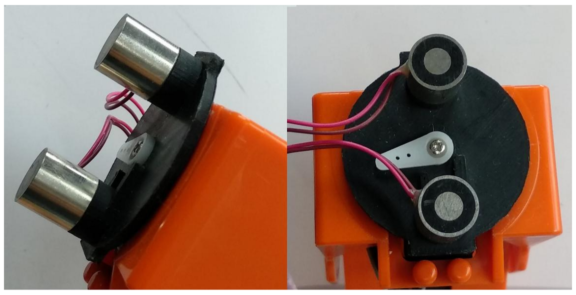


Figure 4.2: The custom designed and 3D printed end effector which uses two electromagnets to pick up the plates of the construction kit.

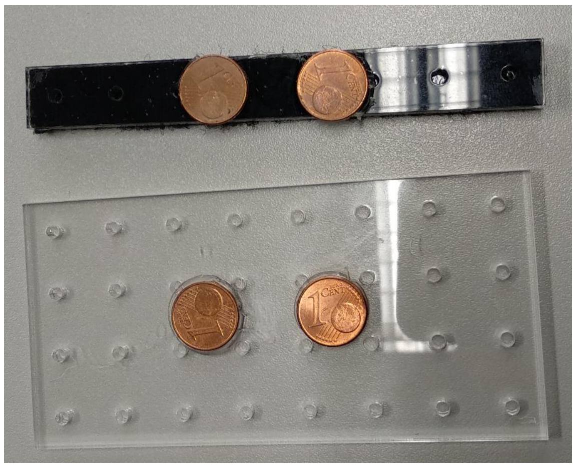


Figure 4.3: The construction kit parts with the two small metal pieces (1-euro cent coins) glued onto it.

4.1.1 Custom end effector

The Braccio comes with a gripper as an end effector, which can be seen in figure 4.1. This end effector cannot be used to pick up and hold the parts of the construction kit. Therefore a custom end effector is designed and 3D printed that fits onto the Braccio. The end effector contains two electromagnets which are used to pick up and hold the parts of the construction kit through the metal pieces on them (figure 4.2).

The original plan was to use a suction cup on the robot to pick up plates, but this would be too heavy to lift for the robot and not stable enough to hold the plates. The electromagnets on the end effector are small and lightweight, which allows the robot to lift a plate and hold it in the required position. The construction kit parts that are used with the Arduino Braccio can be seen in figure 4.3

4.2 Platform design

In this section, I explain each part of the platform that is used with the Braccio. The platform is required because of the limited reach of the robot arm. Placing a plate flat on the base plate limits the range of the robot, therefore the platform needs to move the holder closer to the robot. Placing a plate against the front of the base plate moves the end effector further away, therefore the platform moves further

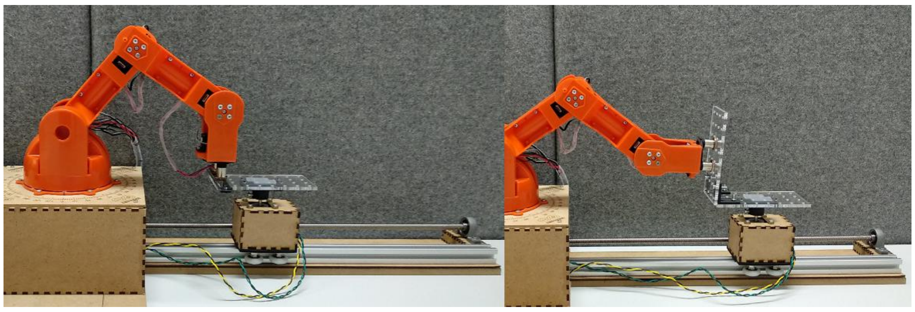


Figure 4.4: The platform and robot positions that are required to place both the plate positions. This image shows why the linear movement of the platform is required.

away as well. Figure 4.4 shows the platform and robot arm position for both plate positions.
This platform holds the model that is built, which it can rotate and move across the linear axis. The robotic arm is placed on top of the platform, to increase the height difference between the model and the robot base. Behind the robot on the platform, the construction kit parts are stored so the robot can pick them up. The Arduino that controls the robot and the electronics that are used in the platform, are placed underneath the robotic arm.

4.2.1 Build platform

The build platform holds the model that the user is building. The base plate of the model is placed on the holder. This holder is placed on a stepper motor, which enables it to rotate. The holder is placed on top of the linear carriage.

Model holder

A custom designed and 3D printed holder is created that holds the base plate of each model. The holder has four pins that fit into the holes of a plate. The bottom of the holder is attached to a stepper motor (NEMA 17). Everything is enclosed in a custom designed laser cut box, which is placed on top of the linear carriage. Figure 4.5 shows the holder and the laser cut box.

Linear carriage

To enable the linear movement of the holder, a linear carriage is built (figure 4.6). The design is based on this customisable small linear carriage generator [16] from Thingiverse. The carriage design is modified to be able to mount the model holder onto it. The final design is 3D printed. The carriage glides along the V-slot rail by the wheels that are mounted under it (as can be seen in figure 4.7).
The carriage works as follows: the stepper motor rotates the lead screw, which moves the nut that is mounted against the carriage. Each rotation of the lead screw moves the platform 8 mm. Controlling the rotation of the stepper motor, controls the linear movement of the carriage.

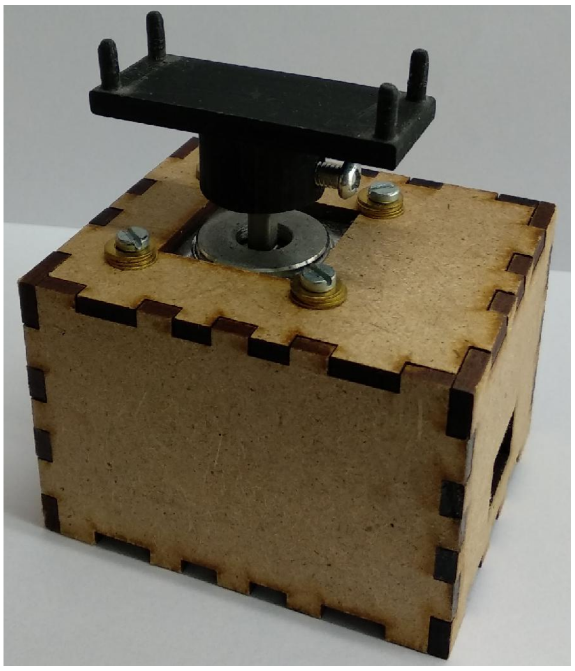


Figure 4.5: The 3D printed holder that holds the model, and the laser cut box that contains the stepper motor which rotates the holder.

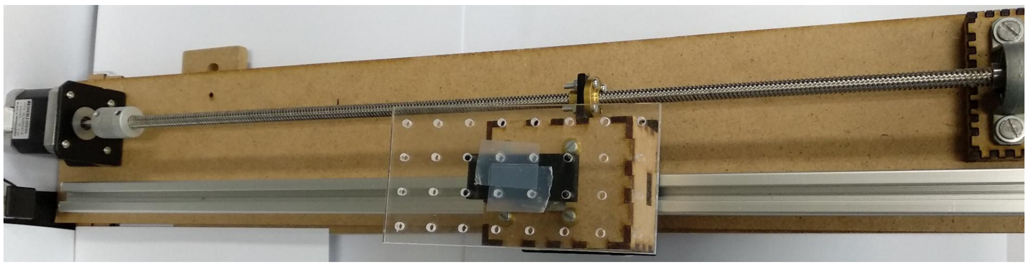


Figure 4.6: The linear carriage that moves the revolving plate holder.

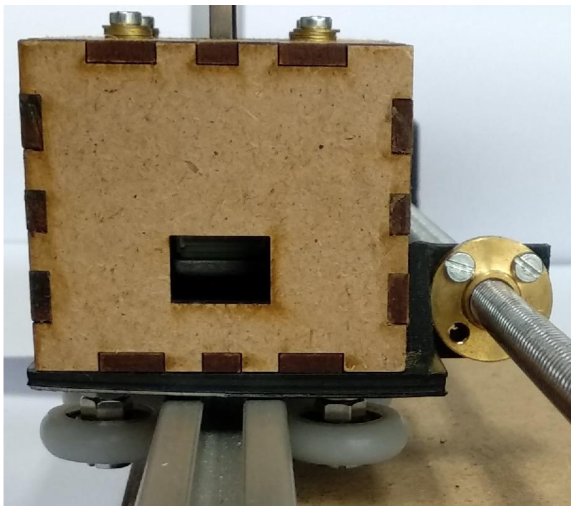


Figure 4.7: Side view of the linear carriage that shows the wheels with which the platform glides along the V-slot rail.

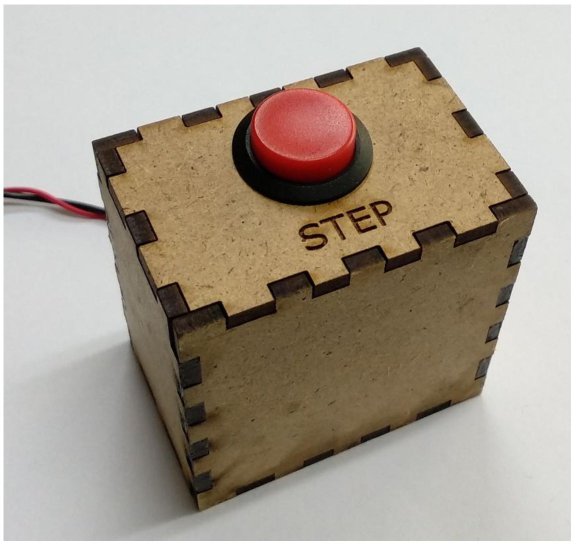


Figure 4.8: The laser cut box that holds the step-button. It is not attached to the platform, therefore the user can place it as desired.

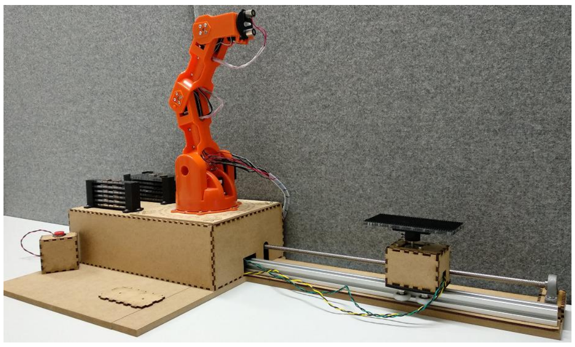


Figure 4.9: The laser cut box on which the construction kit parts are placed and the robot arm is mounted. This box contains the electronics used.

4.2.2 Step-button

A button is used to signal to the robot that a next step in the build process can start. The button is placed in a custom laser cut box. The step-button is not attached to the platform, therefore the user can move the button to a position that is best suited for him. Figure 4.8 shows the step-button.
The possible actions for the robot when pressing the button are: picking up the required plate, holding the plate in the right position, and moving the robot arm out of the way. In the instructions of the tool it is explained when the user should press the button and what the effect is.

4.2.3 Robotic arm platform

The Braccio is placed on top of another laser cut box. Behind the robot are two holders that hold the plates and bars of the construction kit. This ensures that they are always on the same spot and it is easy for the robot to pick them up. Inside the box are the electronics that control the robot arm, the stepper motors, electromagnets and the step-button. The platform can be seen in figure 4.9.

The robotic arm is placed on top of the platform to increase the height between the Braccio and the holder of the build platform. Because the robot arm can reach lower positions more easily than higher positions. This way more positions can be reached, especially in combination with the linear movement of the platform.

4.2.4 The electronics

To power and control the different parts of the platform, an Arduino with the Braccio shield and the following electronics are used:

- 2x NEMA 17 stepper motor [27].
- 2x EasyDriver [30]. These control the stepper motors. They transform the input signals from the Arduino to the corresponding current that the stepper motor receives.
- 2x Electromagnets [16]. These are permanent electromagnets, which means the magnetism is neutralised by a current pulse. They require 12V and deliver 15N holding force. They are controlled by the Arduino using transistors.
- MCP23017: GPIO pin extender [22]. The GPIO extender is required, because the Arduino does not have enough available GPIO pins left when using the Braccio shield.
- AC 240v to DC 14v external power supply. This is used to power the electromagnets and the stepper motors. All other electronics are powered by the Arduino via USB.

4.3 Results

In this section, I explain the components of the tool that are implemented, the support of the robot, and two example models that can be created with the robot.

4.3.1 Implemented components tool

The version of the tool described in chapter 3 is the final version of the tool. With the Braccio, a first version of the tool is used, which only supports the following parts of the construction kit: the plate, the bar and 90 degree angle corner pieces. The tool contains no logic processing. It is used to build the model and calculate the positions. When the user starts the assembly process, the build file is generated. Every time the user presses the step-button, a signal is sent from the Arduino to the tool. The tool responds with the robot commands for the next step in the build process.

4.3.2 Inverse Kinematics implementation

The robot controller code provided by Arduino can only control the Braccio, by sending the required angle that each servo should move to and the speed with which the servos rotate. The tool calculates a x, y, z position for the robotic arm, therefore transformation to the angle of each servo is required. This is done using inverse

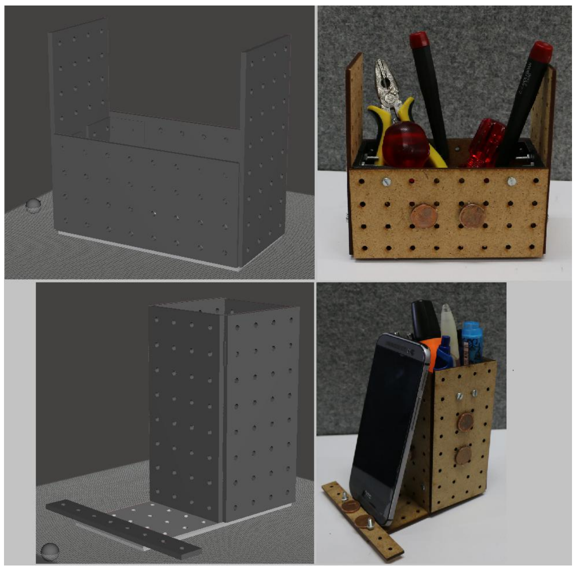


Figure 4.10: Two example models that are designed with the tool (on the left) and built with the Braccio (on the right).

kinematics (IK) calculation on the Arduino that controls the robot. The code that is used to calculate the inverse kinematics is from this project [13].

4.3.3 Platform position calculation

The tool does not calculate the platform position, because it depends on the robotic arm. For the Braccio, the position is calculated on the Arduino by testing if the robot can reach the position. It is tested by checking if any of the servo limits are reached. If any servo limit is reached, the position is out of reach. The platform position is calculated as follows:
Test if the robot can hold the plate in the current position of the holder. If possible the platform stays in position. If it is not, start iterative testing from the first position of the platform until a possible position is found. Each step in the process moves the platform backwards by 20 mm. When a position is found, the platform is sent to that position and the robot arm moves to the given hold position with the plate. When no position is found, the platform is sent to the last position.

4.3.4 Example models

The build process with the Braccio and the corresponding version of the tool, is tested for the two models that can be seen in figure 4.10. There are more models possible, as long as the user stays within the limits of the base plate and the robot.

4.4 Limitations

As explained in this chapter, the Braccio is limited in reach and weight which is why the build platform is used. But even when using the platform, the robot is still limited in the positions it can reach and therefore in the models it can build.

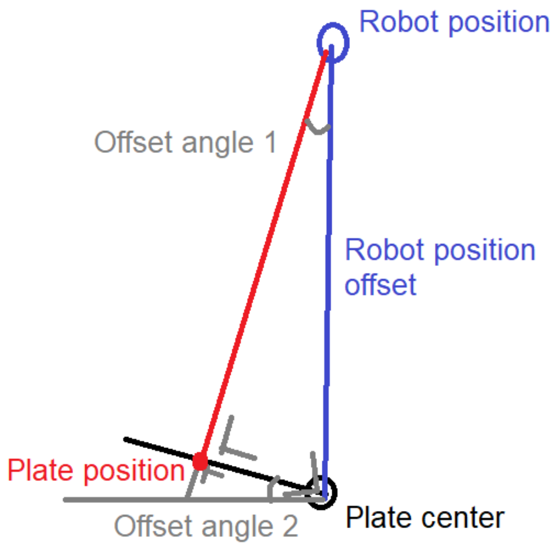


Figure 4.11: Visualisation of how the platform rotation is adjusted when the Braccio has to hold a plate that is off-centre. The vectors visualise the calculation.

4.4.1 Plates at an offset

The Braccio only has 5 degrees of freedom. If we leave out the bottom joint which rotates the robot, the robot can only move in a plane. For example picking up something right behind it and placing it right in front of it. Adding the bottom joint enables the robot to rotate the plane in which it can move. This enables the robot to move to positions off-centre. The movement of the Braccio is still limited, as it cannot move its end effector in one direction without also moving the end effector in another direction. For example, moving the end effector in the horizontal direction will also move the end effector closer to or further from the robot.

This limits the robot from holding plates at an offset from the centre of the holder. The solution to this, is changing the way the platform rotation is calculated. Instead of assuming the robot can reach all positions, I now assume that the robot can only place plates at a 90 degree angle with the base plate of the platform. Therefore to place a plate that is off-centred, the platform will not be 90 degrees but for example 95 degrees. This can be seen in figure 4.11. The offset angle can be calculated by the inverse sinus of the plate position (red line) and the plate centre (blue line). This gives offset angle 1, which is equal to offset angle 2 (the angle that is needed).

4.4.2 Corner piece angles

The tool only supports corner pieces with an angle of 90 degrees. The position calculation code was still in a first version, therefore there were still bugs when trying to build plates at an angle. For example a plate at 45 degrees would result in the robot holding it as a plate of 135 degrees. The limited flexibility of the Braccio also made it harder to support 45 degree angles at most of the possible positions.

4.4.3 Construction kit parts

Each part of the construction kit needs to be modified before the Braccio can pick it up, by hot-glueing metal pieces (e.g. small coins) to it. This means that the final model will have metal pieces on it, although that was not a part of the design.

4.4.4 Plates out of reach

The Braccio is limited in height at which it can hold a horizontal plate (parallel to the base plate). The limit is about 70 mm. This sometimes limits the ability to create models that are designed in the tool.

4.5 Demonstrations

The Braccio in combination with the build platform and the tool are demonstrated on multiple occasions. During the demonstrations, the construction parts are changed to use Velcro tape. This speeds up the assembly process, because tightening the bolts and nuts takes quite a long time. The assembly with the Velcro tape is instant, so the robot is not necessarily needed to hold a part in place. The assistance that the robot arm needs to provide is therefore reduced.

4.5.1 UIST Student Innovation Contest

My thesis started with the UIST Student Innovation Contest, which required a team to come up with a project idea that uses the Arduino Braccio. UIST is a conference on User Interface Software and Technology. The result described in this chapter is the final result that was demonstrated during UIST, by Tom Veuskens. Our team won the honourable mention award for the best implementation.

4.5.2 Science fair: 'Dag van de wetenschap'

The setup was demonstrated during the 'Dag van de wetenschap' at Hasselt University. This time for an audience between five and twelve years old. The children were assisted by the system to build one of the example models. The goal of aiding novice users with the build process was achieved, because almost all the children were able to build the model. During the demonstration we gave the instructions to the children, because the instructions in the tool are in English.

4.6 Conclusion

The Arduino Braccio is a cheap robotic arm that is limited in its reachability and payload, but the modular design makes it very flexible. The custom end effector was easy to install and the Arduino makes it easy to incorporate different electronics with the robotic arm.

The combination of the Braccio and the build platform enables the robotic arm to do much more. For example building the model part by part and therefore bypassing the payload limit. In addition the movement of the platform enables the

robot to build bigger models.
Without the platform the robot can only be used to hold plates in fixed positions. Even with the platform, the Braccio is still limited in the height at which it can hold a plate. This means that building a model with a part too high from the base plate is an issue. The Braccio cannot hold the plate in that position.

To conclude, the first version of the design tool in combination with the cheap Arduino Braccio shows the potential of the design tool. It shows that a fairly basic robot can be used as an alternative for a more expensive robotic arm. The process of building the example models went smoothly during the demonstrations. Anyone can build the model with the robot without needing any knowledge about the model or the build process, which was one of the goals of this project.

Chapter 5

Advanced robotic support with design tool

In this chapter, I explain how the Commonplace Robotics Mover6 robot arm is used in combination with the design tool described in chapter 3. The goal is to create more complex models. I start with the implementation of the first version of the tool with the Mover6 robotic arm. Next, I explain how the new additions to the design tool work. I end this chapter with a discussion on the benefits and limitations of the implementation with the Mover6 robot arm.

5.1 Commonplace Robotics Mover6

The Commonplace Robotics Mover6 robot arm (figure 5.1) is designed especially to meet the needs of schools, universities, and research and development institutions. The payload is specified as 400 g with a reach of 600 mm including the gripper. The robotic arm has 6 degrees of freedom. The extra degree of freedom over the Braccio enables it to move its end effector in one direction only. For example, moving its end effector from left to right, without changing the distance or height from the robot. This enables the Mover6 to hold a plate that is off-centre, without compensation in the rotation of the build platform.

5.1.1 Custom end effector

The Mover6 comes equipped with a two finger gripper end effector, which is not suited to pick up the parts of the construction kit. Therefore a custom end effector is built, as shown in figure 5.2. Instead of using electromagnets, a vacuum suction cup is used to pick up and hold the parts. It enables the robot to pick up the parts without having to attach metal pieces to them. A disadvantage of the suction cup is that the plate cannot be moved when the robot is holding it, because the plate is forced against the pads on the end effector. The electromagnets make it possible to move the plate as long as the metal pieces come into contact with them. This small movement of a plate can compensate for small errors in the hold position of the robot.
The suction cup outer diameter is 15 mm, which is the minimal space that is required to create a vacuum. To make sure that there is enough room to compensate for small position errors, a space of 20 mm is preferred. A bellows suction cup is used, which

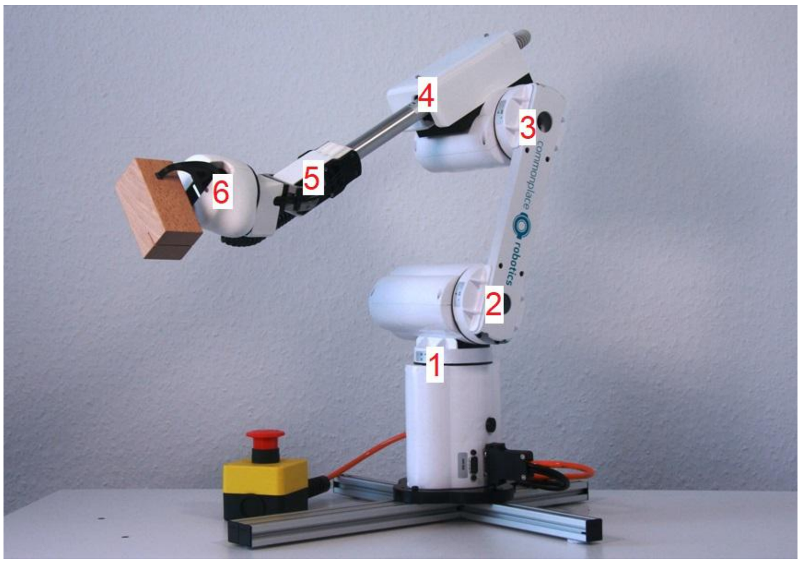


Figure 5.1: The Commonplace Robotics Mover6 robot arm [5]. The numbers indicate the joint that can be controlled.

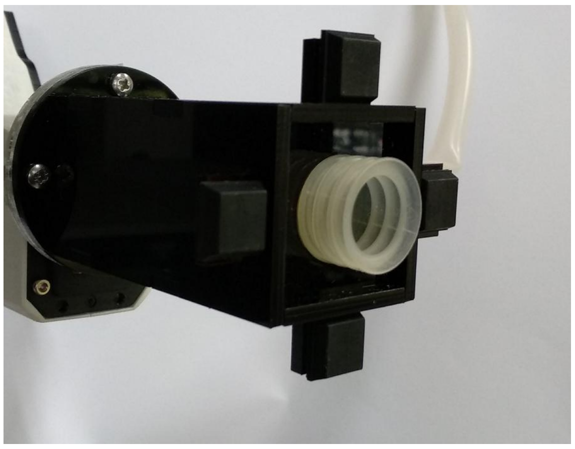


Figure 5.2: The custom end effector fitted with a bellows suction cup, to pick up and hold plates. The pads on each side increase the stability of the plate during movement of the end effector.

helps to compensate for height differences. This ensures that the suction cup can make sufficient contact with the part, even when the end effector is not positioned perfectly level above it. On each side of the suction cup is a pad, the black squares in figure 5.2. When the suction cup creates a vacuum, the plate is forced against the pads causing it to stay still when the end effector moves.
A vacuum pump creates the vacuum for the suction cup. It is powered by an external 24V DC power source. To release the plate from the end effector, a valve is opened which lets air back into the suction cup. The valve and the pump are placed in a custom laser cut box (see figure 5.4) and controlled by relays inside the base of the Mover6. The relays are controlled by the robot control software.

5.1.2 Robot programming

The Mover6 robot is controlled with the included "CPRog - robot control environment [6]". It contains its own implementation of the inverse kinematics of the robotic arm. The movement of the robot is visualised with a 3D simulator, which can also be used for testing. The robot can be controlled in multiple ways using the tool:

- The joint angles can be controlled using the interface of the tool.
- The values of the Cartesian coordinate system can be controlled using the interface.
- Programs that contain commands for the robot can be created in the tool or by code. A program can consist of different types of commands: Cartesian coordinates, joint angles, output signals, wait commands and loops. It is formatted in XML, which makes it easy to create in code.
- A CRI server that the CPRog software starts. It can send all the above commands to the robot. The tool opens a socket server to which the data is sent. It requires at least one alive message every two seconds, otherwise the connection is closed.

I chose to control the Mover6 with the CRI server, sending programs that contain the commands to the robot. The programs are created in a predetermined folder and the CRI server command indicates which program to open and execute. Not all individual commands over the CRI server are implemented [7], which is why I chose to use programs instead.

5.2 Controlling a 6 DoF Robot

The Mover6 has 6 degrees of freedom, which increases the flexibility but also changes the way the robotic arm is controlled. Instead of having one end effector angle that can be controlled like the Braccio, the Mover6 has three angles that determine the position of the end effector, the so-called Euler angles. The position values that the tool calculates, need to be transformed into Euler angles before they can be used with the Mover6.

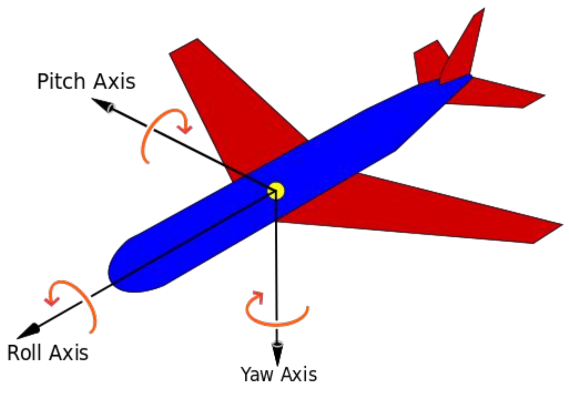


Figure 5.3: Visualisation of the yaw, pitch and roll rotation on a plane [29].

5.2.1 Euler angle

Euler angles are three angles that describe the orientation of a rigid body, with respect to a fixed coordinate system. There are several conventions for Euler angles, depending on the axes on which the rotations are carried out. The most common definition is the so-called x-convention [35]. In this convention, the rotation is given by Euler angles (ϕ, θ, ψ), where:

1. The first rotation is by an angle ϕ about the z-axis
2. The second rotation is by an angle θ about the former x-axis (now x')
3. The third rotation is by an angle ψ about the former z-axis (now z')

This convention is also written as (z-x-z), indicated by the rotation axis.

In robotics, the Tait-Bryan convention is mostly used. In this convention each of the three angles define the rotation around a different cartesian axis. For example x-y-z, the first rotation around the x-axis, the second around the y-axis and the third around the z-axis. The order in which the rotations are executed matters, therefore it should always be specified [29].
When the order of rotation is z-y-x, an alternative way to describe the rotations is yaw, pitch and roll. Figure 5.3 visualises these rotations on a plane.
In the CPRog tool the rotations are indicated as A, B and C. Changing the values in the tool shows the following: A controls roll, B controls pitch, and C controls yaw. This does not match the yaw, picth and roll convention when the order of rotation is z-y-x. The order of rotation that is used in the CPRog tool is most likely x-y-z.

5.2.2 Transformation to Euler angles

The end effector angle calculated in the tool, needs to be transformed into the Euler angles of the Mover6. I manually adjust the joint angles in the CPRog tool, to the possible end effector angles (0, 45, 90, 135) of the robot. For each angle I save the matching Euler angles. The build file indicates the end effector angle. Its corresponding Euler angle is used to control the robot.
An alternative way to do this is by automatically transforming the end effector angle

into the Euler angle convention that is used for the Mover6. I chose not to do it this way, because I could not find an immediate solution for the problem and I did not want to risk it taking up too much time.
The manual transformation is sufficient to test the design tool implementation. The downside is that only a limited amount of angles are supported by the robot. However the design tool only supports the same limited amount of angles, therefore this is not an issue for now.

5.2.3 Singularity

In robotics, a singularity is a condition caused by the collinear alignment of two or more robot axes, resulting in unpredictable robot motion and velocities [1]. When a singularity occurs, there may be infinite possible ways for the inverse kinematics calculation to achieve the same end effector position of the robot. This can result in unpredictable movement of the robot. If a robotic arm is required to maintain a consistent speed across an axis and a singularity occurs, the robot will have to rotate one of the joints 180 degrees in an instant. This requires an infinite velocity, which is not possible.
A 6 degree of freedom robotic arm suffers from these three types of singularities [11]. Figure 5.1 indicates the joint numbers of the robotic arm.

- Wrist singularity: happens when joint 4 and joint 6, the wrist axes, align.
- Shoulder singularity: happens when the centre of the robot wrist aligns with joint 1.
- Elbow singularity: happens when the centre of the robot wrist lies on the same plane as joints 2 and 3.

The movement of the Mover6 stops when a singularity is reached. The CPRog tool indicates what type of singularity is reached. The robotic arm cannot be controlled with the Cartesian coordinates until the singularity is resolved, which can be done with joint angle commands.
Combining the Mover6 with the build platform sometimes causes wrist singularities to occur. Whenever a singularity occurs, I manually adjust the path the robot takes to avoid it next time the command is executed.

5.3 Platform redesign

The platform from chapter 4 is reused with the Mover6 robotic arm. It is used as separate parts and not combined into one platform, because the design is not finalised. That is why all parts are placed on a temporary platform. Figure 5.4 shows the different parts. If the platform design is finalised, a custom-built platform could be created which allows the parts to be in a fixed location, making the setup easy to move.

5.3.1 Build platform

The rotation and linear movement of the holder are required with the Mover6 robot. The rotation is required because the robot is fixed in one place, so it needs a way to

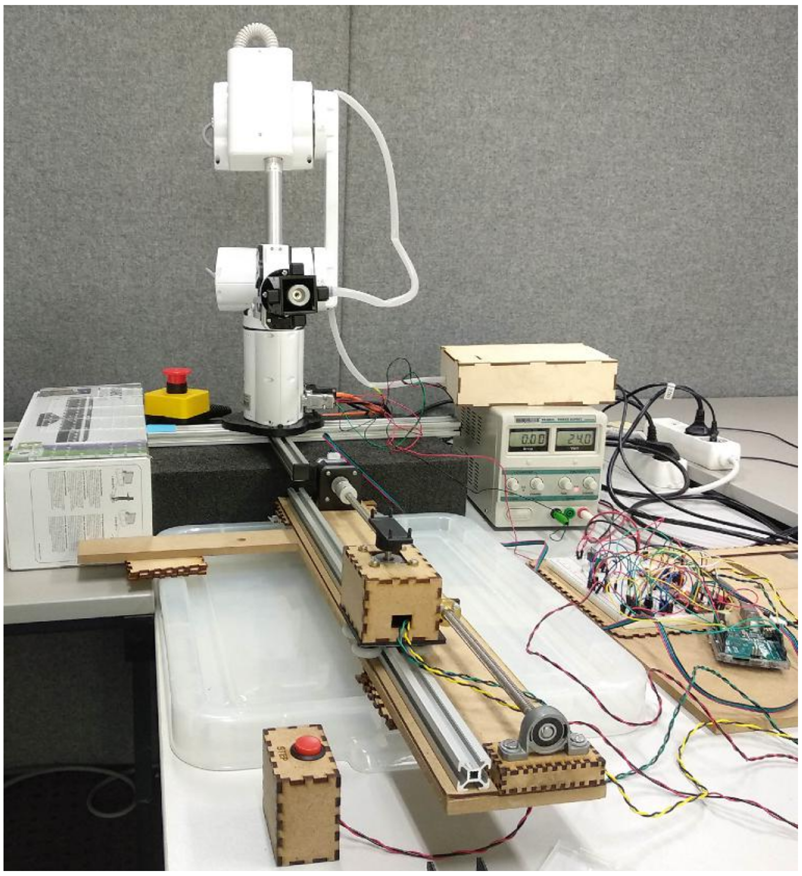


Figure 5.4: The setup of the platform for the Mover6 as it is used to test the implementation.

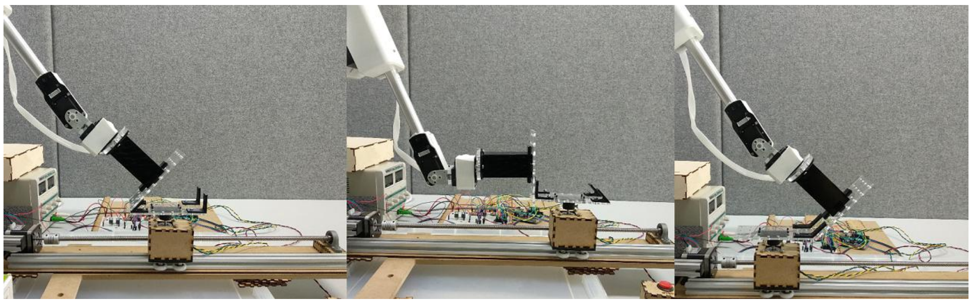


Figure 5.5: The three different positions in which the robot arm can hold a plate. This shows why the linear movement of the holder is required.

reach the other side of the model. The Mover6 can hold parts at an offset from the centre of the holder without compensation in the rotation, because of the 6 DoF in the robotic arm. The linear movement is required because the robot cannot reach all possible positions when the holder is fixed in one position. For example when holding a part at a 90 degree end effector angle, the model needs to be further away from the robot than under a 45 or 135 degree angle (see figure 5.5).

5.3.2 Redesign construction kit

The original parts of the construction kit are made out of MDF, which cannot be picked up by a vacuum suction cup. They do not create an air-tight seal necessary to create a vacuum. The new construction kit parts are therefore made out of Plexiglas.
The second issue with the original parts is that the suction cup needs a 20 mm diameter of space to pick up a part. The original parts however have holes 15 mm apart from each other, which is too close. The solution is to remove the four holes in the middle of the plate (see figure 5.6), which ensures enough space for the suction cup.
The small plate (bar) of the original construction kit is 15 mm wide, which is too small for the suction cup. The new version of the bar is 20 mm wide and has the two middle holes removed (see figure 5.6).

5.3.3 Parts position

The setup with the Mover6 has two ways to pick up parts during the build process. The design tool works with both, because it enables testing and showcasing both.

Predefined positions

The construction kit parts are placed in a predefined spot, which ensures that the robot knows where to pick them up. Instead of designing new holders for the parts, I chose to place them on top of a box and mark the positions.

Custom plates

Building a model that contains custom size plates changes the way the robot picks up the plates. Instead of the parts being placed in predefined positions, the robot

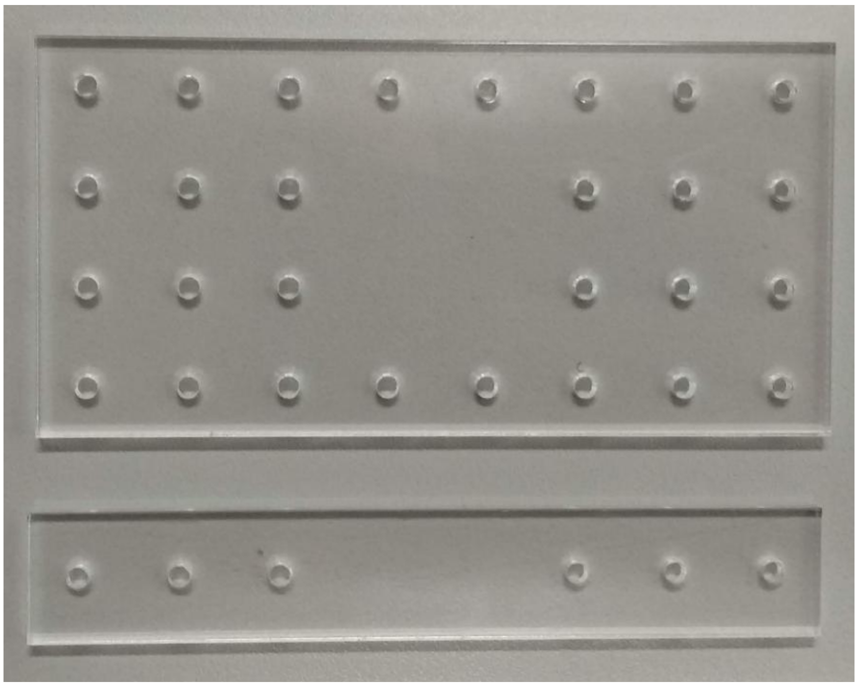


Figure 5.6: The new parts of the construction kit, with extra space in the middle for the suction cup.

knows the positions from the layout of the plates file. The user places the laser cut plate onto the box with one corner aligned. That position is the start position for the robot.

The preferred use case with the Mover6 is using the custom size plates. It enables a novice user to design a model in the tool, extract the plates file to laser cut the required plates, and place them next to the robot. Instead of determining how many of the construction kit parts are needed, creating more if necessary, and placing them in the correct positions.

5.4 Implementation first version

The first step in trying to use the tool with the Mover6, was adapting the first version of the tool (used with the Braccio) to the Mover6 robotic arm. I did this by rewriting the robotic control code into the design tool code. In this section I discuss this process, as well as the issues that occurred.

5.4.1 Input handling

During the build process, the user starts a new step by clicking on the step-button. This is processed on the Arduino and sent through to the tool over a serial connection. The Stephandler class in the tool parses the serial message and sends the right command from the build file to the robot controller.

Synchronisation issues

The Stephandler class continuously parses incoming serial messages. When a message is received, it is sent through to the robot controller for processing. While the robot command is executed, the parsing of incoming messages is halted and all incoming messages are buffered. When the robot command is finished, the parsing continues and all buffered messages get parsed. The Stephandler will immediately start the next step if there is a serial command buffered, which is unwanted behaviour. The user can press the step-button multiple times in case he thinks the step did not start, but the next step should only be able to start when the current step is finished. A second issue is that the step-button will sometimes register one press of the button as multiple presses, and thus start multiple steps after each other.
My solution to this issue is making the processing of the commands work asynchronously. This way, a new command is sent to the robot controller and the Stephandler will immediately continue with processing serial messages. Any incoming command during the time that a step is executed, will now be ignored. It is implemented with the use of C# Background workers. A more robust solution for this issue is rewriting the step handling code with asynchronous message handling in its core functionality.

5.4.2 Horizontal and vertical end effector position

The CPRog tool does not support changing between the horizontal and vertical (0 and 90 degree angle) end effector when controlling the robot with Cartesian coordinates. The reason is that the robot reaches wrist singularity, which stops the movement. Trying to avoid the wrist singularity is not possible because of joint angle limitations.
I solve this by switching between the two positions, by controlling the robot with the joint angles. The inverse kinematics calculation is not used, so wrist singularity is not an issue. The switch between the vertical and horizontal position is straight forward, only joint 5 needs to change. After switching between the positions, the robot can be controlled with Cartesian commands again.

5.4.3 Plate orientation

The build file contains the plate orientation, which indicates if a plate should be placed horizontal or vertical on the model. To switch between these positions, the angle of joint 6 in the robotic arm needs to change. This is however controlled by the Euler angles as well, which requires a transformation from the orientation angle into the Euler angles.
Setting joint 6 into the right orientation is done by changing the joint angles directly and setting the Euler angles to the predefined values. After switching, the Cartesian coordinate commands can be used.
The disadvantage of this solution is that the Mover6 only supports horizontal or vertical plate position. The tool supports plate rotation between 0 and 90 degrees, so this is lost with the Mover6. It is not an issue for the test, because the corner pieces only support orientations of 0 or 90 degrees.

5.4.4 Robot commands

The Mover6 robot is controlled by sending commands to the robot to execute a specified program. The programs are created by the design tool with the information from the build file. The Cartesian coordinate position from the build file is reused for the program. The end effector angle and the plate orientation are used to determine the predefined Euler angles, determined as described before. The end effector angle is the angle on which the part needs to be attached to the model. The plate orientation determines the orientation of the plate on the model (figure 3.9 shows the possible orientations).
Every time a program is executed on the robot, the tool waits until the program is finished before continuing in the code. This ensures that a program is finished before anything else happens, which enables me to use multiple programs for a step in the build process.

5.4.5 Build platform position calculation

The design tool does not calculate the linear movement of the build platform, because the position is dependent on the robotic arm that is used. The Mover6 does not have a way to test if joint limits are reached, which is why the platform position is calculated. The calculation is faster and more precise than the iterative testing of the possible positions.
The idea behind the position calculation is to move the end effector to a fixed position, and move the build platform against the end effector. The calculation uses the x-value of each position. Before the position can be calculated in the tool, determine the following base positions:

- Determine the end effector position in which the robotic arm can best hold the plate under the current angle (see figure 5.7).
- Determine the base position of the build platform. The end of the base plate on the holder should be against the plate that the end effector is holding (see figure 5.7).
- Determine the base position in the design tool. This is the position of a plate in the tool as if it will be placed in the previous step (see figure 5.7).

The position that the holder moves to is calculated as follows:

1. Calculate the offset of the plate in the tool as follows:
   Base position in the tool - current plate position.
2. Calculate the new build platform position as follows:
   Base position of the platform + offset in the object.

5.4.6 Stepper motor control

The design tool controls the build process, which includes sending the platform to the right position and rotation. The serial connection between the tool and the Arduino is used to send messages to the Arduino. The message is parsed on the Arduino and executed. The command contains the required position and rotation for

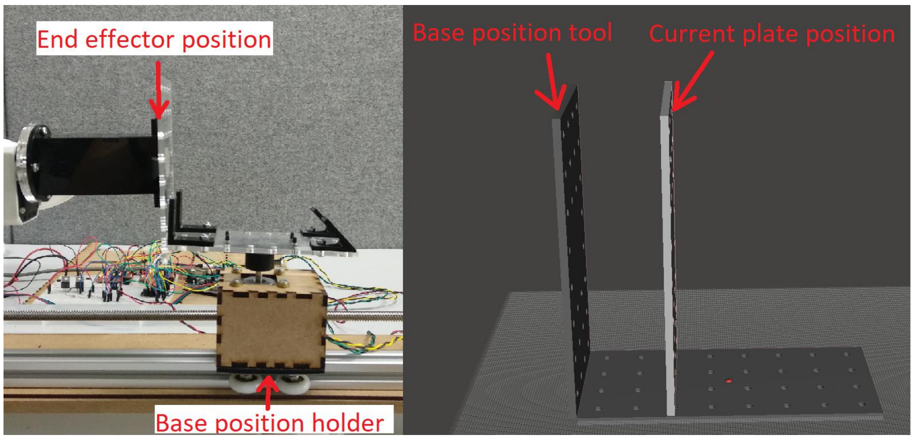


Figure 5.7: Visualisation of the parameters in the build platform position calculation. The end effector position and the corresponding base position of the platform on the left. On the right the base position in the tool and the current plate position.

the platform. The Stephandler in the tool waits for a command completed message from the Arduino before the next step in the build process can start.

With these implemented parts of the design tool, the Mover6 can be controlled and used to build the example models from the first version of the tool.

5.5 Angled plates support

The design tool supports parts under an angle of 0, 45, 90 and 135 degrees. In chapter 3 it is explained how this is implemented in the tool. In this section I explain how parts at an angle of 45 and 135 degrees are supported with the Mover6. The end effector can be positioned in any angle between 0 and 90 degrees. The 135 degree angle cannot be reached because joint 5 is too limited in range, but parts under a 135 degree are still possible.

5.5.1 45 degree angle

The design tool gives the required position for the part, the orientation and the end effector angle. As explained in section 5.2.2, the transformation to Euler angles is predetermined. Moving from the part pick up position to the hold position needs to be done with care, because the end effector is close to wrist singularity. Therefore a joint angle command is used to switch the end effector into the 45 degree angle.

5.5.2 135 degree angle

The end effector cannot hold a part at an angle of 135 degrees, but the inverse angle of 45 degrees is possible. Instead of holding the plate in the position at a 135 degree angle, the plate position at a 45 degree angle is used. The way the robot brings the plate has to change, because the plate needs to be attached to the outside of a corner piece. The robotic arm moves over the corner piece and moves back to hold the plate against the back of the corner piece.

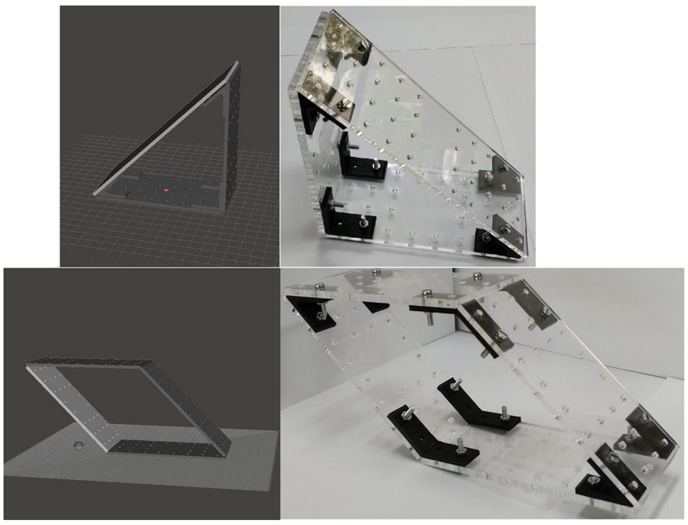


Figure 5.8: Two complex models that are designed with the tool (on the left) and built with the Mover6 (on the right).

This way both plates at an angle of 45 and 135 degrees are supported, although the end effector cannot reach a 135 degree position.

5.6 Benefits

The benefits of using the Mover6 robotic arm are the flexibility, the increased range and the higher payload. The flexibility and increased range make it possible to build bigger models and to support plates at a 45 and 135 angle. The higher payload enables the usage of the suction cup end effector, which is too heavy for the Braccio robotic arm. The suction cup enables the robot to build models with custom size plates. Otherwise the user would have to modify each plate before it could be used with the robotic arm.

5.6.1 Example models

The Mover6 implementation with the tool is tested by designing and building more complex models. These include angled plates at different orientations and even custom plates. Figure 5.8 shows the models I designed and built.

5.7 Limitations

The following limitations occur with the current implementation of the design tool with the Mover6 robot arm.

5.7.1 Advanced robot control

Controlling the 6 DoF robotic arm is more complex, because it requires the use of Euler angles which are not given by the design tool. The following limitations with the implementation are a consequence of this.

Manual transformation Euler angles

The Euler angles are determined by manually setting the end effector angle in the CPRog tool and saving them. This way I can bypass the calculation from the end effector angle to the Euler angles, because I could not find a mathematically correct way of doing it. This is a solution that works for the parts supported in the design tool, however it is not an optimal one. For example, adding a different corner piece angle would include finding the corresponding Euler angles and creating another edge case for it in the robot controller.
If the design tool would give the Euler angles, the complexity of the robot controller would be lower and easier to understand and test.

Singularity avoidance

Avoiding the singularities is a trial and error process: whenever a singularity occurs, the path of the robotic arm is changed and tested again. Determining if all possible positions of the design tool are possible is hard, because this would mean testing each and every position.
The problem is that the execution of a program by the CPRog tool stops when a singularity is reached, and therefore the control over the robot is lost. To avoid singularities I could look for better positions for the build platform and the robot, which reduces the chance of a singularity occurring. The 45 degree end effector position is close to the wrist singularity, which does not change when the platform position is changed. A second solution is to implement a way to detect when the CPRog tool indicates that a singularity occurs, and then send a joint angle command to move the robotic arm out of the singularity.

5.7.2 Robot position bug

The coordinates in the CPRog tool are mapped one-to-one in millimetres on the physical position of the end effector. This allows me to calculate the pick up position for the custom size plates, based on the size of the plates. Given that the plates are always placed against a fixed starting position.
An issue occurs when the end effector x-position gets lower than 100. The one-to-one mapping is not accurate anymore, therefore the end effector position in the tool does not match the physical position. Consequently the suction cup misses the centre of the plate and is not able to pick up the plate.
I noticed this issue only at the end of my thesis, because it only occurs when making a model that uses more than three plates and at least one custom size plate. This means that I did not have the time to investigate the source of the issue or contact Commonplace Robotics about the bug.
I solved the issue by subtracting a certain distance from the offset when the x-position is lower than 100, which makes the end effector move close enough to the centre of the plate to pick it up.

5.8 Conclusion

The flexibility and increased range of the Mover6 make it possible to create more complex models with the design tool. Instead of the basic boxes that could be cre-

ated with the first version of the tool and the Braccio, it is now possible to create multiple complex models. This is why the design tool is expanded with the support for custom size plates, and the support for 45 and 135 degree corner pieces is implemented.
The support for models with custom size plates increases the complexity of the design tool even more. The usage of custom plates is possible because of the suction cup, which enables the robot to pick up parts without modifications from the user. The increased flexibility and reach of the Mover6 make it possible for the robotic arm to pick up the plates from all the different positions.

The goal of using the Mover6 robotic arm to create more complex models designed in the tool, is achieved.

Chapter 6

User study

In this chapter, I describe the small formative study that was conducted as the final part of my thesis. The goal of the user study was getting an impression of the usability of building a model with a robot arm in combination with the tool. The system discussed in chapter 5 was used. Only the assembly of a model with the robot was tested. The assembly instructions were given to the user in the design tool. The design of a 3D model with the tool was not tested, for its usability is not the focus of my thesis.
The user study consisted of a brief overview of the design tool, building the dice tower model with the robot (see figure 6.1), and a questionnaire in combination with an interview. I let the participants make the dice tower because it contains a plate at an angle, uses custom size plates, and is more complex than a basic tray.

6.1 Method

In this section, I discuss the method with which the study was conducted.

6.1.1 Participants

The study was conducted with four participants that all have come in contact with a Makerspace before. Three of them made something at a Makerspace at least once. Participants that have been at a Makerspace were chosen because it is one of the possible places the robotic arm setup could be used, because it contains the equipment to 3D print and laser cut. Three of the participants are computer science

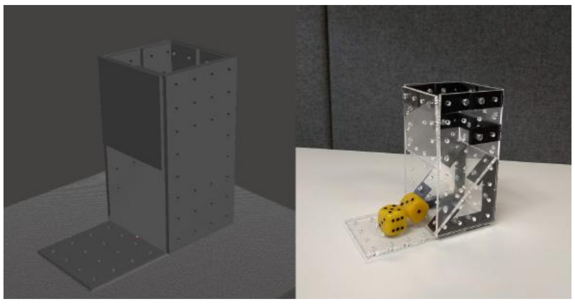


Figure 6.1: The dice tower model that the participants of the user study built.

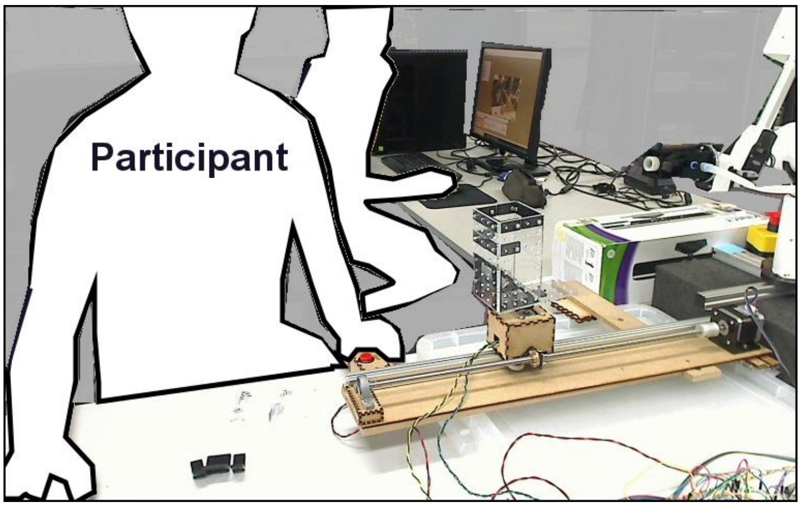


Figure 6.2: The setup during the user study, with on the left the participant and on the right the observer.

students and one is working as a researcher at the Expertise centre for Digital Media (EDM). They are men between 23 and 35 years old.

6.1.2 Apparatus

During the user study, I recorded the participants using a webcam facing the user. This webcam was connected to a second computer setup behind the participant (see figure 6.2). On this computer I made notes in real time using Techsmith Morae Recorder [31].

6.1.3 Procedure

The user study consists of the following three parts.

Introduction

The study starts with getting the informed consent of the participant. I prepared and printed an informed consent form which he could read and sign. It included the request to record the session. The full informed consent form is included in the appendix 6.
Next I explained briefly the idea behind the design tool, and showed him how it can be used to design a 3D model. I opened the dice tower model which the participant would build. I explained to each participant where the instructions during the assembly process are shown, and that the model on the screen visualises the progress and shows which corner pieces should be placed.

Assembly process

I started the assembly process by giving the participant the base plate of the model and showing him where the corner pieces, bolts, nuts and the screwdriver are. The last instruction I gave was to start with attaching the corner pieces to the base plate, as shown in the tool. All other instructions were given by the design tool and it was

up to the participant to execute them correctly.
During the assembly process, I watched the recording of each participant to add notes in real time. The participants were allowed to ask questions during the process if they were not sure how to proceed.

Questionnaire and interview

After the participant was finished with assembling the model, a one-page questionnaire was given. The questionnaire asked about their connection to a Makerspace and about their experience during the assembly, which they answered on a Likert scale from 1 to 5.
I concluded each experiment with a short, semi-structured interview, in which I asked some open ended questions about their experience. I did this with an interview instead of through the questionnaire, so I could ask any follow-up questions if necessary.
Both the questionnaire and the open ended questions are included in the appendix B.

6.2 Results

My findings during the user study are divided into two categories: observations I made during the study, and findings from the questionnaire and interview.

6.2.1 Observations

The position of the build platform and the robotic arm are not required to be perfect. It is sufficient if they are close enough together, on the condition that the build platform and the robotic arm allow small manual adjustments. For example moving the robotic arm by hand, rotating the model or tilting the model.

The robotic arm holds the plate in the required position, after which the user attaches the plate to the model. The idea is that the user attaches the plate with just enough bolts so that the plate stays in place, after which the robotic arm can move away. Moving the robotic arm out of the way gives the user more space to work.
This was not clear to most of the participants, especially in the first step. Some going as far as to attach all the corner pieces on the plate before moving the robot arm away.

The instructions in the tool were indicated as clear by the participants, but they experienced issues during the assembly because they did not read the instructions, missed some information or some information was not available.

The first issue was when the robotic arm picks up a plate. It waits with holding the plate in the desired position until the user presses the step-button again. I included this step, so that the user can make a last check before continuing with the next plate.
This was explained in the instructions but it was not clear to any of the participants. They expected the robotic arm to pick up a plate and hold it in one action. Thus when the robotic arm stopped they were not sure what to do. Some of the

participants thought they had to attach corner pieces to the plate already.
Two of the participants asked me how to proceed in this step. One of them later indicated that he did not notice the instructions immediately because he was not sure if the step was finished. The other participants read the instructions again and saw that they should press the step-button again.

The second issue occurred when the participants were finishing the current step, by tightening the last plate that is added and placing the required corner pieces. During this step, the user is allowed to rotate the model to a position in which he can better reach it. The user however has to return the holder to the original position, before starting a new step. The stepper motor that is used in the build platform has no way of knowing if the user rotated the platform. This means that the rotation is expected to be executed from the original position.
The instructions in the tool do not explain this to the user, which is why three of the participants expected the platform to always rotate to the correct position. One of the participants did not rotate the platform at all because he expected that the platform should stay in his original position.

6.2.2 Questionnaire and interview

The overall experience of building a model with the robot arm was indicated as positive by the participants and they all experienced the robot as an assistance during the assembly. Three out of four would use the setup again to build another model. The fourth participant prefers to do it without the robot, but he saw potential in it for people with less technical knowledge or for children.
None of the participants experienced the system as too slow, when asked about it. One participant used the time when the robot was picking up the next plate to read the instructions and prepare for the next step.
All the participants would use the design tool to design their own 3D model. One participant indicated that it would be easier than creating a 3D design from scratch using other CAD software.

All of the participants indicated that they sometimes had trouble inserting a bolt into a hole, because the end effector was in the way. Especially on the smaller plates. A second issue that the participants indicated was with the very small bolts and nuts, which made it hard to tighten them. The issue was most common in combination with the corner piece at a 45 degree angle. These corner pieces have little room left to insert the bolt or hold the nut in the right place. Some of the positions in which a corner piece had to be placed were hard to reach with your fingers, which also made it harder to attach the plate.

6.3 Conclusion

The study that was executed was only a small one, but it still gave some valuable information. It showed that there is potential in using such a setup to aid users in building a model, even for more technical users. The way the robot arm and the build platform work together to show the user how the model should be built, is helpful. One of the participants even indicated in the interview that he experienced

it as building with a third hand, which is the goal of the project.
The user study also showed that the position of the robotic arm and the platform do not have to be perfect, as long as manual adjustments are possible. This makes the assembly possible even with less precise robotic arms.

During the study, an early indication of issues associated with the process occurred. Most of these could be fixed by improving the instructions and streamlining the build process. The participants provided insights in improvements that can be made to the tool and the process, which are described in chapter 7.

For me the user study was a success. It was not executed with enough participants to make any real conclusions, but it gave valuable insights to my current work. It showed that there is potential in it and ways in which the current setup could be improved.

Chapter 7

Future work

The design tool that is implemented with the Mover6 can still be expanded with more functionality. In this chapter, I list the possible expansions that can be implemented in future work. These are expansions that were discussed during my thesis but were not implemented, or findings and suggestions from the user study.

7.1 Design process

The following expansions can be made to aid the novice user even more during the design process of a 3D model.

7.1.1 Build order

The order in which the model will be built with the robot, is defined by the order in which the user places the parts. This requires the novice user to have an idea of the model that he wants to build before designing it in the tool, so he can design it in the right order. A design that consists of parts in the wrong order requires the user to redesign the model in the correct order.
This increases the barrier of entry for the design tool, because it requires the user to think about the best build order of the 3D model. The ease of use of the design tool would be higher if the best build order is determined by the tool.

7.1.2 Corner pieces

The usage and the design of the corner pieces can be expanded in multiple ways.

Automatic placement

Instead of requiring the novice user to place the corner pieces in the right positions, the design tool determines what the best position is. The user does not have to think about the placement anymore and the tedious placement process is eliminated. The corner pieces are placed when the user clicks the 'Place corner pieces' button. The user can still place and delete corner pieces himself with the existing UI.

7.1.3 Custom plate extraction

An improvement that can be made is to optimise the way the plates are placed. They are placed next to each other, which increases the amount of Plexiglas that is required. The optimal placement is a variant of the bin packing problem. The robot control needs to change so it is able to pick up the plates given by the layout of the generated file.

Hole placement

The generation of the custom plates file can be expanded to only create the required holes in the plates. Instead of having the plate full of holes, there would only be holes where a plate is attached to another plate or to a corner piece.
The final design would not consist of construction kit parts anymore. When designing a model without custom plates, the original construction kit parts are used.

Angle support

The corner pieces are limited to a 45 and 135 degree angle which could be expanded with more angles, for example 30, 60, 120 and 150 degree angles. This would increase the complexity even more of the model that can be designed.

Custom corner pieces

Instead of only using predefined corner pieces, the design tool can be expanded by creating custom corner pieces. The tool would need to automatically place the corner pieces and adjust the angle to fit. Before the user can start the build process, he needs to 3D print the required corner pieces. The .stl file is generated by the tool. During the design process the tool indicates which corner piece to use.
For this to work, the following issues need to be taken into consideration: First, the robot arm that is used needs to be able to support all possible angles. Secondly, there should always be enough space to insert and tighten the bolts and nuts. Thirdly, the user needs to be able to differentiate between all the corner pieces. This requires labelling them during 3D printing, and the user having to find the required piece each time.

7.1.4 Different plate types

The design tool is limited to rectangular plates. This could be expanded in two possible ways.

Cut interactions

The novice user can only add new parts to the design, not remove a part of an existing plate. For example making a square hole in a plate that represents a window. To do this the tool needs to support a cut operation on existing plates in the model. The final model is then built out of custom plates.

Non rectangular plates

Instead of only supporting rectangular plates, the design tool could also support other types of plates, for example triangles. This would increase the complexity of models that can be designed with the tool, but it would also increase the complexity of using the tool.

7.1.5 Variable base plate

Every model that is built with the tool starts with the same base plate. The user should be able to change the size of this plate, which enables different shapes of models to be built. This can be implemented in two ways:
The interface of the tool could give the user the option to change the size of the base plate, within given bound. For example a minimal size so that the plate still fits on the build platform.
A second option is to start without any base plate. The user can choose which type of plate he wants as the base plate.

7.2 Robot interaction

The robot interaction using the design tool, is limited to pressing the step-button to start the next step. The focus of my thesis is not on these interactions, but they can enhance the way the tool is used. The user study also showed ways in which the instructions given to the user could be improved.

7.2.1 Instructions

The instructions given to the user during the build process can be improved so that they are easier to understand, and attract the attention of the user. The improvements are based on the findings of the user study.

Text improvements

The font size of the instructions text can be increased to make it more notable and bold text can be used to highlight the most important parts. With these improvements the instructions should grab more attention from the user.

Corner piece highlighting

The tool shows the model of the current step in the build process, which also shows the corner pieces that have to be placed on the model. The corner pieces that are already placed on the model are also shown, which makes it harder for the user to see at a glance which corner pieces should be placed. A solution would be to highlight the corner pieces that should be placed in the current step, by changing the colour. This also helps differentiate the corner pieces from the plate, because they are all the same grey colour.

Instructions on platform rotation

During the assembly, it can be useful to rotate the platform when attaching the different parts. It is however important that the user returns the platform to the original position before starting the next step. Currently the user does not know that this is required. Possible solutions for this issue are:

- Instructing the user to always return the platform to the original rotation position. The disadvantage is that the user must remember the original rotation. The model in the tool could also rotate to indicate the original rotation.
- Instructing the user to always return the platform to face the robotic arm. To do this the build calculation code needs to be adjusted. The disadvantage being that the user needs to remember this.
- Tracking the rotation of the platform during the assembly process, so that the current rotation is known by the tool. This could be done with a camera above the setup, or by modifying the platform to count the rotation.

Move away the robotic arm

Indicate to the user that the robotic arm can move away as soon as the plate is attached to the model. When the robot is not in the way of the user, it is easier to insert the bolts and tighten the nuts.

Extra step before holding plate

The step in between the robotic arm picking up a plate and holding the plate in the required position, was confusing to the participants of the user study. This can be solved in two ways:

- Combining the two steps into one step, removing the doubt about what to do.
- Indicating more clearly in the instructions what to do in that moment. This requires the user to read the instructions, but keeps the possibility for the user to do a last check before continuing with the next step.

7.2.2 Build process

The way the user progresses to the next step can be implemented in different ways, instead of only with the step-button.

Foot pedal

Progressing to the next step in the build process requires the user to click the step-button, which can only be done if the user has a hand available. A simple solution is to use a foot pedal. This allows the user to progress to the next step without using his hands.

Undo hold step

The tool does not support cancelling the hold step of the robotic arm. For example the user commands the robot to hold the next piece in place, but then notices that he forgot to attach a corner piece. In that case, attaching the missing corner piece is easier without the robot holding the plate. After attaching the corner piece, the user commands the robot to hold the plate in the desired position.
The undo action could be implemented with a separate button or with a gesture.

7.2.3 Implicit gestures

Using an overhead camera, the user can be tracked. The tool could try to deduce when the user is finished with a step in the process, and when to start a new step.

Automatically moving away

Moving the robotic arm out of the way when the plate is attached to the object, which ensures that the user has enough room to tighten the bolts and nuts.
The implicit gesture for this would be when the user picks up the screwdriver that is used to tighten the last part of the bolt.

Automatically starting next step

The robotic arm can start the next step in the build process when the user is finished with the current step. This would mean starting with picking up the next plate and positioning the build platform.
The implicit gesture for this is determined by the hand position of the user. When the users' hands are outside of a rectangle created from the position of the corner pieces, and the bolts and nuts, it is very likely that he is finished with the step.

Rotation adjustment

The position in which the robotic arm holds the plate, could be adjusted to increase the reachability for the user. Instead of the user having to reach around to tighten the bolt, the robotic arm and the platform could rotate. This way it is easier for the user to reach the bolt.

7.3 Conclusion

As you can read in this chapter, there are still a lot of ways in which the design tool and the build process can be expanded. Some of the possibilities have a higher priority than others, for example the custom plate extraction is more important than supporting a cut interaction. Expanding upon the design and build process increases the usability and ease of use for the novice user, which is where the potential of the tool lies.

Chapter 8

Conclusion

The design tool is implemented with two different robotic arms: an Arduino Braccio and a Commonplace Robotics Mover6. The first basic version of the tool was made with the Braccio and the more advanced version was implemented with the Mover6.

The first version of the design tool implemented with the Arduino Braccio, supports the construction kit in combination with 90 degree angle corner pieces. The result was demoed on multiple occasions and was well received.
The main limitations are the limited reach and flexibility of the Braccio robotic arm. The limited flexibility is due to the 5 degrees of freedom of the Braccio. This made it harder to implement different angles of corner pieces. The limited reach of the robotic arm is solved by the build platform that is used. The height in which the robotic arm can hold a plate is still limited. Fixing this requires the build platform to be able to move up and down. The custom end effector that is designed uses electromagnets to pick up the plates, which requires hot-gluing small metal pieces to them.
The main advantage of the Arduino Braccio is the price, which make the robot arm as well as the design tool more accessible.

The second version of the design tool (as described in chapter 3) is implemented in combination with the Commonplace Robotics Mover6 and the build platform.
The first limitation is the missing support for Euler angles in the build calculation of the design tool. Euler angles are required to control the end effector of the Mover6 robotic arm. My solution to this issue is to use predefined Euler angles for the possible angles of the end effector. The second limitation is the occurrence of singularities when moving the robotic arm, which are a consequence of a 6 DoF robotic arm. Singularities limit the possible paths that can be taken with the robotic arm while moving.
The main advantage of the Mover6 is the extra degree of freedom and increased reach of the robotic arm. The design tool supports corner pieces with angles of 45 and 135 degrees and the usage of custom size plates. These make it possible to build more complex models with the design tool. The use of custom size plates is made possible, because the Mover6 has enough payload capacity to hold the custom suction cup end effector. This enables the robotic arm to pick up plates without the user having to modify them.

A small informative user study with four participants was executed as a last part of my thesis. In this study, the participants were asked to build the dice tower model (see figure 6.1) together with the Mover6 robotic arm. After the assembly they were questioned with a short questionnaire and an interview.
The participants experienced the robotic arm as an assistance during the assembly and three of the four would use the setup again to build a model. The study provided insights in the issues occurring during the assembly process, for example the instructions were not always clear. The user study did show the potential in my proposed work.

Chapter 7 describes a lot of possible ways in which the design tool and build process can be expanded. If given the chance, I would implement the following options: First, calculating the best order in which to build the object, which makes it even easier for the novice user to design his 3D model. Secondly, improving the instructions given to the user during the assembly process. Thirdly, look for a way to calculate the Euler angles during the build calculation. This would allow more angles supported by the design tool, and simplify the robot programming code.

The goal of assembling a model together with a robot is achieved with both the Braccio and the Mover6 robotic arm. The tool supports not only the assembly with the robotic arm, but supports the novice user with the design process as well.

Appendix A

Informed Consent

Informed Consent User Study

De bedoeling van deze user study is dat u samen met de robotarm een model opbouwt met behulp van de tool die ik gemaakt heb. Het doel van deze test is het testen van de bruikbaarheid van het systeem en niet om te testen hoe goed u met het systeem werkt. U mag ten alle tijden de test onderbreken of stoppen. Alle data die gedurende de test verzameld wordt zal geanonimiseerd worden.

Indien u hiermee toestemt zullen er tijdens de evaluatie video opnames gemaakt worden. Deze beelden mogen gebruikt worden om uw gebruik van het systeem te observeren. Noch de volledige opnamen, noch fragmenten van de opname zullen publiekelijk gemaakt worden.

Alle data die ik verzamel zal enkel bewaard worden tot na publicatie van mijn eindresultaat, met een uiterlijk limiet van 6 maanden. De data is alleen toegankelijk voor mij (Bram van Deurzen), mijn promotor Prof. Dr. Kris Luyten, en mijn copromotors Dr. Jan Van den Bergh en Prof. Dr. Raf Ramakers.

Als u akkoord gaat met deze voorwaarden, teken dan hieronder.

Naam:

Datum:	Naam onderzoeker: Bram van Deurzen
Handtekening:	Handtekening onderzoeker:

Appendix B

Questionnaire and questions

Vragen user study

Persoonlijke informatie:

Leeftijd:

Geslacht:

Studie / werk:

Vragen over het proces:

Gelieve het antwoord te omcirkelen.

Bent u al ooit in een makerspace (of Fablab) geweest?

Ja Nee

Indien ja, hebt u er al ooit iets gemaakt?

Ja Nee

De robotarm was een hulp tijdens het assembleren.

Helemaal oneens	1	2	3	4	5	Helemaal eens

De instructies in de tool zijn duidelijk.

Helemaal oneens	1	2	3	4	5	Helemaal eens

Het bouwen van het model ging beter omdat de robot de plaat op de juiste plaats vasthoudt.

Helemaal oneens	1	2	3	4	5	Helemaal eens

Uw rol tijdens het assembleren is duidelijk.

Helemaal oneens	1	2	3	4	5	Helemaal eens

De rol van de robot tijdens het assembleren is duidelijk.

Helemaal oneens	1	2	3	4	5	Helemaal eens

Open vragen (tijdens interview):

Zou u de robot opnieuw gebruiken om een model te assembleren? (ja/nee en waarom)

Zou u zelf een model willen ontwerpen met de tool?

Wat was uw ervaring om iets samen met een robot te bouwen?

Was er iets onverwachts gebeurd gedurende het proces?

Zijn er problemen die u ondervonden hebt gedurende het proces?

Wat vond u van de snelheid van het bouwproces?

Heeft u suggesties voor verbeteringen?

Appendix C

Nederlandse samenvatting

Breng je idee tot leven samen met een robot

Heb je al ooit een voorwerp in gedachten gehad dat je graag zou willen maken, maar weet je niet hoe je eraan moet beginnen? Of krijg je nu een idee? Mijn systeem helpt je om het werkelijkheid te maken.

Het systeem is speciaal ontworpen om onervaren gebruikers te ondersteunen. Met behulp van de design tool ontwerp je jouw idee. Daarna bouw je dit model samen met de robot, waarbij deze fungeert als derde hand.

THE BEST OF BOTH WORLDS

Er bestaan veel vooroordelen wanneer het gaat over robots. Zo zouden ze jobs kunnen afnemen van mensen, of veroveren ze zelfs de wereld in sommige films.

Gelukkig is dit niet hoe men in onderzoek of in de industrie naar robots kijkt. Hier wordt er gezocht naar manieren waarop mensen en robots kunnen samenwerken, om zo het beste uit elkaar te halen. Ze hebben namelijk beiden unieke eigenschappen die elkaar ondersteunen. Een robot kan continu geconcentreerd blijven en zware taken uitvoeren, zonder vermoeid te geraken. Een mens kan daarentegen zeer goed omgaan met veranderingen in de omgeving en de robot bijsturen indien nodig.

Mijn thesis heeft als doel om de interactie tussen mens en robot op een kleine schaal uit te voeren, bijvoorbeeld in een plaatselijke Makerspace. De kennis van de robot wordt gebruikt om je te begeleiden doorheen het bouwen, terwijl je zelf de onderdelen monteert.

VAN IDEE TOT MODEL

Een visuele voorstelling van de stappen die je doorloopt met het systeem, kan je zien in figuur C.1.

Het begint allemaal met een idee wat je tot leven wilt brengen. In dit voorbeeld een ‘dice tower’.

De eerste stap is het maken van een 3D model, samen met mijn design tool. In deze tool kan je met behulp van platen uit een bouwpakket een 3D model opbouwen, door de platen op de juiste plaats in het model te positioneren. Er kunnen ook platen van een gekozen grootte ingevoegd worden. De design tool ondersteunt platen onder een hoek van 45, 90 of 135 graden, met behulp van 3D geprinte hoekstukjes.

Nadat je klaar bent met het maken van je 3D model, snijd je de verschillende platen uit met een lasercutter. Hiervoor maakt het systeem automatisch een bestand aan, dat rechtstreeks gebruikt kan worden door de lasercutter.

Daarna leg je de platen, hoekstukjes, bouten en moeren klaar bij de robotarm. Nu heb je alles ter beschikking om samen met de robotarm je 3D model op te bouwen. Hiervoor worden in de design tool automatisch instructies voor de robotarm berekend, op basis van je 3D model. De tool zal je tijdens het bouwen instructies geven.

Samen met de robotarm ga je stap voor stap het model opbouwen. In de design tool krijg je te zien waar je ieder hoekstukje moet plaatsen, voordat de robotarm de volgende plaat mag brengen. Het platform waarop je het model bouwt zal naar de juiste positie bewegen. De robotarm zal de volgende plaat opnemen en deze op de juiste plaats tegen het model houden. Hierdoor kan je met beide handen de plaat vastmaken aan de hoekstukjes.

Wanneer alle stappen doorlopen zijn, kan je het model van het platform nemen en is het klaar om te gebruiken.

SAMEN STERK WERK

Het doel van mijn systeem is om aan te tonen dat je samen met een robotarm aan een model kan werken. De bijhorende design tool maakt het mogelijk om een 3D model te maken, zonder dat je kennis moet hebben van 3D modelleren. De robot zal je tijdens het opbouwen van je eigen model helpen, door de platen van het model op de juiste plaats vast te houden. Hierdoor kan jij zelf met twee handen het model monteren.

"Het voelt echt aan alsof je een derde hand hebt die je helpt tijdens het bouwen", zoals een van de deelnemers omschreef tijdens de gebruikersstudie van mijn systeem.

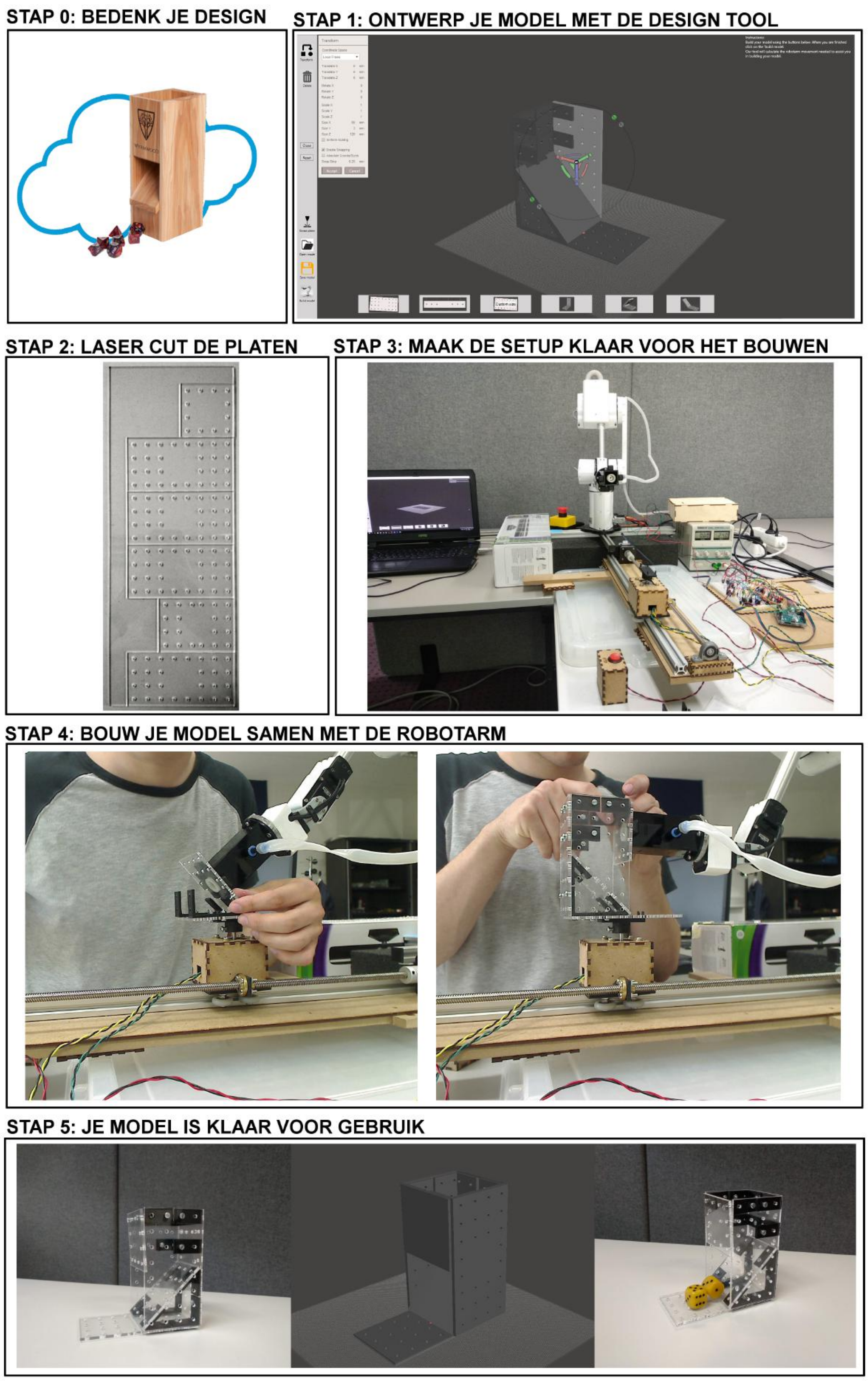

STAP 0: BEDENK JE DESIGN
STAP 1: ONTWERP JE MODEL MET DE DESIGN TOOL
STAP 2: LASER CUT DE PLATEN
STAP 3: MAAK DE SETUP KLAAR VOOR HET BOUWEN
STAP 4: BOUW JE MODEL SAMEN MET DE ROBOTARM
STAP 5: JE MODEL IS KLAAR VOOR GEBRUIK

Figure C.1: Stappen in het proces

Bibliography

[1] ANSI and RIA. “American National Standard for Industrial Robots and Robot Systems — Safety Requirements Robotic Industries Association”. In: . (1999), p. 75. URL: http://www.paragonproducts-ia.com/documents/RIA%20R15%7B%5C_%7D06-1999.pdf.

[2] Arduino. *Tinkerkit Braccio robot*. URL: https://store.arduino.cc/tinkerkit-braccio (visited on 03/10/2018).

[3] Autodesk. *Autodesk Meshmixer*. URL: https://www.meshmixer.com/ (visited on 03/07/2018).

[4] Khelifa Baizid et al. “IRoSim: Industrial Robotics Simulation Design Planning and Optimization platform based on CAD and knowledgeware technologies”. In: *Robotics and Computer-Integrated Manufacturing* 42 (2016). DOI: 10.1016/j.rcim.2016.06.003. URL: http://daneshyari.com/article/preview/413928.pdf.

[5] Commonplace Robotics. *Commonplace Robotics Mover6*. URL: http://www.cpr-robots.com/products/mover6.html (visited on 03/10/2018).

[6] Commonplace Robotics. *CPRog Updates - CPR-robots wiki*. 2018. URL: http://wiki.cpr-robots.com/index.php?title=CPRog%7B%5C_%7DUpdates (visited on 05/01/2018).

[7] Commonplace Robotics. *Robot Interface CRI*. 2016. URL: http://www.cpr-robots.com/download/CRI/CPR%7B%5C_%7DRobotInterfaceCRI.pdf.

[8] empira Software GmbH. *PDFsharp & MigraDoc*. URL: http://www.pdfsharp.net/ (visited on 06/06/2018).

[9] Krzysztof Foit and Grzegorz Ćwikła. “The CAD drawing as a source of data for robot programming purposes – a review”. In: *MATEC Web of Conferences* 94 (Jan. 2017). Ed. by G. Oancea and M.V. Drăgoi, p. 05002. DOI: 10.1051/matecconf/20179405002. URL: http://www.matec-conferences.org/10.1051/matecconf/20179405002.

[10] Bradley Hayes and Brian Scassellati. “Challenges in Shared-Environment Human-Robot Collaboration”. In: (2013). URL: https://pdfs.semanticscholar.org/19fb/e18e8da489b17ebb283ddc7e72af7c3ffd32.pdf.

[11] M J D Hayes, M L Husty, and P J Zsombor-Murray. “Singular Configurations of Wrist-Partitioned 6R Serial Robots: a Geometric Perspective for Users”. In: *Transactions of the Canadian Society for Mechanical Engineering* 26.1 (2003), p. 15. URL: http://faculty.mae.carleton.ca/John%7B%5C_%7DHayes/Papers/KR15SingCSME.pdf.

[12] Mario Hermann, Tobias Pentek, and Boris Otto. "Design principles for industrie 4.0 scenarios". In: *Proceedings of the Annual Hawaii International Conference on System Sciences*. Vol. 2016-March. IEEE, Jan. 2016, pp. 3928–3937. ISBN: 9780769556703. DOI: `10.1109/HICSS.2016.488`. arXiv: `arXiv:1011.1669v3`. URL: `http://ieeexplore.ieee.org/document/7427673/`.

[13] Casper Hofstede. *Arduino Braccio Robot Arm, Raspberry Pi Controlled with Arduino and SSC-32 – Portfolio C.M.Hofstede.* URL: `http://www.cmhofstede.nl/hobby-vrije-tijd/arduino-braccio-robot-arm-raspberry-pi-controlled-with-arduino-and-ssc-32/` (visited on 04/22/2018).

[14] International Federation of Robotics. "Industrial robots - definition and classification". In: (2016), pp. 25–34. URL: `https://ifr.org/img/office/Industrial%7B%5C_%7DRobots%7B%5C_%7D2016%7B%5C_%7DChapter%7B%5C_%7D1%7B%5C_%7D2.pdf`.

[15] International Federation of Robotics. "Service Robots - Definition and Classification WR 2016". In: (2016), pp. 9–12. DOI: `ISO8373:2012`. URL: `http://www.iso.org/iso/iso%7B%5C_%7Dcatalogue/catalogue%7B%5C_%7Dtc/catalogue%7B%5C_%7Ddetail.htm?csnumber=55890.%20https://ifr.org/img/office/Service%7B%5C_%7DRobots%7B%5C_%7D2016%7B%5C_%7DChapter%7B%5C_%7D1%7B%5C_%7D2.pdf`.

[16] Intertec. *Electromagnet magnetic (disconnected) 15 N 12 Vdc 1.8 W from Conrad Electronic UK.* 2018. URL: `https://www.conrad-electronic.co.uk/ce/en/product/1218314/Electromagnet-magnetic-disconnected-15-N-12-Vdc-18-W?ref=searchDetail` (visited on 04/18/2018).

[17] ISO/TC 299. *ISO 8373:2012 - Robots and robotic devices – Vocabulary.* Tech. rep. 2012. URL: `https://www.iso.org/standard/55890.html`.

[18] KUKA. *LBR iiwa — KUKA AG.* 2018. URL: `https://www.kuka.com/en-be/products/robotics-systems/industrial-robots/lbr-iiwa` (visited on 04/14/2018).

[19] Benjamin Lafreniere et al. "Crowdsourced Fabrication". In: *Proceedings of the 29th Annual Symposium on User Interface Software and Technology - UIST '16* (2016), pp. 15–28. DOI: `10.1145/2984511.2984553`. URL: `http://dl.acm.org/citation.cfm?doid=2984511.2984553`.

[20] Ilias El Makrini et al. "Design of a collaborative architecture for human-robot assembly tasks". In: *IEEE International Conference on Intelligent Robots and Systems* 2017-Septe (2017), pp. 1624–1629. ISSN: 21530866. DOI: `10.1109/IROS.2017.8205971`.

[21] Vittorio Megaro et al. "Interactive Design of 3D-Printable Robotic Creatures". In: *ACM SIGGRAPH Asia* (2015). URL: `https://ait.ethz.ch/projects/2015/RobotDesigner/downloads/RobotDesigner-Paper.pdf`.

[22] Microchip Technology. "MCP23017/MCP23S17: 16-Bit I/O Expander with Serial Interface". In: (2007). URL: `https://cdn-shop.adafruit.com/datasheets/mcp23017.pdf`.

[23] Yuki Mori and Takeo Igarashi. "Plushie: An Interactive Design System for Plush Toys". In: *ACM Trans. Graph. Article* 26.10 (2007). DOI: `10.1145/1239451.1239496`. URL: `http://doi.acm.org/10.1145/1239451.12`.

[24] Pedro Neto and Nuno Mendes. "Direct off-line robot programming via a common CAD package". In: *Robotics and Autonomous Systems* 61.8 (Aug. 2013), pp. 896–910. ISSN: 09218890. DOI: 10.1016/j.robot.2013.02.005. URL: https://www.sciencedirect.com/science/article/pii/S0921889013000419.

[25] Pedro Neto, J Norberto Pires, and A Paulo Moreira. "High-level programming and control for industrial robotics: using a hand-held accelerometer-based input device for gesture and posture recognition". In: *Industrial Robot: An International Journal* 37.2 (2010), pp. 137–147. URL: https://arxiv.org/ftp/arxiv/papers/1309/1309.2093.pdf.

[26] Pedro Neto et al. "High-level robot programming based on CAD: dealing with unpredictable environments". In: *Industrial Robot: An International Journal* 39.3 (Apr. 2012), pp. 294–303. DOI: 10.1108/01439911211217125. URL: https://www.emeraldinsight.com/doi/10.1108/01439911211217125.

[27] RepRap. *NEMA 17 Stepper motor - RepRapWiki*. 2018. URL: http://reprap.org/wiki/NEMA%7B%5C_%7D17%7B%5C_%7DStepper%7B%5C_%7Dmotor (visited on 04/21/2018).

[28] Rethink Robotics. *Baxter Collaborative Robots for Industrial Automation — Rethink Robotics*. 2018. URL: http://www.rethinkrobotics.com/baxter/ (visited on 04/14/2018).

[29] D. Rose. *Rotations in Three-Dimensions: Euler Angles and Rotation Matrices*. 2015. URL: http://danceswithcode.net/engineeringnotes/rotations%7B%5C_%7Din%7B%5C_%7D3d/rotations%7B%5C_%7Din%7B%5C_%7D3d%7B%5C_%7Dpart1.html (visited on 04/29/2018).

[30] Brian Schmalz. *Easy Driver Stepper Motor Driver*. 2010. URL: http://www.schmalzhaus.com/EasyDriver/index.html.

[31] TechSmith. *Usability testing with Morae*. 2018. URL: https://www.techsmith.com/morae.html (visited on 06/04/2018).

[32] Nobuyuki Umetani et al. "Pteromys: interactive design and optimization of free-formed free-flight model airplanes". In: *ACM Transactions on Graphics* 33.4 (July 2014), pp. 1–10. DOI: 10.1145/2601097.2601129. URL: http://dl.acm.org/citation.cfm?doid=2601097.2601129.

[33] Nobuyuki Umetani et al. "Sensitive Couture for Interactive Garment Modeling and Editing". In: *SIGGRAPH* (2011). URL: http://www.cs.columbia.edu/cg/pdfs/179-SC.pdf.

[34] Universal Robotics. *Collaborative Industrial Robotic robot Arms — Cobots from UR*. 2018. URL: https://www.universal-robots.com/ (visited on 04/14/2018).

[35] Eric W. Weisstein. "Euler Angles". In: (). URL: http://mathworld.wolfram.com/EulerAngles.html.